
Thesis Submitted for the degree of Doctor of Philosophy at the University of Leicester

by

Vassilios A. Tsachouridis M.Sc (Salford)
Department of Engineering
University of Leicester

February 2002
Abstract

A unified framework for the numerical solution and analysis of all algebraic quadratic matrix equations is presented in this thesis. In a global approach, it is shown how all systems of algebraic quadratic equations can be equivalently considered as special cases of a single generalized algebraic quadratic matrix equation. Hence, the thesis is devoted to the theoretical development of the numerical solution and analysis of this general matrix equation and to the design and implementation of relevant computer software. The essential tools for the developments of the numerical algorithms are the probability-1 homotopy methods. In addition, perturbation theory is used in order to conduct numerical analysis studies of the designed algorithms and to derive error estimates for the computed solutions. All the above are then implemented in software in the form of a MATLAB toolbox.

It is shown how the numerical solution and analysis of many design and analysis problems in engineering and science can be formulated as special cases within the proposed framework. Several problems are considered: classical and generalized algebraic Riccati equations, modern robust control system designs, second order matrix polynomial equations for the solution of the quadratic eigenvalue problem, computation of equilibrium points in noncooperative Nash games in economic systems, chemical kinetics modelling, computation of equilibrium points of quadratic dynamical systems (which often appear in bifurcation and in chaos theory). For the above cases, relevant numerical examples from the literature are used to illustrate the efficiency and success of the proposed framework.

Both theoretical and application issues are analysed and discussed in detail. Because the presentation of the thesis is highly mathematical and several issues are new, all necessary background information is provided in order to make the thesis an independent self-study.
Acknowledgments

I gratefully acknowledge the support, encouragement and understanding of my family all these years. I am grateful to my sister and brother, Helen and Antony, for their spiritual, moral and economic support during my Ph.D studies in Leicester. Without them the completion of this Ph.D wouldn’t have been possible. Particularly, I am grateful to my mother Melli for her spiritual support and encouragement who, especially after the death of my father, helped me to continue and finish my Ph.D research. In addition to the above persons I want to express my deepest regards and thanks to professor I. Postlethwaite (my Ph.D supervisor) for his professionalism, help and guidance through my Ph.D course. Also, many thanks and best wishes to professors M. Konstantinov and P. Petkov for their research cooperation, for many useful scientific discussions and for their hospitality. At last, but not least, I want to thank all of my friends, wherever they are, for the very good times we have had.
To Christian Orthodox Holy Trinity …

“The Single Unchanging Cause Of All Change”.

(Aristotle, ‘physics, = 350 BC’)
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>ii</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>iii</td>
</tr>
<tr>
<td>List of Figures</td>
<td>xi</td>
</tr>
<tr>
<td>List of Tables</td>
<td>xiii</td>
</tr>
<tr>
<td>Notation</td>
<td>xiv</td>
</tr>
<tr>
<td>Chapter 1. Introduction</td>
<td></td>
</tr>
<tr>
<td>1.1 Prologue</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Thesis Style &amp; Outline</td>
<td>5</td>
</tr>
<tr>
<td>1.3 How to Read this Thesis</td>
<td>10</td>
</tr>
<tr>
<td>Chapter 2. Numerical Solution</td>
<td></td>
</tr>
<tr>
<td>2.1 Introduction</td>
<td>11</td>
</tr>
<tr>
<td>2.2 Historical Background of Homotopy Methods</td>
<td>12</td>
</tr>
<tr>
<td>2.3 Probability-1 Homotopy Methods</td>
<td>14</td>
</tr>
<tr>
<td>2.3.1 Probability-1 Homotopy Theorems</td>
<td>19</td>
</tr>
<tr>
<td>2.3.2 Tracking of Homotopy Paths</td>
<td>43</td>
</tr>
<tr>
<td>2.3.3 Further Analysis and Discussions</td>
<td>51</td>
</tr>
<tr>
<td>2.4 Numerical Solution of Equation (2.1)</td>
<td>55</td>
</tr>
<tr>
<td>2.5 Synopsis</td>
<td>59</td>
</tr>
<tr>
<td>Chapter 3. Scaling &amp; Homogeneous Projective Transformations</td>
<td></td>
</tr>
<tr>
<td>3.1 Introduction</td>
<td>60</td>
</tr>
<tr>
<td>3.2 Scaling</td>
<td>62</td>
</tr>
<tr>
<td>3.2.1 Analysis</td>
<td>62</td>
</tr>
</tbody>
</table>
3.2.2 Synthesis...............................................................................................67
3.2.3 Variations on Scaling and Some Useful Observations..........................80
3.2.4 Scalar Numerical Example.................................................................84
3.3 Homogeneous Projective Transformation...............................................88
3.3.1 Homogeneous Projective Transformation for Polynomial Vector Equations.........................................................................................88
3.3.2 Homogeneous Projective Transformation of (3.2) with \( t = n \) and \( l = p \) .................................................................................................96
3.3.3 Numerical Example of the Homogeneous Projective Transformation for (2.1).........................................................................................99
3.3.4 Useful Comments and Observations....................................................100
3.4 Probability-1 Homotopies with Scaling and Homogeneous Projective Transformation for (2.1).................................................................................101
3.4.1 Probability-1 Homotopy Theorems with Scaling and the Homogeneous Projective Transformation for (2.1)...........................................102
3.4.2 Tracking of Homotopy Paths of Theorems (3.5) and (3.6)....................104
3.4.3 Numerical solution of (3.90)..................................................................107
3.5 Synopsis....................................................................................................110

Chapter 4. Numerical Analysis

4.1 Introduction...............................................................................................112
4.2 Rounding Error Analysis..........................................................................114
4.2.1 The Floating Point Number System and Model of Arithmetic.............115
4.2.2 Some Results for Matrix Addition and Multiplication..........................116
4.2.3 Rounding Error Estimates for the Homotopy Equations (3.95) and (3.98).........................................................................................139
4.2.4 Further Discussions.............................................................................147
Chapter 5. Software Design and Implementation

5.1 Introduction ................................................. 172
5.2 Program Inputs and Outputs ................................ 174
   5.2.1 Program Inputs ........................................ 174
   5.2.2 Program Outputs ..................................... 181
5.3 The GQME-Toolbox ........................................... 183
5.4 Possible Failures and Criticism of
   Computed Results ............................................. 190
5.5 Synopsis ...................................................... 192

Chapter 6. Generalized Algebraic Riccati Equations

6.1 Introduction .................................................. 193
6.2 Computation of Special Kind
   Solutions of AREs ................................................. 196
   6.2.1 Directionality Rules .................................... 205
6.3 Numerical Examples .......................................... 213
   6.3.1 Numerical Example 6.1 ......................... 213
   6.3.2 Numerical Example 6.2 ......................... 224
   6.3.3 Numerical Example 6.3 ......................... 233
   6.3.4 Numerical Example 6.4 ......................... 238
6.4 Synopsis ...................................................... 241
Chapter 7. Numerical Solution & Analysis to Scientific Problems

7.1 Introduction ................................................................. 245

7.2 Feedback Nash equilibria in scalar
infinite horizon LQ-games ................................................. 246
7.2.1 Problem Formulation and Solution ............................... 246
7.2.2 Numerical Example 7.1............................................. 247

7.3 Second Order Polynomial Algebraic Matrix
Equation and the Quadratic Eigenvalue Problem .................. 262
7.3.1 Numerical Example 7.2............................................. 264
7.3.2 Numerical Example 7.3............................................. 275

7.4 Example from a study of a Chemical
Kinetics Model ..................................................................... 281
7.4.1 Numerical Example 7.4............................................. 282

7.5 Computation of Equilibrium Points
of the Lorenz Attractor .................................................... 284
7.5.1 Numerical Example 7.5............................................. 294

7.6 Synopsis ........................................................................ 300

Chapter 8. Case Studies in Robust Control System Design:
Applications to Fast Modern Electrotechnological
Manufacturing Processes and Power Systems

8.1 Introduction .................................................................. 301

8.2 The Actuator Saturation Problem
and Possible Solutions .................................................... 302
8.2.1 Preventing Saturation in Asymptotically
Stable Plants via Limited Inputs .................................... 304

8.3 Proposed Control System Design ................................... 313
8.3.1 Fixed Order Observer Based Dynamic
Output Feedback $H_2$/Anti-windup
Controllers with Integrators for Systems
with Amplitude Saturated Control Inputs ......................... 313

8.4 Control System Design Computation .............................. 319

8.5 Robust Control of Fast Modern
Electro-technological Manufacturing

Processes: High Powered Lasers .................................................. 323

8.5.1 System Description and
    Design Specifications ......................................................... 324

8.5.2 Control System Computation ................................................. 325

8.6 Robust Control of a Synchronous
    Electric Generator/Infinite Bus Power System .......................... 327

8.6.1 System Description and Modeling ............................................. 328

8.6.2 Control System Computation ................................................. 332

8.7 Synopsis .................................................................................... 337

    Algebraic Quadratic Matrix Equations

9.1 Introduction ............................................................................. 338

9.2 Coupled Generalized Quadratic
    Algebraic Matrix Equations .................................................. 339

9.3 Applications to Control and
    Game Theory Design Problems .............................................. 341

9.3.1 Multi-objective Control Systems
    Design Computations .......................................................... 341

9.3.2 Computation of Two Player Feedback
    Nash Equilibria in Infinite Horizon
    Multivariable LQ-Games .................................................... 342

9.4 Synopsis .................................................................................... 345

Chapter 10. Epilogue

10.1 General Comments ............................................................. 346

10.2 Future research ................................................................. 347

Bibliography ................................................................................. 450

Index of Theorems Propositions and Lemmas ................................. 468
Appendix A. Algebraic & Differential Topology Results

A.1 Introduction ................................................................. 370
A.2 Set Theory Elements ...................................................... 370
A.3 Nonlinear Algebraic Polynomial
   Systems and Equations ..................................................... 374
A.4 Algebraic & Differential Topology Theorems ..................... 377
A.5 Useful Bibliography ....................................................... 385

Appendix B. Results in Matrix Analysis

B.1 Introduction ................................................................. 386
B.2 Matrix Operators & Algebra ........................................... 386
   B.2.1 Matrix Operators ......................................................... 386
   B.2.2 Algebraic Results ....................................................... 389
B.3 Matrix Calculus ............................................................. 393
B.4 Useful Bibliography ....................................................... 397

Appendix C. The GQME-Toolbox

C.1 Introduction ................................................................. 398
C.2 Software Code ............................................................. 398
C.3 Modified Versions of Major
   Functions of GQME-Toolbox ............................................ 449

Appendix D. Drivers for the Numerical Examples

D.1 Introduction ................................................................. 473
D.2 Drivers ................................................................. 473
List of Figures

Figure 2.1. Geometric interpretation of $\tilde{H}_{\alpha p}(0)$.

Figure 3.1. Geometric interpretation of problem (3.31) for $10^{-201}x^2 - 2x + 2 \times 10^{201} = 0$.

Figure 3.2. Contours $u_X, u_C$, of problem (3.31) for $10^{-201}x^2 - 2x + 2 \times 10^{201} = 0$.

Figure 3.3. Geometric interpretation of problem (3.63) for $10^{-201}x^2 - 2x + 2 \times 10^{201} = 0$.

Figure 3.4. Contours $u_X, u_C$, of problem (3.63) for $10^{-201}x^2 - 2x + 2 \times 10^{201} = 0$.

Figure 5.1. The GQME-Toolbox Structure.

Figure 6.1. Topology of solutions of (6.18).

Figure 6.2. Homotopy paths for example 6.1.

Figure 6.3. Comparison graphs for example 6.1.

Figure 6.4. Comparison graphs for example 6.1.

Figure 6.5. Homotopy paths for example 6.2.

Figure 6.6. Comparison graphs for example 6.2.

Figure 6.7. Comparison graphs for example 6.2.

Figure 6.8. Comparison graphs for examples 6.1, 6.2.

Figure 6.9. Homotopy paths for example 6.3.

Figure 6.10. Homotopy paths for example 6.4.

Figure 7.1. Homotopy paths for example 7.1 (n.s.n.p).

Figure 7.2. Homotopy paths for example 7.1 (n.s.p).

Figure 7.3. Homotopy paths for example 7.1 (s.n.p).

Figure 7.4. Homotopy paths for example 7.1 (s.p).

Figure 7.5. Homotopy paths for example 7.1 (sv.n.p).

Figure 7.6. Homotopy paths for example 7.1 (sv.p).

Figure 7.7. Homotopy paths of finite solutions for example 7.2 (n.s.n.p).

Figure 7.8. Homotopy paths of finite solutions for example 7.2 (n.s.p).

Figure 7.9. Homotopy paths of finite solutions for example 7.2 (s.p).

Figure 7.10. Homotopy paths for example 7.3.

Figure 7.11. Homotopy paths for example 7.4 (n.s.p).

Figure 7.12. Homotopy paths for example 7.4 (s.p).

Figure 7.13. Homotopy paths for example 7.4 (sv.p).

Figure 7.14. Homotopy paths for example 7.5 (n.s.p).
Figure 7.15. Homotopy paths for example 7.5 (s.p).

Figure 8.1. The observer based ‘pseudo linear’ controller (8.48), (8.49) in Laplace domain.

Figure 8.2. Closed loop system (8.50), (8.51) in Laplace domain.

Figure 8.3. Location of the closed loop eigenvalues under state feedback.

Figure 8.4. Closed loop simulation.

Figure 8.5. System under study.

Figure 8.6. Transfer function block diagram of the plant under study in the Laplace domain.

Figure 8.7. Closed loop simulation.
List of Tables

Table 6.1.1. Computation results for example 6.1.
Table 6.1.2. Computation results for example 6.1.
Table 6.2.1. Computation results for example 6.2.
Table 6.2.2. Computation results for example 6.2.
Table 6.3.1. Computation results for example 6.3.
Table 6.3.2. Computation results for example 6.3.
Table 6.4. Computation results for example 6.4.
Table 7.1.1. Computation results for example 7.1.
Table 7.1.2. Computation results for example 7.1.
Table 7.1.3. Computation results for example 7.1.
Table 7.1.4. Computation results for example 7.1.
Table 7.1.5. Computation results for example 7.1.
Table 7.1.6. Computation results for example 7.1.
Table 7.2.1. Computation results for example 7.2.
Table 7.2.2. Computation results for example 7.2.
Table 7.2.3. Computation results for example 7.2.
Table 7.2.4. Computation results for example 7.2.
Table 7.2.5. Computation results for example 7.2.
Table 7.3.1. Computation results for example 7.3.
Table 7.3.2. Computation results for example 7.3.
Table 7.4.1. Computation results for example 7.4.
Table 7.4.2. Computation results for example 7.4.
Table 7.4.3. Computation results for example 7.4.
Table 7.4.4. Computation results for example 7.4.
Table 7.4.5. Computation results for example 7.4.
Table 7.4.6. Computation results for example 7.4.
Table 7.5.1. Computation results for example 7.5.
Table 7.5.2. Computation results for example 7.5.
Table 8.1. Computation results.
Table 8.2. System Nomenclature.
Table 8.3. Computation results.
Notation

Different alphabetical characters and types of fonts are used for different variables in order to avoid confusion. The notation is consistent with that used in other relevant works. Unusual notation not listed below is explained in the main text where this notation is first introduced.

Sets

Sets are denoted by capital letters, or by use of braces (e.g. \{x,y,z\} denotes the set having x, y and z as elements, \{x|x > z\} denotes the set of all x such that x > z, etc).

Symbols used with sets are:

- $\emptyset$: The empty set (set with no elements)
- $x \in \Psi$: $x$ is an element of $\Psi$ (we say, $x$ belong to $\Psi$)
- $x \notin \Psi$: $x$ is not an element of $\Psi$ (we say, $x$ does not belong to $\Psi$)
- $\Psi = \Omega$: $\Psi$ and $\Omega$ are equal (identical, consist of the same elements)
- $\Psi \neq \Omega$: $\Psi$ and $\Omega$ are not equal (different)
- $\Psi \subseteq \Omega$: $\Psi$ is a subset of $\Omega$ (each element of $\Psi$ also belongs to $\Omega$)
- $\Psi \supseteq \Omega$: Same as $\Psi \subseteq \Omega$
- $\Psi \subset \Omega$: $\Psi$ is a proper subset of $\Omega$ (i.e. $\Psi \subseteq \Omega$ and $\Psi \neq \Omega$)
- $\Psi \supset \Omega$: Same as $\Psi \subset \Omega$
- $\Psi \not\subset \Omega$: $\Psi$ is a not a proper subset of $\Omega$
- $\Psi \cup \Omega$: Union of $\Psi$ and $\Omega$. $\Psi \cup \Omega := \{x|x \in \Psi \text{ or } x \in \Omega\}$
- $\bigcup_{i=1}^{n} \Psi_i := \bigcup_{i=1}^{n} \Psi_i := \Psi_1 \cup \Psi_2 \cup \cdots \cup \Psi_n$
- $\Psi \cap \Omega$: Intersection of $\Psi$ and $\Omega$. $\Psi \cap \Omega := \{x|x \in \Psi \text{ and } x \in \Omega\}$
- $\bigcap_{i=1}^{n} \Psi_i := \bigcap_{i=1}^{n} \Psi_i := \Psi_1 \cap \Psi_2 \cap \cdots \cap \Psi_n$
- $\Psi \cap \Omega = \emptyset$: $\Psi$ and $\Omega$ are disjoint sets (sets without common elements)
- $\Psi - \Omega$: Difference of $\Psi$ and $\Omega$. $\Psi - \Omega := \{x|x \in \Psi \text{ and } x \notin \Omega\}$
- $\text{int}\{\Psi\}$: The interior of $\Psi$ (see appendix A)
- $\text{ext}\{\Psi\}$: The exterior of $\Psi$ (see appendix A)
\[ \Psi \] The boundary of \( \Psi \) (see appendix A)

\[ \overline{\Psi} \] The closure of \( \Psi \) (see appendix A)

\[ \Psi \times \Omega = \{(x, y) \mid x \in \Psi, y \in \Omega \text{ and } \Psi \neq \emptyset, \Omega \neq \emptyset \} \]

(\text{the Cartesian product of } \Psi \neq \emptyset \text{ and } \Omega \neq \emptyset)

\[ \sup(\Psi) \] The supremum of \( \Psi \) (i.e. the least upper bound of \( \Psi \))

\[ \inf(\Psi) \] The infimum of \( \Psi \) (i.e. the maximum lower bound of \( \Psi \))

### Spaces

\[ \mathcal{C}, \mathcal{R}, \mathcal{Z}, \mathcal{N} \] The space of complex, real, integer and natural numbers respectively

\[ \mathcal{C}_0, \mathcal{R}_0, \mathcal{Z}_0, \mathcal{N}_0 \]
\[ \mathcal{C}_0 := \mathcal{C} - \{0\}, \mathcal{R}_0 := \mathcal{R} - \{0\}, \mathcal{Z}_0 := \mathcal{Z} - \{0\}, \mathcal{N}_0 := \mathcal{N} - \{0\} \]

\[ \mathcal{R}_+, \mathcal{Z}_+ \]
\[ \mathcal{R}_+ := \mathcal{R} - \{x \in \mathcal{R} \mid x < 0\}, \mathcal{Z}_+ := \mathcal{Z} - \{x \in \mathcal{Z} \mid x < 0\} \]

\[ \mathcal{R}_-, \mathcal{Z}_- \]
\[ \mathcal{R}_- := \mathcal{R} - \{x \in \mathcal{R} \mid x > 0\}, \mathcal{Z}_- := \mathcal{Z} - \{x \in \mathcal{Z} \mid x > 0\} \]

\[ \mathcal{C}^{n \times m}, \mathcal{R}^{n \times m}, \mathcal{Z}^{n \times m}, \mathcal{N}^{n \times m} \] The space of complex and real \( n \times m \) matrices respectively

\[ \mathcal{C}_0^{n \times m}, \mathcal{R}_0^{n \times m}, \mathcal{Z}_0^{n \times m}, \mathcal{N}_0^{n \times m} \]
\[ \mathcal{C}_0^{n \times m} := \mathcal{C}^{n \times m} - \{0_{n \times m}\}, \mathcal{R}_0^{n \times m} := \mathcal{R}^{n \times m} - \{0_{n \times m}\}, \mathcal{Z}_0^{n \times m} := \mathcal{Z}^{n \times m} - \{0_{n \times m}\}, \mathcal{N}_0^{n \times m} := \mathcal{N}^{n \times m} - \{0_{n \times m}\} \]

\[ \mathcal{R}_+^{n \times m}, \mathcal{Z}_+^{n \times m} \]
\[ \mathcal{R}_+^{n \times m} := \mathcal{R}^{n \times m} - \{x \in \mathcal{R}^{n \times m} \mid x = |x|\}, \mathcal{Z}_+^{n \times m} := \mathcal{Z}^{n \times m} - \{x \in \mathcal{Z}^{n \times m} \mid x = |x|\} \]

\[ \mathcal{R}_-^{n \times m}, \mathcal{Z}_-^{n \times m} \]
\[ \mathcal{R}_-^{n \times m} := \mathcal{R}^{n \times m} - \{x \in \mathcal{R}^{n \times m} \mid x = -|x|\}, \mathcal{Z}_-^{n \times m} := \mathcal{Z}^{n \times m} - \{x \in \mathcal{Z}^{n \times m} \mid x = -|x|\} \]

\[ \mathcal{C}^n, \mathcal{R}^n, \mathcal{Z}^n, \mathcal{N}^n \] The space of complex and real \( n \times 1 \) vectors respectively

\[ \mathcal{C}_0^n, \mathcal{R}_0^n, \mathcal{Z}_0^n, \mathcal{N}_0^n \]
\[ \mathcal{C}_0^n := \mathcal{C}^n - \{0^n\}, \mathcal{R}_0^n := \mathcal{R}^n - \{0^n\}, \mathcal{Z}_0^n := \mathcal{Z}^n - \{0^n\} \]

\[ \mathcal{R}_+^n, \mathcal{Z}_+^n \]
\[ \mathcal{R}_+^n := \mathcal{R}^n - \{x \in \mathcal{R}^n \mid x = |x|\}, \mathcal{Z}_+^n := \mathcal{Z}^n - \{x \in \mathcal{Z}^n \mid x = |x|\} \]

\[ \mathcal{R}_-^n, \mathcal{Z}_-^n \]
\[ \mathcal{R}_-^n := \mathcal{R}^n - \{x \in \mathcal{R}^n \mid x = -|x|\}, \mathcal{Z}_-^n := \mathcal{Z}^n - \{x \in \mathcal{Z}^n \mid x = -|x|\} \]
Matrices & Vectors

Matrices are denoted by capital bold face font characters and vectors are denoted by small bold face font characters, in order to distinguish them from scalars. \( \mathbf{0} \) and \( \mathbf{I} \) denote respectively the zero and identity matrices of appropriate dimensions.

\[
\begin{align*}
\mathbf{0}_{n \times m} & \quad \text{The zero } n \times m \text{ matrix (all elements are zero)} \\
\mathbf{0}_n & \quad \text{The zero } n \times n \text{ matrix (all elements are equal to 0)} \\
\mathbf{U}_{n \times m} & \quad \text{The unity } n \times m \text{ matrix (all elements are equal to 1)} \\
\mathbf{I}_n & \quad \text{The identity } n \times n \text{ matrix (all elements in main diagonal are equal to 1 and all the others are equal to 0)} \\
A_{ij} & \quad \text{The } (i,j) \text{-element of } A \text{ (also denoted as } [A]_{ij}) \\
\text{row}_i(A) & \quad \text{The } i \text{ row of } A . \\
\text{col}_i(A) & \quad \text{The } i \text{ column of } A . \\
A^T & \quad \text{The transpose of } A \\
A^{Tc} & \quad \text{The complex conjugate transpose of } A \\
det(A) & \quad \text{The determinant of } n \times n \text{ matrix } A \\
\text{rank}(A) & \quad \text{The rank of } A \\
\text{Re}\text{rank}(A) & \quad \text{The real rank of a complex matrix } A \\
A^{-1} & \quad \text{The inverse of the } n \times n \text{ matrix } A \\
A>0, A \geq 0 & \quad A \text{ is positive definite, semidefinite (nonnegative) respectively} \\
A<0, A \leq 0 & \quad A \text{ is negative definite, semidefinite (nonpositive) respectively} \\
A^n & \quad A^n := A A \cdots A \\
\| & \quad \text{The Absolute Value operator (see appendix B)} \\
E_{10}(\cdot) & \quad \text{Exponent Operator Base-10 (see appendix B)} \\
L_{10}(\cdot) & \quad \text{Logarithmic Operator Base-10 (see appendix B)} \\
S(\cdot) & \quad \text{Sign Operator (see appendix B)} \\
\text{diag}(\cdot) & \quad \text{Vector Diagonalization Operator (see appendix B)} \\
\text{vec}(\cdot) & \quad \text{Vector Matrix Operator (see appendix B)}
\end{align*}
\]
**Notation**

\[
vec^{-1}(\cdot) \quad \text{Inverse Vector Matrix Operator (see appendix B)}
\]

\(\oplus\) \quad \text{Kronecker Sum Operator (see appendix B)}

\(\otimes\) \quad \text{Kronecker Product Operator (see appendix B)}

\(\odot\) \quad \text{Hadamard Product Operator (see appendix B)}

\(\div\) \quad \text{Hadamard Division Operator (see appendix B)}

\(J_{f(x_1,x_2,\ldots,x_n)}^{(i)}\) \quad \text{The Jacobian of vector (or matrix) function } f(x_1,x_2,\ldots,x_n) \text{ with respect to } x_i \text{ (see appendix B)}

\(\nabla f(x_1,x_2,\ldots,x_n)\) \quad \text{The Jacobian of scalar function } f(x_1,x_2,\ldots,x_n) \text{ with respect to } (x_1,x_2,\ldots,x_n) \text{ (see appendix B)}.

\[
\|A\|_F := \left( vec(A)^T vec(A) \right)^{1/2}. \text{ The Frobenious norm of a matrix}
\]

\[
\|x\|_F := \left( vec(x)^T vec(x) \right)^{1/2}. \text{ The 2-norm of a vector}
\]

**Miscellaneous**

**Ker(\cdot)** \quad \text{The Kernel (null space) of a linear transformation}

**C^v** \quad \text{The class of } v \text{-times continuously differentiable manifolds}

**\text{deg}(\cdot)** \quad \text{Topological (Brouwer) degree of a polynomial system (see appendix A)}

\(\langle \cdot \rangle\) \quad \text{The homogeneous polynomial system or equation (see appendix A)}

\[
\frac{d(\cdot)}{dx}\quad \text{The derivative of } (\cdot) \text{ with respect to } x
\]

\[
\frac{\partial(\cdot)}{\partial x}\quad \text{The partial derivative of } (\cdot) \text{ with respect to } x
\]

\[
\|f(s)\|_{\infty} := \text{sup}_s \left\{ |f(s)| \right\}. \text{ The } H_{\infty} \text{-norm of a function } f(s)
\]

**Re(\cdot)** \quad \text{The real part of the complex number, vector or matrix}

**Im(\cdot)** \quad \text{The imaginary part of the complex number, vector or matrix}

\(i\) \quad \text{The imaginary unit (i.e. } i^2 = -1\)

\[
\prod_{i=1}^{n}(\cdot)_{i} := (\cdot)_{1}(\cdot)_{2}\cdots(\cdot)_{n}
\]
The rest of the notation used is standard in the literature.
Chapter 1

Introduction

1.1 Prologue

The birth of algebraic equations goes back to soon after Aristotle established the meaning of unit and Euclid composed his geometry. Through the ages algebraic equations have been used consistently and have evolved through pure and applied mathematics and mechanics.

There is a huge variety of different kinds and forms of algebraic equations, and their importance has never been in doubt. An ancient example is the Pythagoras theorem $x^2 + y^2 = z^2$ [52], while a relatively modern example is the algebraic matrix Riccati equation $A^T X + X A + X D X + C = 0$ [106].

In many cases, algebraic equations, and especially systems of algebraic quadratic equations, do not possess an analytic solution. Therefore, their solutions can only be approximated numerically, via a numerical algorithm. Since this is the case for most practical problems (e.g. engineering designs, scientific analysis), numerous algorithms have been developed for the numerical solution of algebraic quadratic equations and their systems. In addition, some of these algorithms are accompanied with theoretical studies about their numerical analysis (e.g. numerical accuracy and robustness).

The majority of the algorithms that have been developed over the years address specific forms and types of single or systems of algebraic quadratic equations. Hence, most of the relevant theoretical designs and implementations are restricted to hold for special cases.
In order to form a global approach to the numerical solution and analysis of all single and systems of algebraic quadratic equations, this thesis unifies their numerical solution and analysis under a single framework. This justifies the general title of the thesis.

To be more specific, it will be shown that all systems of algebraic quadratic equations can be equivalently considered as special cases of the generalized algebraic quadratic matrix equation (GQME)

\[ f(X) = \sum_{i=1}^{\omega} A_i XB_i + \sum_{i=1}^{\xi} C_i XD_iXE_i + G = 0 \]  

Where, \( \omega, \xi \in \mathbb{N} \), \( \omega + \xi \not= 0 \), \( n, p \in \mathbb{Z}_+ \). \( A_i, C_i \in \mathbb{C}^{n \times n} \), \( B_i, E_i \in \mathbb{C}^{p \times p} \), \( D_i \in \mathbb{C}^{p \times n} \), \( G \in \mathbb{C}^{n \times p} \), \( 0 \in \mathbb{C}^{n \times p} \) are the constant matrix coefficients of the equation and \( X \in \mathbb{C}^{n \times p} \) is the equation’s unknown matrix. Hence, this thesis is devoted to the theoretical development of the numerical solution and analysis of (1.1) and to the design and implementation of relevant computer software. Moreover, the problems of finding all or some numerical solutions to (1.1) and their numerical analysis, is addressed. Being able to find all solutions to (1.1), distinguishes the algorithm as rigorous rather than heuristic.

Equation (1.1) appears in various engineering design problems. Some examples are the classical and generalized algebraic Riccati equations in control system design [106], [188], the second order matrix polynomial equations for the solution of the quadratic eigenvalue problem (used in measuring vibrations in materials [105]) [79], [80]. In addition, many scientific problems such as the computation of equilibrium points in noncooperative Nash games in economic systems, chemical kinetics modelling, the computation of equilibrium points of nonlinear quadratic time invariant dynamical systems (which often appear in bifurcation [91], [10] and in chaos theory [161]), can be reformulated as (1.1).

Furthermore, there is an indication that designs in modern system theory and control are getting more complex and very often reduce to optimization problems the solution of which rely on equations which are special forms of (1.1). Recent examples are the mixed objective robust control system designs [172], [85], [48] and [49], for which it will be shown in later chapters how they can considered as special cases of (1.1) and hence how they can be solved with the proposed framework. Hence, the utility of the present thesis for existing problems and possible future designs is evident.
In some of the problem cases mentioned above, existing numerical methods cannot always provide solutions. The most common reason is the sensitivity of convergence of algorithms with respect to initial conditions. The proposed methods in this work try to overcome this problem. This will be illustrated with numerical examples. Moreover, the proposed numerical algorithms, which are implemented via the designed software, will be compared with other already state-of-the-art algorithms under a common numerical analysis framework and their efficiency will be demonstrated.

Apart from the above applications, (1.1) is attractive from the mathematical viewpoint, since as already mentioned it unifies a large class of systems of algebraic quadratic equations into a single framework. Most importantly, a general solution for this type of generalized equation has never been found before and therefore the present work is novel in that aspect. At this point, it is emphasized that the present thesis is dealing with (1.1) only from the numerical solution and analysis viewpoint. Theoretical results about the existence and the geometry of solutions of (1.1) in its general form are not available yet, as far as this author is aware. This is a big research study in abstract mathematics.

In general, there are two kinds of numerical algorithms for solving equations: discrete and continuous. The realization process of a discrete numerical algorithm is via iterative procedures. The basic idea of the realization of a continuous numerical algorithm is to connect mathematically (e.g. via a mapping) two different problems and to continuously deform the solution of one problem to that of the other. Nowadays, the success of investigating the realization processes of discrete numerical algorithms is apparent. On the other hand, the use of dynamical systems theory and differential equations have recently provided robust and globally convergent continuous numerical algorithms, that can compete and outperform in many cases discrete time algorithms. Continuous time algorithms are also gaining in popularity because of the steady increase in computer power, which is such that differential equations (the hardcore of continuous algorithms) can now be solved fast and efficiently.

The essential tools for the numerical solution of (1.1) that are used in the present work are the probability-1 homotopy methods [5]. These methods have their origin in the abstract mathematical branches of algebraic [147], [16] and differential topology [67], [119] and belong to the category of continuous numerical algorithms. In probability-1 homotopy methods there are two abstract problems. Usually, the one problem is trivial to solve while the other is the original problem that is to be solved. These two problems are connected to each other mathematically via an integral curve
of an ordinary differential equation defined on a manifold, that is a dynamic flow. This flow actually constitutes the rule of how the easy problem is deformed to the original problem. The existence of such a dynamic flow is guaranteed in a probabilistic sense with probability-1 (i.e. almost always). This is something that guarantees, almost always, the global convergence of these kinds of algorithms. Note that global convergence is not usually the case for the majority of other numerical methods, especially when dealing with non-convex problems. In this respect note that (1.1) under specific data can produce non-convex problems and therefore there is a major advantage in choosing probability-1 homotopy methods.

Apart from homotopy methods, dynamic flows are defined for other numerical methods as well. A very interesting survey of such methods is [35]. In this last reference, it is shown that discrete numerical algorithms are considered as special cases of generalized dynamic flows on manifolds. Therefore, all virtually known discrete (iterative) algorithms can be considered as subsets of continuous numerical algorithms. As is stated in [35], another advantage of investigating designs with continuous algorithms is that we can gain fundamental insights into the structure and behaviour of existing discrete methods and, to suggest new improved numerical algorithms. In addition, dynamic flows have many applications to both the numerical algorithms and theoretical studies of open problems [35], [92].

Now, because of the usage of probability-1 homotopy methods (i.e. dynamic flow based methods), it is hoped that the studies within this thesis will contribute to the understanding of how to reach solutions to generalized quadratic matrix algebraic equations and to the understanding of some of the dynamics of such realization processes. Also, with the development of numerical analysis results on the computed solutions of such processes, the thesis contributes to the understanding of the numerical accuracy and robustness of the above methods. Furthermore, as far the author knows, the numerical analysis of computed solutions of homotopy methods for matrix algebraic equations, is very limited if not absent from the literature. This is true also for many engineering and scientific problems which can be viewed as special cases of (1.1), since most of the algorithms provided for their numerical solution are without numerical analysis studies. Hence, no criticism of the computed solutions can be made. It is hoped that the present work will help to provide investigations in these areas of analysis.
1.2 Thesis Style & Outline

Most of the material of this thesis is mathematical. The major outline is in the form of theorems, propositions, lemmas and their proofs followed by discussions and remarks. Additional care has been made to keep the thesis as self-contained as possible. All the issues are analyzed and proven in detail and background information is provided in appendices A-D for the unfamiliar reader. In addition, a rich bibliography is presented to guide the reader into a more detailed study of relevant mathematics and other similar subjects. However, it is expected that the reader will have an essential knowledge of pure and applied mathematics to a level which is usually covered in most engineering undergraduate courses.

Special attention has been given to the software, which is built in Matlab, so that it is free from bugs and runtime errors, and includes all the necessary help features and comments, in order to guide the reader to the proper use and understanding of the code.

The thesis outline is as follows:

- Notation.
  This lists the major symbols used in the mathematical expressions.

- List of Tables.

- List of figures.

- Chapter 2. Numerical Solution
  This chapter presents the mathematical analysis and synthesis of probability-1 homotopy algorithms for the numerical solution of (1.1). A historical background of homotopy methods for the numerical solution of nonlinear algebraic equations is first introduced. The application of these methods for the numerical solution of (1.1) and the established theoretical results follow. In the end of the chapter, relevant conclusions are made.

- Chapter 3. Scaling & Homogeneous Projective Transformations
  This chapter presents the mathematical formulation and solution to the problems of scaling and homogeneous projective transformations. The problem of scaling is presented for a more general dimensioned equation than (1.1), namely
\[
f(X) = \sum_{i=1}^{\omega} A_i X B_i + \sum_{i=1}^{\xi} C_i X D_i X E_i + G = 0, \quad (1.2)
\]

where \( \omega, \xi \in \mathbb{N}, \quad \omega + \xi \neq 0, \quad t, n, p, l \in \mathbb{Z}_+, \quad A_i, C_i \in \mathbb{C}^{t \times n}, \quad B_i, E_i \in \mathbb{C}^{p \times l}, \quad D_i \in \mathbb{C}^{p \times n}, \quad G \in \mathbb{C}^{l \times l}, \quad 0 \in \mathbb{C}^{l \times l} \). Note that (1.2) is the special case of (1.1) with \( t = n \) and \( l = p \). The problem of scaling considers the transformation of (1.2), hence (1.1), into an equivalent equation in order to prevent numerical problems in the computations. This prevents arithmetic problems on a computer by transforming the coefficients and the unknown matrices of the original equation so that they do not have extreme (i.e. very small or very large) values.

After the study of scaling, the concept of homogeneous projective transformations for (1.1) is addressed. These transformations are used to avoid possible solutions at infinity. Homogeneous projective transformations convert the original equation into an equation with no solutions at infinity. With such a transformation, the finite solutions of the transformed equation should correspond to the finite and infinite solutions (if any) of the original equation. Such a transformation provides mathematical formulas for implementing the correspondence of solutions between the original and the transformed equation, under certain hypotheses.

Scaling and the homogeneous projective transformation are independent of any numerical method and therefore they can in general be used along with any numerical method for the equation’s solution. Next, both the scaling and the homogeneous projective transformation are used along with the probability-1 homotopy methods presented in chapter 2. New probability-1 homotopy theorems are presented, which are characterized by their ability to calculate solutions at infinity.

Useful comments and comparisons are made in order to declare similarities and differences as well as advantages and disadvantages with other methods. Conclusions on each method are presented.

• Chapter 4. Numerical Analysis

In this chapter, the accuracy and the stability of the probability-1 homotopy algorithms presented in chapters 2 and 3, are analyzed. The chapter begins by introducing some background information on numerical analysis and outlines essential definitions and concepts. Next it presents the rounding error analysis for the homotopy equations derived in chapter 3 and determines condition number estimates for these equations, which give information about the numerical conditioning of specific problems and their solutions. Backward and forward errors for the homotopy
equations in chapter 3 are then derived. After summarizing earlier results the chapter proceeds with the derivation of criteria for the numerical stability and behaviour of the homotopy algorithms of chapters 3 and 2. The chapter ends with further relevant discussion and conclusions.

It should be emphasized that the analysis of this chapter and the results obtained are novel and not reported elsewhere, as far as the author knows. In addition, this chapter can viewed as a unified framework for the numerical analysis of general quadratic matrix equations, and can be a reference for further studies and research considering the specialization to specific types of equations.

• **Chapter 5. Software Design and Implementation**

In this chapter, the algorithms, which have been developed theoretically in the previous chapters, are implemented in software. The programming of the algorithms is done using the ‘high level’ programming language of MATLAB v. 5.3.1.29215a (R11.1) [115]. MATLAB provides a user-friendly environment that allows even inexperienced users to perform numerical computations easily and quickly. MATLAB functions and programming can be used in an object-oriented manner [139], which is what was done in this thesis. This makes the programming more economical flexible and robust.

The designed software is in the form of a MATLAB toolbox (a collection of separate functions (objects) and driver programs [115]), named *The GQME-Toolbox* (The General Quadratic Matrix Equation Toolbox). The generated code of GQME-Toolbox is given in appendix C. The toolbox structure and its special functions (subroutines) are described in this chapter.

Care has been taken to make the software environment as user friendly as possible and to prevent the user from incorrectly setting up a particular problem. Help features which guide the user to run problems are included within both the toolbox’s main driver and the special functions. Extensive numerical tests have taken place in order to detect and remove runtime errors and bugs, before finalizing the code.

The reader is advised to read the code along with chapter 5, following chapters 3 and 4, in order to understand better the computations. Overall, chapter 5 can be considered to be a manual for the designed software.

• **Chapter 6. Generalized Algebraic Riccati Equations**

This chapter considers the specialization of (1.1) into generalized algebraic Riccati equations. These specific equations play very important roles in stability
analysis, in systems theory, and particularly in modern control system design. An introduction to these equations and their different forms takes place first. Next, some theory is developed for how to extract solutions of a special kind that algebraic Riccati equations usually possess. The computation of such special types of solutions via the GQME-Toolbox is discussed next. In addition, examples with known solutions are solved and comparisons with existing methods given in terms of numerical stability, behaviour and accuracy of computations. Finally, general conclusions are given.

• Chapter 7. Numerical Solution & Analysis to Scientific Problems

Important scientific problems from the area of finance, applied mathematics/mechanics, chemical engineering and chaotic dynamical systems, are formulated and solved in this chapter. Each problem is presented, via a numerical example from the literature, in a separate section. The numerical solution and analysis of the computation of feedback Nash equilibria in scalar infinite horizon LQ-games [51], [15], is considered first. Next, the numerical solution of a second order polynomial algebraic matrix equation [39], [42], [43] and the solution of a quadratic eigenvalue problem [79], [80], which can be used in measuring mechanical vibrations is addressed. An example from chemical kinetics [121] follows. The last example of this chapter, considers the computation of the equilibrium points of the famous Lorenz attractor reported in [161]. Each section contains relevant discussions and general conclusions are given at the end of the chapter. The MATLAB driver m-files for all examples are given in appendix D.

• Chapter 8. Case Studies in Robust Control System Design: Applications to Fast Modern Electro-technological Processes and Power Systems

In this chapter, it will be shown how (1.1) can be used in the design of robust control systems. More specifically, the design of fixed order $H_2$ and anti-windup control systems subject to saturated actuators will be addressed.

After a general discussion about such design problems, the proposed control system design methodology is presented. Next, two case studies from industry are presented. The first case study is the automatic control of the power from a high powered laser, which is used in welding, cutting and surface treatment [158]. Research on this example is relatively recent, and only few control techniques have been applied so far (most of them based on classical PID controller design [158]). The second case study is taken from the area of electrical power systems. More specifically, it
considers the design of a fast and robust control system for a single synchronous
electric generator connected to an infinite bus [50].

In both the above case studies, it is illustrated how the proposed control system
design can be numerically solved using the GQME-Toolbox. The numerical analysis
of the computations, the simulation of the controlled systems and relevant discussions
are reported in the end of each relevant section. General conclusions are given at the
end. This chapter may be of interest to control systems engineers in the area of modern
electro-technological applications.

• Chapter 9. Numerical Solution & Analysis of Coupled Generalized Algebraic
  Quadratic Matrix Equations

This is a short chapter in which it is shown how a system of coupled
generalized algebraic quadratic matrix equations can be equivalently written as
equation (1.1). Therefore, the numerical solution and numerical analysis of such
systems will follow from chapters 2-4.

Systems of coupled algebraic quadratic matrix equations arise in various
multiobjective control system design problems and in multivariable game theory. The
chapter concentrates on two coupled generalized algebraic quadratic matrix equations
and discusses the extension to systems of more than two equations and the solution of
a wide range of control system design and multivariable noncooperative game theory
problems. It is shown how these problems can be formulated and how they can be
solved with the GQME-Toolbox. Discussions on future research are given at the end
of this chapter.

• Chapter 10. Epilogue

This chapter summarizes the main results of the thesis and highlights areas of
future research. Detailed discussions and conclusions have been given in the relevant
sections of earlier chapters. Chapter 10 is therefore relatively short focusing only on
major results.

• Bibliography.
  A detailed bibliography is given.

• Index of Theorems, Propositions and Lemmas.
  This section locates the theorems, propositions and lemmas of the thesis.
Chapter 1  Introduction

• Appendix A. Algebraic & Differential Topology Results

This appendix presents background material from algebraic and differential topology. Many of the results have been generalized in order to be compatible with the problems formulated in this thesis. This appendix can serve as a useful survey in algebraic and differential topology.

• Appendix B. Results in Matrix Analysis

This appendix presents some useful results in matrix analysis. Many of the results presented are novel. This appendix can serve as a useful survey in matrix analysis.

• Appendix C. The GQME-Toolbox

This appendix presents the software code of the GQME-Toolbox. Helpful alternatives to improve speed of the computations are given.

• Appendix D. Drivers for the Numerical Examples

This appendix presents the drivers for the numerical examples of chapters 6, 7 and 8.

1.3 How to Read this Thesis

The main chapters of the thesis are written in such a way that each constitutes a separate self-contained document. Therefore, the reader (especially one familiar with the thesis topics) can independently choose a chapter to read. Chapters 2 to 4 and 9 are theoretical, while chapters 6, 7 and 8 constitute case studies in scientific and engineering problems. Chapter 5 presents the designed software and contains guidelines on how this software can be used. Therefore, the reader who is interested only in using a black box software for the computation of the solution of a relevant problem can view chapter 5 as the software’s manual. However, it is recommended that the best way to read the thesis, especially for the unfamiliar reader, is to follow the orderly sequence of the chapters.
Chapter 2

Numerical Solution

2.1 Introduction

This chapter presents the mathematical analysis and synthesis of algorithms for the numerical solution of the general quadratic algebraic matrix equation

\[ f(X) := \sum_{i=1}^{\omega} A_i X B_i + \sum_{i=1}^{\xi} C_i X D_i X E_i + G = 0 \]

where, \( \omega, \xi \in \mathbb{N}, \ \omega + \xi \neq 0, \ n, p \in \mathbb{Z}_+, \ A_i, C_i \in \mathbb{C}^{n \times n}, \ B_i, E_i \in \mathbb{C}^{p \times p}, \ D_i \in \mathbb{C}^{p \times n}, \)
\( G \in \mathbb{C}^{n \times p}, \ 0 \in \mathbb{C}^{n \times p} \) are the constant matrix coefficients of the equation and \( X \in \mathbb{C}^{n \times p} \) is the equation's unknown matrix. In the sequel, \( 0 \) denotes the zero matrix, vector or number of compatible dimensions. The importance of (2.1) was discussed in chapter 1 of this thesis.

The developed algorithms are based on ideas arising from probability-1 homotopy methods, for the solution of algebraic systems of equations. A primitive homotopy algorithm for a simpler quadratic equation than (2.1), namely the equation

\[ AX + XB + XCX + EXDXF + G = 0, \ A, E \in \mathbb{R}^{n \times n}, \ B, F \in \mathbb{R}^{p \times p}, \ C, D \in \mathbb{R}^{p \times n}, \]
\( G \in \mathbb{R}^{n \times p} \), was developed in [168], where only real solutions \( X \in \mathbb{R}^{n \times p} \) were considered. The algorithms to be presented in this chapter are more robust and sophisticated and they can be used for equations with both complex and/or real coefficients. Also, in contrast to [168] the proposed algorithm is able to compute one, a limited number, or all (when the number is finite) existing solutions, real and/or complex.
Computing all existing solutions discriminates the method from other heuristic methods. Note that computing all existing solutions might not be practical for equations with large dimensions. This is because the number of existing solutions (assuming that there is a finite number of them) in general increases exponentially with the dimension of the equation, according to Bezout’s theorem A.3.1 (see appendix A). For the particular case of (2.1), there can be in general $2^{np}$ geometrically isolated solutions including multiplicities, when there is a finite number of them. Nevertheless, for equations with small dimensions and when computation time is not crucial, computing all existing solutions is practically feasible. This will be illustrated with a numerical example in chapter 7.

The main contents of this chapter can be outlined as follows. An historical background of homotopy methods for the numerical solution of nonlinear algebraic equations is given in section 2.2. Next the mathematical presentation of probability-1 homotopy methods is given in section 2.3. The application of these methods for the numerical solution of (2.1), and the evaluation of respective formulas are given in section 2.4. Finally in section 2.5, some relevant conclusions are made. Throughout this chapter specialized mathematical terms and results are used. For completeness these are presented in appendixes A and B.

### 2.2 Historical Background of Homotopy Methods

General homotopy theory is a branch of topology and is widely used in the studies of differential geometry [67], [119], complex analysis [2], [160], and nonlinear dynamical systems [1], [10]. During the last two decades, the application of homotopy methods has been extended to the field of numerical analysis. They are used for the numerical solution of nonlinear systems of algebraic equations [131], [5], [142]. The philosophy of these methods is also used in interior point polynomial methods for the numerical solution of linear matrix inequalities [127], [24], [63]. Similar to homotopy methods are the continuation [127] and parameter continuation methods [131], the continuous Newton methods [131], and the probability-1 homotopy methods [32]. The differences and similarities between these methods will be explained later in this chapter.

The main reason for using homotopy methods is that they possess good global convergence properties when compared with locally convergent methods such as Newton-Raphson [131], [142], [162] and similar iterative-type methods. More

---

$^\dagger$ See definition A.3.6 in appendix A.
specifically, probability-1 homotopy methods guarantee global convergence, in a probabilistic sense, from arbitrary initial conditions. For this reason the utility of these methods is evident in difficult and non-convex problems. Recent examples of such non convex problems are various multi-objective control system design problems (e.g. mixed $H_2 / H_\infty$ [48], [49], fixed structure $H_2$/anti-windup [170]-[172], etc). Note that the selection of suitable initial conditions for convergence is almost always a problem when dealing with local convergence methods.

In the past, homotopy methods have been used for problems with low dimensions [179]. The reason for this is that homotopy type algorithms are computationally more expensive than locally convergent methods and the additional time required for precision and accuracy of high dimensional problems was not then thought desirable. The main purpose of the algorithms at that time was to obtain good starting points for locally convergent methods, especially when the problem was non-convex and/or not well understood and hence no clue for its solutions was available. At present, computation time seems less of a problem. The computer power of today is relatively cheap and more importantly it is still growing fast. One could argue that the problem has been reversed. Hence, it might be better to use directly an accurate homotopy method instead of spending human analysis time in finding suitable initial conditions for a problem using locally convergent methods.

At this point, it should be said that any numerical method (including those presented in this thesis) on its own will have difficulties in some cases. For a particular problem, one method may be better than another. There is a large variety of different methods and equal respect should be paid to all of them.

When the problem of the numerical solution of (2.1) is formulated as an optimization problem (e.g. norm minimization of $\| f(X) \|^2_F$ to zero, [162], [53]), this can produce a non-convex optimization problem (depending on its data). Therefore, a wide range of convex numerical methods by default can not be used. For non-convex problems, convergence is laborious for any method in general, most of the time. This problem is not an issue within the formulation of the present work, since the proposed algorithm is independent of the convexity of objective functions used in optimization based methods.

 Generally, the homotopy approach to the solution of an algebraic equation (original problem), is to first solve an easy equation (easy problem) which may be similar or not to the original, and then to continuously deform the easy problem and its solution, to the original problem and its solution. For this purpose an appropriate mapping (homotopy mapping) needs to be constructed. Based on this philosophy,
theorems and propositions can be developed in order to guarantee convergence and
good numerical properties. The same logic applies to probability-1 homotopy methods
under the characteristic of global convergence. As was stated earlier, probability-1 and
general homotopy methods for the numerical solution of algebraic equations have
gained respect during the last twenty years. The application of these methods
exclusively to the numerical solution of the algebraic matrix equation (2.1) is novel,
and is the main contribution of this thesis.

The history of homotopy-type methods, and homotopy theory in general, is a
large subject, too large to cover in detail in an engineering thesis. From the extensive
literature, the interested reader can find excellent references in [3]-[5], [16], [32], [33],
[55], [56], [110], [122], [124],[131], [142], [179], [182]. Probability-1 homotopy
methods are presented next.

2.3 Probability-1 Homotopy Methods

In order to customize the presentation of probability-1 homotopy methods for solving
(2.1), the numerical solution of the nonlinear algebraic polynomial vector equation†
(2.2), shown below, is primarily considered.

\[ F(x) = 0, \quad x \in \mathbb{C}^m, \quad m \in \mathbb{Z}_+ \]  \quad (2.2)

For the sequel, define the open unit interval as \( U := (0,1) \) and its closure as \( \bar{U} \). Hence
\( \bar{U} = [0,1] \).

Definition 2.1;

Consider the mappings \( F: \mathbb{C}^m \rightarrow \mathbb{C}^m \) and \( F_0: \mathbb{C}^m \times \mathbb{C}^m \times \mathbb{C}^m \rightarrow \mathbb{C}^m, \quad m \in \mathbb{Z}_+ \).

A continuous mapping \( H: \mathbb{C}^m \times \bar{U} \times \mathbb{C}^m \times \mathbb{C}^m \rightarrow \mathbb{C}^m, \) is called a homotopy from \( F_0 \)
to \( F \) if \( H(x,0,\alpha,\beta) = F_0(x,\alpha,\beta) \) and \( H(x,1,\alpha,\beta) = F(x) \)
\( \forall (x,\alpha,\beta) \in \mathbb{C}^m \times \mathbb{C}^m \times \mathbb{C}^m \). \( F_0 \) is said to be homotopic to \( F \) and \( H(x,\varepsilon,\alpha,\beta) = 0 \)
is called the homotopy equation of the homotopy variable \( \varepsilon \in \bar{U} \).

Definition 2.2 [109]:

† See definition A.3.4 in appendix A.
Consider the mappings \( F: \mathbb{C}^m \to \mathbb{C}^m \) and \( F_0: \mathbb{C}^m \to \mathbb{C}^m, \ m \in \mathbb{Z}_+ \). A continuous mapping \( H_{\alpha\beta}: \mathbb{C}^m \times \mathbb{U} \to \mathbb{C}^m \) is called a homotopy from \( F_0 \) to \( F \) if \( H_{\alpha\beta}(x,0) = F_0(x) \) and \( H_{\alpha\beta}(x,1) = F(x) \forall x \in \mathbb{C}^m \). \( F_0 \) is said to be homotopic to \( F \) and \( H_{\alpha\beta}(x,\varepsilon) = 0 \) is called the homotopy equation of the homotopy variable \( \varepsilon \in \mathbb{U} \).

Note that, \( H_{\alpha\beta} \) can be considered as a special case of \( H \) with \( \alpha, \beta \in \mathbb{C}^m \) fixed.

**Definition 2.3 [180]:**
The \( C^2 \) mapping \( H: \mathbb{C}^m \times \mathbb{U} \times \mathbb{C}^m \times \mathbb{C}^m \to \mathbb{C}^m, \ m \in \mathbb{Z}_+ \) is said to be transversal to zero if either \( H^{-1}(0) = \emptyset \), or \( H^{-1}(0) \neq \emptyset \) and \( \forall (x, \varepsilon, \alpha, \beta) \in H^{-1}(0) \), \( \text{Re} \left( \text{rank} \left( J_{H(x, \varepsilon, \alpha, \beta)} \right) \right) = 2m \),

where

\[
H^{-1}(0) = \left\{ (x, \varepsilon, \alpha, \beta) \in \mathbb{C}^m \times [0,1] \times \mathbb{C}^m \times \mathbb{C}^m \mid H(x, \varepsilon, \alpha, \beta) = 0 \right\}
\]  

(2.3)

and \( J_{H(x, \varepsilon, \alpha, \beta)} \) is the Jacobian matrix of \( H(x, \varepsilon, \alpha, \beta) \) with respect to \( (x, \varepsilon, \alpha, \beta) \).

Similarly for \( H_{\alpha\beta} \) the definition 2.4 below applies.

**Definition 2.4 [180]:**
The \( C^2 \) mapping \( H_{\alpha\beta}: \mathbb{C}^m \times \mathbb{U} \to \mathbb{C}^m, \ m \in \mathbb{Z}_+ \) is said to be transversal to zero if either \( H_{\alpha\beta}^{-1}(0) = \emptyset \), or \( H_{\alpha\beta}^{-1}(0) \neq \emptyset \) and

\[
\forall (x, \varepsilon) \in H_{\alpha\beta}^{-1}(0), \ \text{Re} \left( \text{rank} \left( J_{H_{\alpha\beta}(x, \varepsilon)} \right) \right) = 2m
\]

where

\[
H_{\alpha\beta}^{-1}(0) = \left\{ (x, \varepsilon) \in \mathbb{C}^m \times [0,1] \mid H_{\alpha\beta}(x, \varepsilon) = 0 \right\}
\]  

(2.4)

and \( J_{H_{\alpha\beta}(x, \varepsilon)} \) is the Jacobian matrix of \( H_{\alpha\beta}(x, \varepsilon) \) with respect to \( (x, \varepsilon) \).

\[ \text{A mapping is said to be } C^v, \ v \in \mathbb{Z}_+, \text{ if it is } v \text{ times continuously differentiable.} \]

\[ \text{For the operator } \text{Re} \left( \text{rank} \left( \right) \right) \text{ see definition B.2.1 in appendix B.} \]
According to definition 2.2 a homotopy method for the solution of \( F(x) = 0 \), \( x \in \mathbb{C}^m \), is to initially solve an easy problem, say \( F_0(x) = 0, \ x \in \mathbb{C}^m \) and then to continuously deform the easy problem and its solution to the original problem and its solution \( F(x) = 0, \ x \in \mathbb{C}^m \). To do this a homotopy mapping \( H \) needs to be synthesized.

Now, the parameter continuation methods follow the same strategy. For the continuation and continuous Newton methods the process is similar with parameters similar to \( \varepsilon \) not to be limited in \( \overline{U} \).

Based on the above philosophies, when using parameter continuation and homotopy methods, some problems can arise during the deformation of the easy problem to the original. More specifically, the curves \( \mu \) (usually called homotopy paths) formed by the zeros (solutions) of \( H(x, \varepsilon, \alpha, \beta) = 0 \), starting from \( (x(0), 0, \alpha, \beta) \) (where, \( H(x(0), 0, \alpha, \beta) = F_0(x(0), \alpha, \beta) = 0 \) \( \forall (x, \varepsilon, \alpha, \beta) \in \mathbb{C}^m \times \overline{U} \times \mathbb{C}^m \times \mathbb{C}^m \),

(i) may have turning points,
(ii) may bifurcate,
(iii) may cross and overlap each other,
(iv) may not exist for some \( \varepsilon \in \overline{U} \).
(v) may diverge to infinity or spiral endlessly for some \( \varepsilon \in U \).
(vi) may form closed loops for some \( \varepsilon \in U \).

The above problems (i)-(vi) will be referred to as problems (i)-(vi) in the sequel of this thesis. Problems (i)-(vi) can also be observed if we start tracing curves \( \mu \) backwards i.e. starting from \( (x(1), 1, \alpha, \beta) \) (where, \( H(x(1), 1, \alpha, \beta) = F(x(1)) = 0 \)). Hence the solution set \( \widetilde{H}^{-1}(0) \) of a homotopy mapping \( H \), during the deformation process from \( \varepsilon = 0 \) to \( \varepsilon = 1 \), is defined as the collection of all existing homotopy paths \( \mu \). In mathematical terminology,

\[
\widetilde{H}^{-1}(0) = \{(x, \varepsilon, \alpha, \beta) \in \mathbb{C}^m \times \overline{U} \times \mathbb{C}^m \times \mathbb{C}^m | H(x, \varepsilon, \alpha, \beta) = 0 \}. \tag{2.5}
\]

Similarly by fixing \( \alpha, \beta \in \mathbb{C}^m \), the solution set \( \widetilde{H}^{-1}_{\alpha\beta}(0) \) of the homotopy mapping \( H_{\alpha\beta} \) is defined as

\[
\widetilde{H}^{-1}_{\alpha\beta}(0) = \{(x, \varepsilon) \in \mathbb{C}^m \times \overline{U} | H(x, \varepsilon) = 0 \}. \tag{2.6}
\]
It is apparent from (2.5) and (2.6) that \( H^{-1}_a(0) \subset H^{-1}_a(0) \), \( H^{-1}_b(0) \subset H^{-1}_b(0) \), \( H^{-1}_a(0) \subset H^{-1}_a(0) \), \( H^{-1}_b(0) \subset H^{-1}_b(0) \). A geometric interpretation of \( H^{-1}_a(0) \) is given in figure 2.1, with problems (i)-(vi) indicated as dotted curves.

Now, the difference between homotopy and parametric continuation methods is that the former consider monotonically increasing homotopy equations as functions of \( \varepsilon \) while the latter are not limited by the monotonicity property of the homotopy equation. In other words, paths can have turning points for \( \varepsilon \in U \) with homotopy methods. In the case of continuation and Newton continuation methods similar problems to (i)-(vi) apply.

Because of the above problems, the main consideration when constructing homotopy mappings \( H \) is to ensure almost always (i.e. with probability 1), that every solution \( x(0) \) of the easy problem \( F_0(x, \alpha, \beta) = 0 \) is connected to a solution \( x(1) \) of the original problem via smooth continuous homotopy paths \( \mu \). Such mappings are

---

**Figure 2.1.** Geometric interpretation of \( H^{-1}_a(0) \).
known as probability-1 homotopy mappings. Hence, this is why the probability-1 homotopy methods are superior to all continuation based methods.

In order to include in the analysis, the characteristic of global convergence from almost every (i.e. with probability-1) solution $x(0)$ of easy problems $F_0(x, \alpha, \beta) = 0$ with a specific structure, the homotopy mapping $H$ instead of $H_{\alpha\beta}$ is used in the analysis and synthesis of results. The difference between the two mappings, although they are very often designed identical in structure, is that a specific easy problem $F_0(x) = 0$ in the homotopy $H_{\alpha\beta}$ is parameterized with the variables $\alpha, \beta \in C^m$ as the easy problem $F_0(x, \alpha, \beta) = 0$ in the homotopy $H$. Very often, the above parameterization is done in such a way that the structure of the easy problem $F_0(x) = 0$ is preserved. An immediate result of this structure preservation, is the same structure for $H_{\alpha\beta}$ and $H$. In fact only the solutions $x(0)$ are changing as a function of $\alpha, \beta \in C^m$ i.e. $x(0) = f(\alpha, \beta)$. Hence, as stated previously, the homotopy $H_{\alpha\beta}$ can be viewed as a special case of the homotopy $H$ for a specific $\alpha, \beta \in C^m$. Thus, ensuring that $H$ is a probability-1 homotopy we actually cover almost all possible homotopies $H_{\alpha\beta}$ in which each easy problem solution $x(0)$ corresponds to a specific value of $\alpha, \beta \in C^m$, i.e. $F_0(x) = F_0(x, \alpha, \beta)$. Therefore, global convergence characteristics can be guaranteed for a family of easy problems with the same structure. Now note that the set $\tilde{H}^{-1}(0)$ in (2.5) defines all possible homotopy paths $\mu$ with respect to all possible easy problem solutions.

If the mapping equation of $F$ is already a function of the homotopy variable $\varepsilon$, the mapping $F$ is itself a homotopy mapping assuming $F$ is continuous. This type of homotopy is called a natural parameter homotopy. For the case that $F$ is not a natural parameter homotopy, it is mathematically always possible to define an artificial parameter homotopy $H$ with variable $\varepsilon$. Natural homotopies may appear in the study of physical problems and bifurcation problems in dynamical systems theory [96], [145], [91], with $\varepsilon$ being the bifurcation parameter.

Furthermore, there is an indication that artificial homotopies are more computationally efficient than the natural homotopies [182], [183]. This advantage can be attributed to the structure of artificial homotopies which can be built efficiently under the mathematical analysis and derivation of certain conditions. This is almost always the case for probability-1 homotopy mappings, since the majority of these
mappings are artificial parameter homotopies. Hence, it is very often wise to use artificial parameter probability-1 homotopies even for natural parameter homotopy problems, for which the global convergence characteristics cannot be guaranteed in many cases.

Frequently used artificial homotopy mappings, are the so called line homotopies having homotopy equation

\[ H(x, \epsilon) = \epsilon F(x) - (1 - \epsilon) F_0(x). \]

Line homotopies are characterized by their good generic properties both from computational [182], [183] and theoretical [110] view points. They are used in many theoretical studies in topology such as constructions of fundamental groups and deformations of planes into smooth objects (e.g. torus) [6], proofs of theorems in analysis (e.g. the Poincare-Bohl theorem [131]), studies in manifold orientations [119], [1]. Line homotopies are also the basis for developing the numerical algorithms in this work.

Sometimes the easy problem is chosen such that its mathematical structure is similar to the original problem, with the hope that the initial condition will be effectively close to the original solution and hence an easy deformation process will result with small computational effort. In general, there is no rigorous mathematical justification of how to make such an appropriate choice of the easy problem. Assuming that no theoretical results about the existence and geometry of solutions of the original problem are available, such a choice is far from easy. Nevertheless in specific cases where knowledge about the geometry of the solutions is available, it might be possible to choose an easy problem with solution close to the original following mathematical analysis of the original problem (e.g using structural stability theory [67], [10] and/or perturbation theory [125], [93]). As will be shown later, no such choice is necessary for convergence for the probability-1 methods and therefore the methods are not restricted by such considerations, although these might be useful for some particular problems.

The synthesis of probability-1 homotopy mappings can be done via certain theorems called probability-1 homotopy theorems. For the numerical solution of (2.1), two such theorems are stated next.

### 2.3.1 Probability-1 Homotopy Theorems

**Theorem 2.1 (probability-1 fixed point polynomial homotopy):**

Suppose that \( F(x) = 0, \ x \in \mathbb{C}^m, \ m \in \mathbb{Z}_+ \) is an algebraic polynomial system of total
degree\(^t \; d\) with its solution set \(S_x\) not containing solutions at infinity\(^t\). Let \(H_{\alpha\beta}\) be a homotopy mapping as in definition 2.2 with homotopy equation

\[
H_{\alpha\beta}(x, \varepsilon) := \varepsilon F(x) - \gamma (1 - \varepsilon) F_0(x)
\]  

(2.7)

where \(\gamma \in C_x\), and with easy problem

\[
F_0(x) = \alpha \circ x - \beta^\star.
\]  

(2.8)

Furthermore suppose that \(H_{\alpha\beta}\) is \(C^2\) and complex analytic\(^t\) in \(x\). Then there is a set \(S_{\alpha\beta} \subseteq C^n \times C^m\) of measure zero\(^t\) such that \(\forall (\alpha, \beta) \in \left\{ C^n \times C^m \right\} - S_{\alpha\beta} \), \(H_{\alpha\beta}(0)\) in (2.6) contains a unique smooth homotopy path (curve) which connects the zero solution \(\beta + \alpha\), of \(F_0(x) = 0\), with one solution of \(F(x) = 0\) and with the problems (i)-(vi) not applying for this particular path.

**Proof.**

Let \(H\) be a homotopy mapping as in definition 2.1 with homotopy equation

\[
H(x, \varepsilon, \alpha, \beta) := \varepsilon F(x) - \gamma (1 - \varepsilon) F_0(x, \alpha, \beta), \quad \gamma \in C_x.
\]

and with easy problem \(F_0(x, \alpha, \beta) \equiv F_0(x) = \alpha \circ x - \beta\). Because \(H_{\alpha\beta}\) is a restriction of \(H\) with \(\forall (\alpha, \beta) \in \left\{ C^n \times C^m \right\} - S_{\alpha\beta} \) fixed, it follows that \(H\) is also \(C^2\) and complex analytic in \(x\) as \(H_{\alpha\beta}\) is (by assumption). Hence, form

\[
J_H(x, \varepsilon, \alpha, \beta) \in C^{n \times (3m + 1)}
\]

as shown next in (2.9).

\[
J_H(x, \varepsilon, \alpha, \beta) = 
\begin{bmatrix}
\frac{\partial \left[ H(x, \varepsilon, \alpha, \beta) \right]}{\partial x^T} & \frac{\partial \left[ H(x, \varepsilon, \alpha, \beta) \right]}{\partial \varepsilon} & \frac{\partial \left[ H(x, \varepsilon, \alpha, \beta) \right]}{\partial \alpha^T} & \frac{\partial \left[ H(x, \varepsilon, \alpha, \beta) \right]}{\partial \beta^T}
\end{bmatrix}
\]

(2.9)

Also from (2.7) and (2.8), \(\frac{\partial \left[ H(x, \varepsilon, \alpha, \beta) \right]}{\partial \beta^T} = \gamma (1 - \varepsilon) I_m\), where \(I_m \in \mathbb{R}^{m \times m}\) is the identity matrix. Therefore, since \(\gamma \in C_x\) by assumption, it follows that

\(^t,\dagger,\ddagger\) For the definitions, total degree, solution at infinity and set of measure zero see definitions A.3.2, A.3.9 and A.2.14 respectively in appendix A.

\(^t\circ\) is the Hadamard product operator (see appendix B).

\(^t\) A complex mapping is said to be complex analytic when its equation is a function of a complex variable and its derivative exists wherever the function is defined [2].
\[
\frac{\partial [H(x, \varepsilon, \alpha, \beta)]}{\partial \beta^T} \in \mathbb{C}^m, \ \forall \varepsilon \in [0,1).
\] (2.10)

Now (2.10), in view of (2.9), results in \( \text{Re} \text{rank}(J_H(x, \varepsilon, \alpha, \beta)) = 2m \), and hence by definition 2.3, \( H \) is transversal to zero. Since \( H \) is proven to be transversal to zero, the parameterized Sard's theorem A.4.5 (see appendix A) implies that \( H_{\alpha \beta} \), with \( \forall (\alpha, \beta) \in \left\{ \left[ \mathbb{C}^m \times \mathbb{C}^m \right] - S_{\alpha \beta} \right\} \) fixed, subject to definition 2.4, is also transversal to zero.

With \( H_{\alpha \beta} \) transversal to zero, the parameterized Sard's theorem A.4.5 (see appendix A), despite the polynomial structure of (2.7) in the present case, implies that \( \forall (\alpha, \beta) \in \left\{ \left[ \mathbb{C}^m \times \mathbb{C}^m \right] - S_{\alpha \beta} \right\} \) fixed, \( H_{\alpha \beta}^{-1}(0) \) in (2.4) consists of smooth disjoint curves which have no end points in \( \mathbb{C}^m \times U \) and have finite arc length in any compact subset of \( \mathbb{C}^m \times U \) [180]. In other words, these curves can have both end points in \( \mathbb{C}^m \times \{0\} \), \( \mathbb{C}^m \times \{1\} \) (problem (i)), can be unbounded with one end point either in \( \mathbb{C}^m \times \{0\} \) or in \( \mathbb{C}^m \times \{1\} \) (problem (v)), and can form closed loops lying entirely in \( \mathbb{C}^m \times U \) (problem (vi)). Also since \( H_{\alpha \beta}^{-1}(0) \subset \tilde{H}_{\alpha \beta}^{-1}(0) \), \( \tilde{H}_{\alpha \beta}^{-1}(0) \) includes all the previous cases plus the case of curves having one point in \( \mathbb{C}^m \times \{0\} \) and the other in \( \mathbb{C}^m \times \{1\} \) (desirable homotopy paths). Hence, \( H_{\alpha \beta} \) being transversal to zero rules out for all paths in \( H_{\alpha \beta}^{-1}(0) \), \( \tilde{H}_{\alpha \beta}^{-1}(0) \), the problems (ii) and (iii) and (v) subject to endlessly spiraling paths in \( \mathbb{C}^m \times U \). \( H_{\alpha \beta} \) being \( C^2 \) and complex analytic in \( x \) (by assumption), obviously rules out problem (iv) for all paths in \( H_{\alpha \beta}^{-1}(0) \), \( \tilde{H}_{\alpha \beta}^{-1}(0) \).

Problems (i), (v) (subject to paths tending to infinity) and (vi), presumably can still exist.

Define,
\[
\Omega_{\alpha \beta} := \left\{ H_{\alpha \beta}^{-1}(0) \cap \left[ \mathbb{C}^m \times [\delta \xi, 1 - \delta \eta] \right] \left( \alpha, \beta \right) \in \left\{ \left[ \mathbb{C}^m \times \mathbb{C}^m \right] - S_{\alpha \beta} \right\}, \delta \xi, \delta \eta \in U \right\}.
\]

Taking into account the polynomial structure in (2.7), it will be shown next that \( \tilde{\Omega}_{\alpha \beta} \neq \emptyset \). \( \tilde{\Omega}_{\alpha \beta} \neq \emptyset \) is equivalent to \( H_{\alpha \beta}(x, \varepsilon) = 0 \) having no solutions at infinity.

\footnote{See definition A.2.13 in appendix A.}
∀ ε ∈ [δ, 1 - δ]. The proof of this result is divided into two cases of (2.7) with \( d = 1 \) and \( d > 1 \).

a) \( d = 1 \).

In this case, (2.7) can be written equivalently as

\[
H_{αβ}(x, ε) = [εA - γ(1 - ε)diag(α)]x + γ(1 - ε)β,
\]

where \( A ∈ C^{m×m} \). Since, \( H_{αβ} \) is transversal to zero, \( \text{Re} \text{rank} \left( J_{H_{αβ}(x, ε)} \right) = 2m \).

where \( J_{H_{αβ}(x, ε)} := \left[ \frac{∂[H_{αβ}(x, ε)]}{∂x^T} \right] \frac{∂[H_{αβ}(x, ε)]}{∂ε} \). Recall that problems (ii)-(iv) and (v) subject to endlessly spiraling paths in \( C^m \times U \) do not occur in \( H_{αβ}^{-1}(0), \tilde{H}_{αβ}^{-1}(0) \).

From the above, it follows that \( ∀ ε ∈ [0, 1] \), there is a continuous and differentiable with respect to \( ε \) solution \( x = x(ε) \) of \( H_{αβ}(x, ε) = 0 \); i.e. there exists a \( C^1 \) mapping \( x : [0, 1] → C^m, x = x(ε) \) such that \( H_{αβ}(x(ε), ε) = 0 \). Hence, \( H_{αβ}(x(ε), ε) = 0 \) being continuously differentiable \( ∀ ε ∈ [0, 1] \), implies

\[
\frac{d}{dε} \left[ H_{αβ}(x(ε), ε) \right] = 0 ⇔ J_{H_{αβ}(x(ε), ε)} \left[ \frac{dx(ε)}{dε} \right] = 0.
\]

Where, \( J_{H_{αβ}(x(ε), ε)} := \left[ \frac{∂[H_{αβ}(x(ε), ε)]}{∂x^T} \right] \frac{∂[H_{αβ}(x(ε), ε)]}{∂ε} \).

The above, shows that \( \left[ \frac{dx(ε)}{dε} \right] \) is orthogonal to \( J_{H_{αβ}(x, ε)} \). Furthermore, since

\[
\text{Re} \text{rank} \left( J_{H_{αβ}(x, ε)} \right) = \text{Re} \text{rank} \left( J_{H_{αβ}(x(ε), ε)} \right) = 2m,
\]

taking into account the definition of B.2.1 of \( \text{Re} \text{rank}(·) \) in appendix B, it follows equivalently that
\[
\begin{align*}
\begin{bmatrix}
\text{Re} \left( \frac{\partial H_{\text{cap}}(x, \epsilon)}{\partial x(\epsilon)^T} \right) & -\text{Im} \left( \frac{\partial H_{\text{cap}}(x, \epsilon)}{\partial x(\epsilon)^T} \right) & \text{Re} \left( \frac{\partial H_{\text{cap}}(x, \epsilon)}{\partial \epsilon} \right) & -\text{Im} \left( \frac{\partial H_{\text{cap}}(x, \epsilon)}{\partial \epsilon} \right) \\
\text{Re} \left( \frac{d x(\epsilon)^T c}{d \epsilon} \right) & -\text{Im} \left( \frac{d x(\epsilon)^T c}{d \epsilon} \right) & 1 & 0 \\
\text{Im} \left( \frac{d x(\epsilon)^T c}{d \epsilon} \right) & \text{Re} \left( \frac{d x(\epsilon)^T c}{d \epsilon} \right) & 0 & 1 \\
\end{bmatrix}
= 2m + 2 \\
\begin{bmatrix}
\text{Re} \left( \frac{\partial H_{\text{cap}}(x, \epsilon)}{\partial x(\epsilon)^T} \right) & -\text{Im} \left( \frac{\partial H_{\text{cap}}(x, \epsilon)}{\partial x(\epsilon)^T} \right) & \text{Re} \left( \frac{\partial H_{\text{cap}}(x, \epsilon)}{\partial \epsilon} \right) & -\text{Im} \left( \frac{\partial H_{\text{cap}}(x, \epsilon)}{\partial \epsilon} \right) \\
\text{Re} \left( \frac{d x(\epsilon)^T c}{d \epsilon} \right) & -\text{Im} \left( \frac{d x(\epsilon)^T c}{d \epsilon} \right) & 1 & 0 \\
\text{Im} \left( \frac{d x(\epsilon)^T c}{d \epsilon} \right) & \text{Re} \left( \frac{d x(\epsilon)^T c}{d \epsilon} \right) & 0 & 1 \\
\end{bmatrix}
= 2m + 2 \\
\begin{bmatrix}
\text{Re} \left( \frac{\partial H_{\text{cap}}(x, \epsilon)}{\partial \epsilon} \right) & -\text{Im} \left( \frac{\partial H_{\text{cap}}(x, \epsilon)}{\partial \epsilon} \right) & \text{Re} \left( \frac{\partial H_{\text{cap}}(x, \epsilon)}{\partial \epsilon} \right) & -\text{Im} \left( \frac{\partial H_{\text{cap}}(x, \epsilon)}{\partial \epsilon} \right) \\
\text{Re} \left( \frac{d x(\epsilon)^T c}{d \epsilon} \right) & -\text{Im} \left( \frac{d x(\epsilon)^T c}{d \epsilon} \right) & 1 & 0 \\
\text{Im} \left( \frac{d x(\epsilon)^T c}{d \epsilon} \right) & \text{Re} \left( \frac{d x(\epsilon)^T c}{d \epsilon} \right) & 0 & 1 \\
\end{bmatrix}
= 2m + 2
\end{align*}
\]
By the determinant formula (B.2.25) (see appendix B), the last relation is equivalent to

\[
\det(I_2) \det \left( \begin{array}{c} \text{Re} \left( \frac{\partial H_{\alpha\beta}(x(\epsilon), \epsilon)}{\partial \epsilon} \right) & \text{Im} \left( \frac{\partial H_{\alpha\beta}(x(\epsilon), \epsilon)}{\partial \epsilon} \right) \\ \text{Im} \left( \frac{\partial H_{\alpha\beta}(x(\epsilon), \epsilon)}{\partial \epsilon} \right) & \text{Re} \left( \frac{\partial H_{\alpha\beta}(x(\epsilon), \epsilon)}{\partial \epsilon} \right) \end{array} \right) \neq 0
\]

(2.11)
Now, let $i^2 = -1$ be the imaginary unit. Then

\[ J_{\alpha\beta}(x(\varepsilon),\varepsilon) \left[ \frac{d[x(\varepsilon)]}{d\varepsilon} \right] = 0 \iff \left[ \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial \varepsilon} \right] \left[ \frac{d[x(\varepsilon)]}{d\varepsilon} \right] = 0 \]

\[ \Leftrightarrow \text{Re} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial \varepsilon} \right) + \text{Im} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial \varepsilon} \right) = 0 \]

Interchanging rows and columns, (2.12) can be written equivalently as equation (2.13), below.

Augmenting (2.12) and (2.13) it follows that

\[ \begin{bmatrix} \text{Re} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial \varepsilon} \right) & -\text{Im} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial \varepsilon} \right) & \text{Im} \left( \frac{d[x(\varepsilon)]}{d\varepsilon} \right) \\ \text{Im} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial \varepsilon} \right) & \text{Re} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial \varepsilon} \right) & -\text{Re} \left( \frac{d[x(\varepsilon)]}{d\varepsilon} \right) \end{bmatrix} = - \begin{bmatrix} \text{Re} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial \varepsilon} \right) \\ \text{Im} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial \varepsilon} \right) \end{bmatrix} \]

(2.13)
In view of (2.14), (2.11) becomes equivalently

\[
\begin{vmatrix}
\text{Re} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial [x(\varepsilon)^T]} \right) & -\text{Im} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial [x(\varepsilon)^T]} \right) \\
\text{Im} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial [x(\varepsilon)^T]} \right) & \text{Re} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial [x(\varepsilon)^T]} \right)
\end{vmatrix}
= \begin{cases}
I_{2m}^+ & \text{if } d x(\varepsilon)^T \\
I_{2m}^- & \text{if } d x(\varepsilon)^T
\end{cases}
\neq 0
\]

\[
\Rightarrow \text{det} \begin{vmatrix}
\text{Re} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial [x(\varepsilon)^T]} \right) & -\text{Im} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial [x(\varepsilon)^T]} \right) \\
\text{Im} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial [x(\varepsilon)^T]} \right) & \text{Re} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial [x(\varepsilon)^T]} \right)
\end{vmatrix}
= \begin{cases}
I_{2m}^+ & \text{if } d x(\varepsilon)^T \\
I_{2m}^- & \text{if } d x(\varepsilon)^T
\end{cases}
\neq 0
\]
\[
\begin{aligned}
\text{det} & \begin{pmatrix}
\text{Re} \left( \frac{\partial [H_{\alpha\beta}(x(e),\epsilon)]}{\partial [x(e)^T]} \right) & -\text{Im} \left( \frac{\partial [H_{\alpha\beta}(x(e),\epsilon)]}{\partial [x(e)^T]} \right) \\
\text{Im} \left( \frac{\partial [H_{\alpha\beta}(x(e),\epsilon)]}{\partial [x(e)^T]} \right) & \text{Re} \left( \frac{\partial [H_{\alpha\beta}(x(e),\epsilon)]}{\partial [x(e)^T]} \right)
\end{pmatrix} \\
\text{det} & \begin{pmatrix}
\text{Re} \left( \frac{d[x(e)]}{d\epsilon} \right) & -\text{Im} \left( \frac{d[x(e)]}{d\epsilon} \right) \\
\text{Im} \left( \frac{d[x(e)]}{d\epsilon} \right) & \text{Re} \left( \frac{d[x(e)]}{d\epsilon} \right)
\end{pmatrix}
\end{aligned}
\]

\[
\begin{aligned}
&\Rightarrow
\begin{pmatrix}
\text{Re} \left( \frac{\partial [H_{\alpha\beta}(x(e),\epsilon)]}{\partial [x(e)^T]} \right) & -\text{Im} \left( \frac{\partial [H_{\alpha\beta}(x(e),\epsilon)]}{\partial [x(e)^T]} \right) \\
\text{Im} \left( \frac{\partial [H_{\alpha\beta}(x(e),\epsilon)]}{\partial [x(e)^T]} \right) & \text{Re} \left( \frac{\partial [H_{\alpha\beta}(x(e),\epsilon)]}{\partial [x(e)^T]} \right)
\end{pmatrix} \neq 0 \text{ and }

\begin{pmatrix}
\text{Re} \left( \frac{d[x(e)]}{d\epsilon} \right) & -\text{Im} \left( \frac{d[x(e)]}{d\epsilon} \right) \\
\text{Im} \left( \frac{d[x(e)]}{d\epsilon} \right) & \text{Re} \left( \frac{d[x(e)]}{d\epsilon} \right)
\end{pmatrix} \neq 0
\end{aligned}
\]

Hence,

\[
\begin{aligned}
\text{det} & \begin{pmatrix}
\text{Re} \left( \frac{\partial [H_{\alpha\beta}(x(e),\epsilon)]}{\partial [x(e)^T]} \right) & -\text{Im} \left( \frac{\partial [H_{\alpha\beta}(x(e),\epsilon)]}{\partial [x(e)^T]} \right) \\
\text{Im} \left( \frac{\partial [H_{\alpha\beta}(x(e),\epsilon)]}{\partial [x(e)^T]} \right) & \text{Re} \left( \frac{\partial [H_{\alpha\beta}(x(e),\epsilon)]}{\partial [x(e)^T]} \right)
\end{pmatrix} \\
\text{det} & \begin{pmatrix}
\text{Re} \left( \frac{d[x(e)]}{d\epsilon} \right) & -\text{Im} \left( \frac{d[x(e)]}{d\epsilon} \right) \\
\text{Im} \left( \frac{d[x(e)]}{d\epsilon} \right) & \text{Re} \left( \frac{d[x(e)]}{d\epsilon} \right)
\end{pmatrix} = 0
\end{aligned}
\]

and therefore,

\[
\text{Re} \, \text{rank} \left( \frac{\partial [H_{\alpha\beta}(x(e),\epsilon)]}{\partial [x(e)^T]} \right) = 2m.
\]

From (2.16) it follows that

\[
\begin{aligned}
\text{det} & \left( \frac{\partial [H_{\alpha\beta}(x(e),\epsilon)]}{\partial [x(e)^T]} \right) \neq 0 \Leftrightarrow \text{det} \left( [\epsilon A - \gamma(1-\epsilon)\text{diag}(\alpha)] \right) \neq 0.
\end{aligned}
\]

From (2.17), it is apparent that there is a unique solution to \( H_{\alpha\beta}(x(e),\epsilon) = 0 \); namely,

\[
x(e) = -\gamma(1-\epsilon)[\epsilon A - \gamma(1-\epsilon)\text{diag}(\alpha)]^{-1}\beta = 0.
\]

Moreover, \( H_{\alpha\beta} \) is norm-coercive\(^*\), since \( \forall (\epsilon, \alpha, \beta) \in [0,1-\delta\eta] \times \{C^m_\alpha \times C^m_\alpha - S_{\alpha\beta}\}, \)

\[
\lim_{\|x(e)\|_2 \to \infty} \left( \|\epsilon A - \gamma(1-\epsilon)\text{diag}(\alpha)\| x(e) + \gamma(1-\epsilon)\beta \right)_2
\]

\(^*\) See definition a.4.13 in appendix A.
\[
\lim_{\|x(\varepsilon)\|_2 \to \infty} \left(\| \varepsilon A - \gamma(1 - \varepsilon) \text{diag}(\alpha) \| x(\varepsilon)\|_2 \right) = \infty.
\]

Hence the unique solution \( x(\varepsilon) = -\gamma(1 - \varepsilon) [\varepsilon A - \gamma(1 - \varepsilon) \text{diag}(\alpha)]^{-1} \beta = 0 \) is not at infinity. Therefore, for \( d = 1 \)

\[
\forall (x, \varepsilon) \in \left\{ H^{-1}_{\alpha\beta}(0) \cap \left[ C_m \times [0,1 - \delta \eta] \right] \right\}, (\alpha, \beta) \in \left\{ \left[ C_m \times C_m^m \right] - S_{\alpha\beta} \right\}, \delta \eta \in U \}
\]

and therefore \( \forall (x, \varepsilon) \in \Omega_{\alpha\beta}, d = 1, H_{\alpha\beta}(x, \varepsilon) = 0 \) has no solutions at infinity. Hence \( \Omega_{\alpha\beta} \neq \emptyset \).

b) \( d > 1 \).

In this case, the definition of solutions at infinity in appendix A is used. According to this, (2.7) has solutions at infinity in \( \Omega_{\alpha\beta} \), if the homogeneous\(^1\) equation of \( H_{\alpha\beta}(x, \varepsilon) = 0 \), denoted as \( \tilde{H}_{\alpha\beta}(x, \varepsilon) = 0 \) has at least one solution \( (\tilde{x}, \tilde{\varepsilon}) \) such that \( (\tilde{x}, \tilde{\varepsilon}) \in \Omega - \{(0,0)\} \). Now, from (2.7) with \( d > 1 \) it follows that \( \tilde{H}_{\alpha\beta}(x, \varepsilon) = \varepsilon \tilde{F}(x) \), where \( \tilde{F}(x) \) is the homogeneous part\(^2\) of \( F(x) \). Therefore,

\[
\tilde{H}_{\alpha\beta}(x, \varepsilon) = 0 \iff \varepsilon \tilde{F}(x) = 0 \iff \tilde{F}(x) = 0.
\]

From the last equivalence it is apparent that the solutions of \( \tilde{H}_{\alpha\beta}(x, \varepsilon) = 0 \) are the same as the solutions of \( \tilde{F}(x) = 0 \). Therefore, \( H_{\alpha\beta}(x, \varepsilon) = 0 \) has solutions at infinity in \( \Omega_{\alpha\beta} \) if and only if \( F(x) = 0 \) has solutions at infinity. But by assumption \( F(x) = 0 \) has no solutions at infinity. Hence, for \( d > 1 \),

\[\forall \varepsilon \in [\delta \xi, 1 - \delta \eta], \quad H_{\alpha\beta}(x, \varepsilon) = 0 \]

has no solutions at infinity and this implies \( \Omega_{\alpha\beta} \neq \emptyset \).

From the above so far it has been proven that homotopy paths cannot tend to infinity in \( \Omega_{\alpha\beta} \). Now as \( \varepsilon \to 1 \) and moreover \( \varepsilon \notin [\delta \xi, 1 - \delta \eta] \), \( \tilde{H}_{\alpha\beta}(x, \varepsilon) \to \tilde{F}(x) \).

Therefore, similarly to the previous paragraph, \( H_{\alpha\beta}(x, \varepsilon) = 0 \) has solutions at infinity in

\[
\hat{\Omega}_{\alpha\beta} := \Omega_{\alpha\beta} \cup \left\{ \tilde{H}_{\alpha\beta}^{-1}(0) \cap \left[ C_m \times (1 - \delta \eta, 1) \right] \right\}, (\alpha, \beta) \in \left\{ \left[ C_m \times C_m^m \right] - S_{\alpha\beta} \right\}, \delta \eta \in U \}
\]

and only if \( F(x) = 0 \) has solutions at infinity. But by assumption, \( F(x) = 0 \) has no solutions at infinity. Therefore \( \hat{\Omega}_{\alpha\beta} \neq \emptyset \) and obviously \( \Omega_{\alpha\beta} \cup \hat{\Omega}_{\alpha\beta} \neq \emptyset \). The last

\(^{1}\) See definition A.3.5 in appendix A.

\(^{2}\) See definition A.3.3 in appendix A.
result means that paths cannot tend to infinity for $\forall \varepsilon \in [\delta r, 1]$. Also at $\varepsilon = 0$, there is a unique solution $x(0) = \beta + \alpha$ not at infinity $\forall (\alpha, \beta) \in \left(\left\{C_u^m \times C_u^m\right\} - S_{\alpha \beta}\right)$ fixed. Hence the only case for paths to tend to infinity is when $\varepsilon \to 0$ and $\varepsilon \notin [\delta r, 1]$, $H_{\alpha \beta}(x, \varepsilon) = 0$. The last situation is false for the case of (2.7) with $d = 1$ as showed in a) previously and it is true for $d > 1$, since $H_{\alpha \beta}(x, \varepsilon) = 0$ has more than one solution for $\varepsilon > 0$ and $C^m \times U$ has no end points.

Hence, so far it has been proven that homotopy paths can tend to infinity only as $\varepsilon \to 0$. Now, because of the uniqueness of the solution $x(0) = \beta + \alpha$ and the continuity and smoothness of $H_{\alpha \beta}$, the uniqueness of a smooth bounded and continuous homotopy path in $\widetilde{H}^{-1}_{\alpha \beta}(0)$, emanating from $x(0) = \beta + \alpha$ at $\varepsilon = 0$ and terminating at a solution $x(1)$ (of $F(x) = 0$) at $\varepsilon = 1$, follows $\forall (\alpha, \beta) \in \left(\left\{C_u^m \times C_u^m\right\} - S_{\alpha \beta}\right)$ fixed. For this particular path, problems (ii)-(v) do not occur.

Problems (i) and (vi) are ruled out as a consequence of the assumption of $H_{\alpha \beta}$ being $C^2$ and complex analytic in $x$, and $F(x) = 0$ being an algebraic polynomial system. This is basically a consequence of the Cauchy-Riemann equations (see (A.3.5), (A.3.6) in appendix A) for $H$. The proof of this particular result is given next. At this point it should be noted that through this proof, the ruling out of problem (ii) will also be vindicated.

Note that (2.11)-(2.14) in case a) and the results that these equations generate are generic and independent of $d$. Now from $H_{\alpha \beta}$ being $C^2$, the continuity and smoothness of paths, and with problems (ii)-(v) eliminated as before, it follows that $\forall \varepsilon \in [0, 1]$, there is a continuous and differentiable with respect to $\varepsilon$ solution $x(x(\varepsilon))$ of $H_{\alpha \beta}(x, \varepsilon) = 0$; that is, there exists a $C^1$ mapping $x: [0, 1] \to C^m$, $x = x(\varepsilon)$ such that $H_{\alpha \beta}(x(\varepsilon), \varepsilon) = 0$. Hence, since $H_{\alpha \beta}(x(\varepsilon), \varepsilon) = 0$ is obviously continuously differentiable $\forall \varepsilon \in [0, 1]$, it follows that

$$\frac{d}{d\varepsilon}[H_{\alpha \beta}(x(\varepsilon), \varepsilon)] = 0.$$  \hspace{1cm} (2.18)
Recall that \( H_{\alpha \beta} \) is complex analytic by assumption, and \( H_{\alpha \beta}(x(\epsilon), \epsilon) = 0 \) can be equivalently written as

\[
H_{\alpha \beta}(x(\epsilon), \epsilon) = u_{\alpha \beta}(\Re(x(\epsilon)), \Im(x(\epsilon)), \epsilon) + v_{\alpha \beta}(\Re(x(\epsilon)), \Im(x(\epsilon)), \epsilon)i = 0 + 0i \tag{2.19}
\]

where

\[
u_{\alpha \beta}(x(\epsilon), \epsilon) = \Re\left(H_{\alpha \beta}(x(\epsilon), \epsilon)\right) \tag{2.20}
\]

\[
u_{\alpha \beta}(x(\epsilon), \epsilon) = \Im\left(H_{\alpha \beta}(x(\epsilon), \epsilon)\right) \tag{2.21}
\]

From complex analysis [2], [160] it is known that (2.19), in view of (2.20) and (2.21), can be equivalently considered as the real system

\[
P_{\alpha \beta}(x_r(\epsilon), \epsilon) := \begin{bmatrix} u_{\alpha \beta}(\Re(x(\epsilon)), \Im(x(\epsilon)), \epsilon) \\ v_{\alpha \beta}(\Re(x(\epsilon)), \Im(x(\epsilon)), \epsilon) \end{bmatrix} = 0 \tag{2.22}
\]

where, \( x_r(\epsilon) := \begin{bmatrix} \Re(x(\epsilon)) \\ \Im(x(\epsilon)) \end{bmatrix} \) and \( x_r \in \mathbb{R}^{2m} \). Hence (2.18) in view of (2.19) becomes

\[
\frac{d}{d\epsilon} P_{\alpha \beta}(x_r(\epsilon), \epsilon) = 0. \tag{2.23}
\]

Equation (2.22) defines, in general, a nonlinear dynamical system in the real Euclidean space \( \mathbb{R}^{2m} \). Hence, the real and imaginary parts of the homotopy paths in \( H_{\alpha \beta}^{-1}(0) \) are determined uniquely by the dynamic flow (trajectories) of (2.23), in \( \mathcal{C}^m \times [0,1] \) with initial condition the easy problem solution \( x_r(0) := \begin{bmatrix} \Re(x(0)) \\ \Im(x(0)) \end{bmatrix} \).

\( H_{\alpha \beta}(x(0), 0) = F_0(x(0), \alpha, \beta) = 0 \). Now, since \( \forall (\alpha, \beta) \in \left\{ \mathcal{C}^m \times \mathcal{C}^m \right\} - S_{\alpha \beta} \), \( H_{\alpha \beta} \) is transversal to zero, \( \Re\text{rank} \left( J_{H_{\alpha \beta}}(x(\epsilon), \epsilon) \right) = 2m \). \( J_{H_{\alpha \beta}}(x(\epsilon), \epsilon) \in \mathcal{C}^{m \times (m+1)} \).

From (2.18) it follows that

\[
\frac{d}{d\epsilon} [H_{\alpha \beta}(x(\epsilon), \epsilon)] = 0 \Leftrightarrow J_{H_{\alpha \beta}}(x, \epsilon) \left[ \frac{d}{d\epsilon} [x(\epsilon)] \right] = 0. \tag{2.24}
\]

From (2.24) the tangent to \( J_{H_{\alpha \beta}}(x(\epsilon), \epsilon) \) vector \( \left[ \frac{d}{d\epsilon} [x(\epsilon)] \right] \) satisfies,

\[
\left[ \frac{d}{d\epsilon} [x(\epsilon)] \right] \in \text{Ker} \left( J_{H_{\alpha \beta}}(x(\epsilon), \epsilon) \right). \]
Chapter 2  

Hence, as in case a) (2.16) follows and

\[
\det \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial [x^T(\varepsilon)]} \right) \neq 0, \quad \forall \varepsilon \in [0,1).
\]  

(2.25)

By the equivalence of (2.19) and (2.22) and in view of (2.12), (2.23) can be written equivalently, without loss of generality, in real Euclidean space \( \mathbb{R}^m \) as shown next.

\[
J_{P_{\alpha\beta}(x_r(\varepsilon),\varepsilon)} = 0 \iff \left[ \frac{\partial [P_{\alpha\beta}(x_r(\varepsilon),\varepsilon)]}{\partial x_r(\varepsilon)} \right] \left[ \frac{\partial [P_{\alpha\beta}(x_r(\varepsilon),\varepsilon)]}{\partial x_r(\varepsilon)^T} \right] = 0
\]  

(2.26)

where,

\[
\frac{\partial [P_{\alpha\beta}(x_r(\varepsilon),\varepsilon)]}{\partial [x_r(\varepsilon)^T]} = \begin{bmatrix}
\text{Re} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial [x(\varepsilon)^T]} \right) & -\text{Im} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial [x(\varepsilon)^T]} \right) \\
\text{Im} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial [x(\varepsilon)^T]} \right) & \text{Re} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial [x(\varepsilon)^T]} \right)
\end{bmatrix}
\]

and

\[
\frac{\partial [P_{\alpha\beta}(x_r(\varepsilon),\varepsilon)]}{\partial x_r(\varepsilon)} = \begin{bmatrix}
\text{Re} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial x_r(\varepsilon)} \right) \\
\text{Im} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial x_r(\varepsilon)} \right)
\end{bmatrix}, \quad \frac{\partial [P_{\alpha\beta}(x_r(\varepsilon),\varepsilon)]}{\partial x_r(\varepsilon)^T} = \begin{bmatrix}
\text{Re} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial x_r(\varepsilon)} \right) \\
\text{Im} \left( \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial x_r(\varepsilon)} \right)
\end{bmatrix}.
\]

Since (2.15) holds, it is apparent that

\[
\text{rank} \left( \frac{\partial [P_{\alpha\beta}(x_r(\varepsilon),\varepsilon)]}{\partial [x_r(\varepsilon)^T]} \right) = 2m, \quad \forall \varepsilon \in [0,1) \quad \text{and therefore}
\]

\[
\det \left( \frac{\partial [P_{\alpha\beta}(x_r(\varepsilon),\varepsilon)]}{\partial [x_r(\varepsilon)^T]} \right) \neq 0, \quad \forall \varepsilon \in [0,1)
\]  

(2.27)

Equation (2.27) (and hence (2.25)), in view of (2.26)) implies that no bifurcations ([91], [10]) occur in paths of \( H_{\alpha\beta}^{-1}(0) \) and this is why problem (ii) is ruled out.
The orientation of the homotopy paths now in $F^{-1}(0)$, is determined by

$$\text{sign det} \left( \frac{\partial P_{\alpha \beta}(x_r(\varepsilon), \varepsilon)}{\partial x_r(\varepsilon)^T} \right), \forall \varepsilon \in [0,1).$$

In other words,

$$\text{if } \text{sign det} \left( \frac{\partial P_{\alpha \beta}(x_r(\varepsilon), \varepsilon)}{\partial x_r(\varepsilon)^T} \right) = \text{const}, \forall \varepsilon \in [0,1),$$

then the homotopy paths do not have turning points and do not form closed loops in $F^{-1}(0)$. Using the Cauchy-Riemann equations (see definition A.3.10 in appendix A),

$$\text{det} \left( \frac{\partial P_{\alpha \beta}(x_r(\varepsilon), \varepsilon)}{\partial x_r(\varepsilon)^T} \right)$$

becomes equivalently

$$\text{det} \left( \frac{\partial P_{\alpha \beta}(x_r(\varepsilon), \varepsilon)}{\partial x_r(\varepsilon)^T} \right) = \text{det} \left[ \begin{bmatrix} \phi & -\rho \\ \rho & \phi \end{bmatrix} \right].$$

(2.28)

where in view of (A.3.5), (A.3.6) in definition A.3.10 (see appendix A)

$$\varphi = \frac{\partial [\text{Re}(x(\varepsilon)), \text{Im}(x(\varepsilon))]}{\partial [\text{Re}(x(\varepsilon)^T)]} = \text{Re} \left( \frac{\partial [H_{\alpha \beta}(x(\varepsilon), \varepsilon)]}{\partial [x(\varepsilon)^T]} \right)$$

and

$$\rho = \frac{\partial [\text{Re}(x(\varepsilon)), \text{Im}(x(\varepsilon))]}{\partial [\text{Re}(x(\varepsilon)^T)]} = \text{Im} \left( \frac{\partial [H_{\alpha \beta}(x(\varepsilon), \varepsilon)]}{\partial [x(\varepsilon)^T]} \right).$$

Because (2.27), equation Equation (2.28) in terms of proposition B.2.2 in appendix B, results in

$$\text{det} \left( \frac{\partial P_{\alpha \beta}(x_r(\varepsilon), \varepsilon)}{\partial x_r(\varepsilon)^T} \right) > 0.$$

Therefore, \text{sign} \left( \text{det} \left( \frac{\partial P_{\alpha \beta}(x_r(\varepsilon), \varepsilon)}{\partial x_r(\varepsilon)^T} \right) \right) = 1, \forall \varepsilon \in [0,1).$$

Hence problems (i) and (vi) are ruled out in $F^{-1}(0)$, and since the above results hold generically \(\forall (\alpha, \beta) \in \left\{ \left[ C_o^m \times C_o^m \right] - S_{\alpha \beta} \right\}\), these problems are ruled out from $F^{-1}(0)$ too.
Finally note, that the above results hold \( \forall (\alpha, \beta) \in \left[ \left( \mathbb{C}_m \times \mathbb{C}_m \right) - S_{\alpha\beta} \right] \), with \( S_{\alpha\beta} \subseteq \mathbb{C}_m \times \mathbb{C}_m \) having measure zero, or in other words with probability-1. Hence the proof is complete.

At this point it should be said that similar theorems can be found in various studies, (e.g. [182], [183], [178], [60], [61], [187]). Because the above theorem guarantees the deformation of the unique solution of the easy problem \( F_0(x) \equiv F_0(x, \alpha, \beta) = 0 \) to a geometrically isolated solution (fixed point) of the polynomial system (2.2) with probability one, this theorem is referred to as the probability-1 fixed-point polynomial homotopy theorem.

The next theorem 2.2, is the consequence of results stated in [122].

**Theorem 2.2 (probability-1 polynomial homotopy):**

Suppose that \( F(x) = 0 \), \( x \in \mathbb{C}_m \), \( m \in \mathbb{Z}_+ \) is an algebraic polynomial system of total degree \( d \) that has no solutions at infinity and has \( r \) geometrically isolated solutions. Let \( H_{\alpha\beta} \) be a homotopy mapping as in definition 2.2 with homotopy equation

\[
H_{\alpha\beta}(x, \epsilon) = \epsilon F(x) - (1 - \epsilon)F_0(x)
\]

where \( \gamma \in \mathbb{C}_m \), and with easy problem

\[
F_0(x) = \alpha \cdot \begin{bmatrix} x_1^{d_1} & x_2^{d_2} & \cdots & x_m^{d_m} \end{bmatrix}^T - \beta.
\]

where \( d_i, i = 1, \ldots, m \) is the total degree of the polynomial system being the i-th row element of \( F(x) = 0 \). Furthermore, suppose that \( H_{\alpha\beta} \) is \( C^2 \) and complex analytic in \( x \). Then there is a set \( S_{\alpha\beta} \subseteq \mathbb{C}_m \times \mathbb{C}_m \) of measure zero such that \( \forall (\alpha, \beta) \in \left[ \left( \mathbb{C}_m \times \mathbb{C}_m \right) - S_{\alpha\beta} \right] \) fixed, \( H_{\epsilon\beta}^{-1}(0) \) in (2.6) contains \( r \leq d \) smooth homotopy paths (curves) connecting \( r \) geometrically isolated solutions of \( F_0(x) = 0 \) with the \( r \) geometrically isolated solutions (counting multiplicities) of \( F(x) = 0 \) in 1-1 relationship; and \( d - r \) smooth homotopy paths connecting \( d - r \) geometrically isolated solutions of \( F_0(x) = 0 \) with \( \sigma < d - r \), \( \sigma \in \mathbb{Z}_+ \) non geometrically isolated solutions of \( F(x) = 0 \) in 1-1 relationship. For all these particular paths problems (i)-(vi) do not apply.
Proof.

The proof follows the same logic as in theorem 2.1. More specifically, let $H$ be a homotopy mapping as in definition 2.1 with homotopy equation

$$H(x, \varepsilon, \alpha, \beta) = \varepsilon F(x) - \gamma(1 - \varepsilon) F_0(x, \alpha, \beta), \quad \gamma \in \mathbb{C}_n,$$

and with easy problem $F_0(x, \alpha, \beta) = F(x) = \alpha \cdot x - \beta$. Because $H_{\alpha \beta}$ is a restriction of $H$ with $\forall (\alpha, \beta) \in \left[\mathbb{C}_n^m \times \mathbb{C}_n^m\right] - S_{\alpha \beta}$ fixed, it follows that $H$ is also $C^2$ and complex analytic in $x$ as $H_{\alpha \beta}$ is (by assumption). Hence, form

$$J_H(x, \varepsilon, \alpha, \beta) \in \mathbb{C}^{m \times (3m+1)}$$

as shown next in (2.9).

$$J_H(x, \varepsilon, \alpha, \beta) = \begin{bmatrix}
\frac{\partial [H(x, \varepsilon, \alpha, \beta)]}{\partial \alpha^T} & \frac{\partial [H(x, \varepsilon, \alpha, \beta)]}{\partial \varepsilon} & \frac{\partial [H(x, \varepsilon, \alpha, \beta)]}{\partial \alpha^T} & \frac{\partial [H(x, \varepsilon, \alpha, \beta)]}{\partial \beta^T}
\end{bmatrix}
\tag{2.31}
$$

Also from (2.7) and (2.8),

$$\frac{\partial [H(x, \varepsilon, \alpha, \beta)]}{\partial \beta^T} = \gamma(1 - \varepsilon)I_m, \quad \text{where } I_m \in \mathbb{R}^{m \times m} \text{ is the identity matrix. Therefore, since } \gamma \in \mathbb{C}_n \text{ by assumption, it follows that}$$

$$\frac{\partial [H(x, \varepsilon, \alpha, \beta)]}{\partial \beta^T} \in \mathbb{C}^{m}, \quad \forall \varepsilon \in [0,1). \tag{2.32}
$$

Now (2.31), in view of (2.32) results in $\text{Re} \text{rank}(J_H(x, \varepsilon, \alpha, \beta)) = 2m$, and hence by definition 2.3 $H$ is transversal to zero. Since $H$ is proven to be transversal to zero, the parameterized Sard’s theorem A.4.5 (see appendix A) implies that $H_{\alpha \beta}$, with $\forall (\alpha, \beta) \in \left[\mathbb{C}_n^m \times \mathbb{C}_n^m\right] - S_{\alpha \beta}$ fixed, subject to definition 2.4 is also transversal to zero.

With $H_{\alpha \beta}$ being transversal to zero, the parameterized Sard’s theorem, despite the polynomial structure of (2.29) in the present case, implies that $\forall (\alpha, \beta) \in \left[\mathbb{C}_n^m \times \mathbb{C}_n^m\right] - S_{\alpha \beta}$ fixed, $H_{\alpha \beta}^{-1}(0)$ in (2.4) consists of smooth disjoint curves which have no end points in $\mathbb{C}^m \times U$ and have finite arc lengths in any compact subset of $\mathbb{C}^m \times U$. In other words, these curves can have both end points in $\mathbb{C}^m \times \{0\}$, $\mathbb{C}^m \times \{1\}$ (problem (i)), can be unbounded with one end point either in $\mathbb{C}^m \times \{0\}$ or in $\mathbb{C}^m \times \{1\}$ (problem (v)), and can form closed loops lying
entirely in $C^n \times U$ (problem (vi)). Also since $H_{\alpha\beta}^{-1}(0) \subset \tilde{H}_{\alpha\beta}^{-1}(0)$, $\tilde{H}_{\alpha\beta}(0)$ includes all the previous cases plus the case of curves having one point in $C^n \times \{0\}$ and the other in $C^n \times \{1\}$ (desirable homotopy paths). Hence, $H_{\alpha\beta}$ being transversal to zero rules out for all paths in $H_{\alpha\beta}^{-1}(0)$, $\tilde{H}_{\alpha\beta}^{-1}(0)$, the problems (ii) and (iii) and (v) subject to endlessly spiraling paths in $C^n \times U$. $H_{\alpha\beta}$ being $C^2$ and complex analytic in $x$ (by assumption), obviously rules out problem (iv) for all paths in $H_{\alpha\beta}^{-1}(0)$, $\tilde{H}_{\alpha\beta}(0)$.

Problems (i), (v) subject to paths tending to infinity, and (vi), presumably can still exist.

Define,

$$\Theta_{\alpha\beta} := \left\{ \left\{ H_{\alpha\beta}^{-1}(0) \cap \left[ C^n \times [0.1-\delta \eta] \right] \right\} (x,\beta) \in \left\{ \left\{ C^n \times C^n \right\} - S_{\alpha\beta} \right\}, \delta \eta \in U \right\}.$$ 

Taking into account the polynomial structure in (2.24), it will be shown next that $\Theta_{\alpha\beta} \neq \emptyset$, $\Theta_{\alpha\beta} \neq \emptyset$ is equivalent to $H_{\alpha\beta}(x,\varepsilon) = 0$ having no solutions at infinity $\forall \varepsilon \in [0.1-\delta \eta]$. The definition of solutions at infinity in appendix A is used.

According to this, (2.29) has solutions at infinity in $\Theta_{\alpha\beta}$, if the homogeneous equation of $H_{\alpha\beta}(x,\varepsilon) = 0$, denoted as $\tilde{H}_{\alpha\beta}(x,\varepsilon) = 0$ has at least one solution $(\bar{x},\bar{\varepsilon})$ such that $(\bar{x},\bar{\varepsilon}) \in \Theta_{\alpha\beta} - \{(0,1)\}$. Now, from (2.29) it follows that $\tilde{H}_{\alpha\beta}(x,\varepsilon) = \varepsilon F(x) - \gamma (1-\varepsilon) \tilde{F}(0)$, where $F(x)$ and $\tilde{F}(0)$ are the homogeneous parts of $F(x)$ and $F_0(x)$ respectively. From (2.30) it is clear that $\tilde{F}_0(x) = \alpha \left[ x_1^{d_1}, x_2^{d_2}, \ldots, x_m^{d_m} \right]^T$. Suppose that $H_{\alpha\beta}(x,\varepsilon) = 0$ has at least one solution at infinity in $\Theta_{\alpha\beta}$, then $\exists \varepsilon \in [0.1-\delta \eta]$ such that $\tilde{H}_{\alpha\beta}(x(\varepsilon),\varepsilon) = 0$, $x(\varepsilon) \neq 0$. Without loss of generality $x(\varepsilon)$ can be ordered with respect to its non zero elements. Hence assuming that there are $k \in [1,m]$ non zero elements of $x(\varepsilon)$, $\tilde{H}_{\alpha\beta}(x(\varepsilon),\varepsilon) = 0$ is equivalent to

$$\tilde{H}_{\alpha\beta} \left[ \begin{array}{c} x_k(\varepsilon) \\ 0_{(m-k) \times 1} \end{array} \right], \varepsilon = \delta \left[ \begin{array}{c} x_k(\varepsilon) \\ 0_{(m-k) \times 1} \end{array} \right]$$ 

$$-\gamma (1-\varepsilon) \left[ \begin{array}{c} \alpha_k(\varepsilon) \\ \alpha_{(m-k) \times 1} \end{array} \right] \left[ \begin{array}{c} x_1(\varepsilon)^{d_1} \\ x_2(\varepsilon)^{d_2} \\ \vdots \\ x_k(\varepsilon)^{d_k} \\ 0_{1 \times (m-k)} \end{array} \right] \gamma^T = 0$$
\[ \Rightarrow \alpha_k = \frac{\tilde{e}}{\gamma(1-\tilde{e})} \begin{bmatrix} 1 \\ x_{1j}(\tilde{e})^d \\ x_{2j}(\tilde{e})^d \\ \vdots \\ x_{kj}(\tilde{e})^d \end{bmatrix}^T = 0 \]

where,

\[ x_j(\tilde{e}) = \begin{bmatrix} x_{1j}(\tilde{e}) \\ x_{2j}(\tilde{e}) \\ \vdots \\ x_{kj}(\tilde{e}) \end{bmatrix} \quad \alpha_j = \begin{bmatrix} \alpha_{1j} \\ \alpha_{2j} \\ \vdots \\ \alpha_{kj} \end{bmatrix}, \quad j \in \mathbb{Z}_+ \]

\[ c_{xkj} = 1, \quad c \in \mathbb{C}, \quad \alpha_{(m-k)j} = \begin{bmatrix} \alpha_{(k+1)j} \\ \alpha_{(k+2)j} \\ \vdots \\ \alpha_{mj} \end{bmatrix}^T. \]

\[ x(\tilde{e}) = \begin{bmatrix} x_k(\tilde{e}) \\ \mathbf{0}_{1 \times (m-k)} \end{bmatrix}^T, \quad F(x_k(\tilde{e})) = H_{\alpha_{k-1}}(x(1), \cdot), \quad \text{and} \quad \mathbf{0}_{1 \times j} \text{ is the zero row vector with dimensions } 1 \times j. \]

From the above it is obvious that

\[ \Lambda_{\alpha_{k-1}}(x_{k-1}(\tilde{e}), \tilde{e}) = \varepsilon F_k(x_{k-1}(\tilde{e})) \]

\[ -\gamma(1-\tilde{e}) \alpha_{k-1} \begin{bmatrix} c_{d1} x_{1j}(\tilde{e})^d \\ c_{d2} x_{2j}(\tilde{e})^d \\ \vdots \\ c_{dk-1} x_{(k-1)j}(\tilde{e})^d \end{bmatrix}^T = 0 \]

is the homotopy equation at \((x_{k-1}(\tilde{e}), \tilde{e}) = (x_{k-1}(\tilde{e}), \tilde{e})\) of the homotopy mapping \(\Lambda_{\alpha_{k-1}} : C_+^{k-1} \times U \rightarrow C^{k-1}\).

Considering \(\alpha_k\) as a varying parameter, the homotopy \(\Lambda : C_+^{k-1} \times U \times C_+^{k-1} \rightarrow C^{k-1}\) is defined with homotopy equation \(\Lambda(x_{k-1}(\tilde{e}), \tilde{e}, \alpha_{k-1}) = \alpha_{k-1}(x_{k-1}(\tilde{e}), \tilde{e}) = 0\).

Hence,

\[ J_{\Lambda}(x_{k-1}(\tilde{e}), \tilde{e}, \alpha_{k-1}) = \begin{bmatrix} \frac{\partial \Lambda(x_{k-1}(\tilde{e}), \tilde{e}, \alpha_{k-1})}{\partial x(\tilde{e})^T} \\ \frac{\partial \Lambda(x_{k-1}(\tilde{e}), \tilde{e}, \alpha_{k-1})}{\partial \tilde{e}} \\ \frac{\partial \Lambda(x_{k-1}(\tilde{e}), \tilde{e}, \alpha_{k-1})}{\partial \alpha_{k-1}} \end{bmatrix} \]

Since \(x_k \in C_+^{n}\) and \(\gamma \in C_+^{n}\) we have

\[ \frac{\partial \Lambda(x_{k-1}(\tilde{e}), \tilde{e}, \alpha_{k-1})}{\partial \alpha_{k-1}^T} = \]
\[
\tilde{\gamma}(1-\varepsilon)\alpha_{K-1}^T \begin{bmatrix}
c_{d1}x_{11}(\varepsilon)^{d1} & c_{d1}x_{12}(\varepsilon)^{d1} & \cdots & c_{d1}x_{1(k-1)1}(\varepsilon)^{d1} \\
c_{d2}x_{21}(\varepsilon)^{d2} & c_{d2}x_{22}(\varepsilon)^{d2} & \cdots & c_{d2}x_{2(k-1)1}(\varepsilon)^{d2} \\
\vdots & \vdots & \ddots & \vdots \\
c_{dk-1}x_{(k-1)1}(\varepsilon)^{dk-1} & \cdots & c_{dk-1}x_{(k-1)(k-2)1}(\varepsilon)^{dk-1} & c_{dk-1}x_{(k-1)(k-1)1}(\varepsilon)^{dk-1}
\end{bmatrix} = 0
\]

Therefore \(\text{Re}\{\text{rank} \left(J_{\alpha(x_{k-1}f,\varepsilon,\alpha_{K-1})}\right)\} = 2k-2\), \(\forall \alpha \in C^m_v\), and \(\forall \varepsilon \in [0,1-\delta\eta]\).

Hence, \(\Lambda\) is transversal to zero and by the parameterized Sard’s theorem so is \(\Lambda_{\alpha\beta}\).

Now, let \(\Lambda_{\alpha\beta}^{-1}(0):= \left\{(x_{k-1}f,\varepsilon) \in C^m_v \times [0,1) \mid \Lambda_{\alpha\beta}(x_{k-1}f,\varepsilon) = 0\right\}\). By the transversality theorem, either \(\Lambda_{\alpha\beta}^{-1}(0) = \emptyset\) or \(\Lambda_{\alpha\beta}^{-1}(0)\) consists of no singular solutions \(\forall \varepsilon \in [0,1)\) and hence \(\forall \varepsilon \in [0,1-\delta\eta]\). If \(\Lambda_{\alpha\beta}^{-1}(0) = \emptyset\) then \(\Lambda_{\alpha\beta}(x_{k-1}f,\varepsilon) = 0\) has no solutions and this is obviously true for \(\tilde{H}_{\alpha\beta}(x_{k}(\tilde{e}),\tilde{e}) = 0\), which implies that \(H_{\alpha\beta}(x_{k}(\varepsilon),\varepsilon) = 0\) has no solutions at infinity.

If \(\Lambda_{\alpha\beta}^{-1}(0)\) consists of no singular solutions \(\forall \varepsilon \in [0,1-\delta\eta]\), then these must be geometrically isolated and by Bezout’s theorem cannot be more than \(2^{k-1}\) (including multiplicities) in number. In this case, \(\alpha_{k1}\) satisfy

\[
\alpha_{k1} = \frac{\varepsilon}{\tilde{\gamma}(1-\varepsilon)} \left[0_{1 \times (k-1)} \ 1 \right] F_k(x_{k}(\varepsilon)).
\]

Now, define \(S_{\alpha_{k1}} := \left\{\frac{\varepsilon}{\tilde{\gamma}(1-\varepsilon)} \left[0_{1 \times (k-1)} \ 1 \right] F_k(x_{k}(\varepsilon)) \mid c \in C, k \in [1,m]\right\}\).

Since \(S_{\alpha_{k1}}\) consists of a finite number of elements, it can be indexed and therefore is countable. Hence \(S_{\alpha_{k1}}\) has measure zero (see definition A.2.14 in appendix A). It is obvious that \(\tilde{H}_{\alpha\beta}(x_{k}(\tilde{e}),\tilde{e}) = 0\) cannot have solutions \(\forall \alpha_{k1} \not\in S_{\alpha_{k1}}\). Without loss of generality, \(S_{\alpha\beta}\) can extend to include the cases of \(\alpha_{k1} \in S_{\alpha_{k1}}\) and still be of measure zero. Since the above results hold \(\forall k \in [1,m]\),

\[
\forall (\varepsilon, \alpha, \beta) \in [0,1-\delta\eta] \times \left\{\left\{C_v^m \times C_v^m\right\} - S_{\alpha\beta}\right\}, \tilde{H}_{\alpha\beta}(x_{k}(\tilde{e}),\tilde{e}) = 0
\]

cannot have solutions. Hence, \(\tilde{H}_{\alpha\beta}(x_{k}(\tilde{e}),\tilde{e}) = 0\) cannot have nonzero solutions and therefore, the assumption that \(\forall (\varepsilon, \alpha, \beta) \in [0,1-\delta\eta] \times \left\{\left\{C_v^m \times C_v^m\right\} - S_{\alpha\beta}\right\}, H(x(\varepsilon),\varepsilon) = 0\) has at least one solution at infinity in \(\Theta_{\alpha\beta}\) is false. Hence, \(\hat{\Theta}_{\alpha\beta} \neq \emptyset\), \(\hat{\Theta}_{\alpha\beta} \neq \emptyset\) where, \(\hat{\Theta}_{\alpha\beta} := \Theta_{\alpha\beta} - \left\{C_v^m \times 0\right\}\).
Now as $\delta \eta \to 0$, $\tilde{H}_{\alpha \beta}(x, \varepsilon) \to \tilde{F}(x)$.

Furthermore, at $\varepsilon = 1$, $\tilde{H}_{\alpha \beta}(x, \varepsilon) = 0 \iff \varepsilon \tilde{F}(x) = 0 \iff F(x) = 0$. From the last equivalence it is apparent that the solutions of $\tilde{H}_{\alpha \beta}(x, \varepsilon) = 0$ are the same as the solutions of $\tilde{F}(x) = 0$. Therefore, $H_{\alpha \beta}(x, \varepsilon) = 0$ has solutions at infinity at $\varepsilon = 1$ if and only if $F(x) = 0$ has solutions at infinity. But by assumption $F(x) = 0$ has no solutions at infinity. Hence, the solution set $S_x$ of $F(x) = 0$ has no solutions at infinity and therefore $\hat{S}_x \neq \emptyset$. Note also, that $\tilde{H}_{\alpha \beta}^{-1}(0) \neq \emptyset$, since it can be assumed that $\text{int}\{H_{\alpha \beta}^{-1}(0)\} = \hat{\Theta}_{\alpha \beta}$. Finally, it is obvious that $\tilde{H}_{\alpha \beta}^{-1}(0) \neq \emptyset$, since $\tilde{H}_{\alpha \beta}^{-1}(0) = H_{\alpha \beta}^{-1}(0) \cup S_x$. So $\tilde{H}_{\alpha \beta}^{-1}(0)$ contains smooth bounded homotopy paths with the problems (ii)-(v) not holding.

Because $H_{\alpha \beta}(x,0) = 0$ has $d$ different geometrically isolated solutions (counting multiplicities) $\forall(\alpha, \beta) \in \left\{\left\{C_{\alpha'}^m \times C_{\beta'}^m\right\} - S_{\alpha \beta}\right\}$, there are $d$ paths emanating from $C_{\alpha'}^m \times \{0\}$. Now by Bezout's theorem (see appendix A), the total number of geometrically isolated solutions of $F(x) = 0$ is no more than $d$: i.e. $r \leq d$. Since $F(x) = 0$ has no solutions at infinity, each geometrically isolated solution $\hat{x} \in \hat{S}_x$ with multiplicity $m_{\hat{x}}$ is determined as a solution of $H_{\alpha \beta}(x,1) = 0$. By definition of multiplicity of a geometrically isolated solution (see appendix A) the perturbation with respect to $\varepsilon$, $H_{\alpha \beta}(x,1 - \delta \varepsilon) = 0$ with $\delta \varepsilon \in (0,1)$ being arbitrarily small, determines $m_{\hat{x}}$ nonsingular geometrically isolated solutions in $H_{\alpha \beta}^{-1}(0)$, arbitrarily close to $\hat{x}$. Therefore each geometrically isolated solution of $H_{\alpha \beta}(x,1) = 0$ is connected to a number of paths from $H_{\alpha \beta}^{-1}(0)$ equal to its multiplicity.

From the above, it follows that there are $r$ paths from $H_{\alpha \beta}^{-1}(0)$ converging to the $r$ geometrically isolated solutions (counting multiplicities) of $F(x) = 0$ in $C_{\alpha'}^m \times \{1\}$. Now the number of non geometrically isolated solutions of $F(x) = 0$ can in general be more than $d - r$. Also, when the solution is non geometrically isolated

---

*For $\text{int}\{\}$, see definition A.2.7 in appendix A.*
then it is singular by definition A.3.7 (see appendix A) and its multiplicity is more than 1. Since these solutions are not at infinity by assumption, it follows by the continuity of $H_{\alpha\beta}$ and from $H_{\alpha\beta}^{-1}(0)$ being bounded that there will be at least two paths converging to each such solution. In other words there will be at least two solutions from $d-r$ solutions of the easy problem connected to a non geometrically isolated solution of $F(x) = 0$. Hence the total number of such paths is $\sigma < d - r$, $\sigma \in \mathbb{Z}_+$. 

Hence, $H_{\alpha\beta}^{-1}(0)$ contains $r \leq d$ smooth homotopy paths connecting $r$ geometrically isolated solutions of $F_0(x) = 0$, with the $r$ geometrically isolated solutions (counting multiplicities) of $F(x) = 0$ in 1-1 relationship, and $d-r$ smooth homotopy paths connecting $d-r$ geometrically isolated solutions of $F_0(x) = 0$, with $\sigma < d - r$, $\sigma \in \mathbb{Z}_+$ non geometrically isolated solutions of $F(x) = 0$ in 1-1 relationship. For all these particular paths problems (ii)-(v) do not apply. Note that the 1-1 relationship is immediate from problems (ii)-(v) not occurring.

Problems (i) and (vi) are ruled out as a consequence of the assumption of $H_{\alpha\beta}$ being $C^2$ and complex analytic in $x$, and $F(x) = 0$ being an algebraic polynomial system. This is basically a consequence of the Cauchy-Riemann equations (see (A.3.5), (A.3.6) in appendix A) for $H_{\alpha\beta}$. The proof of this particular result is exactly as in theorem 2.1, and the result is independent of the different structures of the easy problem in homotopy equations (2.7), (2.8) and (2.29), (2.30).

Finally note that the above results hold: $\forall (\alpha, \beta) \in \left\{ (C^n \times C^n) - S_{\alpha\beta} \right\}$ fixed, with $S_{\alpha\beta} \subset C^n \times C^n$ having measure zero, or in other words with probability-1. Hence the proof is complete.

Because theorem 2.2 guarantees the deformation of $r$ solutions of the easy problem $F_0(x) \equiv F_0(x, \alpha, \beta) = 0$ to all geometrically isolated solutions of the polynomial system (2.2) with probability one, this theorem is referred as the probability-1 polynomial homotopy theorem. The results of theorem 2.2 can be more specialized when $F(x) = 0$ has the additional property of a finite number of solutions assuming that all of them are geometrically isolated. The respective results are summarized as lemma 2.1 below.
Lemma 2.1:

Suppose that \( F(x) = 0 \), \( x \in \mathbb{C}^m \), \( m \in \mathbb{Z}_+ \) is an algebraic polynomial system of total degree \( d \) that has only a finite number of geometrically isolated solutions and no solutions at infinity. Let \( H_{\alpha \beta} \) be a homotopy mapping as in theorem 2.2 with the same properties. Then there is a set \( S_{\alpha \beta} \subset \mathbb{C}^m \times \mathbb{C}^m \) of measure zero such that

\[
\forall (\alpha, \beta) \in \left\{ \left( \mathbb{C}^m \times \mathbb{C}^m \right) - S_{\alpha \beta} \right\}, \quad \tilde{H}^{-1}_{\alpha \beta}(0) \text{ contains exactly } d \text{ smooth homotopy paths (curves) connecting } d \text{ geometrically isolated solutions of } F_0(x) = 0, \text{ with the } d \text{ solutions (counting multiplicities) of } F(x) = 0 \text{ in 1-1 relationship. For all these particular paths problems (i)-(vi) do not apply.}
\]

Proof.

Obviously this is a special case of theorem 2.2, taking into account that \( F(x) = 0 \) has exactly \( d \) solutions counting multiplicities, according to Bezout’s theorem (see appendix A). It is worth mentioning that for the present case the property of no turning back points in \( \tilde{H}^{-1}_{\alpha \beta}(0) \), apart from the method used in theorem 2.2, can also be proven using results from topological degree theory (see definition A.4.1 of the Brouwer degree in appendix A and [110], [119]) for the mapping \( H_{\alpha \beta} \) and the homotopy invariance theorem A.4.1 (see appendix A and [131], [110]). This proof is given next.

Initially recall, from the proof of theorem 2.2, that \( \tilde{H}^{-1}_{\alpha \beta}(0) \neq \emptyset \). Obviously \( H^{-1}_{\alpha \beta}(0) \) by its definition is open too (see definition A.2.5 in appendix A). Let the set in equation (2.33), be defined as

\[
\Psi_{\alpha \beta} = \left\{ \left[ \begin{array}{c} \text{Re}(y) \\ \text{Im}(y) \end{array} \right] : y \in \mathbb{C}^m, \left[ \begin{array}{c} \text{Re}(y) \\ \text{Im}(y) \end{array} \right] < p \\ + \max_{\varepsilon \in U} \left[ \begin{array}{c} \text{Re}(x) \\ \text{Im}(x) \end{array} \right] : (x, \varepsilon) \in \tilde{H}^{-1}_{\alpha \beta}(0), p \in \mathbb{R}_+^m \right\}
\]

(2.33)

Clearly it follows from (2.33) that \( \Psi_{\alpha \beta} \) is open and \( \overline{\Psi}_{\alpha \beta} \subset \mathbb{R}_+^m \). The closure \( \overline{\Psi}_{\alpha \beta} \) and boundary \( \partial \Psi_{\alpha \beta} \) of \( \Psi_{\alpha \beta} \) are then given by (2.34) and (2.35), shown below:
\[\psi_{\alpha\beta} = \begin{bmatrix} \text{Re}(y) \\ \text{Im}(y) \end{bmatrix} y \in \mathbb{C}^m, \begin{bmatrix} \text{Re}(y) \\ \text{Im}(y) \end{bmatrix} \leq p \]

\[+ \max_{\varepsilon \in \mathbb{U}} \left\{ \begin{bmatrix} \text{Re}(x) \\ \text{Im}(x) \end{bmatrix} \left( x, \varepsilon \right) \in \mathbb{H}^{-1}_{\alpha\beta}(0), p \in \mathbb{R}^{2m}_+ \right\} \] (2.34)

\[\psi_{\alpha\beta} = \begin{bmatrix} \text{Re}(y) \\ \text{Im}(y) \end{bmatrix} y \in \mathbb{C}^m, \begin{bmatrix} \text{Re}(y) \\ \text{Im}(y) \end{bmatrix} = p \]

\[+ \max_{\varepsilon \in \mathbb{U}} \left\{ \begin{bmatrix} \text{Re}(x) \\ \text{Im}(x) \end{bmatrix} \left( x, \varepsilon \right) \in \mathbb{H}^{-1}_{\alpha\beta}(0), p \in \mathbb{R}^{2m}_+ \right\} \] (2.35)

Now consider the homotopy mapping

\[P_{\alpha\beta} : \psi_{\alpha\beta} \times \mathbb{U} \to \mathbb{R}^{2m} \] (2.36)

with homotopy equation (2.22).

Under (2.20)-(2.22) the following are obviously true.

\[(x_r(\varepsilon), \varepsilon) \in \psi_{\alpha\beta} \times \mathbb{U} \Leftrightarrow (x(\varepsilon), \varepsilon) \in \mathbb{H}^{-1}_{\alpha\beta}(0)\]

\[\forall (x(\varepsilon), \varepsilon) \notin \mathbb{H}^{-1}_{\alpha\beta}(0) \Rightarrow H_{\alpha\beta}(x(\varepsilon), \varepsilon) \neq 0 \]

\[H_{\alpha\beta}(x(\varepsilon), \varepsilon) \neq 0 \Leftrightarrow \forall (x(\varepsilon), \varepsilon) \notin \mathbb{H}^{-1}_{\alpha\beta}(0)\]

\[H_{\alpha\beta}(x(\varepsilon), \varepsilon) \neq 0 \Leftrightarrow P_{\alpha\beta}(x_r(\varepsilon), \varepsilon) \neq 0 \] (2.37)

Hence \(P_{\alpha\beta}(x_r(\varepsilon), \varepsilon) \neq 0\), \(\forall (x_r(\varepsilon), \varepsilon) \in \psi_{\alpha} \times \mathbb{U}\), and the homotopy invariance theorem A.4.1 in appendix A, provided that \(\text{deg}(P_{\alpha\beta}(x_r(\varepsilon), \varepsilon), \psi_{\alpha\beta}, 0) = \text{const}\), \(\forall \varepsilon \in \mathbb{U}\). Therefore \(\text{deg}(P_{\alpha\beta}(x_r(\varepsilon), \varepsilon), \psi_{\alpha\beta}, 0) = \text{deg}(P_{\alpha\beta}(x_r(0), 0), \psi_{\alpha\beta}, 0)\). By definition,

\[\text{deg}(P_{\alpha\beta}(x_r(\varepsilon), \varepsilon), \psi_{\alpha\beta}, 0) = \sum_{i=1}^{d} \text{sign} \left( \det \left( \frac{\partial P_{\alpha\beta}(x_r(\varepsilon), \varepsilon)}{\partial x_r(\varepsilon)^T} \right) \right|_{x_r(\varepsilon) = x_{\varepsilon i}} \right) \] (2.38)

where, \(x_{\varepsilon i} \in S_{x_{\varepsilon}}\) and \(S_{x_{\varepsilon}} := \left\{ x_{\varepsilon} \in \mathbb{R}^{2m} | x_{\varepsilon} := \begin{bmatrix} \text{Re}(x(\varepsilon)) \\ \text{Im}(x(\varepsilon)) \end{bmatrix} \in \mathbb{H}_{\alpha\beta}(x(\varepsilon), \varepsilon) = 0 \right\} \).
Since \( \text{sign}\left( \det \left( \frac{\partial P_{\alpha\beta}(x_r(\varepsilon), \varepsilon)}{\partial [x_r(\varepsilon)^T]} \right) \right) = 1 \) \( \forall \varepsilon \in [0,1) \) from the proof of theorem 2.1 (also true for theorem 2.2), it follows that \( \text{deg}\left(P_{\alpha\beta}(x_r(\varepsilon), \varepsilon), \Psi_{\alpha\beta}, 0 \right) \geq 1 \). But at \( \varepsilon = 0 \) \( (2.38) \) becomes
\[
\text{deg}\left(P_{\alpha\beta}(x_r(0), 0), \Psi_{\alpha\beta}, 0 \right) = \sum_{i=1}^{d} \text{sign}\left( \det \left[ \frac{\partial P_{\alpha\beta}(x_r(0), 0)}{\partial x_r(0)^T} \right]_{x_r(0) = x_{0i}} \right)
\]
where \( x_{0i} \in S_{x_0} \) and \( S_{x_0} := \left\{ x_0 \in \mathbb{R}^{2m} \mid x_0 \cdot \begin{bmatrix} \text{Re}(x(0)) \\ \text{Im}(x(0)) \end{bmatrix}, H_{\alpha\beta}(x(0), \varepsilon) = 0 \right\} \).

Therefore from \( (2.38) \) and \( (2.39) \) it is obvious that \( \text{deg}\left(P_{\alpha\beta}(x_r(\varepsilon), \varepsilon), \Psi_{\alpha\beta}, 0 \right) \in [1, d] \).

By definition, the Brouwer degree can be considered as an integer number equal to the sum of integers, each of which defines a path orientation for a specific solution \( x_i(\varepsilon) \) in \( H_{\alpha\beta}^{-1}(0) \). Since this number happens to be a constant positive integer in the present case, the path’s orientation is preserved in \( H_{\alpha\beta}^{-1}(0) \). Hence paths do not turn back in \( H_{\alpha\beta}^{-1}(0) \), and this is generically true for every \( \forall (\alpha, \beta) \in \{C_w^n \times C_w^n \} - S_{\alpha\beta} \) fixed.

Once again, it is reminded that the specific structures of \( \tilde{H}^{-1}(0) \) and \( \tilde{H}_{\alpha\beta}^{-1}(0) \) are consequences of the transversality theorem and hold with probability 1. Hence, any result related to the structure of \( \tilde{H}^{-1}(0) \) or \( \tilde{H}_{\alpha\beta}^{-1}(0) \) is valid in a probabilistic sense with the same probability 1.

### 2.3.2 Tracking of Homotopy Paths

The major computational task in a probability-1 homotopy algorithm is to track the curves \( \mu \), which lead to the solutions, for specific \( (\alpha, \beta) \in \{C_w^n \times C_w^n \} - S_{\alpha\beta} \). One method of doing this continuously is to view the homotopic deformation process as a dynamic flow. For this purpose recall \( (2.16) \) from the proof of theorem 2.1 (also holds for theorem 2.2). Hence, the dynamical system below is well defined.
It is obvious that desirable homotopy paths $\mu$ can be viewed as trajectories of the initial value problem (IVP) (2.40), emanating from $\varepsilon = 0$ and terminating at $\varepsilon = 1$. For given easy problem data, the IVP initial condition $x(0)$ can be determined as the easy problem solution and with respect to this the IVP (2.40) has a unique solution. Hence, solutions to the original equation $F(x) = 0$, can be obtained by integrating (2.40) from $\varepsilon = 0$ to $\varepsilon = 1$. In other words the solutions obtained are given by,

$$
\frac{d(x(e))}{d\varepsilon} = -\left(\frac{\partial H_{\alpha\beta}(x(e),\varepsilon)}{\partial x^T(e)}\right)^{-1} \frac{\partial H_{\alpha\beta}(x(e),\varepsilon)}{\partial \varepsilon}
$$

(2.40)

$$
x(0) \in \mathcal{C}^{\eta}, \quad F_0(x(0)) = F_0(x(0),\alpha,\beta) = 0
$$

An alternative continuous tracking can be implemented by parameterizing (2.22) with the pseudo-arc length $s$ of $\mu$ [180], [179], [182]. With this technique the homotopy variable $\varepsilon$ is actually replaced with a pseudo-arc length parameter function $\varepsilon(s)$, with $s$ being the new homotopy parameter. Effectively the homotopy paths $\mu$ are parameterized with $s$. To do this, first, a relation between $s$ and $\varepsilon$ is defined by

$$
ds := \left(\frac{d[x_r(\varepsilon(s))]^2}{d\varepsilon(s)} + 1\right)^{\frac{1}{2}} d[\varepsilon(s)]
$$

(2.42)

The function $s = f(\varepsilon)$ is evaluated from (2.42) as

$$
s = \int_0^1 \left(\left(\frac{d[x_r(\varepsilon(s))]^2}{d[\varepsilon(s)]} + 1\right)^{\frac{1}{2}} d[\varepsilon(s)].
$$

(2.43)

Now since $x = x(\varepsilon(s)) = x(s)$ and $\varepsilon = \varepsilon(s)$, the respective homotopy mappings are parameterized identically in $s$ as $P_{\alpha\beta}(x_r(\varepsilon(s)),\varepsilon(s)) = 0$. Note that $P_{\alpha\beta}$ in (2.36) is
Chapter 2 Numerical Solution 45

$C^2$, transversal to zero (obviously true with probability-1 in theorems 2.1, 2.2, because of (2.27)), and the mapping

$$x_r : \mathbb{R}^+ \cup \{0\} \rightarrow \mathbb{R}^{2n}, \ P_{\alpha\beta}(x_r(\epsilon(s)), \epsilon(s)) = 0.$$  

$$x_r = x_r(\epsilon(s)) = \begin{bmatrix} \Re(x(\epsilon(s))) \\ \Im(x(\epsilon(s))) \end{bmatrix} = x_r(\epsilon(s)) = \begin{bmatrix} \Re(x(\epsilon(s))) \\ \Im(x(\epsilon(s))) \end{bmatrix}. $$

is $C^1$. Hence, with $(\alpha, \beta) \in \left\{ \left\{ C^m_\alpha \times C^m_\beta \right\} - S_{\alpha\beta} \right\}$ fixed

$$\frac{d}{ds} \left[ P_{\alpha\beta}(x_r(\epsilon(s)), \epsilon(s)) \right] = J P_{\alpha\beta}(x_r(\epsilon(s)), \epsilon(s)) \begin{bmatrix} \frac{d[x_r(\epsilon(s))]}{d[s]} \\ \frac{d[\epsilon(s)]}{d[s]} \end{bmatrix} = 0 \quad (2.44)$$

where, $P_{\alpha\beta}(x_r(\epsilon(s)), \epsilon(s))$ is the restriction of $P(x_r(\epsilon(s)), \epsilon(s), \alpha, \beta) = 0$ with

$(\alpha, \beta) \in \left\{ \left\{ C^m_\alpha \times C^m_\beta \right\} - S_{\alpha\beta} \right\}$ fixed and

$$J P_{\alpha\beta}(x_r(\epsilon(s)), \epsilon(s)) = \begin{bmatrix} \frac{\partial P_{\alpha\beta}[x_r(\epsilon(s)), \epsilon(s)]}{\partial [x_r(\epsilon(s))]^T} & \frac{\partial P_{\alpha\beta}[x_r(\epsilon(s)), \epsilon(s)]}{\partial [\epsilon(s)]} \end{bmatrix} \quad (2.45)$$

Note also that because of (2.42), we have

$$\left\| \frac{d[x_r(\epsilon(s))]}{ds} \right\|_2^2 = \left\| \frac{d[x_r(\epsilon(s))]}{d[\epsilon(s)]} \right\|_2^2 \left\| \frac{d[\epsilon(s)]}{d[s]} \right\|_2^2 + \left( \frac{d[\epsilon(s)]}{d[s]} \right)^2 \left\| \frac{d[x_r(\epsilon(s))]}{d[\epsilon(s)]} \right\|_2^2 + \left( \frac{d[\epsilon(s)]}{d[s]} \right)^2 \left\| \frac{d[x_r(\epsilon(s))]}{d[\epsilon(s)]} \right\|_2^2 \right\|_2^2 = 1. \quad (2.46)$$

Now from (2.42)-(2.46) and from $P_{\alpha\beta}(x_r(\epsilon(s)), \epsilon(s))$ the dynamical system below is well defined:
It is obvious that the parameterized homotopy paths $\mu$ in this case can be viewed as trajectories of the initial value problem (IVP) (2.47), emanating from $s_0 = 0$ and terminating at $s = s_1$, where $\varepsilon(s_1) = 1$. For a given initial condition $x(0)$ (i.e. easy problem solution), the IVP (2.47) has a unique solution.

From (2.44), it is apparent that

$$
\left[ \frac{d[x_r(\varepsilon(s))]}{ds} \right] \in \text{Ker} \left\{ J_{P_{\alpha\beta}}(x_r(\varepsilon(s)),\varepsilon(s)) \right\}.
$$

Furthermore it follows that the vector

$$
\left[ \frac{d[x_r(\varepsilon(s))]}{ds} \right],
$$

is a tangent to $J_{P_{\alpha\beta}}(x_r(\varepsilon(s)),\varepsilon(s))$ and therefore the matrix

$$
\left[ \frac{d[x_r(\varepsilon(s))]}{ds} \right] \frac{d[\varepsilon(s)]}{ds}
$$

is non-singular for all $s$, since

$$
\left[ \frac{d[x_r(\varepsilon(s))]}{ds} \right] \frac{d[\varepsilon(s)]}{ds}
$$

is orthogonal to $m$ linearly independent rows of $J_{P_{\alpha\beta}}(x_r(\varepsilon(s)),\varepsilon(s))$. Therefore it must be that

$$
\text{sign} \det \left[ \left[ \begin{array}{cc} J_{P_{\alpha\beta}}(x_r(\varepsilon(s)),\varepsilon(s)) \\
\frac{d[x_r(\varepsilon(s))]}{ds} & \frac{d[\varepsilon(s)]}{ds} \end{array} \right] \right] = \text{const} \neq 0, \forall s \in [0,s_1], \varepsilon(s_1) = 1.
$$

This is referred to as preserved path orientation. For convention, positive and negative signs refer to positive and negative path orientations respectively. At this point, note the difference with the tracking method using (2.40) where the path's
orientation by convention is always positive according to the Brouwer degree (see proof of theorem 2.2).

Hence, in contrast to (2.40) the derivatives \( \frac{d[x_r(\varepsilon(s))]}{ds} \) and \( \frac{d[\varepsilon(s)]}{ds} \) are evaluated from the numerical solution to

\[
\left[ \frac{d[x_r(\varepsilon(s))]}{ds} \right] \in \text{Ker}\left\{ J_{\alpha\beta}(x_r(\varepsilon(s)),\varepsilon(s)) \right\}.
\]

Among positive and negative sign solutions of the last problem the ones which satisfy

\[
\text{sign det} \left[ \begin{bmatrix} \frac{d[x_r(\varepsilon(s))]}{ds} & \frac{d[\varepsilon(s)]}{ds} \end{bmatrix} \right] = \text{sign det} \left[ \begin{bmatrix} \frac{d[x_r(0)]}{ds} & \frac{d[\varepsilon(0)]}{ds} \end{bmatrix} \right].
\]

\( \forall s \in [0,s_1], \varepsilon(s_1) = 1 \), are selected.

When \( H_{\alpha\beta} \) is complex analytic the function \( \varepsilon = f(s) \) evaluated from (2.43), is a monotonically increasing function with respect to \( \varepsilon \), since the homotopy paths do not have turning points. In the last case, as was shown in section 2.3.2,

\[
\text{sign det} \left[ \begin{bmatrix} \frac{\partial P_{\alpha\beta}(x_r(\varepsilon),\varepsilon)}{\partial x_r(\varepsilon)} \end{bmatrix} \right] = 1, \quad \forall \varepsilon \in [0,1), \quad \forall (\alpha,\beta) \in \left\{ \mathcal{C}_e^{\alpha} \times \mathcal{C}_e^{\beta} \right\} - S_{\alpha\beta} \text{ fixed.}
\]

Now note that

\[
\text{sign det} \left[ \begin{bmatrix} \frac{\partial P_{\alpha\beta}(x_r(\varepsilon(s)),\varepsilon(s))}{\partial x_r(\varepsilon(s))} \end{bmatrix} \right] = \text{sign det} \left[ \begin{bmatrix} \frac{\partial P_{\alpha\beta}(x_r(\varepsilon),\varepsilon)}{\partial x_r(\varepsilon)} \end{bmatrix} \right] = 1.
\]

Moreover it is obvious that \( s \geq 1 \).

Now suppose \( H_{\alpha\beta} \) is a strictly real valued homotopy mapping (hence, not complex analytic) and all conditions in theorems 2.1 and 2.2 are such that everything is restricted to real Euclidean spaces plus the additional condition of \( H_{\alpha\beta}^{-1}(0) \) being bounded. Then theorems 2.1 and 2.2 can still be used in real Euclidean spaces, with problems (i) and (vi) possible but not considered as problems any more. For this reason, pseudo-arc length parameterization is necessary in order to track paths which have turning points, as a monotonically increasing function of \( s = f(\varepsilon) \). However, the inverse function \( \varepsilon = f^{-1}(s) \) is not always monotonically increasing, and therefore
concerned, although closed path loops can now be defined in $H_{\alpha\beta}^{-1}(0)$ for $\forall \varepsilon \in U$, $\varepsilon(s)$ can increase or decrease as homotopy paths are traced. As far as problem (vi) is concerned, although closed path loops can now be defined in $H_{\alpha\beta}^{-1}(0)$ for $\forall \varepsilon \in U$, these will never be intersected by the homotopy paths, and hence they will be avoided. The pseudo-arc length parameterization, is very often used when real arithmetic is used for real valued homotopies having turning points [183].

Based on the previous analysis a real valued probability-1 homotopy theorem, is given below.

**Theorem 2.3:**
Consider the mappings $F: \mathbb{R}^m \rightarrow \mathbb{R}^m$ and $F_0: \mathbb{R}^m \rightarrow \mathbb{R}^m$, $m \in \mathbb{Z}_+$. Suppose that $F(x) = 0$, $x \in \mathbb{R}^m$ is a real algebraic polynomial system of total degree $d$ with its solution set $S_x$ not containing solutions at infinity and having at least one real solution, and $F_0(x) = \alpha \cdot x - \beta$, $(\alpha, \beta) \in \mathbb{R}_+^m \times \mathbb{R}_+^m$. Now, let $H_{\alpha\beta}: \mathbb{R}^m \times \mathbb{U} \rightarrow \mathbb{R}^m$ be a real homotopy mapping from $F_0$ to $F$, such that $H_{\alpha\beta}(x,0) = F_0(x)$ and $H_{\alpha\beta}(x,1) = F(x)$ $\forall (x, \alpha, \beta) \in \mathbb{R}_+^m \times \mathbb{R}_+^m \times \mathbb{R}_+^m$, and with the homotopy equation $H_{\alpha\beta}(x, \varepsilon) = \varepsilon F(x) - \gamma(1-\varepsilon)F_0(x), \gamma \in \mathbb{R}_+$. Furthermore, suppose that $H$ is $C^2$ and that $H_{\alpha\beta}^{-1}(0) \neq \emptyset$. $H_{\alpha\beta}^{-1}(0) = \{(x, \varepsilon) \in \mathbb{R}_+^m \times [0,1] | H_{\alpha\beta}(x, \varepsilon) = 0\}$. Then there is a set $S_{\alpha\beta} \subset \mathbb{R}_+^m \times \mathbb{R}_+^m$ of measure zero such that $\forall (\alpha, \beta) \in \{\left[\mathbb{R}_+^m \times \mathbb{R}_+^m\right] - S_{\alpha\beta}\}$ the set $H_{\alpha\beta}^{-1}(0) = \{(x, \varepsilon) \in \mathbb{R}_+^m \times \mathbb{U} | H_{\alpha\beta}(x, \varepsilon) = 0\}$ contains a unique smooth homotopy path (curve) which connects the zero solution $\beta + \alpha$, of $F_0(x) = 0$, with one solution of $F(x) = 0$.

**Proof.**

The proof is apparent from section 2.3.2 and from previous analysis in pseudo-arc-length parameterization in section 2.3.3. Also similar results can be found in [181]-[183].

When tracing the homotopy paths via a continuous tracking method, the question which may arise is whether or not the respective dynamic flow (2.40) has equilibrium points for $\varepsilon \in \mathbb{U}$. If this is the case, problems of bifurcation and end-
points for $\varepsilon \in [0,1)$ can take place. The above question is answered next with theorem 2.4.

**Theorem 2.4:**
Let the conditions of theorems 2.1 or 2.2 be satisfied. Then with probability-1, (2.40), can have equilibrium points $x_\varepsilon$ for $\varepsilon \in [0,1)$, if and only if $x(0) = x(1)$. In this case the equilibrium points define the line $x_\varepsilon = y_\varepsilon = x(\varepsilon) = x(0), \forall \varepsilon \in \overline{U}$.

**Proof.**
Let the homotopy equation according to theorems 2.1 and 2.2 be

$$H_{\alpha\beta}(x,\varepsilon) := \varepsilon F(x) - \gamma (1 - \varepsilon) F_0(x),$$

and define $J_\alpha := \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial [x(\varepsilon)^T]}$.

$$J_\alpha \varepsilon := \frac{\partial [H_{\alpha\beta}(x(\varepsilon),\varepsilon)]}{\partial \varepsilon}.$$ \(\Phi(x,\varepsilon) := J_\alpha^{-1} J_\varepsilon\). Under the validity of the conditions of theorems 2.1 and 2.2, $J_\alpha^{-1}$ exists $\forall \varepsilon \in [0,1)$ with probability-1. Now

$$J_\varepsilon = F(x) + \gamma F_0(x)$$ \hspace{1cm} (2.48)

The equilibrium points of (2.48) define the set

$M := \{(x,\varepsilon) \in \mathbb{R}^n \times \overline{U} \mid \Phi(x,\varepsilon) = 0\}$ [145]. Hence, $\forall \varepsilon \in [0,1)$,

$$\Phi(x,\varepsilon) = 0 \iff J_\alpha^{-1} J_\varepsilon = 0 \iff J_\alpha (J_\alpha^{-1} J_\varepsilon) = J_\alpha 0 \iff J_\varepsilon = 0.$$ \hspace{1cm} (2.49)

(2.48) in view of (2.49) becomes,

$$F(x) + \gamma F_0(x) = 0.$$ \hspace{1cm} (2.50)

Now $\forall \varepsilon \in [0,1)$, $H_{\alpha\beta}(x,\varepsilon) = 0$, and therefore

$$\varepsilon F(x) - \gamma (1 - \varepsilon) F_0(x) = 0.$$ \hspace{1cm} (2.51)

Equations (2.50) and (2.51) are two polynomial systems equal by identity. Hence it must be that either $\varepsilon = 1$ and $\varepsilon - 1 = 1 \Rightarrow \varepsilon = 2$, which is a contradiction, or $F(x) = F_0(x) = 0$, $\forall \varepsilon \in [0,1)$. In the latter case, since $F_0(x) = 0$ for $x = x(0)$ and $F(x) = 0$ for $x = x(1)$, it follows that $x = x(0) = x(1)$. Hence, when the easy problem solution is the original problem solution, (2.40) can have equilibrium points and these define the line $x_\varepsilon = y_\varepsilon = x(\varepsilon) = x(0), \forall \varepsilon \in \overline{U}$. 

\[\blacksquare\]
Instead of using an integration method for tracking $\mu$, another way is to discretise $\mu$ in a finite number of points and to use iterative local convergence methods for the computation of each such point [131], [182]. Thus, for specific $(\alpha, \beta) \in \left( C^\infty \times C^\infty \right) - S_{\alpha\beta}$, starting with a known initial point $(x(0),0)$ as the solution of $H_{\alpha\beta}(x(0),0) = 0$, evaluating a point $(x(\bar{e}), \bar{e})$ by solving $H_{\alpha\beta}(x(\bar{e}), \bar{e}) = 0$, then fixing a step $\delta e > 0$ and solving $H_{\alpha\beta}(x(\bar{e} + \delta e), \bar{e} + \delta e) = 0$ for $x(\bar{e} + \delta e)$ (using a locally convergent iterative method, with initial condition the solution $x(\bar{e})$ obtained in the previous step), each step of the overall tracking procedure of $\mu$ is evaluated as $e$ is increased monotonically from 0 to 1. This approach is called a discrete tracking method.

A major problem that may appear with discrete tracking methods is that, highly nonlinear structures on $H_{\alpha\beta}(x(0),0) = 0$ result in very small domains of attraction when applying locally convergent methods for tracking each point of $\mu$. This implies a very small increment step for the homotopy variable. Hence, the computational effort of tracking $\mu$ can be much bigger than when using an integration scheme with continuous tracking. Very small increment steps for discrete tracking methods, is the case for stiff IVPs. Note that in many cases iterative Newton type methods are unable to track the homotopy curve $\mu$ since their local iterations diverge to infinity overflowing the numerical process, or trapping into closed loops or orbits with infinite arc length (e.g. fractals, strange attractors [138]).

Finally, a combination of both discrete and continuous tracking methods can prove to be useful in practice. In such a combination, the IVP numerical routine has error corrector at each step, a local convergence method (e.g. Newton-Raphson) for the numerical solution of $H_{\alpha\beta}(x(0),0) = 0$. The advantage is that the error at each integration step of the IVP routine is explicitly known as the error to the solution of $H_{\alpha\beta}(x(0),0) = 0$, which $\forall e \in \bar{U}$ can be reduced to a desirable accuracy by solving $H_{\alpha\beta}(x(0),0) = 0$ with a local convergence method. Note that $\forall e \in \bar{U}$, the initial condition of the local convergence method is the one computed by the IVP routine and there is a big chance of being within the domain of attraction of the local convergence solver. Similarly, discrete tracking methods can be applied for pseudo-arc parameterized homotopies [182], [183].
2.3.3 Further Analysis and Discussions

From section 2.3.1, it is apparent that theorem 2.1 can be used in cases for which only one geometrically isolated solution of \( F(x) = 0 \) is required, while theorem 2.2 can be used in cases where the complete set of geometrically isolated solutions of \( F(x) = 0 \) is required.

Theorems 2.1 and 2.2 have been synthesized and are valid in Euclidean complex spaces in general. Hence computations will take place in complex arithmetic. Even when the data of \( F(x) = 0 \) are real matrices, complex arithmetic is guaranteed with \( y \in \mathbb{C} - \mathbb{R} \). The mathematical superiority of complex Euclidean spaces, is apparent in the proofs of theorems 2.1 and 2.2 in proving and guaranteeing the property of no turning points of homotopy paths. Also, the analysis in complex spaces is necessary for computing complex solutions. A simple way to appreciate this is to restrict theorems 2.1 and 2.2 to real arithmetic and to try to compute the solution of the scalar equation \( x^2 + x + 1 = 0 \). Clearly, for this example any algorithm based on theorems 2.1 and 2.2 and restricted to real arithmetic will fail, since there are only complex solutions to the problem. Moreover, in some cases during the deformation process of a particular problem via a homotopy mapping, one solution may be real for some intervals of \( \varepsilon \) and ‘explode’ to a complex solution in subsequent intervals, and vice versa. This can be observed even for scalar second order polynomials [56]. In contrast to their superiority in theoretical analysis, complex Euclidean spaces require complex arithmetic in computer implementations. For old generation computer machines, this increases the number of arithmetic operations and therefore increases the computational burden and accumulative round-off errors of algorithms. For modern computer machines, complex arithmetic is now a built in hardware feature in the computer’s processor and hence the computational burden is close to that of real arithmetic. Now, if the sufficient conditions of the previous theorems can be proven in real Euclidean spaces, then real arithmetic can be used with no problems and it is usually preferable. As will be shown in chapter 6, this can be the case when (2.1) is specialized to a classical algebraic Riccati equation with real data.

Theorems 2.1 and 2.2 also hold, if \( \varepsilon \) is replaced with any continuous and differentiable function of \( \varepsilon \), \( f(\varepsilon) \), such that \( \varepsilon \in [\varepsilon_1, \varepsilon_2] \) and \( f(\varepsilon_1) = 0 \), \( f(\varepsilon_2) = 1 \). Some examples are \( \varepsilon^l \) \( \forall l \in \mathbb{Z}_+ \) with \( \varepsilon \in \overline{U} \), \( \sin(\varepsilon) \) with \( \varepsilon \in \left[ 0, \frac{\pi}{2} \right] \), etc. Also another example [131] is \( 1 - e^{-\varepsilon} \) with \( \varepsilon \in [0, +\infty) \), in which case the deformation of the easy problem to the original problem will take infinite time, assuming \( \varepsilon \) is the time
variable. Hence one can make experiments with modified homotopy mappings, all of them converging with probability-1.

The assumption of \( F(x) = 0 \) having no solutions at infinity is a strong assumption which might not be satisfied in many problems. This condition however could be relaxed under certain conditions by modifying the homotopy mappings in a way that will provide \( H_{\alpha \beta}^{-1}(0) \) bounded with probability-1. This can be done via a homogeneous projective transformation. The most important feature of this transformation is that it provides analytical formulas showing the correspondence between the solutions of \( F(x) = 0 \) (including finite solutions and solutions at infinity) and the finite solutions of the modified homotopy equation at \( \varepsilon = 1 \). This will be presented in chapter 3. For the moment it is only said that the properties of this transformation are subject to \( F(x) = 0 \) having a finite number of solutions, and this is a deep result in general, since it is related to the problem of determining the geometry of solutions.

With a condition in theorems 2.1 or 2.3 not satisfied, any attempt at computing solutions will be heuristic. Nonetheless, in practice, even when \( F(x) = 0 \) has an infinite number of solutions, algorithms based on theorems 2.1 and 2.2 can still obtain some of the solutions of \( F(x) = 0 \). Such good numerical properties of respective algorithms have been recognized for some time, and very often attributed to the structure of artificial fixed-point and polynomial homotopies [182]. This will be apparent through numerical examples in later chapters of this thesis.

In view of theorems 2.1, 2.2 and 2.4, \( J_x := \frac{\partial H_{\alpha \beta}(x(\varepsilon), \varepsilon)}{\partial x(\varepsilon)^T} \) not only can be nonsingular with probability-1 but moreover, it cannot be zero with the same probability. A more popularized fact than the parameterized Sard's theorem (see appendix A) which explains this property is the independence principle [123]; it is almost impossible for a common solution to an algebraic system of two equations, chosen independently, in one unknown, to exist. In other words if the data of each equation are chosen independently from the other, then these equations have no common solution. This is true even for the case of a system of two linear first order equations, for which the data of each equation are chosen say by two independent persons. For the case of theorems 2.1, 2.2 and 2.4, if \( J_x = 0 \), there is the system of two equations \( J_x = 0 \) and \( H_{\alpha \beta}(x(\varepsilon), \varepsilon) = 0 \), in the unknown \( x(\varepsilon) \). According to the
independence principle above, if \( \alpha \) and \( \beta \) are chosen independently from the data of \( J_x \) in \( H_{\alpha\beta}(x(\varepsilon), \varepsilon) = 0 \) and independent of each other (e.g. at random), then the last system is almost impossible to have a common solution. This result carries more force when one takes into account that \( \beta \), enters in \( 0 \) but not in \( J_x \).

Similarly, the independence principle applies to theorem 2.3 as well. (2.27) is a consequence of the parameterized Sard’s theorem A.4.5 (see also theorems A.4.3, A.4.4 in appendix A) and for (2.27) to hold it is necessary that \( J_x \neq 0 \). Hence, the independence principle holds with probability-1 because it is a consequence of the parameterized Sard’s theorem A.4.5.

In the very rare cases where \( J_x = 0 \) and \( H_{\alpha\beta}(x(\varepsilon), \varepsilon) = 0 \) simultaneously, homotopy paths can overlap each other. Note however that such cases can easily be eliminated by small perturbations of the respective \( \alpha \) and \( \beta \). In terms of numerical computation, if some \( \alpha \) and \( \beta \) result in \( J_x = 0 \) in theory, it will be very difficult to have \( J_x = 0 \) in practice, because of finite computations. Hence, path crossing (overlapping) cases will almost never occur in practice even though this could be the case in theory. This argument can be presented mathematically and the resulting analysis is known as simplicial pivoting theory [56]. Because of this some researchers allow paths to cross each other, knowing a priori that the numerical process will not be affected from singularities of \( J_x \). This good numerical property is attributed to the general structure of line homotopies and is not general for any homotopy mapping. Also, it should be borne in mind that theorems 2.1-2.4 guarantee that such bad path behaviour is unlikely to occur with probability-1.

In contrast to the above, the guarantee of bad parameter sets \( S_{\alpha\beta} \subset C^n \times C^m \), such that \( (\alpha, \beta) \in S_{\alpha\beta} \) with \( S_{\alpha\beta} \) of measure zero, provided by theorems 2.1 and 2.2, does not definitely mean that a computer algorithm will not encounter them. This case is more evident for real valued homotopies as in theorem 2.3. This is because the set of computer numbers is finite. Hence, practical experience and tests with specific computer arithmetic units must be made. Note that with respect to the author’s knowledge, no such problems have been reported in the literature for homotopy methods tested on various computer machines.

If the easy problem results in a solution which is far away from the original problem solution, then this will normally produce large arc length for the homotopy paths \( \mu \). Also, depending on the numerical data of the original equation this can also
produce a stiff IVP, and therefore additional care must be taken with the numerical integration routine. Situations like these can effectively produce large computation times. Also, when the initial conditions result in an easy problem solution close to the original, then homotopy paths \( \mu \) might have small arc lengths, the IVP might not be stiff and the computation times can be small. At this point it should be said that the above speculation is not true in general and it depends on the polynomial structure of \( F(x) \). Normally, the numerical solution of the IVP will generate paths that are approximations to the actual ones. How good these approximations are, depends on the accuracy of the numerical solution of the IVP.

One major disandvantage of the pseudo-arc length parameterized homotopies is that the end point of the IVP (2.47) is not known a priori, since \( \varepsilon = f^{-1}(s) \) is evaluated implicitly during the numerical solution of (2.47). Hence, the numerical integration is stopped when \( f^{-1}(s_1) = \varepsilon(s_1) \geq 1 \). In practice, it is more realistic to have \( f^{-1}(s_1) = \varepsilon(s_1) > 1 \). Therefore, a new numerical problem of recovering the solution at \( \varepsilon = 1 \) with an initial condition the solution at \( \varepsilon(s_1) > 1 \), should be started after the end of the IVP. This additional numerical problem is known as the homotopy end game, and usually local convergence solvers are used [156], [182], [183]. A problem arises when the last IVP step is big enough to end the path tracking with a pseudo-arc length much bigger than the one which corresponds to \( \varepsilon(s_1) = 1 \). In this case, there is the danger for the homotopy end game to have an initial condition out of the domain of attraction of the locally convergent equation solver that is being used.

When using complex arithmetic and (2.40) the bounds of the variable \( \varepsilon \) in (2.40) are known a prioriy and more importantly the end point corresponds accurately to the desired solution point. Hence, removing complex arithmetic and using pseudo-arc length parameterization, in order to reduce the computational effort arising from complex arithmetic, may have a negative effect. In the author's experience with the algorithms presented in this work, using complex arithmetic and (2.40) is faster and sometimes more robust and more accurate than using pseudo-arc length parameterizations and (2.47). This will be illustrated with numerical examples in later chapters and hence will contradict the habit of many researchers of using (2.47) instead of (2.40) [182], [183], [5]. To this end, note that there is no mathematical justification to say that (2.47) is better than (2.40) in terms of numerical implementation and therefore numerical experiments which test both (2.40) and
(2.47), should be made for each particular problem in order to make a fair assessment of which method performs best.

### 2.4 Numerical Solution of Equation (2.1)

In this section, the synthesis of probability-1 homotopy algorithms for the numerical solution of (2.1) will be presented. Specific results from matrix analysis and calculus which will be used in the sequel are provided in appendix B.

Consider the problem of numerically solving (2.1). Using the vec(·) operator and formulas (B.2.11), (B.2.14) in appendix B, (2.1) can be transformed into a vector equation as shown below:

\[
\text{vec}(f(X)) = \text{vec} \left( \sum_{i=1}^{\omega} A_i X B_i + \sum_{i=1}^{\xi} C_i X D_i X E_i + G \right) = 0
\]

\[
\Leftrightarrow \text{vec}(f(X)) = \text{vec} \left( \sum_{i=1}^{\omega} A_i X B_i \right) + \text{vec} \left( \sum_{i=1}^{\xi} C_i X D_i X E_i \right) + \text{vec}(G) = 0
\]

\[
\Leftrightarrow \text{vec}(f(X)) = \left[ \sum_{i=1}^{\omega} (B_i^T \otimes A_i) + \sum_{i=1}^{\xi} \left( E_i^T \otimes C_i \right) \left( I_p \otimes (X D_i) \right) \right] \text{vec}(X) + \text{vec}(G) = 0
\]

(2.52)

Define

\[
x := \text{vec}(X)
\]

(2.53)

\[
F(x) := \text{vec}(f(X)).
\]

(2.54)

Then (2.52) becomes

\[
F(x) = \left[ \sum_{i=1}^{\omega} (B_i^T \otimes A_i) + \sum_{i=1}^{\xi} \left( E_i^T \otimes C_i \right) \left( I_p \otimes (X D_i) \right) \right] x + \text{vec}(G) = 0
\]

(2.55)

It is apparent from (2.55) that matrix equation (2.1) can be written as a vector polynomial system (2.55) similar to (2.2) with \( m = np \), \( d = 2 \); i.e. (2.55) is in general a vector polynomial system with total degree 2. At this point note that (2.55) (hence (2.1)) can have in general at least \( 2^{np} \) solutions including multiplicities, according to Bezout’s theorem (see appendix A). Now, the probability-1 homotopy theory developed in the previous sections of this chapter can be directly applied to (2.55).

Solving (2.55) with respect to the unknown vector \( x \in \mathbb{C}^m \), the original matrix
unknown $X \in \mathbb{C}^{n \times p}$ of (2.1) is recovered via the $\text{vec}^{-1}(\cdot)$ operator as shown in (2.56) below

$$X = \text{vec}^{-1}(x) \quad (2.56)$$

Theorems 2.1-2.4 and lemma 2.1 will guarantee that the algorithms based on the numerical solution of the IVP (2.40) and (2.47), will converge with probability-1. All that is left to show are the formulas of partial derivatives and Jacobians that make up (2.40) and (2.47). For (2.55), these formulas are derived straightforwardly using the formulas (B.3.3)-(B.3.9) in appendix B.

For the IVP (2.40), the following formulas hold:

$$\frac{\partial H_{\alpha \beta}(x(\varepsilon), \varepsilon)}{\partial x^T(\varepsilon)} = \frac{d[F(x(\varepsilon))]}{d[x^T(\varepsilon)]} - \gamma(1-\varepsilon)\frac{d[F_0(x(\varepsilon), \alpha, \beta)]}{d[x^T(\varepsilon)]} \quad (2.57)$$

$$\frac{\partial H_{\alpha \beta}(x(\varepsilon), \varepsilon)}{\partial \varepsilon} = F(x(\varepsilon)) + \gamma F_0(x(\varepsilon), \alpha, \beta) \quad (2.58)$$

$$\frac{d[F(x(\varepsilon))]}{d[x^T(\varepsilon)]} = \frac{d}{d[x^T(\varepsilon)]}\left[\sum_{i=1}^{\omega}(B_i^T \otimes A_i) + \sum_{i=1}^{\xi}\left((E_i^T \otimes C_i)(I_p \otimes (X(\varepsilon)D_i))\right)x(\varepsilon) + \text{vec}(G)\right]$$

$$= \frac{d}{d[x^T(\varepsilon)]}\left[\sum_{i=1}^{\omega}(B_i^T \otimes A_i) + \sum_{i=1}^{\xi}\left((E_i^T \otimes C_i)(I_p \otimes (X(\varepsilon)D_i))\right)x(\varepsilon)\right]$$

$$= \frac{d}{d[x^T(\varepsilon)]}\left[\sum_{i=1}^{\omega}(B_i^T \otimes A_i)x(\varepsilon)\right] + \frac{d}{d[x^T(\varepsilon)]}\left[\sum_{i=1}^{\xi}\left((E_i^T \otimes C_i)(I_p \otimes (X(\varepsilon)D_i))\right)x(\varepsilon)\right]$$

$$= \sum_{i=1}^{\omega}\left(\frac{d[(B_i^T \otimes A_i)x(\varepsilon)]}{d[x^T(\varepsilon)]}\right) + \sum_{i=1}^{\xi}\left(\frac{d[(E_i^T \otimes C_i)(I_p \otimes (X(\varepsilon)D_i))x(\varepsilon)]}{d[x^T(\varepsilon)]}\right)$$

$$= \sum_{i=1}^{\omega}\left(\frac{d[(B_i^T \otimes A_i)x(\varepsilon)]}{d[x^T(\varepsilon)]}\right)$$

$$= \sum_{i=1}^{\omega}(B_i^T \otimes A_i) \quad (2.59)$$

$$\sum_{i=1}^{\omega}\left(\frac{d[(B_i^T \otimes A_i)x(\varepsilon)]}{d[x^T(\varepsilon)]}\right) = \sum_{i=1}^{\omega}(B_i^T \otimes A_i) \quad (2.60)$$
\[
\sum_{i=1}^{\xi} \left( \frac{d}{d\left(x^T(e)\right)} \left[ (E_i^T \otimes C_i) \left( I_p \otimes (X(e)D_i) \right) x(e) \right]\right) = \sum_{i=1}^{\xi} \left( \frac{d}{d\left(x^T(e)\right)} \left[ \left( I_p \otimes (X(e)D_i) \right) \frac{d[x(e)]}{d[x_{II}(e)]} \right] \right) \ldots \frac{d}{d\left[x_{m1}(e)\right]} \left( (X(e)D_i) + \left( X(e)D_i \right)^T \otimes I_n \right) \right) (2.61)
\]

Also for (2.61), subject to (B.3.10) in appendix B, a more compact alternative formula is (2.62), shown below.

\[
\sum_{i=1}^{\xi} \left( \frac{d}{d\left(x^T(e)\right)} \left[ (E_i^T \otimes C_i) \left( I_p \otimes (X(e)D_i) \right) x(e) \right]\right) = \sum_{i=1}^{\xi} \left( \frac{d}{d\left(x^T(e)\right)} \left[ \left( I_p \otimes (X(e)D_i) \right) \frac{d[x(e)]}{d[x_{II}(e)]} \right] \right) \ldots \frac{d}{d\left[x_{m1}(e)\right]} \left( (X(e)D_i) + \left( X(e)D_i \right)^T \otimes I_n \right) \right) (2.62)
\]

Now, for \( F_0(x(e),\alpha,\beta) \) we have two formulas subject to (2.8) (under the homotopy mapping (2.7)) and to (2.30) (under the homotopy mapping (2.29)). These are presented next as formulas (2.63) and (2.64) respectively.

\[
F_0(x(e),\alpha,\beta) = \alpha \circ x(e) - \beta
\]

\[
F_0(x(e),\alpha,\beta) = \alpha \circ \begin{pmatrix} \epsilon^2_{11} & \epsilon^2_{21} & \ldots & \epsilon^2_{m1} \end{pmatrix}^T - \beta = \alpha \circ x(e) \circ x(e)
\]

Hence, for \( \frac{d[F_0(x(e),\alpha,\beta)]}{d[x^T(e)]} \) with respect to (2.63) and (2.64), it follows that

\[
\frac{d[F_0(x(e),\alpha,\beta)]}{d[x^T(e)]} = \frac{d[\alpha \circ x(e) - \beta]}{d[x^T(e)]} = \text{diag}(\alpha)
\]

\[
\frac{d[F_0(x(e),\alpha,\beta)]}{d[x^T(e)]} = \frac{d[\alpha \circ \begin{pmatrix} \epsilon^2_{11} & \epsilon^2_{21} & \ldots & \epsilon^2_{m1} \end{pmatrix}^T - \beta]}{d[x^T(e)]} = \frac{d[\alpha \circ x(e) \circ x(e) - \beta]}{d[x^T(e)]} = 2\text{diag}(\alpha \circ x(e)).
\]
From the above it is obvious that under (2.59)-(2.66), (2.57) and (2.58), the IVP (2.40) is well defined under the assumptions of the probability-1 theorems of section 2.3.

For the IVP (2.47), the following formulas hold according to section 2.3 (see proof of theorem 2.1).

\[
\begin{align*}
\frac{\partial_{x_{r}(\epsilon(s))} P_{\alpha\beta}(x_{r}(\epsilon(s)),\epsilon(s))}{\partial x_{r}(\epsilon(s))^T} &= 
\begin{bmatrix}
\text{Re} \left( \frac{\partial}{\partial x_{r}(\epsilon(s))^T} P_{\alpha\beta}(x_{r}(\epsilon(s)),\epsilon(s)) \right) \\
\text{Im} \left( \frac{\partial}{\partial x_{r}(\epsilon(s))^T} P_{\alpha\beta}(x_{r}(\epsilon(s)),\epsilon(s)) \right)
\end{bmatrix}
\end{align*}
\]

(2.67)

\[
\begin{align*}
\frac{\partial_{\epsilon(s)} P_{\alpha\beta}(x_{r}(\epsilon(s)),\epsilon(s))}{\partial \epsilon(s)} &= 
\begin{bmatrix}
\text{Re} \left( \frac{\partial}{\partial \epsilon(s)} P_{\alpha\beta}(x_{r}(\epsilon(s)),\epsilon(s)) \right) \\
\text{Im} \left( \frac{\partial}{\partial \epsilon(s)} P_{\alpha\beta}(x_{r}(\epsilon(s)),\epsilon(s)) \right)
\end{bmatrix}
\end{align*}
\]

(2.68)

Therefore, (2.67) and (2.68), under the substitution \( \epsilon \rightarrow \epsilon(s) \) in (2.57) and (2.58), are well defined and therefore (2.47) is well defined.

Hence, it is clear from the above that probability-1 homotopy algorithms can be implemented via the tracking of the homotopy paths methods of section 2.3.2.

Formulas (2.52)-(2.68) have analytical expressions. This is a big advantage, since implicit evaluation of derivative functions during a numerical process can in general cause numerical inaccuracies [53]. A drawback perhaps of the above formulation, is the creation of high dimensional problems and hence relatively large computation times. This is because of the use of Kronecker products. However, all the formulas have a nice structure and are amenable to parallel computer programming [30]. Also, for symmetric forms of (2.1), the symmetric parts can be removed and therefore the computational burden can be reduced.

Concluding, it should be said that all the results and relevant discussions which have taken place so far in this chapter apply to the case of (2.55) and hence (2.1). Finally, when (2.1) has some particular specialized forms, instead of using the vector function’s derivatives as in this section, it is helpful to use matrix Frechet derivatives directly. Note that (2.62) is a result of Frechet derivative under the vec(·) operator as is shown in appendix B. Further comments about this will be made in chapters 6 and 7.
2.5 Synopsis

This chapter has presented the mathematical analysis and synthesis of probability-1 homotopy algorithms for the numerical solution of the general quadratic algebraic matrix equation (2.1). The presentation is rather mathematical and for this reason, technical details and definitions are provided in appendices A and B for the reader unfamiliar with this subject. A rich bibliography is also given.

The majority of the analysis was focused on vector polynomial systems (2.2), since (2.1) can mathematically be viewed as a general second order vector polynomial system. More specifically, (2.1) is equivalent to

\[ f(x) := C + L(x) + Q(x) = 0, \]

where

\[ L(x) := Ax, \]
\[ Q(x) := \begin{bmatrix} x^T B_1 x \\ \vdots \\ x^T B_m x \end{bmatrix}. \]

Now, \( L(x) \) and \( Q(x) \) in (2.70) and (2.71) are known in the literature as the generalized linear and quadratic vector operators respectively. When (2.1) is a symmetric equation, then \( L(x) \) and \( Q(x) \) are the vectorizations of the well known generalized Lyapunov and Riccati operators respectively [99], [100].

The results on the homotopy methods hold generally for \( n \)-order vector polynomial systems. Therefore, \( n \)-order matrix polynomial equations can be solved similarly, as soon as they have been transformed into the equivalent vector equation. Hence, chapter 2 is characterized by the general applicability of the derived probability-1 homotopy methods for the numerical solution of general polynomial matrix equations.

The algorithms are able to compute one, a limited number or all (when the number is finite) solutions, real and/or complex. Note that computing all solutions, discriminates the method as rigorous rather than simply heuristic.

It is worth mentioning again that all the algorithms so far assume the strong condition of (2.55) and hence (2.1) having no solutions at infinity. This condition, however, will be relaxed in chapter 3.
3.1 Introduction

This chapter presents the mathematical formulation and solution to the problems of scaling and homogeneous projective transformations. The problem of scaling of the following general quadratic algebraic matrix equation (3.1), is considered first:

\[ f(X) = \sum_{i=1}^{\omega} A_i X B_i + \sum_{i=1}^{\xi} C_i X D_i X E_i + G = 0 \]  

(3.1)

where, \( \omega, \xi \in \mathbb{N}, \omega + \xi \neq 0, \ t, n, p, l \in \mathbb{Z}_+, \ A_i, C_i \in \mathbb{C}^{t \times n}, \ B_i, E_i \in \mathbb{C}^{p \times l}, \ D_i \in \mathbb{C}^{p \times p}, \ G \in \mathbb{C}^{t \times l}, \ 0 \in \mathbb{C}^{t \times l} \) are the constant matrix coefficients of the equation and \( X \in \mathbb{C}^{n \times p} \) is the equation’s unknown matrix. As in chapter 2, \( 0 \) denotes the zero matrix, vector or number of compatible dimensions. Scaling involves transforming the equation to prevent numerical problems in the computations. After the study of scaling, the concept of homogeneous projective transformations for (2.1) is addressed. This transformation is used to avoid solutions at infinity. Note that, (2.1) is the special case of (3.1) with \( t = n \) and \( l = p \). The difference is that (3.1) is equivalent to a system of \( tl \) scalar equations with \( np \) unknowns, whereas for (2.1), the number of resulting scalar equations and the number of unknowns are both equal to \( np \). For the latter case, results similar to the probability-1 homotopy results in chapter 2 are developed, with respect to scaling and the homogeneous projective transformation.

Scaling and the homogeneous projective transformation are independent of any numerical method and therefore they can in general be used along with any numerical method for the equation’s solution.
When the probability-1 homotopy algorithms of chapter 2 are used for the numerical solution of (2.1), scaling and homogeneous projective transformations can be useful. A major drawback of the probability-1 homotopy algorithms in chapter 2 occurs when (2.1) has solutions at infinity and the algorithms try to compute such solutions. In these cases, the computation requires the tracking of homotopy paths which tend to infinity (i.e. paths with infinite arc length) and therefore infinite computation time. Hence, a decision of when to stop the numerical process must be taken. This is really a hard decision to make, because a finite solution with an arbitrarily large norm might demand large computational time. Therefore, when no knowledge about the geometry of the equation’s solution is available, there is always the danger of stopping the algorithm’s numerical process on its way to computing a finite solution, by incorrectly assuming the solution to be at infinity.

One way to avoid this problem is to transform the original equation into an equation with no solutions at infinity. With such a transformation, the finite solutions of the transformed equation should correspond to the finite and infinite solutions (if any) of the original equation. Such a transformation which provides mathematical formulae for implementing the correspondence of solutions between the transformed and the original equation, under certain hypotheses, is the homogeneous projective transformation. For scalar polynomial systems, a homogeneous projective transformation is presented in [121]. The application of the results of [121] to (2.1) are original and presented for the first time in this chapter.

The purpose of scaling is to prevent arithmetic problems on a computer by transforming the coefficients and the unknown matrices of the original equation so that they do not have extreme values (either very small or very large). This kind of transformation differs from the homogeneous projective transformation. Scaling assigns predetermined values to the coefficients and the unknown matrices of the original equation. The homogeneous projective transformation changes the structure of the original equation into a homogeneous equation.

There is no rigorous mathematical theory to guide the scaling of equations (3.1) and (2.1), in their general form, and no proof of efficiency. Nevertheless, numerical experience shows that the proposed scaling affects the curvature and arc length of the homotopy paths associated with the original equation, when using the probability-1 homotopy algorithms of chapter 2. Hence, there is an effect on computation time. This will be shown via numerical examples in other chapters of this thesis. Usually, it is a good practice when solving a polynomial equation to scale the coefficients and the variables in such a way that the magnitudes of the equation’s coefficients are minimized. There are several approaches for doing this. The one followed in the present chapter is to equivalently transform (3.1) into a magnitude-type exponential vector equation. Then the problem of scaling is to minimize the sum of the squares of the coefficient exponents of the resulting vector equation. Thus, the problem of scaling becomes an optimization problem; more specifically, the minimization of the Frobenious norm of a matrix
function. The variables of this matrix function, are actually the scale factors for the scalar equations (provided from the exponential magnitude-type vector equation) and the scale factors for the respective equations’ unknowns.

Both scaling and homogeneous projective transformations contribute to smooth numerical operations and in many cases reduce computation times. However, the reader should bear in mind that scaling and homogeneous projective transformations are not always effective. For example, it is possible for the scaling and/or the homogeneous projective transformation to produce an ill conditioned numerical problem.

Finally, it should be said that scaling strategies for special cases of (2.1), such as the algebraic Riccati equation for continuous time systems, have been developed in the past. The positive effect of scaling for these special cases can be rigorously justified for the particular method used. In the case of algebraic Riccati equations, the scaling strategy is to change the equation’s data by a single scalar multiplier in such a way that the resulting Hamiltonian matrix [106] provides a stable numerical process. These special strategies are based on well known numerical linear algebra results [64], [184], [185] and can be found in [94] and [132]. An experimental illustration is outlined in [135].

The structure of this chapter will be as follows. Section 3.2 will present the mathematical formulation and solution to the problem of scaling (3.1). Next, the concept of homogeneous projective transformation, for the same equation, is introduced in section 3.3. The probability-1 homotopy theorems of chapter 2, adapted for the solution of the scaled and homogeneous projective transformed equation (2.1) will be stated and proved in section 3.4. Finally, conclusions are given in section 3.5. Throughout the chapter specialized mathematical terms and results are used. These are presented in appendices A and B.

3.2 Scaling

In order to define and then to solve the problem of scaling, a major part of this section is given to analysis and synthesis. Possible variations of the scaling problem are discussed and a simple scalar example is given.

3.2.1 Analysis

Consider (3.1) and let $U_X \in \mathbb{R}^{n \times p}$ and $U_C \in \mathbb{R}^{r \times l}$. (3.1) can be scaled with respect to its unknown and coefficient matrices as in equation (3.2) shown below.

$$
\bar{f}(\bar{X}) := E_{10}(U_C) \circ \left( \sum_{i=1}^{\omega} (A_i(E_{10}(U_X) \circ X)B_i) + \sum_{i=1}^{\xi} (C_i(E_{10}(U_X) \circ X)D_i(E_{10}(U_X) \circ X)E_i) + G \right) = 0
$$

(3.2)
where the matrix operator $E_{10}(\cdot)$ is defined in appendix B and $E_{10}(U_X)$, $E_{10}(U_C)$ are the scaling factors for the equation’s unknown and coefficient matrices respectively.

Equation (3.2) defines a different problem from (3.1), and it is referred to as the *scaled equation*. It is apparent from (3.2) that the original unknown $X \in \mathbb{C}^{n \times p}$ in (3.1) can be recovered from $\overline{X} \in \mathbb{C}^{n \times p}$ by

$$X = E_{10}(U_X) \circ \overline{X}. \quad (3.3)$$

Using the matrix operators $L_{10}(\cdot)$ and $S(\cdot)$ (see appendix B), (3.2) is transformed into the equation (3.4):

$$f(X) = E_{10}(U_C) \circ \sum_{i=1}^{\omega} [S(A_i) \circ E_{10}(L_{10}(A_i))][E_{10}(U_X) \circ \overline{X}][S(B_i) \circ E_{10}(L_{10}(B_i))]
\quad + E_{10}(U_C) \circ \sum_{i=1}^{\xi} [S(C_i) \circ E_{10}(L_{10}(C_i))][E_{10}(U_X) \circ \overline{X}][S(D_i) \circ E_{10}(L_{10}(D_i))]
\quad + \overline{X} = 0 \quad (3.4)$$

Because of the nature of operators $L_{10}(\cdot)$ and $S(\cdot)$, (3.4) is equivalent to $\text{vec}(3.2)$. Now, (3.4) can be viewed as a magnitude-type vector equation of $\text{vec}(3.2)$. To this end, note that (3.4) is not an actual magnitude equation of $\text{vec}(3.2)$, because $(3.3) \neq \text{vec}(3.2)$.

Using the $\text{vec}(\cdot)$ operator (see appendix B), (3.4) can be transformed into the vector equation

$$\overline{F}(\overline{x}) = \text{vec}(f(\overline{X})) \quad (3.5)$$

where

$$\overline{x} := \text{vec}(\overline{X}). \quad (3.6)$$

Moreover, let $U_{k \times v} \in \mathbb{Z}^{k \times v}_+$ be the unity matrix with dimensions $k \times v$, and define

$$u_X := \text{vec}(U_X) \quad (3.7)$$

$$u_C := \text{vec}(U_C). \quad (3.8)$$

Hence, in view of (3.4), (3.5) becomes equivalently

$$\overline{F}(\overline{x}) = \text{vec}\left[ E_{10}(U_C) \circ \sum_{i=1}^{\omega} [S(A_i) \circ E_{10}(L_{10}(A_i))][E_{10}(U_X) \circ \overline{X}][S(B_i) \circ E_{10}(L_{10}(B_i))] \right]
\quad + \text{vec}\left[ E_{10}(U_C) \circ \sum_{i=1}^{\xi} [S(C_i) \circ E_{10}(L_{10}(C_i))][E_{10}(U_X) \circ \overline{X}][S(D_i) \circ E_{10}(L_{10}(D_i))] \right]$$
Chapter 3 Scaling & Homogeneous Projective Transformations 64

\( (E_{10}(U_X) \circ \bar{X})(S(E_i) \circ E_{10}(L_{10}(E_i))) + \text{vec}(E_{10}(U_C) \circ G) = 0 \) (3.9)

(3.9) via (B.2.5) (see appendix B) is equivalent to

\[
\bar{F}(\bar{x}) = \sum_{i=1}^{n_0} L_i + \sum_{i=1}^{n_1} Q_i + E_{10}(u_C) \circ S(\text{vec}(G)) \circ \text{vec}(E_{10}(L_{10}(G))) = 0
\] (3.10)

where

\[
L_i := \text{vec}(E_{10}(U_C) \circ ((S(A_i) \circ E_{10}(L_{10}(A_i)))(E_{10}(U_X) \circ \bar{X})(S(B_i) \circ E_{10}(L_{10}(B_i)))))
\] (3.11)

\[
Q_i := \text{vec}(E_{10}(U_C) \circ ((S(C_i) \circ E_{10}(L_{10}(C_i)))(E_{10}(U_X) \circ \bar{X})(S(D_i) \circ E_{10}(L_{10}(D_i))))
\] + \begin{align*}
&+ (E_{10}(U_X) \circ \bar{X})(S(E_i) \circ E_{10}(L_{10}(E_i))))
\end{align*} (3.12)

Now, (3.11) via (B.2.1) and (B.2.17) (see appendix B) is equivalent to

\[
L_i = E_{10}(u_C) \circ \left( \left( (S(B_i) \circ E_{10}(L_{10}(B_i)))^T \otimes (S(A_i) \circ E_{10}(L_{10}(A_i))) \right)(E_{10}(u_X) \circ \bar{x}) \right)
\] (3.13)

where

\[
S_{B_i A_i} := S(B_i^T) \otimes S(A_i)
\] (3.14)

\[
L_{B_i A_i} := L_{10}(B_i^T) \oplus L_{10}(A_i)
\] (3.15)

With respect to (B.2.19), (B.2.15), (B.2.3) and (B.2.21) (see appendix B), (3.13) is equivalent to

\[
L_i = E_{10}(u_C) \circ \left( \left( U_d \times I(E_{10}(u_X))^T \circ S_{B_i A_i} \circ E_{10}(L_{B_i A_i}) \right) \bar{x} \right)
\]

\[
\Leftrightarrow L_i = E_{10}(u_C) \circ \left( \left( E_{10}(U_d \times I u_X^T) \circ S_{B_i A_i} \circ E_{10}(L_{B_i A_i}) \right) \bar{x} \right)
\]

\[
\Leftrightarrow L_i = E_{10}(u_C) \circ \left( \left( S_{B_i A_i} \circ E_{10}(U_d \times I u_X^T + L_{B_i A_i}) \right) \bar{x} \right)
\]

\[
\Leftrightarrow L_i = \left( S_{B_i A_i} \circ E_{10}(U_d \times I u_X^T + u_C U_l \times np + L_{B_i A_i}) \right) \bar{x}
\]

\[
\Leftrightarrow L_i = (S_{B_i A_i} \circ E_{10}(S L_i)) \bar{x}
\] (3.16)

where

\[
S_{L_i} := U_d \times I u_X^T + u_C U_l \times np + L_{B_i A_i}
\] (3.17)
Working similarly, (3.12) via (B.2.12) and (B.2.17) (see appendix B) is equivalent to

\[ Q_i = E_{10}(u_C) \circ \text{vec}\left( \left( S(E_i) \circ E_{10}(L_{10}(E_i)) \right)^T \otimes \left( S(C_i) \circ E_{10}(L_{10}(C_i)) \right) \right) \]

\[ \left[ \left( E_{10}(U_X) \otimes (E_{10}(U_X) \otimes \bar{X}) \right)^T \circ \left( S(D_i) \circ E_{10}(L_{10}(D_i)) \right) \right] \]

\[ \Leftarrow Q_i = E_{10}(u_C) \circ \left( \left( S(E_i^T) \otimes S(C_i) \right) \circ E_{10}(L_{10}(E_i^T) \otimes (L_{10}(C_i)) \right) \]

\[ \left( E_{10}(U_X) \otimes \bar{X} \right)^T \otimes (E_{10}(U_X) \otimes \bar{X}) \text{vec} \left( \left( S(D_i) \circ E_{10}(L_{10}(D_i)) \right) \right) \]

\[ \Leftarrow Q_i = E_{10}(u_C) \circ \left( S(E_i^T) \otimes S(C_i) \right) \circ E_{10}(L_{10}(E_i^T) \otimes (L_{10}(C_i)) \right) \]

\[ \left( E_{10}(U_X) \otimes \bar{X} \right)^T \otimes (E_{10}(U_X) \otimes \bar{X}) \text{vec} \left( \left( S(D_i) \circ E_{10}(L_{10}(D_i)) \right) \right) \]

where

\[ S_{E_iC_i} := S(E_i^T) \otimes S(C_i) \]  \hspace{1cm} (3.19)

\[ L_{E_iC_i} := L_{10}(E_i^T) \otimes L_{10}(C_i). \]  \hspace{1cm} (3.20)

With respect to (B.2.19) and (B.2.3) (see appendix B), (3.18) is equivalent to

\[ Q_i = E_{10}(u_C) \circ \left( S_{E_iC_i} \circ E_{10}(L_{E_iC_i}) \right) \]

\[ \left( \left( U_{np \times l^{vec}}(S(D_i)) \right)^T \circ \left( E_{10}(U_X^T \otimes U_X) \circ \left( \bar{X}^T \otimes \bar{X} \right) \right) \right) \text{vec} \left( E_{10}(L_{10}(D_i)) \right) \]

\[ \Leftarrow Q_i = E_{10}(u_C) \circ \left( S_{E_iC_i} \circ E_{10}(L_{E_iC_i}) \right) \]

\[ \left( E_{10}(U_X^T \otimes U_X) \circ \left( \bar{X}^T \otimes \bar{X} \right) \circ \left( U_{np \times l^{vec}}(S(D_i)) \right)^T \right) \text{vec} \left( E_{10}(L_{10}(D_i)) \right) \]

\[ \Leftarrow Q_i = E_{10}(u_C) \circ \left( S_{E_iC_i} \circ E_{10}(L_{E_iC_i}) \right) \]

\[ \left( E_{10}(U_X^T \otimes U_X) \circ \left( \bar{X}^T \otimes \bar{X} \right) \circ \text{vec} \left( E_{10}(L_{10}(D_i)) \right) \right) \]

\[ \text{vec} \left( E_{10}(L_{10}(D_i)) \right) \]

where

\[ S_{D_i} := U_{np \times l^{vec}}(S(D_i)) \]  \hspace{1cm} (3.21)

Furthermore, in view of (B.2.23) (see appendix B), (3.21) becomes equivalently

\[ Q_i = E_{10}(u_C) \circ \left( \left( U_{d \times l^{vec}}(S_{D_i}) \right)^T \circ \left( U_{l \times np} \otimes S_{E_iC_i} \right) \right) \]
\[ \mathbb{E}_{10} \left( U_{1 \times np} \otimes L_{E_i C_i} + U_{d \times I^{vec}} \left( u_X^T \oplus u_X + U_{np \times 1^{vec}} (L_{10}(D_i))^T \right) \right) \vec{X}^T \otimes \vec{X} \]

\[ \Rightarrow Q_i = \mathbb{E}_{10} (u_C) \circ \left( [S_{E_i D_i C_i}] \right) \]

\[ \mathbb{E}_{10} \left( U_{1 \times np} \otimes L_{E_i C_i} + U_{d \times I^{vec}} \left( u_X^T \oplus u_X + U_{np \times 1^{vec}} (L_{10}(D_i))^T \right) \right) \vec{X}^T \otimes \vec{X} \]

where

\[ S_{E_i D_i C_i} := \left( U_{d \times I^{vec}} (S_{D_i}) \right)^T \left( U_{1 \times np} \otimes S_{E_i C_i} \right) \]

Finally, (3.22) via (B.2.3), (B.2.18) and (B.2.21) (see appendix B) gives equivalently

\[ Q_i = \left( S_{E_i D_i C_i} \circ \mathbb{E}_{10} \left( S_{Q_i} \right) \right) \vec{X}^T \otimes \vec{X} \]

where

\[ S_{Q_i} := U_{1 \times np} \otimes L_{E_i C_i} + U_{d \times I^{vec}} \left( u_X^T \oplus u_X + U_{np \times 1^{vec}} (L_{10}(D_i))^T \right)^T + u_C U_{I \times (np)^2} \]

Now, from (B.2.3) (see appendix B) we have

\[ \mathbb{E}_{10} (u_C) \circ \mathbb{S} \left( \vec{vec}(G) \right) \circ \vec{vec}(\mathbb{E}_{10} (L_{10}(G))) = \mathbb{S}_{G} \circ \mathbb{E}_{10} (S_{C}) \]

where

\[ \mathbb{S}_{C} := u_C + L_{10} (\vec{vec}(G)) \]

\[ \mathbb{S}_{G} := \mathbb{S} (\vec{vec}(G)) \]

From the above analysis, and more precisely from (3.16), (3.24) and (3.26), it is apparent that (3.10) can be equivalently written as

\[ \bar{F}(\bar{x}) = \left( \sum_{i=1}^{\omega} (S_{B_i A_i} \circ \mathbb{E}_{10} (S_{L_i})) \right) \bar{x} + \left( \sum_{i=1}^{\xi} (S_{E_i D_i C_i} \circ \mathbb{E}_{10} (S_{Q_i})) \right) \vec{X}^T \otimes \vec{X} + \mathbb{S}_{G} \circ \mathbb{E}_{10} (S_{G}) = 0 \]

Now (3.29) is by construction a vector equation containing the magnitudes of the coefficients of the original equation (3.1) and the scaling factors as the powers \( S_{L_i} \), \( S_{Q_i} \) and \( S_{C} \). As mentioned in section 3.1, the general idea in the present approach is to minimize the magnitudes.
of the coefficients of the original equation (3.1) with respect to the scaling factors given by (3.7) and (3.8). In order to formalize such a minimization define

$$S(uX, uC) = \sum_{i=1}^{\omega} \|SL_i\|_F^2 + \sum_{i=1}^{\xi} \|SQ_i\|_F^2 + \|SC\|_F^2.$$  \hspace{1cm} (3.30)

Hence, the problem of scaling is mathematically stated as:

**Scaling Problem:** Minimize $S(uX, uC)$ with respect to $uX$ and $uC$, i.e.

$$\min_{(uX, uC)} \{S(uX, uC)\}.$$  \hspace{1cm} (3.31)

At this point note that, the operator $L_{10}(\cdot)$ by definition (see appendix B) will scale accordingly very small or very large values in (3.29). Moreover, the factors $S_{B_iA_i}$, $S_{E_iD_iC_i}$ and $S_G$ because of the definition of $S(\cdot)$ (see appendix B), are already normalized since

$$\text{sign}(z) = \frac{z}{|z|}, \quad |z| \geq \Re(z) \quad \text{and} \quad |z| \geq \Im(z), \forall z \in \mathbb{C}.$$  \hspace{1cm} (3.32)

Hence, the problem of scaling should concentrated to $S_{L_i}$, $S_{Q_i}$ and $S_C$, and this is done via (3.31).

### 3.2.2 Synthesis

The synthesis process is summarized in theorem 3.1 below, which solves the scaling problem (3.31).

**Theorem 3.1:**

A unique global solution to the scaling problem (3.31) is given by

$$\begin{bmatrix} uX \\ uC \end{bmatrix} = K^{-1}M$$  \hspace{1cm} (3.32)

where

$$K := \begin{bmatrix} 2\zeta U_{np \times np} + (2\zeta \ln p + \omega \zeta)I_{np} & (2\zeta np + \omega)U_{np \times d} \\ (2\zeta np + \omega)U_{d \times np} & (\zeta np)^2 + \omega np + 1 \end{bmatrix}$$  \hspace{1cm} (3.33)

and

$$M := \begin{bmatrix} \sum_{i=1}^{\xi} L_{10}(D_i^T) + (U_{1 \times d} \otimes B_i)\text{vec} \left( \sum_{i=1}^{\xi} (L_{E_iC_i})^T \right) + \sum_{i=1}^{\omega} (L_{B_iA_i})^T U_{d \times I} \\ npU_{d \times np} \text{vec} \left( \sum_{i=1}^{\xi} L_{10}(D_i^T) + np \left( \sum_{i=1}^{\xi} (L_{E_iC_i}) \right) U_{np \times 1} + \sum_{i=1}^{\omega} (L_{B_iA_i}U_{np \times 1}) + \text{vec}(L_{10}(G)) \right) \end{bmatrix}$$  \hspace{1cm} (3.34)
\[ B_{I} := p(I_n \otimes U_{n \times n}) + n(U_{p \times p} \otimes I_n) \quad (3.35) \]

**Proof.**

The proof of this theorem is constructive. The sufficient conditions [53] for the minimization problem (3.31) to have strict or local minimum points are

\[ \nabla S(u_{X}, u_{C}) = 0 \quad (3.36) \]

and

\[ \nabla^{2} S(u_{X}, u_{C}) > 0 \quad (3.37) \]

where \( \nabla S(u_{X}, u_{C}) \in \mathbb{R}^{np+tl} \), \( \nabla^{2} S(u_{X}, u_{C}) \in \mathbb{R}^{(np+tl) \times (np+tl)} \), and \( \nabla^{2} S(u_{X}, u_{C}) > 0 \) means \( \nabla^{2} S(u_{X}, u_{C}) \) is a positive definite matrix.

By completing all the algebra on the left hand side of (3.36), i.e. writing analytically the Frobenious norms, grouping appropriate terms and differentiating with respect to \( u_{X}, u_{C} \), equation (3.38) below emerges:

\[ K \begin{bmatrix} u_{X} \\ u_{C} \end{bmatrix} - M = 0 \quad (3.38) \]

where \( K \) and \( M \) are given by (3.33) and (3.34) respectively. Hence, (3.36) is equivalent to (3.38) and this is an immediate consequence of the operator \( \nabla (\cdot) \) in (3.30) and nothing more.

Furthermore, \( \nabla^{2} S(u_{X}, u_{C}) = \nabla \nabla S(u_{X}, u_{C}) \) and from the left hand side of (3.38) it follows that \( \nabla \nabla S(u_{X}, u_{C}) = \nabla \left( K \begin{bmatrix} u_{X} \\ u_{C} \end{bmatrix} - M \right) = K \). Therefore,

\[ \nabla^{2} S(u_{X}, u_{C}) = K \quad (3.39) \]

Now it can be shown that \( K > 0 \). This result is given as lemma 3.1 of theorem 3.2 both given at the end of the present proof. Hence, from \( K > 0 \) and (3.37) follows. Moreover, \( K > 0 \) implies that \( K^{-1} \) exists and is unique. Hence (3.38) has a unique solution given by (3.32). Therefore, since (3.36) and (3.37) are satisfied under a unique solution point given by (3.32), this point is the global minimum of the minimization problem (3.31).

\[ \text{\textsuperscript{\dagger}} \text{For } \nabla \cdot \text{ see section B.3 in appendix B. } \nabla^{2} 
\text{:= } \nabla \nabla \cdot \]
In order to prove $K > 0$ in lemma 3.1, two preliminary results, proposition 3.1 and theorem 3.2, are required.

**Proposition 3.1:**

Let $W := \rho U_{\sigma \times \sigma} + k I_{\sigma}$, $\sigma \in \mathbb{Z}_+$, $\rho \in \mathbb{R}_+ \cup \{0\}$, $k \in \mathbb{R}_+$, $W \in \mathbb{R}^{\sigma \times \sigma}$.

Then $W > 0$ with eigenvalues $\xi_1 = \xi_2 = \ldots = \xi_{\sigma-1} = k$, $\xi_\sigma = \rho \sigma + k$. Furthermore, $\det(W) = (\rho \sigma + k)^{\sigma-1} \neq 0$ and $W^{-1} = \frac{-\rho}{(\rho \sigma + k)k} U_{\sigma \times \sigma} + \frac{1}{k} I_{\sigma}$.

**Proof.**

If $\rho = 0$, the proof is obvious since $W = k I_{\sigma}$, $k \in \mathbb{R}_+$. For $\rho \in \mathbb{R}_+$ the proof is given next.

Let $W = \begin{bmatrix} \rho + k & \rho & \rho & \cdots & \rho \\ \rho & \rho + k & \rho & \cdots & \rho \\ \rho & \rho & \rho + k & \cdots & \rho \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \rho & \rho & \rho & \cdots & \rho + k \end{bmatrix}$.

By definition, $W \in \mathbb{R}^{\sigma \times \sigma}$ are linearly independent vectors in $\mathbb{R}^\sigma$ for all $\rho \in \mathbb{R}_+$ and $k \in \mathbb{R}_+$. Therefore, $W \in \mathbb{R}^{\sigma \times \sigma}$ is by construction a nonsingular matrix, which implies that $\forall \sigma \in \mathbb{Z}_+$, $\forall \rho \in \mathbb{R}_+$ and $\forall k \in \mathbb{R}_+$, $\det(W) \neq 0$ and $W^{-1}$ is well defined.

The proof of the expression for $\det(W)$ is arrived at by induction as follows.

For $\sigma = 1$, $\det(W) = (\rho + k)^{1-1} = \rho + k$, which is true since $W = \rho I + k I = \rho + k$ and so $\det(W) = \det(\rho + k) = \rho + k$. Assuming that $\det(W) = (\rho \sigma + k)^{\sigma-1}$ holds for $\sigma \in \mathbb{Z}_+$, it will be shown that it also holds for $\sigma + 1$; i.e. it will be proven that $\det(W) = \det(\rho U_{(\sigma+1) \times (\sigma+1)} + k I_{\sigma + 1}) = (\rho (\sigma + 1) + k)^{\sigma+1-1} = (\rho \sigma + \rho + k)^{\sigma}$.

This part of the proof is constructive

$$
\det(W) = \det\left[\rho U_{(\sigma+1) \times (\sigma+1)} + k I_{\sigma + 1}\right] = \det\left[\begin{bmatrix} \rho U_{\sigma \times \sigma} + k I_{\sigma} & \rho U_{\sigma \times I} \\ \rho U_{I \times \sigma} & k + \rho \end{bmatrix}\right]
$$

$$
= (-1)^{2\sigma} \det\left[\begin{bmatrix} k + \rho & \rho U_{I \times \sigma} \\ \rho U_{\sigma \times I} & \rho U_{\sigma \times \sigma} + k I_{\sigma} \end{bmatrix}\right].
$$

(3.40)

Since $\rho + k \neq 0$ (immediate from $\forall \rho \in \mathbb{R}_+$, $k \in \mathbb{R}_+$), (3.40), via (B.2.25) (see appendix B), is equivalent to
\[ \det(W) = \det(\rho \sigma \times \sigma + k I_\sigma - \rho U_{\sigma \times 1}(\rho + k)^{-1} \rho U_{I \times \sigma}) \]

\[ = (\rho + k) \det(\rho U_{\sigma \times \sigma} + k I_\sigma - \frac{\rho^2}{k + \rho} U_{\sigma \times \sigma}) \]

\[ = (\rho + k) \det\left( \frac{\rho^2 + \rho k - \rho^2}{k + \rho} U_{\sigma \times \sigma} + k I_\sigma \right) = (\rho + k) \det\left( \frac{\rho k}{k + \rho} U_{\sigma \times \sigma} + k I_\sigma \right). \quad (3.41) \]

(3.41) via the last assumption \( \det(W) = (\rho \sigma + k) \sigma^{-1} \) holds for \( \sigma \in \mathbb{Z}_+ \) and with \( \rho \leftarrow \frac{\rho k}{k + \rho} \), is equivalent to

\[ \det(W) = (\rho + k) \det\left( \frac{\rho k}{k + \rho} \sigma + k \right) \sigma^{-1} = (\rho + k) \left( \frac{\rho k \sigma + \rho k + k^2}{k + \rho} \right) \sigma^{-1} \]

\[ = \frac{\rho + k}{\rho + k} k(\rho \sigma + \rho + k) k^{-1} = (\rho \sigma + k) k^\sigma. \]

Hence, it has been proven that \( \det(W) = (\rho \sigma + k) k^\sigma \), and hence that \( \det(W) = (\rho \sigma + k) k^{-1} \neq 0 \).

To show that \( W > 0 \), it is enough to show that all its eigenvalues are strictly positive.
Now the eigenvalues of \( W \) are determined as the roots \( \xi_1, \xi_2, \ldots, \xi_\sigma \) of its characteristic polynomial being equal to zero [83], namely \( \det(W - \lambda I_\sigma) = 0 \) with respect to \( \lambda \).

Hence,

\[ \det(W - \lambda I_\sigma) = 0 \iff \det(\rho U_{\sigma \times \sigma} + k I_\sigma - \lambda I_\sigma) = 0 \]

\[ \iff \det(\rho U_{\sigma \times \sigma} + (k - \xi) I_\sigma) = 0. \quad (3.42) \]

If \( \xi = k \), then (3.42) becomes \( \det(\rho U_{\sigma \times \sigma}) = 0 \) since all columns of \( \rho U_{\sigma \times \sigma} \) are linearly independent vectors in \( \mathbb{R}^\sigma \forall \rho \in \mathbb{R}_+ \). Therefore, \( \xi = k \) is an eigenvalue of \( W \) and moreover \( \xi > 0 \) since \( k \in \mathbb{R}_+ \).

If \( \xi \neq k \), then \( k - \xi \neq 0 \). Also, if \( \rho + k - \xi = 0 \iff \xi = \rho + k \) then (3.42) gives
Chapter 3  Scaling & Homogeneous Projective Transformations  71

\[
\det(\rho U_{\sigma \times \sigma} - \rho I_{\sigma}) = 0 \iff \det \begin{bmatrix}
0 & \rho & \cdots & \rho \\
\rho & 0 & \cdots & \rho \\
\rho & \rho & 0 & \cdots \\
\vdots & \vdots & \ddots & \vdots \\
\rho & \rho & \rho & \cdots & 0
\end{bmatrix} = 0.
\]
The last result is a contradiction since

\[
\begin{bmatrix}
0 & \rho & \cdots & \rho \\
\rho & 0 & \cdots & \rho \\
\rho & \rho & 0 & \cdots \\
\vdots & \vdots & \ddots & \vdots \\
\rho & \rho & \rho & \cdots & 0
\end{bmatrix}
\]

\[\forall \rho \in \mathbb{R}_+, \text{ all columns of}\]

hence, the determinant of this matrix is different from zero. Therefore, \(\rho + k\) is not an eigenvalue of \(W\), and so \(\rho + k - \xi \neq 0\) with respect to (3.42). From the above, it is proven next by induction that with respect to \(\xi \neq k\) and (3.42),

\[
\det(\rho U_{\sigma \times \sigma} + (k - \xi)I_{\sigma}) = (\rho \sigma + k - \xi)(k - \xi)^{\sigma - 1}. 
\]

For \(\sigma = 1\) \(\det(\rho l + (k - \xi)l) = (\rho l + k - \xi)(k - \xi)^{-1} = \rho + k - \xi\), which is obviously true.

Assuming that \(\det(\rho U_{\sigma \times \sigma} + (k - \xi)I_{\sigma}) = (\rho \sigma + k - \xi)(k - \xi)^{\sigma - 1}\) holds for \(\sigma \in \mathbb{Z}_+\) it will be shown that it is also true for \(\sigma + 1\); i.e. it will be proven that

\[
\det(\rho U_{(\sigma + 1) \times (\sigma + 1)} + (k - \xi)I_{\sigma + 1}) = (\rho(\sigma + 1) + k - \xi)(k - \xi)^{\sigma + 1 - 1} = (\rho \sigma + \rho + k - \xi)(k - \xi)^{\sigma}
\]

This proof is again constructive

\[
\det(\rho U_{(\sigma + 1) \times (\sigma + 1)} + (k - \xi)I_{\sigma + 1}) = \det \begin{bmatrix}
\rho U_{\sigma \times \sigma} + (k - \xi)I_{\sigma} & \rho U_{\sigma \times I} \\
\rho U_{I \times \sigma} & k - \xi + \rho
\end{bmatrix}
\]

\[
= (-1)^{2\sigma} \det \begin{bmatrix}
k - \xi + \rho & \rho U_{I \times \sigma} \\
\rho U_{\sigma \times I} & \rho U_{\sigma \times \sigma} + (k - \xi)I_{\sigma}
\end{bmatrix}.
\]

Since because of (3.42) and \(\rho + k \neq 0\) we have \(\rho + k - \xi \neq 0\) (\(k \in \mathbb{R}_+\)), then (3.44), via (B.2.25) (see appendix B), is equivalent to

\[
\det(\rho + k - \xi) \det \left( \rho U_{\sigma \times \sigma} + (k - \xi)I_{\sigma} - \rho U_{\sigma \times I}(\rho + k - \xi)^{-1}\rho U_{I \times \sigma} \right)
\]

\[
= (\rho + k - \xi)\det \left( \rho U_{\sigma \times \sigma} + (k - \xi)I_{\sigma} - \frac{\rho^2}{k - \xi + \rho} U_{\sigma \times \sigma} \right)
\]

\[
= (\rho + k - \xi)\det \left( \frac{\rho^2(k - \xi) - \rho^2}{k - \xi + \rho} U_{\sigma \times \sigma} + (k - \xi)I_{\sigma} \right)
\]

\[
= (\rho + k - \xi)\det \left( \frac{\rho(k - \xi)}{k - \xi + \rho} U_{\sigma \times \sigma} + (k - \xi)I_{\sigma} \right). 
\]

(3.45)
(3.45) via the previous assumption \( \det(\rho U_{\sigma \times \sigma} + (k - \zeta)I_{\sigma}) = (\rho \sigma + k - \zeta)(k - \zeta)^{\sigma - 1} \) holds for \( \sigma \in \mathbb{Z}_+ \) and with \( \rho \rightarrow \frac{\rho(k - \zeta)}{k - \zeta + \rho} \), becomes equivalently

\[
(\rho + k - \zeta) \det \left( \frac{\rho(k - \zeta)}{k - \zeta + \rho} \sigma + k - \zeta \right)(k - \zeta)^{\sigma - 1} = \frac{(\rho + k - \zeta)(\rho \sigma + \rho + k - \zeta)(k - \zeta)^{\sigma - 1}}{(\rho + k - \zeta)(k - \zeta)^{\sigma - 1}} = (\rho \sigma + \rho + k - \zeta)(k - \zeta)^{\sigma - 1}.
\]

Therefore, (3.43) is proven. Now (3.43) in view of (3.42) with respect to \( \zeta \neq k \), becomes equivalently,

\[
(\rho \sigma + k - \zeta)(k - \zeta)^{\sigma - 1} = 0.
\]

Since \( \zeta \neq k \), (3.46) has a unique solution with multiplicity 1, \( \zeta = \rho \sigma + k \). Therefore, \( \zeta = \rho \sigma + k \) is an eigenvalue of \( W \) with multiplicity 1. Moreover, because \( \sigma \in \mathbb{Z}_+ \), \( \forall \rho \in \mathbb{R}_+ \) and \( k \in \mathbb{R}_+ \), it is apparent that \( \zeta = \rho \sigma + k > 0 \).

Note that it has been previously shown that \( \forall \zeta \neq k \) there is a unique solution of multiplicity 1, and for \( \zeta = k \) (3.42) can provide this solution \( (\zeta = k) \) only. Furthermore, since \( W \) is nonsingular, (3.42) has \( \sigma \) solutions not at infinity (bounded). Therefore the only valid possibility is for (3.42) to have the roots, \( \zeta = \rho \sigma + k \) with multiplicity 1 and \( \zeta = k \) with multiplicity \( \sigma - 1 \). Hence, the eigenvalues of \( W \) are \( \zeta_1 = \zeta_2 = \ldots = \zeta_{\sigma - 1} = k > 0 \), \( \zeta_\sigma = \rho \sigma + k > 0 \). It is obvious that \( W > 0 \).

Now, noting that \( U_{\sigma \times \sigma}^2 = U_{\sigma \times \sigma} U_{\sigma \times \sigma} = \sigma U_{\sigma \times \sigma} \), it follows that

\[
WW = (\rho U_{\sigma \times \sigma} + kI_\sigma)(\rho U_{\sigma \times \sigma} + kI_\sigma) = \rho^2 U_{\sigma \times \sigma}^2 + 2\rho k U_{\sigma \times \sigma} + k^2 I_\sigma
\]

\[
= \rho^2 \sigma U_{\sigma \times \sigma} + 2\rho k U_{\sigma \times \sigma} + k^2 I_\sigma = \left(\rho^2 \sigma + 2\rho k\right)U_{\sigma \times \sigma} + k^2 I_\sigma
\]

\[
\Leftrightarrow WW = \left(\rho^2 \sigma + 2\rho k\right)U_{\sigma \times \sigma} + k^2 I_\sigma.
\]

Also,

\[
WU_{\sigma \times \sigma} = (\rho U_{\sigma \times \sigma} + kI_\sigma)U_{\sigma \times \sigma} = \rho U_{\sigma \times \sigma}^2 + k U_{\sigma \times \sigma}
\]

\[
= \rho \sigma U_{\sigma \times \sigma} + kU_{\sigma \times \sigma} = (\rho \sigma + k)U_{\sigma \times \sigma}
\]

\[
\Leftrightarrow WW = (\rho \sigma + k)U_{\sigma \times \sigma}
\]
\[ U_{\sigma \times \sigma} = \frac{1}{\rho \sigma + k} WU_{\sigma \times \sigma}. \]  

(3.48)

In view of (3.48), (3.47) becomes equivalently,

\[
WW = \frac{\rho^2 \sigma^2 + 2\rho k}{\rho \sigma + k} WU_{\sigma \times \sigma} + k^2 I_{\sigma} \iff W^{-1}WW = \frac{\rho^2 \sigma^2 + 2\rho k}{\rho \sigma + k} W^{-1}WU_{\sigma \times \sigma} + k^2 W^{-1}I_{\sigma}.
\]

\[ \iff W = \frac{\rho^2 \sigma^2 + 2\rho k}{\rho \sigma + k} U_{\sigma \times \sigma} + k^2 W^{-1} \iff W^{-1} = \frac{1}{k^2} W - \frac{\rho^2 \sigma^2 + 2\rho k}{(\rho \sigma + k)k^2} U_{\sigma \times \sigma} \]

\[ \iff W^{-1} = \frac{1}{k^2} (\rho U_{\sigma \times \sigma} + kI_{\sigma}) - \frac{\rho^2 \sigma^2 + 2\rho k}{(\rho \sigma + k)k^2} U_{\sigma \times \sigma} \iff W^{-1} = \frac{\rho}{k^2} U_{\sigma \times \sigma} + \frac{1}{k} I_{\sigma} - \frac{\rho^2 \sigma^2 + 2\rho k}{(\rho \sigma + k)k^2} U_{\sigma \times \sigma} \]

\[ \iff W^{-1} = \frac{\rho^2 \sigma^2 + \rho k - \rho^2 \sigma^2 - 2\rho k}{(\rho \sigma + k)k^2} U_{\sigma \times \sigma} + \frac{1}{k} I_{\sigma} \iff W^{-1} = \frac{-\rho}{(\rho \sigma + k)} U_{\sigma \times \sigma} + \frac{1}{k} I_{\sigma}. \]

After proving proposition 3.1, theorem 3.2 is also required before proving lemma 3.1

**Theorem 3.2:**

Let the symmetric real matrix

\[ V := \begin{bmatrix} 
\rho U_{\sigma \times \sigma} + kI_{\sigma} & \nu U_{\sigma \times q} \\
\nu U_{q \times \sigma} & \lambda I_q
\end{bmatrix}, \quad V \in \mathbb{R}^{(\sigma+q) \times (\sigma+q)}, \quad V = V^T, \]

\[ \rho \in \mathbb{N}, \quad k, \lambda, \nu, \sigma, q \in \mathbb{Z}^+. \] Then \( V > 0 \) if and only if \( \sigma \nu^2 - (k + \rho \sigma)\lambda < 0 \). Furthermore, the eigenvalues \( \xi_1, \xi_2, \ldots, \xi_{\sigma+q-2} \in \mathbb{Z}^+ \), \( \xi_{\sigma+q-1}, \xi_{\sigma+q} \in \mathbb{R}_+ \) of \( V \) are equal to

\[ \xi_1 = \xi_2 = \ldots = \xi_{\sigma-1} = k, \quad \xi_{\sigma} = \rho \sigma + k, \quad \xi_{\sigma+1} = \ldots = \xi_{\sigma+q-2} = \lambda, \]

\[ \xi_{\sigma+q-1} = \frac{1}{2} \left( \rho \sigma + k + \lambda + \left( \rho^2 \sigma^2 + 2\rho \sigma k - 2\rho \sigma \lambda + k^2 - 2k \lambda + \lambda^2 + 4 \sigma \nu^2 \right)^{\frac{1}{2}} \right), \]

\[ \xi_{\sigma+q} = \frac{1}{2} \left( \rho \sigma + k + \lambda - \left( \rho^2 \sigma^2 + 2\rho \sigma k - 2\rho \sigma \lambda + k^2 - 2k \lambda + \lambda^2 + 4 \sigma \nu^2 \right)^{\frac{1}{2}} \right). \]

**Proof.**

Under the theorem's assumptions, it follows from proposition 3.1 that \( \rho U_{\sigma \times \sigma} + kI_{\sigma} > 0 \) and moreover \( (\rho U_{\sigma \times \sigma} + kI_{\sigma})^{-1} > 0 \). Now define

\[ Z := \begin{bmatrix} I_{\sigma} & 0_{\sigma \times q} \\
-\nu U_{q \times \sigma} (\rho U_{\sigma \times \sigma} + kI_{\sigma})^{-1} & I_q
\end{bmatrix}. \]

It follows that
Again from proposition 3.1 and \((\rho U_{\sigma \times \sigma} + kl_\sigma)^{-1} = -\frac{\rho}{(\rho \sigma + k)} U_{\sigma \times \sigma} + \frac{1}{k} I_\sigma > 0\), the last equality gives

\[
ZVZ = \begin{bmatrix}
\rho U_{\sigma \times \sigma} + kl_\sigma & 0_{\sigma \times q} \\
0_{q \times \sigma} & \lambda I_q - \frac{\rho v^2}{(\rho \sigma + k)} U_{q \times q} - \frac{v^2}{k} U_{q \times q} U_{\sigma \times q}
\end{bmatrix}.
\] (3.49)

But by construction \(U_{q \times q} U_{\sigma \times \sigma} = \sigma U_{q \times \sigma}\) and \(U_{q \times \sigma} U_{\sigma \times q} = \sigma U_{q \times q}\). Hence (3.49) becomes equivalently

\[
ZVZ = \begin{bmatrix}
\rho U_{\sigma \times \sigma} + kl_\sigma & 0_{\sigma \times q} \\
0_{q \times \sigma} & \lambda I_q - \frac{\rho v^2}{(\rho \sigma + k)} U_{q \times q} - \frac{v^2}{k} U_{q \times q} U_{\sigma \times q}
\end{bmatrix}.
\] (3.50)

From (3.50), it is apparent that \(V\) can be diagonalised as a block diagonal matrix with block matrices \(\rho U_{\sigma \times \sigma} + kl_\sigma\) and \(\lambda I_q - \frac{v^2}{(\rho \sigma + k)} U_{q \times q}\). It is well known [83] that

\[V > 0 \iff \rho U_{\sigma \times \sigma} + kl_\sigma > 0 \quad \text{and} \quad \lambda I_q - \frac{v^2}{(\rho \sigma + k)} U_{q \times q} > 0\].

From proposition 3.1, \(\rho U_{\sigma \times \sigma} + kl_\sigma > 0\).
To show that \( \mathbf{\lambda}_q = \frac{\nu^2 \sigma}{\rho \sigma + k} \mathbf{U}_q \times q > 0 \), it is enough to show that all its eigenvalues \( \varepsilon_1, \varepsilon_2, \ldots, \varepsilon_q \) are strictly positive. Now, the eigenvalues of \( \mathbf{\lambda}_q = \frac{\nu^2 \sigma}{\rho \sigma + k} \mathbf{U}_q \times q \) are determined as the roots of its characteristic equation [83], namely

\[
det \left( \mathbf{\lambda}_q - \frac{\nu^2 \sigma}{\rho \sigma + k} \mathbf{U}_q \times q \right) = 0 \quad \text{with respect to } \varepsilon \, \text{. Hence,}
\]

\[
det \left( \mathbf{\lambda}_q - \frac{\nu^2 \sigma}{\rho \sigma + k} \mathbf{U}_q \times q - \mathbf{I}_q \right) = 0 \Leftrightarrow \det \left( \mathbf{\lambda}_q - \frac{\nu^2 \sigma}{\rho \sigma + k} \mathbf{U}_q \times q - \mathbf{I}_q \right) = 0
\]

\[
\Leftrightarrow \det \left( -\frac{\nu^2 \sigma}{\rho \sigma + k} \mathbf{U}_q \times q + (\lambda - \varepsilon) \mathbf{I}_q \right) = 0 \, . \quad (3.51)
\]

Assume \( \varepsilon \neq \lambda \). It is proven next, by induction, that

\[
det \left( -\frac{\nu^2 \sigma}{\rho \sigma + k} \mathbf{U}_q \times q + (\lambda - \varepsilon) \mathbf{I}_q \right) = \left( -\frac{\nu^2 \sigma}{\rho \sigma + k} + \lambda - \varepsilon \right) (\lambda - \varepsilon)^{q-1} \, . \quad (3.52)
\]

For \( q = 1 \),

\[
\det \left( -\frac{\nu^2 \sigma}{\rho \sigma + k} \mathbf{U}_q \times q + (\lambda - \varepsilon) \mathbf{I}_q \right) = \left( -\frac{\nu^2 \sigma}{\rho \sigma + k} + \lambda - \varepsilon \right) (\lambda - \varepsilon) = -\frac{\nu^2 \sigma}{\rho \sigma + k} + \lambda - \varepsilon,
\]

which is obviously true. Assuming that

\[
\det \left( -\frac{\nu^2 \sigma}{\rho \sigma + k} \mathbf{U}_q \times q + (\lambda - \varepsilon) \mathbf{I}_q \right) = \left( -\frac{\nu^2 \sigma}{\rho \sigma + k} + \lambda - \varepsilon \right) (\lambda - \varepsilon)^{q-1} \text{ holds for } q \in \mathbb{Z}_+ \text{ it will be shown that it is also true for } q + 1 \, ; \text{i.e. it will be proven that}
\]

\[
\det \left( -\frac{\nu^2 \sigma}{\rho \sigma + k} \mathbf{U}_{q+1} \times (q+1) + (\lambda - \varepsilon) \mathbf{I}_{q+1} \right) = \left( -\frac{\nu^2 \sigma}{\rho \sigma + k} + \lambda - \varepsilon \right) (\lambda - \varepsilon)^q
\]

This proof is again constructive

\[
\det \left( -\frac{\nu^2 \sigma}{\rho \sigma + k} \mathbf{U}_{q+1} \times (q+1) + (\lambda - \varepsilon) \mathbf{I}_{q+1} \right) = \det \left( \begin{bmatrix}
-\frac{\nu^2 \sigma}{\rho \sigma + k} \mathbf{U}_q \times q + (\lambda - \varepsilon) \mathbf{I}_q \\
-\frac{\nu^2 \sigma}{\rho \sigma + k} \mathbf{U}_q \times q + (\lambda - \varepsilon) \mathbf{I}_q \\
\end{bmatrix}
\right)
\]

\[
= (-1)^2 q \det \left( \begin{bmatrix}
\lambda - \varepsilon - \frac{\nu^2 \sigma}{\rho \sigma + k} \\
\frac{\nu^2 \sigma}{\rho \sigma + k} \mathbf{U}_q \times q
\end{bmatrix}
\right).
\]

From (3.51), \( \varepsilon \neq \lambda \) and the last assumption
Chapter 3  Scaling & Homogeneous Projective Transformations 76

\[
(\det -\frac{v^2\sigma}{\rho\sigma+k}U_{q}\times q + (\lambda - \zeta)I_{q}) = \left( -\frac{v^2\sigma}{\rho\sigma+k} + \lambda - \zeta \right)(\lambda - \zeta)^q = \text{ holds for } q \in \mathbb{Z}_+ \text{ ) it follows that}
\]

\[
\det \left( -\frac{v^2\sigma}{\rho\sigma+k}U_{q}\times q + (\lambda - \zeta)I_{q} \right) = 0 \iff \left( -\frac{v^2\sigma}{\rho\sigma+k} + \lambda - \zeta \right)(\lambda - \zeta)^q = 0 \iff -\frac{v^2\sigma}{\rho\sigma+k} + \lambda - \zeta = 0
\]

Therefore, \(-\frac{v^2\sigma}{\rho\sigma+k} + \lambda - \zeta \neq 0\) (note that \(q \in \mathbb{Z}_+\), and (3.53), via (B.2.25) (see appendix B),

becomes equivalently

\[
\det \left( -\frac{v^2\sigma}{\rho\sigma+k}U_{(q+1)\times (q+1)} + (\lambda - \zeta)I_{(q+1)} \right)
\]

\[
= \det \left( \lambda - \zeta - \frac{v^2\sigma}{\rho\sigma+k} \right) \det \left( -\frac{v^2\sigma}{\rho\sigma+k}U_{q}\times q + (\lambda - \zeta)I_{q} - \frac{v^2\sigma}{\rho\sigma+k}U_{q}\times 1 \left( \lambda - \zeta - \frac{v^2\sigma}{\rho\sigma+k} \right)^{-1} \right) U_{q}\times 1 \right) \right)
\]

\[
= \left( \lambda - \zeta - \frac{v^2\sigma}{\rho\sigma+k} \right) \det \left( -\frac{v^2\sigma}{\rho\sigma+k} \left( 1 + \frac{v^2\sigma}{\rho\sigma+k} \left( \lambda - \zeta - \frac{v^2\sigma}{\rho\sigma+k} \right)^{-1} \right) U_{q}\times q + (\lambda - \zeta)I_{q} \right)
\]

\[
= \left( \lambda - \zeta - \frac{v^2\sigma}{\rho\sigma+k} \right) \det \left( -\frac{v^2\sigma}{\rho\sigma+k} \left( \lambda - \zeta - \frac{v^2\sigma}{\rho\sigma+k} \right)^{-1} U_{q}\times q + (\lambda - \zeta)I_{q} \right) \right)
\]

(3.54)

via the previous assumption

\[
(\det -\frac{v^2\sigma}{\rho\sigma+k}U_{q}\times q + (\lambda - \zeta)I_{q}) = \left( -\frac{v^2\sigma}{\rho\sigma+k} + \lambda - \zeta \right)(\lambda - \zeta)^q = \text{ holds for } q \in \mathbb{Z}_+ \text{ ) and with}
\]

\[
\frac{v^2\sigma}{\rho\sigma+k} \leftrightarrow \frac{v^2\sigma}{\rho\sigma+k} \left( \lambda - \zeta - \frac{v^2\sigma}{\rho\sigma+k} \right)^{-1}
\]

becomes equivalently

\[
\det \left( -\frac{v^2\sigma}{\rho\sigma+k}U_{(q+1)\times (q+1)} + (\lambda - \zeta)I_{(q+1)} \right)
\]

\[
= \left( \lambda - \zeta - \frac{v^2\sigma}{\rho\sigma+k} \right) \left( \frac{v^2\sigma}{\rho\sigma+k} \left( \lambda - \zeta - \frac{v^2\sigma}{\rho\sigma+k} \right)^{-1} \right) + (\lambda - \zeta)q^{-1}
\]
\[
\begin{align*}
&= \left( -\frac{\nu^2 \sigma}{\rho \sigma + k} \right) (\lambda - \zeta) \mathbf{q} + \left( \lambda - \zeta - \frac{\nu^2 \sigma}{\rho \sigma + k} \right) (\lambda - \zeta) (\lambda - \zeta)^{q-1} \\
&= \left( -\frac{\nu^2 \sigma}{\rho \sigma + k} \right) (\lambda - \zeta) (\lambda + 1) + (\lambda - \zeta)^2 (\lambda - \zeta)^{q-1} \\
&= \left( -\frac{\nu^2 \sigma (q + 1)}{\rho \sigma + k} + (\lambda - \zeta) \right) (\lambda - \zeta)^q
\end{align*}
\]

Therefore, (3.52) is proven. Now, (3.52) in view of (3.51) with respect to \( \zeta \neq k \), becomes equivalently
\[
\left( -\frac{\nu^2 \sigma q}{\rho \sigma + k} + (\lambda - \zeta) \right) (\lambda - \zeta)^{q-1} = 0 \iff \left( -\frac{\nu^2 \sigma q}{\rho \sigma + k} + (\lambda - \zeta) \right) = 0
\]

(3.55)

Since \( \rho \sigma + k > 0 \), (3.55) has a unique solution with multiplicity 1, \( \zeta = \frac{(\rho \sigma + k) \lambda - \nu^2 \sigma q}{\rho \sigma + k} \).

Therefore, \( \zeta = \frac{(\rho \sigma + k) \lambda - \nu^2 \sigma q}{\rho \sigma + k} \) is an eigenvalue of \( \lambda \mathbf{q} - \frac{\nu^2 \sigma}{\rho \sigma + k} \mathbf{U}_{q \times q} \) with multiplicity 1.

Now, \( \frac{(\rho \sigma + k) \lambda - \nu^2 \sigma q}{\rho \sigma + k} > 0 \) if and only if \( \nu^2 \sigma q - (\rho \sigma + k) \lambda < 0 \).

If \( \zeta = \lambda \), then (3.51) becomes \( \det \left( -\frac{\nu^2 \sigma}{\rho \sigma + k} \mathbf{U}_{q \times q} \right) = 0 \), which is valid since \( \frac{\nu^2 \sigma}{\rho \sigma + k} \neq 0 \) and therefore all columns of \( -\frac{\nu^2 \sigma}{\rho \sigma + k} \mathbf{U}_{q \times q} \) are linearly dependent vectors in \( \mathbb{R}^q \). Therefore,

\( \zeta = \lambda \) is an eigenvalue of \( \lambda \mathbf{q} - \frac{\nu^2 \sigma}{\rho \sigma + k} \mathbf{U}_{q \times q} \) and moreover, \( \zeta > 0 \) since \( \lambda \in \mathbb{Z}_+ \).

Hence, it has been shown that \( \forall \zeta \neq \lambda \) there is a unique solution of multiplicity 1, and for \( \zeta = \lambda \) (3.51) can provide this solution (\( \zeta = \lambda \)) only. Furthermore, \( \lambda \mathbf{q} - \frac{\nu^2 \sigma}{\rho \sigma + k} \mathbf{U}_{q \times q} \) is nonsingular (i.e. its columns are linearly independent vectors in \( \mathbb{R}^q \)), because \( \lambda \neq 0 \) and \( \frac{\nu^2 \sigma}{\rho \sigma + k} \neq 0 \). Therefore, (3.51) has \( q \) solutions not at infinity (bounded). Hence, the only valid possibility is for (3.51) to have the roots, \( \zeta = \frac{(\rho \sigma + k) \lambda - \nu^2 \sigma q}{\rho \sigma + k} \) with multiplicity 1 and \( \zeta = \lambda \).
with multiplicity $q - 1$. Hence, the eigenvalues of $\lambda I_q - \frac{\nu^2 \sigma}{\rho \sigma + k} U_q \times q$ are

$$\xi_1 = \xi_2 = \ldots = \xi_{q-1} = \lambda > 0, \quad \xi_q = \frac{(\rho \sigma + k)\lambda - \nu^2 \sigma q}{\rho \sigma + k} > 0$$

if and only if $\nu^2 \sigma q - (\rho \sigma + k)\lambda < 0$ and it is obvious that $\lambda I_q - \frac{\nu^2 \sigma}{\rho \sigma + k} U_q \times q > 0$ if and only if $\nu^2 \sigma q - (\rho \sigma + k)\lambda < 0$. Finally, it follows that $V > 0$ if and only if $\nu^2 \sigma q - (\rho \sigma + k)\lambda < 0$.

The eigenvalues of $V$ are determined from the roots of its characteristic equation with respect to $\xi$, which is shown below.

$$\det \begin{bmatrix} \rho U_{\sigma \times \sigma} + (k - \xi)I_{\sigma} & vU_{\sigma \times q} \\ vU_{q \times \sigma} & (\lambda - \xi)I_q \end{bmatrix} = 0$$

Assuming $\xi \neq \lambda, \xi \neq k$, (3.56) via (B.2.25) (see appendix B) and proposition 3.1 becomes equivalently

$$\det((\rho U_{\sigma \times \sigma} + (k - \xi)I_{\sigma})) \det(\lambda I_q - \nu^2 U_q \times \sigma (\rho U_{\sigma \times \sigma} + (k - \xi)I_{\sigma})^{-1} U_q \times q) = 0$$

$$\Leftrightarrow (\rho \sigma + k - \xi)(k - \xi)^{\sigma - 1} \det(\lambda I_q - \nu^2 \frac{\rho \sigma^2 \nu^2}{(\rho \sigma + k - \xi)(k - \xi)} U_{\sigma \times \sigma} + \frac{1}{(k - \xi)} I_{\sigma}) U_q \times q = 0$$

$$\Leftrightarrow (\rho \sigma + k - \xi)(k - \xi)^{\sigma - 1} \det(\lambda I_q - \nu^2 \frac{-\sigma q \nu^2}{(\rho \sigma + k - \xi)(k - \xi)} + (\lambda - \xi)(\lambda - \xi)^{q - 1}) = 0$$

$$\Leftrightarrow (k - \xi)^{\sigma - 1}(\lambda - \xi)^{q - 1}(\rho \sigma (\lambda - \xi) + (k - \xi)(\lambda - \xi) - \sigma q \nu^2) = 0$$

$$\Leftrightarrow (k - \xi)^{\sigma - 1}(\lambda - \xi)^{q - 1}(\xi^2 - (\rho \sigma + k + \lambda)\xi + k \lambda + \rho \sigma \lambda - \sigma q \nu^2) = 0$$

Under the assumptions $\xi \neq \lambda$ and $\xi \neq k$, (3.57) has two only roots (each one of them having multiplicity 1); these are the roots of

$$(\xi^2 - (\rho \sigma + k + \lambda)\xi + k \lambda + \rho \sigma \lambda - \sigma q \nu^2) = 0$$

and are equal to

$$\xi_{\sigma + q - 1} = \frac{1}{2} \left[ \rho \sigma + k + \lambda + \left( \rho^2 \sigma^2 + 2 \rho \sigma k - 2 \rho \sigma k + k^2 - 2 k \lambda + \lambda^2 + 4 \sigma q \nu^2 \right)^{\frac{1}{2}} \right].$$
\[ \xi_{\sigma+q} = \frac{1}{2} \left( \rho \sigma + k + \lambda - \left( \rho^2 \sigma^2 + 2 \rho \sigma k - 2 \rho \sigma \lambda + k^2 - 2 k \lambda + \lambda^2 + 4 \sigma \eta \nu^2 \right)^{\frac{1}{2}} \right). \]

Now, note that both these roots are real since \( V = V^T \). Also, because \( \rho \in \mathbb{N}, k, \lambda, \nu, \sigma, q \in \mathbb{Z}^+ \) it follows that \( \left( \rho^2 \sigma^2 + 2 \rho \sigma k - 2 \rho \sigma \lambda + k^2 - 2 k \lambda + \lambda^2 + 4 \sigma \eta \nu^2 \right) \in \mathbb{Z}^+ \). Moreover, it is obvious that \( \xi_{\sigma+q-1} > 0 \). Furthermore,

\[ \xi_{\sigma+q} > 0 \iff \frac{1}{2} \left( \rho \sigma + k + \lambda - \left( \rho^2 \sigma^2 + 2 \rho \sigma k - 2 \rho \sigma \lambda + k^2 - 2 k \lambda + \lambda^2 + 4 \sigma \eta \nu^2 \right)^{\frac{1}{2}} \right) > 0 \]

\[ \implies \rho \sigma + k + \lambda > \left( \rho^2 \sigma^2 + 2 \rho \sigma k - 2 \rho \sigma \lambda + k^2 - 2 k \lambda + \lambda^2 + 4 \sigma \eta \nu^2 \right)^{\frac{1}{2}} \]

\[ \implies \left( \rho \sigma + k + \lambda \right)^2 > \rho^2 \sigma^2 + 2 \rho \sigma k - 2 \rho \sigma \lambda + k^2 - 2 k \lambda + \lambda^2 + 4 \sigma \eta \nu^2 \]

\[ \implies \rho^2 \sigma^2 + k^2 + 2 \rho \sigma k + \lambda^2 + 2 \rho \sigma \lambda + 2 k \lambda > \rho^2 \sigma^2 + 2 \rho \sigma k - 2 \rho \sigma \lambda + k^2 - 2 k \lambda + \lambda^2 + 4 \sigma \eta \nu^2 \]

\[ \implies \left( \rho \sigma + k \right) \lambda > \sigma i \nu^2 \iff \nu^2 \sigma i - (\rho \sigma + k) \lambda < 0 \]

Therefore, \( \xi_{\sigma+q-1} > 0 \) and \( \xi_{\sigma+q} > 0 \) if and only if \( \nu^2 \sigma i - (\rho \sigma + k) \lambda < 0 \).

If \( \xi \neq \lambda \) and \( \xi \to k \) then \( \xi \neq \xi_{\sigma+q-1} \) and \( \xi \neq \xi_{\sigma+q} \), and from (3.57) it follows \( (k - \xi)^{\sigma-1} = 0 \). Hence, \( \xi = k > 0 \) is an eigenvalue of \( V \) with multiplicity \( \sigma - 1 \). Similarly, if \( \xi \neq k \) and \( \xi \to \lambda \) then \( \xi \neq \xi_{\sigma+q-1} \) and \( \xi \neq \xi_{\sigma+q} \), and from (3.57) it follows \( (\lambda - \xi)^{\sigma-1} = 0 \).

Hence, \( \xi = \lambda > 0 \) is an eigenvalue of \( V \) with multiplicity \( q - 1 \). From the above it is clear that the eigenvalues \( \xi_1, \xi_2, \ldots, \xi_{\sigma+q-2} \in \mathbb{Z}^+ \), \( \xi_{\sigma+q-1}, \xi_{\sigma+q} \in \mathbb{R}_+ \) of \( V \) are equal to \( \xi_1 = \xi_2 = \ldots = \xi_{\sigma-1} = k \), \( \xi_\sigma = \rho \sigma + k \), \( \xi_{\sigma+1} = \ldots = \xi_{\sigma+q-2} = \lambda \),

\[ \xi_{\sigma+q-1} = \frac{1}{2} \left( \rho \sigma + k + \lambda + \left( \rho^2 \sigma^2 + 2 \rho \sigma k - 2 \rho \sigma \lambda + k^2 - 2 k \lambda + \lambda^2 + 4 \sigma \eta \nu^2 \right)^{\frac{1}{2}} \right), \]

\[ \xi_{\sigma+q} = \frac{1}{2} \left( \rho \sigma + k + \lambda - \left( \rho^2 \sigma^2 + 2 \rho \sigma k - 2 \rho \sigma \lambda + k^2 - 2 k \lambda + \lambda^2 + 4 \sigma \eta \nu^2 \right)^{\frac{1}{2}} \right). \]

With theorem 3.2 and proposition 3.1 proven, it is now possible to prove lemma 3.1 and hence the uniqueness of a minimum solution to the scaling problem.
Chapter 3 Scaling & Homogeneous Projective Transformations 80

Lemma 3.1;
Let \( K \) defined as in (3.33), then \( K > 0 \).

Proof.

Defining, \( \rho:=2\xi t \), \( k:=2\xi ln p + \omega t \), \( v:=2\xi np + \omega \), \( \lambda:=\xi(np)^2 + \omega np + 1 \), \( \sigma:=np \) and \( q:=tl \), it follows that
\[
K = \begin{bmatrix}
\rho U_{\sigma \times \sigma} + k I_{\sigma} & v U_{\sigma \times q} \\
v U_{q \times \sigma} & \lambda I_q
\end{bmatrix}
\]
and \( \rho \in \mathbb{N} \), \( k, \lambda, v, \sigma, q \in \mathbb{Z}_+ \).

Furthermore
\[
\sigma q v^2 - (k + \rho \sigma) \lambda < 0 \Leftrightarrow np t l (2\xi np + \omega)^2 - (2\xi ln p + \omega t + 2\xi ln p) (\xi(np)^2 + \omega np + 1) < 0
\]
\[
\Leftrightarrow -\xi n^2 p^2 t l \omega - 4\xi np t l - \omega t l < 0 \Leftrightarrow -\xi n^2 p^2 t l \omega + 4\xi np t l + \omega t l > 0.
\]
The last inequality is true since \( \omega, \xi \in \mathbb{N} \), \( \omega + \xi \neq 0 \). Therefore, \( \sigma q v^2 - (k + \rho \sigma) \lambda < 0 \) and \( K > 0 \) according to theorem 3.2. 

3.2.3 Variations on Scaling and Some Useful Observations

Matrix \( K \) in (3.33) is independent of the data of (3.1) and depends only on the dimensions of (3.1). This means that for a particular class of problems with the same structure in (3.1), \( K \) needs to be calculated once only. In contrast, \( M \) in (3.34), depends on the data (coefficient matrices) of (3.1) and therefore it needs to be calculated every time for problems with different numerical data.

Now scaling problem (3.31) can be modified by adding various constraints. One such additional constraint is, for example, the minimization of the difference between the magnitudes of the coefficients of the original equation (3.31). Hence, a new problem emerges and is referred to as the scaling problem with magnitude variation. To state this problem mathematically,

define \( S_{LQC} \in \mathbb{R}^{t ln p + t l (np)^2 + t l} \) as

\[
S_{LQC}(u_X, u_C) := \left( \begin{array}{c}
\vec{\left( \sum_{i=1}^{\ell} S_{L_i} \right)} \\
\vec{\left( \sum_{i=1}^{\ell} S_{Q_i} \right)} \\
\vec{(S_C)}
\end{array} \right)^T
\]

and let \( S^e_{LQC} \in \mathbb{R}^{\ell} \), \( \ell \leq t ln p + t l (np)^2 + t l \), \( \ell \in \mathbb{Z}_+ \), be the matrix resulting from \( S_{LQC}(u_X, u_C) \) in (3.58) by removing its zero elements. Then,

\[
S_e(u_X, u_C) := S(u_X, u_C) + \sum_{i=1}^{\ell-1} \sum_{j=i+1}^{\ell} \left( S^e_{LQC}(u_X, u_C) \right)_{i1} - \left( S^e_{LQC}(u_X, u_C) \right)_{j1}
\]
Hence, the problem of scaling with magnitude variation is stated as follows:

**Scaling Problem with Magnitude Variation:** Minimize $S_v(u_x, u_c)$ with respect to $u_x$ and $u_c$, i.e.

$$
\min_{(u_x, u_c)} \left\{ S_v(u_x, u_c) \right\}.
$$

(3.60)

For the problem (3.60), there is no guarantee that a minimum will always exist. Moreover, it is not apparent that any analytic pattern similar to (3.32) can provide an analytic solution for all possible problems under (3.31). Nevertheless, (3.60) can be analyzed for a particular problem and a minimum or maximum could be found, as a solution of a linear equation similar to (3.32) (for the problem (3.31)). In order to automate somehow the process of the derivation of such an equation, one can use symbolic computation. Now, whether a solution to (3.60) is unique or not, depends on the number of solutions of the resulting linear equation (similar to (3.32) for the problem (3.31)). Moreover, the solution/solutions $(u_x, u_c) = (u_x^*, u_c^*)$ are obtained by minimizing or maximizing (3.60), if respectively,

$$
S_v(u_x, u_c)|_{(u_x, u_c) = (0, 0)} > S_v(u_x, u_c)|_{(u_x, u_c) = (u_x^*, u_c^*)}
$$

or

$$
S_v(u_x, u_c)|_{(u_x, u_c) = (0, 0)} < S_v(u_x, u_c)|_{(u_x, u_c) = (u_x^*, u_c^*)}.
$$

Another possible variation of (3.31) is to force the constraints

$$
[u_x]_{ij} = u_{x_s}, \quad u_{x_s} \in \mathbb{R}, \quad \forall i = 1, \ldots, n, \quad \forall j = 1, \ldots, p
$$

(3.61)

and

$$
[u_c]_{ij} = u_{c_s}, \quad u_{c_s} \in \mathbb{R}, \quad \forall i = 1, \ldots, t, \quad \forall j = 1, \ldots, l.
$$

(3.62)

This problem is referred to as the *sub-optimal scaling problem*, and it is defined as:

**Sub-optimal Scaling Problem:** Minimize $S(u_x, u_c)$ with respect to $u_x$ and $u_c$ and with respect to (3.61) and (3.62) i.e.

$$
\min_{(u_x, u_c)} \left\{ S(u_x, u_c) \right\}.
$$

(3.63)
The solution of (3.63) can be obtained from theorem 3.3 below.

**Theorem 3.3:**

A unique global solution to the sub-optimal scaling problem (3.63) is given by

\[
\begin{bmatrix}
  u_X \\
  u_C
\end{bmatrix} = \begin{bmatrix}
  u_{Xs} \\
  u_{Cs}
\end{bmatrix} = K_s^{-1} M_s
\]  

(3.64)

where,

\[
K_s := \begin{bmatrix}
  4\xi l(np)^2 + \alpha \ln p & 2\xi l(np)^2 + \alpha \ln p \\
  2\xi l(np)^2 + \alpha \ln p & \xi l(np)^2 + \alpha \ln p + t
\end{bmatrix}
\]  

(3.65)

\[
M_s := \begin{bmatrix}
  U_{l \times np} \left( tlB_l \vec{v} \left( \sum_{i=1}^{\xi} L_{10} \left( D^T_i \right) \right) + (U_{l \times tl} \otimes B_l) \vec{v} \left( \sum_{i=1}^{\xi} (L_{E_i} C_i)^T \right) + \sum_{i=1}^{\omega} (L_{B_i A_i})^T U_{l \times 1} \right) \\
  U_{l \times tl} \left( npU_{l \times np} \vec{v} \left( \sum_{i=1}^{\xi} L_{10} \left( D^T_i \right) \right) + np \left( \sum_{i=1}^{\xi} (L_{E_i} C_i) \right) U_{np \times 1} + \sum_{i=1}^{\omega} (L_{B_i A_i} U_{np \times 1}) + \vec{v}(L_{10}(G)) \right)
\end{bmatrix}
\]  

(3.66)

and \(B_l\) is given by (3.35).

**Proof.**

The proof is constructive. The sufficient conditions [53] for the minimization problem (3.63) to have a global minimum point are

\[\nabla S(u_X, u_C) = 0\]  

(3.67)

and

\[\nabla^2 S(u_X, u_C) > 0\]  

(3.68)

where, \(\nabla S(u_X, u_C) \in \mathbb{R}^2\), \(\nabla^2 S(u_X, u_C) \in \mathbb{R}^{2 \times 2}\), and \(\nabla^2 S(u_X, u_C) > 0\).

By doing the algebra on the left hand side of (3.67), i.e. writing analytically the Frobenious norms, grouping appropriate terms considering (3.61) and (3.62), and differentiating with respect to \(\begin{bmatrix}
  u_X \\
  u_C
\end{bmatrix} = \begin{bmatrix}
  u_{Xs} \\
  u_{Cs}
\end{bmatrix}\), equation (3.69) below emerges.

\[\begin{bmatrix}
  u_{Xs} \\
  u_{Cs}
\end{bmatrix} = K_s^{-1} M_s = 0\]  

(3.69)

where \(K_s\) and \(M_s\) are given by (3.65) and (3.66) and (3.67) respectively. Hence (3.67) is equivalent to (3.69) and this is an immediate consequence of the operator \(\nabla \cdot\) in (3.67) and nothing more.
Furthermore, $\nabla^2 S(u_X, u_C) = \nabla \nabla S(u_X, u_C)$ and from the left hand side of (3.69) it follows that $\nabla \nabla S(u_X, u_C) = \nabla \left( K_s \begin{bmatrix} u_X \\ u_C \end{bmatrix} - M_s \right) = K_s$. Therefore,

$$\nabla^2 S(u_X, u_C) = K_s$$

(3.70)

Now, since $K_s = K_s^T$, $4\xi_l(np)^2 + \alpha l \ln p > 0$, $\xi_l(np)^2 + \alpha l \ln p + t_l > 0$ and

$$\det(K_s) = \det \begin{bmatrix} 4\xi_l(np)^2 + \alpha l \ln p & \xi_l(np)^2 + \alpha l \ln p \\ 2\xi_l(np)^2 + \alpha l \ln p & \xi_l(np)^2 + \alpha l \ln p + t_l \end{bmatrix}$$

$$= \left( 4\xi_l(np)^2 + \alpha l \ln p \right) \left( \xi_l(np)^2 + \alpha l \ln p + t_l \right) - \left( 2\xi_l(np)^2 + \alpha l \ln p \right)$$

$$= \left( 4\xi_l(np)^2 + \alpha l np \right)(t_l)^2 > 0,$$

it follows that $K_s > 0$. Hence, from (3.70), (3.68) follows. Moreover, $K_s > 0$ implies that $K_s^{-1}$ exists and is unique. Hence (3.69) has a unique solution given by (3.64). Therefore, since (3.67) and (3.38) are satisfied under a unique solution point given by (3.64), this point is the global minimum of the minimization problem (3.63). 

If the additional constraint of minimizing the difference between the magnitudes of the coefficients of the original equation (3.31), is introduced, the resulting problem is referred to as the sub-optimal scaling problem with magnitude variation (3.71), and is stated next.

**Sub-optimal Scaling Problem with Magnitude Variation:** Minimize $S_v(u_X, u_C)$ with respect to $u_X$ and $u_C$ and with respect to (3.61) and (3.62) i.e.

$$\min_{(u_X, u_C)} \left\{ S_v(u_X, u_C) \right\}. \tag{3.71}$$

For problem (3.71), as with problem (3.60), there is no guarantee that a minimum will always exist. Hence, whether a solution to (3.71) is unique or not, depends on the number of solutions of the resulting linear equation (similar to (3.69) for the problem (3.60)). Also, as in the case of the problem (3.60), the solution/solutions $(u_X, u_C) = (u_{X*}, u_{C*})$ are obtained by minimizing or maximizing (3.71) if respectively,
Chapter 3 Scaling & Homogeneous Projective Transformations

Finally, additional restrictions such as symmetric $\mathbf{u}_X$ and $\mathbf{u}_C$, when $n = p$ and $t = l$, can be posed in many ways. For example, after solving (3.31), or (3.60), then the upper triangular parts of $\mathbf{u}_X$ and $\mathbf{u}_C$ are set equal to their lower triangular parts, in order to create symmetry. This arbitrary setting is referred to as symmetric scaling. Again, for the last case there is no guarantee for the existence of a minimum solution point. Hence this should be tested a posteriori as in problem (3.60).

Considering all the above, the reader should bare in mind, that each scaling problem has its own different objective function, hence a straightforward comparison between the previous variations of the method cannot be mathematically posed. The only thing one can do is to make posteriori empirical comparisons for a particular numerical problem, in terms of a common feature of the numerical process for the solution of (3.2). Such a feature could be, for example, the computational time or the accuracy of the computed solution of (3.2).

The results on $K > 0$ and $K_s > 0$ (i.e. existence of a global minimum solution) in the problems (3.31) and (3.63) respectively, are novel and are reported for the first time in this thesis. As far as the author is aware, such results are not reported anywhere in the literature for similar scaling techniques [182], [183]. Finally, it is reminded that, no matter which variation of scaling method is used, the original solution of (3.1) is recovered from (3.3).

3.2.4 Scalar Numerical Example

In this subsection, the scaling problems (3.31) and (3.60) are implemented for a scalar second order algebraic equation, for the sake of visualization. For all scalar problems, note that problem (3.63) is equivalent to (3.31) and problem (3.60) is equivalent to (3.71). Also the conditions of symmetric scaling do not apply, since $\mathbf{u}_X$ and $\mathbf{u}_C$ are scalars.

Consider equation (3.1) with data, $\omega = \xi = 1$, $t = n = p = l = 1$, $A_I = -2$, $B_I = D_I = E_I = 1$, $C_I = 10^{-201}$, $G = 2 \times 10^{201}$; that is, consider

$$10^{-201} x^2 - 2x + 2 \times 10^{201} = 0.$$ 

According to theorem 3.1, the solution to the problem (3.31) (i.e. (3.32)) and the global minimum are found to be

$$\begin{bmatrix} \mathbf{u}_X \\ \mathbf{u}_C \end{bmatrix} = \begin{bmatrix} 201.150514997832 \\ -201.351201661608 \end{bmatrix}, \quad S(\mathbf{u}_X, \mathbf{u}_C)(\mathbf{u}_X, \mathbf{u}_C) = (\mathbf{u}_X, \mathbf{u}_C) = 0.01510317638158.
Furthermore, \( S(u_X . u_C) \mid (u_X . u_C) = (0, 0) = 80923.19529637350 > S(u_X . u_C) \mid (u_X . u_C) = (u_X . u_C) \).

For problem (3.63) it is found that:
\[
\nabla S(u_X . u_C) = 0 \iff \begin{bmatrix} 11 & 3 \\ 3 & 3 \end{bmatrix} \begin{bmatrix} u_X \\ u_C \end{bmatrix} - \begin{bmatrix} 1608.60205999133 \\ -0.60205999133 \end{bmatrix} = 0.
\]
Since \( \begin{bmatrix} 11 & 3 \\ 3 & 3 \end{bmatrix} \) > 0, there is a global minimum solution. From the last equation, it is apparent that this solution is
\[
\begin{bmatrix} u_X * \\ u_C * \end{bmatrix} = \begin{bmatrix} 201.150514997832 \\ -201.351201661608 \end{bmatrix}.
\]
The minimum obtained is found to be
\[
S_v(u_X . u_C) \mid (u_X . u_C) = (u_X . u_C) = 0.06041270552630430.
\]
Also, \( S_v(u_X . u_C) \mid (u_X . u_C) = (0, 0) = 323692.4187092608 > S_v(u_X . u_C) \mid (u_X . u_C) = (u_X . u_C) \).

Geometric interpretations of problems (3.31) and (3.60), for the present example, are shown in figures 3.1-3.4. In these figures, the graphs on the left represent the original problem (i.e. minimization of \( S = S(u_X . u_C) \) in figures 3.1, 3.2, and of \( S_v = S_v(u_X . u_C) \) in figures 3.3, 3.4). The graphs on the right represent the minimization of \( S^{01} = S^{01}(u_X . u_C) \) in figures 3.1, 3.2, and the minimization of \( S_v^{01} = S_v^{01}(u_X . u_C) \) in figures 3.3, 3.4. The last representation is done in order to “intensify” the existence of the global minimum since
\[
S(u_X . u_C) \mid (u_X . u_C) = (u_X . u_C) > 1, S_v(u_X . u_C) \mid (u_X . u_C) = (u_X . u_C) > 1.
\]
For the above example, the solutions to problems (3.31) and (3.60) are calculated to be equal to each other.

Note that when (3.1) is not a scalar equation then it is impossible to construct graphs as in figures 3.1-3.4. Nevertheless, scaling can always be criticised numerically, by the minimum values of the objective functions \( S(u_X . u_C) \) obtained.

According to (3.2), and since both problems (3.31) and (3.60) have the same solution, there is one resulting scaled equation,
\[
0.8908987181402324 \bar{x}^2 - 1.259921049894845 \bar{x} + 0.8908987181404071 = 0,
\]
with solutions
\[
\bar{x}_1 = 0.7071067811866169 + 0.7071067811866169i,
\]
\[
\bar{x}_2 = 0.7071067811866169 - 0.7071067811866169i.
\]

The actual solutions to the original equation are \( x_1 = 10^{201} + 10^{201}i \), \( x_2 = 10^{201} - 10^{201}i \). For both scaling problems (3.31) and (3.60) the exact solutions are recovered as
\[
10^{201} \bar{x}_1 = 9.9999999999999999 \times 10^{200} + 9.9999999999999999 \times 10^{200}i.
\]
Figure 3.1. Geometric interpretation of problem (3.31) for $10^{-201}x^2 - 2x + 2 \times 10^{201} = 0$.

Figure 3.2. Contours $u_X, u_C$, of problem (3.31) for $10^{-201}x^2 - 2x + 2 \times 10^{201} = 0$. 
Figure 3.3. Geometric interpretation of problem (3.63) for $10^{-201}x^2 - 2x + 2 \times 10^{201} = 0$.

Figure 3.4. Contours $u_X$, $u_C$, of problem (3.63) for $10^{-201}x^2 - 2x + 2 \times 10^{201} = 0$. 
and
\[ 10^{12} x_2 = 9.999999999999999 \times 10^{200} - 9.999999999999999 \times 10^{200} i. \]

When calculating the original equation's solutions in a finite precision arithmetic machine, there is the danger of calculating zero solutions since they might be assumed zero with respect to the machine’s minimum floating point number (2.23 × 10^{-308} for the present computation). Similar hazards can be observed subject to overflows with respect to a machine’s maximum floating point number (1.79 × 10^{308} for the present computation). In the present example, state-of-the-art methods fail to compute the roots of the original equation.

More specifically, the function roots, in Matlab 5.3.0.10183 (R11) software (which is a robust eigenvalue-based solver [115]) for the present example provide 2 × 10^{201} and 0 as the roots of the original equation, which is obviously totally wrong. However, working with the solutions of the scaled equations, there is an error (arising from machine precision) of
\[ (-1.359713261610924 \pm 1.359713261610924) \times 10^{185}. \]

Nevertheless this is more convenient, since the resulting solutions at least preserve the nature of solutions (a pair of complex conjugate numbers).

Hence, the above example, clearly illustrates how scaling can prevent catastrophic numerical operations, subject to numerical underflows or overflows.

3.3 Homogeneous Projective Transformation

In this section, the problem of homogeneous projective transformation is presented. Initially, in section 3.3.1, the main result is stated for vector polynomial equations as in (2.2) and then in section 3.3.2 the result is adapted to the scaled equation (3.2) with \( t = n \) and \( l = p \). Note that (2.1) is a special case of (3.2) with \( u_X = 0 \) and \( u_C = 0 \). Hence, the results obtained for (3.2) with \( t = n \) and \( l = p \), will cover (2.1) as well. In section 3.3.3, a numerical example will illustrate the homogeneous projective transformation of (2.1). Finally, section 3.3.4 includes some useful discussions on the method.

3.3.1 Homogeneous Projective Transformation for Polynomial Vector Equations

Consider the polynomial vector equation,
\[ F(x) = 0, \ x \in \mathbb{C}^m, \ m \in \mathbb{Z}_+. \]  (3.72)
(3.72) is an algebraic polynomial system. Let the total degree* of (3.72) be \( d = \prod_{i=1}^{m} d_i \), where \( d_i \) is the total degree of the polynomial system being the \( i \)-th row element of (3.72). In the sequel, it is assumed that solution set \( S_x \) of (3.72) consists of a finite number of solutions (i.e. has measure zero†). Also, some of the solutions in \( S_x \) might be at infinity‡.

In order to include in the analysis the possibility of solutions at infinity, consider the algebraic polynomial system (3.73) below.

\[
\hat{F}(\hat{x}) = 0, \quad \hat{F}_{i1}(\hat{x}) = (L(\hat{x}))^{d_i} F_{i1} \left( \left( L(\hat{x}) \right)^{-1} \hat{x} \right), \quad i = 1, \ldots, m
\]

(3.73)

where \( \hat{x} \in \mathbb{C}^m, m \in \mathbb{Z}^+ \) and

\[
L(\hat{x}) := \mathbf{A}_a \hat{x} + \mathbf{a}_{(m+1)}, \quad (3.74)
\]

\[
\mathbf{A}_a := \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{m1} \end{bmatrix}. \quad (3.75)
\]

\( \forall \mathbf{A}_a \in \mathbb{C}^{1 \times m}, \mathbf{a}_{(m+1)} \in \mathbb{C}^n. \)

The polynomial system (3.73) can be viewed as the homogeneous algebraic polynomial system‡‡

\[
\hat{F}_{i1}(y) = y^{d_i}_{(m+1)} F_{i1} \left( \begin{bmatrix} y_{i1} \\ \vdots \\ y_{m1} \end{bmatrix} \right), \quad i = 1, \ldots, m
\]

(3.76)

where

\[
y := \begin{bmatrix} \hat{x} \\ L(\hat{x}) \end{bmatrix}, \quad y \in \mathbb{C}^{m+1}. \quad (3.77)
\]

Therefore, (3.76) is a homogenization of (3.72) from \( \mathbb{C}^m \) to \( \mathbb{C}^{m+1} \). Now, (3.73) is a projection of (3.76) from \( \mathbb{C}^{m+1} \) to \( \mathbb{C}^m \). Hence, (3.73) is referred as the homogeneous projective transformation of (3.72).

Now it can be proved, under the assumption of finiteness of the number of solutions for both (3.72) and (3.73), that (3.73) has no solutions at infinity almost everywhere. Also, there is a relationship, provided by analytical formulas, between the finite and infinite solutions of (3.72) with the finite solutions of (3.73). These results are presented and proved in theorem 3.4.

---

* For the definitions, total degree, solution at infinity, set of measure zero and homogeneous algebraic polynomial system, see definitions A.3.2, A.3.9, A.2.14 and A.3.1 respectively in appendix A.

†, ‡, ‡‡ For the definitions, total degree, solution at infinity, set of measure zero and homogeneous algebraic polynomial system, see definitions A.3.2, A.3.9, A.2.14 and A.3.1 respectively in appendix A.
Theorem 3.4:
Suppose that (3.72) and (3.73) each have finite number of solutions (i.e. their solution sets $S_x$ and $S_S$ are both finite and of measure zero). Then:

a) Each of $S_x$ and $S_S$ has exactly $d$ (the total degree of (3.72) and (3.73)) solutions counting multiplicities.

b) There is a 1-1 correspondence from the solutions of (3.73) to the solutions of (3.72), given as follows:
- Each finite nonsingular (singular) solution $\hat{x}_k \in S_S$, $k \in \mathbb{Z}_+$, $k \leq d$ with multiplicity $m_{\hat{x}_k}$, corresponds to the nonsingular (singular) solution $x_k \in S_x$ with multiplicity $m_{x_k} = m_{\hat{x}_k}$ such that $L(\hat{x}_k) \neq 0$.

$$ x_k = (L(\hat{x}_k))^{-1} \hat{x}_k. $$

- Each nonsingular (singular) solution $\hat{x}_k \in S_S$, $k \in \mathbb{Z}_+$, $k \leq d$ with multiplicity $m_{\hat{x}_k}$, corresponds to the nonsingular (singular) solution $x_k \in S_x$ at infinity with multiplicity $m_{x_k} = m_{\hat{x}_k}$ such that $L(\hat{x}_k) = 0$.

c) (3.73) has no solutions at infinity almost $\forall A_2 \in \mathbb{C}^{1 \times m}$ (i.e. with probability-1).

Proof.

a) By construction, both (3.72) and (3.73) are algebraic vector polynomial equations. Under the assumption that each has a finite number of solutions, result a) follows from Bezout’s theorem A.3.1 (see appendix A).

b) By construction it is obvious that, if $L(\hat{x}) = 0$, then $\bar{F}(\hat{x}) = 0 \iff \bar{F}(\hat{x}) = 0$; i.e. the solutions of (3.73) are identical to the solutions of the homogeneous equation of (3.72). Now, $\hat{x} = 0 \iff L(\hat{x}) = a_{l(m+1)}$ and since $a_{l(m+1)} \in \mathbb{C}$, it follows that $L(\hat{x}) \neq 0$, which is a contradiction to the hypothesis $L(\hat{x}) = 0$. Therefore, $\hat{x} \neq 0$ and $\bar{F}(\hat{x}) = 0$, which by the definition A.3.9 of a solution at infinity (see appendix A), imply that $\hat{x}$ corresponds to a solution of (3.72) $x$ at infinity. Since from a) there can be no more than $d$ solutions, the 1-1 relationship follows. At this point, note that nothing has been proven yet about whether $\hat{x}$ is at infinity or is a finite solution of (3.73). This is tackled in the proof of c).
Now, if \( L(\hat{x}) \neq 0 \), then it is apparent from the structure of (3.72) and (3.73) that 
\[
x = (L(\hat{x}))^{-1} \hat{x}.
\]
To prove the correspondence from the solutions of (3.73) to the solutions of (3.72) is equivalent to prove 
\[
\hat{x}_1 = \hat{x}_2 \Rightarrow x_1 = x_2, \ \forall \hat{x}_1, \hat{x}_2, x_1, x_2 \in C^m.
\]
This proof is shown next.

It is obvious from (3.74) and \( x = (L(\hat{x}))^{-1} \hat{x} \) that
\[
\hat{x}_1 = \hat{x}_2 \Rightarrow \Lambda_a \hat{x}_1 = \Lambda_a \hat{x}_2 \Rightarrow \Lambda_a \hat{x}_1 + \Lambda_l(m+1) = \Lambda_a \hat{x}_2 + \Lambda_l(m+1)
\]
\[
\Rightarrow L(\hat{x}_1) = L(\hat{x}_2) \Rightarrow L(\hat{x}_1) \hat{x}_1 = L(\hat{x}_2) \hat{x}_2
\]
\[
\Rightarrow L(\hat{x}_1) \hat{x}_1 = L(\hat{x}_2) \hat{x}_2. \tag{3.78}
\]
Now since \( x = (L(\hat{x}))^{-1} \hat{x} \), (3.78) imply \( x_1 = x_2 \).

Finally, since from a) there can be no more that \( d \) solutions, the 1-1 relationship follows.

Note that so far the above analysis is general \( \forall x \in S_x, \ \forall \hat{x} \in \hat{S}_x \), hence it holds \( \forall x_k \in S_x \) and \( \forall \hat{x}_k \in \hat{S}_x \), \( k \in \mathbb{Z}_+, \ k \leq d \).

Now, for \( \hat{x} = \hat{x}_k, \ \hat{x}_k \in \hat{S}_x, \ k \in \mathbb{Z}_+, \ k \leq d \) with multiplicity \( m_{\hat{x}_k} \), the corresponding \( x = x_k, \ x_k \in S_x \) has multiplicity \( m_{x_k} = m_{\hat{x}_k} \). This can easily be verified from the definition A.3.8 of the multiplicity (see appendix A). More specifically, since \( \hat{x}_k \in \hat{S}_x \) has multiplicity \( m_{\hat{x}_k} \) by assumption, it is clear that by a small random perturbation of the data of (3.73), \( \hat{x}_k \in \hat{S}_x \) resolves into a collection of \( m_{\hat{x}_k} \) nonsingular (because of the Sard’s density theorem A.4.2 in appendix A) solutions of (3.73) near \( \hat{x}_k \in \hat{S}_x \). Applying the same perturbation in (3.72), all the above analysis by construction provides the corresponding \( x_k \in S_x \) resolving into a collection of \( m_{x_k} = m_{\hat{x}_k} \) nonsingular solutions of (3.72) near \( x_k \in S_x \). Note that the above holds generally for nonsingular and singular solutions and \( \forall \Lambda_a \in C^{1 \times m} \).

To complete the proof of b), the only thing left to show is that each nonsingular solution of (3.73) corresponds to a nonsingular solution of (3.72), and that each singular solution of (3.73) corresponds to a singular solution of (3.72). This is done next.

If \( L(\hat{x}) \neq 0 \) then as proven previously, \( x = (L(\hat{x}))^{-1} \hat{x} \), and from (3.72) \( F_{ii}(x) = 0, \forall i = 1, \ldots, m \). Hence, from (3.73) it follows for \( i = 1, \ldots, m \) that
\[
\frac{d\left[F_{ij}\left(\hat{x}\right)\right]}{d\hat{x}^T} = \left[d\left[(L(\hat{x}))^{d_i} F_{ij}\left((L(\hat{x}))^{-1} \hat{x}\right)\right]\right]
\]

\[
= \left[d\left[(L(\hat{x}))^{d_i}\right]\right] F_{ij}\left((L(\hat{x}))^{-1} \hat{x}\right) + \left[(L(\hat{x}))^{d_i}\right] \frac{dF_{ij}}{d\hat{x}^T}\left((L(\hat{x}))^{-1} \hat{x}\right)
\]

\[
= \left[(L(\hat{x}))^{d_i}\right] d\left[F_{ij}\left((L(\hat{x}))^{-1} \hat{x}\right)\right] + \left[(L(\hat{x}))^{d_i}\right] \frac{dF_{ij}}{d\hat{x}^T}\left((L(\hat{x}))^{-1} \hat{x}\right)
\]

\[
= \left[(L(\hat{x}))^{d_i}\right] d\left[F_{ij}\left((L(\hat{x}))^{-1} \hat{x}\right)\right] \left[(L(\hat{x}))^{-1} \hat{x}\right] d\left[(L(\hat{x}))^{-1} \hat{x}\right]
\]

\[
= \left[(L(\hat{x}))^{d_i}\right] d\left[F_{ij}\left((L(\hat{x}))^{-1} \hat{x}\right)\right] \left[(-1)^2 \hat{x} A_a + (L(\hat{x}))^{-1} I_m\right]
\]

\[
= \left[(L(\hat{x}))^{d_i-2}\right] \frac{d[F_{ij}(x)]}{d\hat{x}^T}\left((L(\hat{x}))I_m - \hat{x} A_a\right).
\]

Hence,

\[
\frac{d\left[F(\hat{x})\right]}{d\hat{x}^T} = \begin{bmatrix}
\left[(L(\hat{x}))^{d_1}\right] & 0 & \cdots & 0 \\
0 & \left[(L(\hat{x}))^{d_2}\right] & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & \left[(L(\hat{x}))^{d_m}\right]
\end{bmatrix} \frac{d[F(x)]}{d\hat{x}^T}\left((L(\hat{x}))I_m - \hat{x} A_a\right)
\]

(3.79)

From (3.79) and \(L(\hat{x}) \neq 0\) it follows that

\[
\det\left(\frac{d\left[F(\hat{x})\right]}{d\hat{x}^T}\right) = 0 \iff \det\left(\frac{d[F(x)]}{d\hat{x}^T}\right)\det((L(\hat{x}))I_m - \hat{x} A_a) = 0.
\]

(3.80)

Also, by (3.74)

\[
(L(\hat{x}))I_m - \hat{x} A_a = \begin{bmatrix}
(L(\hat{x}) - \hat{x}_{11} A_{11}) & -\hat{x}_{11} A_{12} & \cdots & -\hat{x}_{11} A_{1m} \\
-\hat{x}_{21} A_{11} & (L(\hat{x}) - \hat{x}_{21} A_{22}) & \cdots & -\hat{x}_{21} A_{2m} \\
\vdots & \vdots & \ddots & \vdots \\
-\hat{x}_{m1} A_{11} & -\hat{x}_{m1} A_{21} & \cdots & (L(\hat{x}) - \hat{x}_{m1} A_{mm})
\end{bmatrix}
\]

(3.81)

If \(\hat{x} = 0\) (i.e. \(\hat{x}_{ij} = 0, \forall i = 1,\ldots,m\)) or \(A_a = 0\) (i.e. \(A_{ij} = 0, \forall i = 1,\ldots,m\)), then (3.81) gives

\[
(L(\hat{x}))I_m - \hat{x} A_a = A_{j(m+1)} I_m
\]

and because \(A_{j(m+1)} \in C_*\), \((L(\hat{x}))I_m - \hat{x} A_a\) is a nonsingular matrix \(\forall A_a \in C^{1 \times m}\). Now assume that some of \(\hat{x}_{ij}\)’s or some of \(A_{ij}\)’s are zero. Then dividing
each i-row corresponding to a nonzero $\hat{x}_{ij}$ with $\hat{x}_{ij}$ and dividing each i-column corresponding to a nonzero $a_{ij}$ with $a_{ij}$ and performing elementary operations on rows and columns of

$$(L(\hat{x})I_m - \hat{x}A_a)$$

appropriately, the following matrix results:

$$W_L := \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\frac{a_{1j}x_{1j}}{a_{1(m+1)}} & -\frac{a_{2j}x_{2j}}{a_{2(m+1)}} & -\frac{a_{3j}x_{3j}}{a_{3(m+1)}} & \cdots & 1 \end{bmatrix} \quad (3.82)$$

where $W_L \in \mathbb{C}^{n \times m}$. Note, that with respect to the above operations on $(L(\hat{x})I_m - \hat{x}A_a)$,

$$\det(L(\hat{x})I_m - \hat{x}A_a) = c \det(W_L) \text{ and } c \in \mathbb{C}.$$

From (3.82), it follows that, $\det(W_L) = 1 \neq 0$. Hence, $\det(L(\hat{x})I_m - \hat{x}A_a) \neq 0$; i.e.

$$(L(\hat{x})I_m - \hat{x}A_a) \text{ is a nonsingular matrix } \forall A_a \in \mathbb{C}^{l \times m}.$$

Since, $(L(\hat{x})I_m - \hat{x}A_a)$ is a nonsingular matrix $\forall A_a \in \mathbb{C}^{l \times m}$ and $\forall \hat{x} \in \mathbb{C}^m$, (3.80) becomes equivalently

$$\det \left( \frac{d[\hat{F}(\hat{x})]}{d\hat{x}^T} \right) = 0 \iff \det \left( \frac{d[F(x)]}{dx^T} \right) = 0 \Rightarrow \det \left( L(\hat{x})I_m - \hat{x}A_a \right) = 0 \iff \det \left( \frac{d[F(x)]}{dx^T} \right) = 0.$$ 

Therefore, it is proven that if $L(\hat{x}) \neq 0$ then $\forall A_a \in \mathbb{C}^{l \times m}$ and $\forall \hat{x} \in \mathbb{C}^m$

$$\det \left( \frac{d[\hat{F}(\hat{x})]}{d\hat{x}^T} \right) = 0 \iff \det \left( \frac{d[F(x)]}{dx^T} \right) = 0 \quad \text{and obviously } \det \left( \frac{d[\hat{F}(\hat{x})]}{d\hat{x}^T} \right) \neq 0 \iff \det \left( \frac{d[F(x)]}{dx^T} \right) \neq 0.$$

The last two equivalences, show that if $\hat{x} = \hat{x}_k$, $\hat{x}_k \in \hat{S}_\zeta$, $k \in \mathbb{Z}^+$, $k \leq d$ is a singular (nonsingular) solution of (3.73), then the corresponding $x = x_k$, $x_k \in \tilde{S}_\zeta$ is also a singular (nonsingular) solution of (3.72).

If $L(\hat{x}) = 0$, then $\hat{x} = \hat{x}_k$, $\hat{x}_k \in \hat{S}_\zeta$, $k \in \mathbb{Z}^+$, $k \leq d$ is a solution at infinity. By definition of such a solution (see definition A.3.9 in appendix A) it follows that, $\hat{x}_k \neq 0$ and $\tilde{F}(\hat{x}_k) = 0$, where $\tilde{F}(\hat{x}_k) = 0$ is the homogeneous equation (see definition A.3.5 in appendix A) of $\tilde{F}(\hat{x}_k) = 0$. Now, by construction of (3.72)-(3.75), it is apparent that
\[ L(\hat{x}) = A_\alpha \hat{x}, \quad \left( (L(\hat{x}))^{d_i} \right) = (L(\hat{x}))^{d_i} = (A_\alpha \hat{x})^{d_i} , \quad \left( (L(\hat{x}))^{-1} \right) = \left( (L(\hat{x}))^{-1} \right) = (A_\alpha \hat{x})^{-1} \text{ where, } \]

denotes as previously the homogeneous part of an algebraic vector polynomial. From the last equations it follows, for \( i = 1, \ldots, m \), that

\[ \tilde{F}_{ii}(\hat{x}) = \left( (L(\hat{x}))^{d_i} \right) F_{ii} \left( (L(\hat{x}))^{-1} \right) \hat{x} \]

\[ \iff \tilde{F}_{ii}(\hat{x}) = (A_\alpha \hat{x})^{d_i} F_{ii} \left( (A_\alpha \hat{x})^{-1} \right) \hat{x} \]

\[ \text{(3.83)} \]

Now, \( L(\hat{x}_k) = 0 \iff A_\alpha \hat{x}_k = -a_{l(m+1)} \). Hence, from (3.83) it follows that

\[ \tilde{F}_{ii}(\hat{x}_k) = (A_\alpha \hat{x}_k)^{d_i} F_{ii} \left( A_\alpha \hat{x}_k \right)^{-1} \hat{x}_k \iff \tilde{F}_{ii}(\hat{x}_k) = (-a_{l(m+1)})^{d_i} F_{ii} \left( (-a_{l(m+1)})^{-1} \hat{x}_k \right) \]

\[ \iff \tilde{F}_{ii}(\hat{x}_k) = \left( (-a_{l(m+1)})^{d_i+1} \right) \hat{x}_k. \quad \text{(3.84)} \]

Furthermore, from (3.84)

\[ \frac{d \left[ \tilde{F}(\hat{x}_k) \right]}{d \hat{x}_k} = \begin{bmatrix} d \left[ \tilde{F}_{11}(\hat{x}_k) \right] \\ d \left[ \tilde{F}_{21}(\hat{x}_k) \right] \\ \vdots \\ d \left[ \tilde{F}_{m1}(\hat{x}_k) \right] \end{bmatrix} = \begin{bmatrix} (-a_{l(m+1)})^{d_1+1} & 0 & \cdots & 0 \\ 0 & (-a_{l(m+1)})^{d_2+1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & (-a_{l(m+1)})^{d_m+1} \end{bmatrix} \frac{d \left[ F(x_k) \right]}{d \hat{x}_k} \]

and since \( a_{l(m+1)} \in C_\infty \), the last equality gives

\[ \det \left( \frac{d \left[ \tilde{F}(\hat{x}_k) \right]}{d \hat{x}_k} \right) = 0 \iff \det \left( \frac{d \left[ F(x_k) \right]}{d \hat{x}_k} \right) = 0 \text{ and also, } \det \left( \frac{d \left[ \tilde{F}(\hat{x}_k) \right]}{d \hat{x}_k} \right) \neq 0 \iff \det \left( \frac{d \left[ F(x_k) \right]}{d \hat{x}_k} \right) \neq 0. \]

The last two equivalences, show that always \( \forall A_\alpha \in C^{1\times m} \), if \( \hat{x} = \hat{x}_k, \hat{x}_k \in S_{\hat{x}}, k \in Z_+ \), \( k \leq d \) is a singular (nonsingular) solution of (3.73), then the corresponding \( x = x_k \). \( x_k \in S_x \) is also a singular (nonsingular) solution of (3.72).

Hence part b) has been proven.

c) If \( \hat{x} = \hat{x}_k, k \leq d \) (from a) is a solution of (3.73) at infinity, then (3.83) holds. Now, \( \hat{x} = \hat{x}_k \) if \( A_\alpha \hat{x}_k \neq 0 \) then from (3.83), \( \tilde{F}(\hat{x}_k) = 0 \iff F \left( (A_\alpha \hat{x}_k)^{-1} \hat{x}_k \right) = 0 \). Obviously in the last case, the corresponding solution to \( x = x_k \) of (3.72) is given by \( x_k = (A_\alpha \hat{x}_k)^{-1} \hat{x}_k \). But with respect to b) it must be \( x_k = (A_\alpha \hat{x}_k + a_{l(m+1)})^{-1} \hat{x}_k \). Therefore, since both
$x_k = (A_a \hat{x}_k)^{-1} \hat{x}_k$ and $x_k = \left( A_a \hat{x}_k + a_{l(m+1)} \right)^{-1} \hat{x}_k$ must be valid, it follows that $a_{l(m+1)} = 0$, which is a contradiction since $a_{l(m+1)} \in C_2$.

Hence, it must be that $A_a \hat{x}_k = 0$. In this case from (3.83) it follows that $\tilde{F}(\hat{x}_k) = 0$ and therefore $\hat{x}_k \neq 0$ is a solution at infinity. But, $A_a \hat{x}_k = 0$ holds for $A_a \in S_{A_{a_k}}$ with $S_{A_{a_k}} := \left\{ A_a \in C^{1 \times m} \mid \sum_{i=1}^{m} x_{k_1}A_{a_1} = 0, \ x_k \neq 0 \right\}$ being a set of measure zero as a proper hyperplane in $C^{1 \times m}$ [123]. Since, $k \leq d$ (from a)) it is apparent that $S_{A_a} := \bigcup_k S_{A_{a_k}}$ has measure zero as well. Therefore, from the above it follows that (3.73) has solutions at infinity $\forall A_a \in S_{A_a}$, with $S_{A_a}$ having measure zero. Hence, almost $\forall A_a \in C^{1 \times m}$ (i.e. with probability-1), (3.73) has no solutions at infinity.

A similar result, with respect to a different structure of homogeneous projective transformation and under different assumptions can be found in [121]. In [121], there is a more conservative assumption of $A_a \neq 0$, and the proof of the similar results is different than the one in theorem 3.4. Finally, the strategy used in the present proof, can be used to prove the result in [121] as well. In this respect, note that the presentation of such a proof is not necessary for this work and hence is not reported in this thesis.

From theorem 3.4, it is clear that all the solutions of (3.72) can be obtained from the solutions of (3.73). Now, since (3.73) is proven to have no solutions at infinity with probability-1, then the homogeneous projective transformation (3.73) transforms the original equation (3.72) into an equation without any solutions at infinity. Of course, this is valid under the essential assumption of the finiteness of the number of solutions for both (3.72) and the homogeneous projective transformation (3.73). To this end, note that if the assumption that (3.72) has a finite number of solutions is removed, then $S_{A_a} := \bigcup_k S_{A_{a_k}}$ in the proof of c) in theorem 3.4, is not in general a set of measure zero. Hence, the probabilistic feature of c) in theorem 3.4 collapses.

As pointed out in [121], in practice it is easy to satisfy $S_{A_a}$ having measure zero. The main consideration here is the assumption of (3.73) having finite number of solutions. This condition might not always be true (e.g. cases of systems with a finite number of finite solutions and infinite number of solutions at infinity). Nevertheless, practical experience with real world problems shows that most of the time this is not the case [121].
So far the homogeneous projective transformation was presented for (3.72). Next its application
to (3.2), is presented.

3.3.2 Homogeneous Projective Transformation of (3.2) with \( t=n \) and \( l=p \)

Equation (3.2) with \( t=n \) and \( l=p \), can be converted equivalently into an algebraic polynomial
vector equation as shown below.

\[
\text{vec}(\tilde{f}(\tilde{x})) = \text{vec} \left[ E_{10}(U_C) \circ \left( \sum_{i=1}^{\omega} (A_i (E_{10}(U_X) \circ \tilde{x}) B_i) + \sum_{i=1}^{\xi} (C_i (E_{10}(U_X) \circ \tilde{x}) D_i (E_{10}(U_X) \circ \tilde{x}) E_i) + G \right) \right] = 0
\]

\[
\Leftrightarrow \text{vec}(\tilde{f}(\tilde{x})) = \text{vec} (E_{10}(U_C))
\]

\[
\Leftrightarrow \text{vec}(\tilde{f}(\tilde{x})) = \text{vec}(E_{10}(U_C))
\]

\[
\omega \text{vec} \left( \sum_{i=1}^{\omega} (A_i (E_{10}(U_X) \circ \tilde{x}) B_i) + \sum_{i=1}^{\xi} (C_i (E_{10}(U_X) \circ \tilde{x}) D_i (E_{10}(U_X) \circ \tilde{x}) E_i) + G \right) = 0
\]

From, (3.6)-(3.8) and defining

\[
\tilde{F}(\tilde{x}) := \text{vec}(\tilde{f}(\tilde{x}))
\]

(3.84) gives equivalently,

\[
\tilde{F}(\tilde{x}) = \text{vec}(E_{10}(U_C))
\]

(3.85)

\[
\left( \sum_{i=1}^{\omega} (B_i^T \otimes A_i) + \sum_{i=1}^{\xi} \left( (E_i^T \otimes C_i) (E_{10}(U_X) \circ \text{vec}^{-1}(\tilde{x})) D_i \right) \right) \text{vec}(E_{10}(U_X) \circ \tilde{x}) + \text{vec}(G) = 0
\]

(3.86)

Since \( t=n \) and \( l=p \), (3.86) can be viewed as a special case of (3.72) with, \( F(x) \leftarrow \tilde{F}(\tilde{x}) \),
\( x \leftarrow \tilde{x} \), \( x \in \mathbb{C}^m, m=np \) and with \( d=2np \). Hence, the homogeneous projective transformation
of (3.86), can be defined in view of (3.73) as

\[
\hat{F}(\hat{x}) := E_{10}(U_C)
\]

(3.87)

\[
\left( L(\hat{x}) \sum_{i=1}^{\omega} (B_i^T \otimes A_i) + \sum_{i=1}^{\xi} \left( (E_i^T \otimes C_i) (E_{10}(U_X) \circ \text{vec}^{-1}(\hat{x})) D_i \right) \right) \text{vec}(E_{10}(U_X) \circ \hat{x}) + L(\hat{x})^2 \text{vec}(G) = 0
\]

where \( \hat{x} := \text{vec}(\hat{X}) \), \( \hat{x} \in \mathbb{C}^{np} \)
\[
L(\hat{x}) = A_a \hat{x} + a_{f(m+l)},
\]
(3.88)

\[A_a\] is given by (3.75) and \(a_{f(m+l)} \in \mathbb{C}^s\).

Now theorem 3.4 in terms of (3.87) implies the following proposition.

**Proposition 3.2:**

Suppose that (3.86) and (3.87) both have a finite number of solutions (i.e. their solution sets \(\tilde{S}_x\) and \(\hat{S}_x\) are both finite and of measure zero). Then:

a) Each of \(\tilde{S}_x\) and \(\hat{S}_x\) has exactly \(d = 2^{mp}\) (the total degree of (3.86) and (3.87)) solutions.

b) There is a 1-1 correspondence from the solutions of (3.87) to the solutions of (3.86), given as follows:

- Each finite nonsingular (singular) solution \(\tilde{x}_k \in \tilde{S}_x\), \(k \in \mathbb{Z}^+\), \(k \leq d\) with multiplicity \(m_{\tilde{x}_k}\) corresponds to the nonsingular (singular) solution \(\bar{x}_k \in S_x\) with multiplicity \(m_{\bar{x}_k} = m_{\tilde{x}_k}\) such that \(L(\tilde{x}_k) \neq 0\).

\[\bar{x}_k = \left( L(\tilde{x}_k) \right)^{-1} \tilde{x}_k.\]

- Each finite nonsingular (singular) solution \(\hat{x}_k \in \hat{S}_x\), \(k \in \mathbb{Z}^+\), \(k \leq d\) with multiplicity \(m_{\hat{x}_k}\) corresponds to the nonsingular (singular) solution \(\hat{x}_k \in \hat{S}_x\) at infinity with multiplicity \(m_{\hat{x}_k} = m_{\hat{x}_k}\) such that \(L(\hat{x}_k) = 0\).

c) \(3.87\) has no solutions at infinity almost everywhere \(\forall A_a \in \mathbb{C}^{l \times m}\) (i.e. with probability-1).

**Proof.**

The proof is an immediate consequence of theorem (3.4), in view of (3.86) and (3.87).

From the above, it is obvious that the homogeneous projective transformation of (3.2) with \(t = n\) and \(l = p\), can be defined as

\[
\hat{f} \left( \hat{X} \right) = \text{vec}^{-1} \left( \hat{F} (\hat{x}) \right),
\]
(3.89)

Furthermore from (3.84) it follows that
Now, because of (3.88), the results of proposition (3.2) imply the following proposition.

**Proposition 3.3:**

Let \( t = n \) and \( l = p \), and suppose that (3.2) and (3.89) both have a finite number of solutions (i.e. their solution sets \( S^* \) and \( S_\infty \) are both finite and of measure zero). Then,

a) Each of \( S^* \) and \( S_\infty \) has exactly \( d = 2np \) (the total degree of (3.2) and (3.89)) solutions.

b) There is a 1-1 correspondence from the solutions of (3.89) to the solutions of (3.2), given as follows:

- Each finite nonsingular (singular) solution \( \hat{X}_k \in S^*_\hat{X} \), \( k \in \mathbb{Z}_+ \), \( k \leq d \) with multiplicity \( m_{\hat{X}_k} \), corresponds to the nonsingular (singular) solution \( \bar{X}_k \in S_{\bar{X}} \) with multiplicity \( m_{\bar{X}_k} = m_{\hat{X}_k} \) such that \( L(\hat{X}_k) \neq 0 \),

\[
\bar{X}_k = \left( L(\hat{X}_k) \right)^{-1} \text{vec}(\hat{X}_k).
\]

- Each finite nonsingular (singular) solution \( \hat{X}_k \in S_{\hat{X}} \), \( k \in \mathbb{Z}_+ \), \( k \leq d \) with multiplicity \( m_{\hat{X}_k} \), corresponds to the nonsingular (singular) solution \( \bar{X}_k \in S_{\bar{X}} \) at infinity with multiplicity \( m_{\bar{X}_k} = m_{\hat{X}_k} \) such that \( L(\hat{X}_k) = 0 \).

\[
X_k \in S_{\bar{X}} \text{ at infinity with multiplicity } m_{X_k} = m_{\hat{X}_k} \text{ such that } L(\hat{X}_k) = 0.
\]

c) (3.89) has no solutions at infinity almost \( \forall \mathbf{A}_d \in C^{l \times m} \) (i.e. with probability-1).

**Proof.**

The proof is obvious from proposition (3.2).
setting $L(\hat{x})=1$ and $U_X = 0$ and $U_C = 0$ for (2.1). Now, solving (3.89), the finite solutions to (3.2) and (2.1) can be recovered from

$$\bar{X} = \left(L(\hat{x})\right)^{-1} \hat{X}$$

and

$$X = \left(L(\hat{x})\right)^{-1} \left(E_1^0(U_X) \circ \hat{X}\right)$$

respectively. While the cases where $L(\hat{x})=0$, correspond to solutions at infinity.

### 3.3.3 Numerical Example of the Homogeneous Projective Transformation for (2.1)

Consider (2.1) with data, $\omega = 1$, $\xi = 2$, $n = 2$, $p = 1$, $A_I = \begin{bmatrix} 1 & -1 \\ 0 & 0 \end{bmatrix}$, $B_I = 1$, $C_I = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}$, $D_I = [0 \ 1]$, $E_2 = -2$, $C_2 = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}$, $D_2 = [1 \ 1]$, $E_1 = 1$, $G = [0 \ -4]$, $X := x = \begin{bmatrix} x_1 | x_2 \end{bmatrix}$.

Hence

$$F(x) = f(X) := \begin{bmatrix} f_{11}(x) \\ f_{21}(x) \end{bmatrix} = A_I X B_I + C_I X D_I X E_I + C_2 X D_2 X E_2 + G = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

(3.93)

It is apparent that (3.93) is the algebraic polynomial system of the straight line $f_{11}(x) = x_{11} - x_{21} = 0$ and the circle $f_{21}(x) = x_{11}^2 + x_{21}^2 - 4 = 0$. In general, a system of two conics in two variables (i.e. in the plane) can have at the most four real finite solutions [91]; these are their intersection points. In the case of (3.93), there are two real solutions $x_1 := \begin{bmatrix} x_{11} \\ x_{12} \end{bmatrix}$ and $x_2 := \begin{bmatrix} x_{21} \\ x_{22} \end{bmatrix}$ at the intersection points $(x_{11}, x_{12}) = (-\sqrt{2}, -\sqrt{2})$ and $(x_{21}, x_{22}) = (\sqrt{2}, \sqrt{2})$ of the line with the circle, and there are no complex solutions.

Assuming that there is a finite number of solutions (e.g. $2np = 2^2 = 4$ solutions), the other two solutions $x_3 := \begin{bmatrix} x_{31} \\ x_{32} \end{bmatrix}$ and $x_4 := \begin{bmatrix} x_{41} \\ x_{42} \end{bmatrix}$ are at infinity. Using proposition 3.2, the projective transformation for (3.93) results in

$$F(\hat{x}) = \hat{f}(\hat{X}) := \begin{bmatrix} \hat{f}_{11}(\hat{X}) \\ \hat{f}_{21}(\hat{X}) \end{bmatrix} = A_I (L(\hat{x}) \hat{X}) B_I + C_I \hat{X} D_I \hat{X} E_I + C_2 \hat{X} D_2 \hat{X} E_2 + (L(\hat{x}))^2 G = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

(3.94)
where \( L(\hat{x}) = \begin{bmatrix} -0.95013 & -0.23114 \end{bmatrix}\hat{x} - 0.60684 \) with coefficients randomly chosen via MATLAB’s V.5.3 pseudo-random generator \( \text{rand} \). Now, (3.94) is the algebraic polynomial system of the two conics

\[
\hat{f}_{11}(\hat{x}) = -0.95013\hat{x}_{11}^2 + 0.23114\hat{x}_{21}^2 + 0.71899\hat{x}_{11}\hat{x}_{21} - 0.60684\hat{x}_{11} + 0.60684\hat{x}_{21} = 0
\]

and

\[
\hat{f}_{21}(\hat{x}) = -2.6109880676\hat{x}_{11}^2 + 0.7862972016\hat{x}_{21}^2 - 1.7569043856\hat{x}_{11}\hat{x}_{21}
\]

\[-4.6126151136\hat{x}_{11} - 1.1221198828\hat{x}_{21} - 1.4730191424 = 0.\]

According to proposition 3.2, the assumption of (3.94) having a finite number of solutions is required. In reality this is the case since (3.94) has four finite solutions: two real \( \hat{x}_1 := \begin{bmatrix} \hat{x}_{11} \\ \hat{x}_{12} \end{bmatrix} \), \( \hat{x}_2 := \begin{bmatrix} \hat{x}_{21} \\ \hat{x}_{22} \end{bmatrix} \), at the intersection points of the two conics, and two complex solutions \( \hat{x}_3 := \begin{bmatrix} \hat{x}_{31} \\ \hat{x}_{32} \end{bmatrix} \), \( \hat{x}_4 := \begin{bmatrix} \hat{x}_{41} \\ \hat{x}_{42} \end{bmatrix} \). These solutions, have been calculated via MATLAB’s V.5.3 Symbolic Computation toolbox as

\[
\begin{align*}
(\hat{x}_{11}, \hat{x}_{12}) &= (-1.2798124694668606252151625928291, -1.2798124694668606252151625928291) \\
(\hat{x}_{21}, \hat{x}_{22}) &= (-0.32135353982321101901540348501873, -0.32135353982321101901540348501873) \\
(\hat{x}_{31}, \hat{x}_{32}) &= (-0.60300495846662238453443677609891 + 0.14669420616123593398933800261807i, -0.14669420616123593398933800261807i) \\
(\hat{x}_{41}, \hat{x}_{42}) &= (-0.60300495846662238453443677609891 - 0.14669420616123593398933800261807i, -0.14669420616123593398933800261807i)
\end{align*}
\]

Now, \( \hat{x}_3, \hat{x}_4 \) are solutions at infinity since \( L(\hat{x}_k) = 0 \) for \( k = 3,4 \). Also, the finite solutions \( \hat{x}_1, \hat{x}_2 \) correspond to the finite solutions \( x_1, x_2 \) respectively and \( x_k = (L(\hat{x}_k))^{-1}\hat{x}_k \) for \( k = 1,2 \).

Hence, proposition 3.2 has been illustrated through the present example. We emphasize again the a priori assumption that each of (3.93) and (3.94) has a finite number of solutions (actually four solutions each). In this example, these assumptions do as they hold in a wide range of practical problems and as numerical experience [121] shows.

### 3.3.4 Useful Comments and Observations

In the previous sections the homogeneous projective transformation has been synthesized for general total degree algebraic vector equations and for the equations (3.2) and (2.1). It became apparent, that the method could be used independently of the numerical solution method used for the solution of the respective equations. It is now more clear how solutions at infinity are not
a problem in terms of calculations. This will be established with the probability-1 homotopy theorems in the next section. The only major assumptions for a successful (with probability-1) homogeneous projective transformation, are the assumptions on the finiteness of the number of solutions of the equations to which it is applied and the probabilistic assumption on the coefficients of (3.74). If these assumptions are not satisfied, there is no guarantee with probability-1 that a numerical method, in which the homogeneous projective transformation is invoked, will be robust. But, as it will be shown via numerical benchmarks in later chapters, even when some of the assumptions, subject to finiteness of solutions, is violated, the probability-1 homotopy methods of the next section still remain efficient.

A drawback of the homogeneous projective transformation is that it can produce an ill conditioned problem. This problem is a function of the numerical method used for the numerical solution of the respective equations as well as the numerical conditioning of the equations themselves. In the case of homotopy methods and particularly the methods of p[182], [183] for ill conditioned problems, it was observed that the use of homogeneous projective transformation damaged the numerical accuracy of the solutions. Hence, extra care should be taken when using the homogeneous projective transformation in situations like the above.

### 3.4 Probability-1 Homotopies with Scaling and Homogeneous Projective Transformation for (2.1)

It was stated earlier, when the probability-1 homotopy algorithms of chapter 2 are used for the numerical solution of (2.1), scaling and the homogeneous projective transformation can be useful. Especially when (2.1) has solutions at infinity and the algorithms try to compute such solutions, the homogeneous projective transformation transforms the original equation into an equation with no solutions at infinity.

Now, the homogeneous projective transformation should take place on the homotopy equations only in the original equation or in both the original equation and the easy problem. Both, settings are valid from the mathematical viewpoint. Particularly the last setting, (i.e. with homogeneous projective transformation applied in both the original and easy problem equations), may sometimes result in a homotopy equation with the original problem being close to the easy problem in terms of structure. Therefore, this may result in homotopy paths with small arc lengths compared to the case of a non transformed easy problem equation. Of course

---

† The author would like to thank professor L. T. Watson (one of the developers of [182], [183]), for bringing to his attention ill conditioning problems of homogeneous projective transformations, observed in practice via Hompack software [182], [183], and for the fruitful discussions about the present method, parts of which were presented in [171].
Chapter 3 Scaling & Homogeneous Projective Transformations

This is a heuristic assumption, but the author’s computational experience with the problem shows that for the majority of problems tested this assumption has been vindicated. For only this reason, the software code has been designed with projective transformation to take place in both the original and the easy problems. More discussions on this matter will take place in chapter 5, where the software design will be presented.

Now, the scaling applied in the original equation only. Note that the structure of the easy problem will in general chosen in random.

Note that, the original equation under scaling and homogeneous projective transformation is namely equation (3.90) in section 3.3. Hence, the probability-1 homotopy theorems for (3.90), will be defined and new probability-1 homotopy theorems will be established in section 3.4.1. Finally, the tracking of homotopy paths and the numerical solution to (3.90), will be reported in sections 3.4.2 and 3.4.3. Sections 3.4.1 and 3.4.2 are similar in presentation to sections 2.3.1 and 2.3.2 in chapter 2. Hence, for details about the technicalities in probability-1 homotopy theorems and relevant discussions, the reader should refer to chapter 2.

### 3.4.1 Probability-1 Homotopy Theorems with Scaling and the Homogeneous Projective Transformation for (2.1)

With respect to (3.90) the probability-1 homotopy theorems, similar to 2.1 and 2.2, are as follows.

**Theorem 3.5 (probability-1 fixed point polynomial homotopy with scaling and homogeneous projective transformation):**

Let equations (2.1) and (3.90) have finite solution sets \( S_X \) and \( S_{\tilde{X}} \) respectively. Let \( \hat{H}_{\alpha \beta} \) be a homotopy mapping similar to definition 2.2 in chapter 2. That is

\[
\hat{H}_{\alpha \beta}(\tilde{x}, \varepsilon) = \varepsilon \hat{F}(\tilde{x}) - \gamma (1-\varepsilon) \hat{F}_0(\tilde{x})
\]

where \( \gamma \in \mathbb{C} \) and \( \hat{F}(\tilde{x}) \) is given by (3.87), and let the easy problem be either

\[
\hat{F}_0(\tilde{x}) = \alpha \odot \tilde{x} - \beta
\]

or

\[
\hat{F}_0(\tilde{x}) = \alpha \odot (L(\tilde{x})\tilde{x}) - (L(\tilde{x}))^2 \beta.
\]

Furthermore, suppose that \( \hat{H}_{\alpha \beta} \) is \( C^2 \) and complex analytic in \( x \). Then there are sets \( S_{\alpha \beta} \subset C^{np} \times C^{np} \) and \( S_{A_\beta} \subset C^{1 \times m} \), both having measure zero, such that

\[
\forall (\alpha, \beta) \in \left\{ C^{np} \times C^{np} \right\} - S_{\alpha \beta} \quad \text{and} \quad \forall A_\beta \in C^{1 \times m} - S_{A_\beta}.
\]
\( \hat{H}_{\alpha \beta}^{-1}(0) = \left\{ (\hat{x}, \varepsilon) \in \mathbb{C}^{np} \times [0,1] \mid \hat{H}_{\alpha \beta}(\hat{x}, \varepsilon) = 0 \right\} \) contains a unique smooth homotopy path (curve) which connects the zero solution \((\beta + \alpha)\) of (3.96) (if (3.96) is used in 3.95)) or the zero solution \(A_{(np+1)}(I_{np} - (\beta + \alpha)A_a)^{-1}(\beta + \alpha)\) of (3.97), (if (3.97) is used in 3.95)) to one solution of (3.90) and with the problems (i)-(vi) in chapter 2 not applying for this particular path. Furthermore, the solution of (2.1), if it is finite, is recovered from (3.92), while the case where the solution to (3.92) satisfies \(L(\hat{x}) = 0\), corresponds to a solution of (2.1) at infinity.

**Proof.**

The proof of this result is immediate from theorem (2.1) in chapter 2 and proposition 3.3, since the results proven there hold in exactly the same way for both (3.96) and (3.97). When (3.96) is used in (3.95) the present theorem can be considered a special case of theorem (2.1) in chapter 2.

**Theorem 3.6 (probability-1 polynomial homotopy with scaling and homogeneous projective transformation):**

Let equations (2.1) and (3.90) have finite (i.e. with measure zero) solution sets \(S_X\) and \(S_X^{-}\) respectively. Assume that (3.90) has \(r \leq 2np\) geometrically isolated solutions. Let the homotopy mapping, similar to definition 2.2 in chapter 2, be

\[
\hat{H}_{\alpha \beta}(\hat{x}, \varepsilon) = \varepsilon \hat{F}(\hat{x}) - (1 - \varepsilon)\hat{F}_0(\hat{x})
\]

(3.98)

where \(\gamma \in \mathcal{C}_a\) and \(\hat{F}(\hat{x})\) is given by (3.87), and let the easy problem be either

\[
\hat{F}_0(\hat{x}) = \alpha \circ \hat{x} \circ \hat{x} - \beta
\]

(3.99)

or

\[
\hat{F}_0(\hat{x}) = \alpha \circ \hat{x} \circ \hat{x} - (L(\hat{x}))^2 \beta.
\]

(3.100)

Furthermore, suppose that \(\hat{H}_{\alpha \beta}\) is \(C^2\) and complex analytic in \(x\). Then there are sets \(S_{\alpha \beta} \subset \mathbb{C}^{np} \times \mathbb{C}^{np}\) and \(S_{A_a} \subset \mathbb{C}^{l \times m}\), both having measure zero, such that

\[
\forall (\alpha, \beta) \in \left\{ \{C^{np} \times C^{np}\} - S_{\alpha \beta} \right\} \quad \text{and} \quad \forall A_a \in \mathbb{C}^{l \times m} - S_{A_a}.
\]

\(\hat{H}_{\alpha \beta}^{-1}(0) = \left\{ (\hat{x}, \varepsilon) \in \mathbb{C}^{np} \times [0,1] \mid \hat{H}_{\alpha \beta}(\hat{x}, \varepsilon) = 0 \right\}\) contains smooth homotopy paths (curves) connecting \(r\) geometrically isolated solutions of \(\hat{F}_0(\hat{x}) = 0\) with the \(r\) geometrically isolated
solutions (counting multiplicities) of $\tilde{F}(\tilde{x}) = 0$ in 1-1 relationship; and connecting $r - 2^{np}$ non-geometrically isolated solutions of $\tilde{F}_0(\tilde{x}) = 0$ with the $r - 2^{np}$ non-geometrically isolated solutions and in 1-1 relationship. For all these particular paths, problems (i)-(vi) in chapter 2 do not apply. Furthermore, the finite solutions of (2.1) are recovered from (3.92) while the cases where solutions of (3.92) satisfy $L(\tilde{x}) = 0$, correspond to solutions of (2.1) at infinity.

**Proof.**

The proof of this result is immediate from theorem (2.2) in chapter 2 and proposition 3.3, since the results proven there hold in exactly the same way for both (3.99) and (3.100). When (3.99) used in (3.98) the present theorem can be considered a special case of theorem (2.2) in chapter 2. 

With the theorems above, the main goal i.e. to design an appropriate homotopy mapping in which homotopy paths will not diverge to infinity, has been achieved. Note also that the theorems hold in a probabilistic sense (with probability-1), under the satisfaction of their assumptions. The major assumptions in these theorems are the finiteness of the number of solutions for (2.1) and (3.90). For many practical problems these assumptions are very often satisfied as pointed out in [121]. Nevertheless, even when some of these assumptions are not satisfied, computational experience shows that the above theorems are still able to calculate some solutions, justifying the robustness of the method.

### 3.4.2 Tracking of Homotopy Paths of Theorems (3.5) and (3.6)

For theorems (3.5) and (3.6), all the technical results and much of the discussion of section 2.3.2 and 2.3.4, are straightforwardly verified with respect to (3.95) and (3.98).

First of all, discrete tracking methods are obvious to apply. For a continuous tracking method, an initial value problem (IVP) similar to (2.40) is defined as

$$\begin{cases}
\frac{d(\tilde{x}(\varepsilon))}{d\varepsilon} = \left( \frac{\partial \hat{H}_a f(\tilde{x}(\varepsilon),\varepsilon)}{\partial \tilde{x}^T(\varepsilon)} \right)^{-1} \frac{\partial \hat{H}_a f(\tilde{x}(\varepsilon),\varepsilon)}{\partial \varepsilon} \\
\tilde{x}(0) \in C^{np}, \quad F_0(\tilde{x}(0)) \equiv F_0(\tilde{x}(0),\alpha,\beta) = 0
\end{cases}$$

In view of (3.101), the desirable homotopy paths $\mu$ in theorems 3.5, 3.6 can be viewed as trajectories of the IVP (3.101), emanating from $\varepsilon = 0$ and terminating at $\varepsilon = 1$. For given easy problem data, the IVP initial condition $\tilde{x}(0)$ is determined as the easy problem solution. This is
the unique solution $\hat{x}(0) = a_{l(np+1)}^{-1}(I_{np} - (\beta + \alpha)A_a)^{-1}(\beta + \alpha)$ in view of theorem 3.5 and the

$2np$ solutions $\hat{x}(0) = \pm a_{l(np+1)}^{-1}(I_{np} + (\beta + \alpha)^2 A_a)^{-1}(\beta + \alpha)^2$ in view of theorem 3.6. With

respect to each $\hat{x}(0)$, the IVP (3.101) has a unique solution. Hence solutions to the original

equation $\hat{F}(\hat{x}) = 0$ given by (3.87), can be obtained by integrating (3.101) from $\varepsilon = 0$ to $\varepsilon = 1$.

In other words, the solutions obtained are given by

$$V \int_{x(0)}^{x(1)} d\hat{x}(\varepsilon)$$

$$= \frac{1}{2} \left[ \frac{\partial}{\partial \varepsilon} \frac{\partial}{\partial \varepsilon} \right]^{-1} \left[ \frac{\partial}{\partial \varepsilon} \frac{\partial}{\partial \varepsilon} \right] d\varepsilon$$

where $\hat{x}(0) = a_{l(np+1)}^{-1}(I_{np} - (\beta + \alpha)A_a)^{-1}(\beta + \alpha)$ for theorem 3.5

and $\hat{x}(0) = \pm a_{l(np+1)}^{-1}(I_{np} + (\beta + \alpha)^2 A_a)^{-1}(\beta + \alpha)^2$ for theorem 3.6.

For (3.101), note that $\hat{x}(0)$ tends to provide geometrically isolated solutions for

$\hat{F}_0(\hat{x}) = 0$ given by either (3.96) or (3.97) in theorem 3.5 and (3.99) or (3.100) in theorem 3.6.

This nature of the $\hat{x}(0)$ solutions is apparent from (3.96), (3.97), (3.99), (3.100) and it holds

$\forall (\alpha, \beta) \in \{C_u^{np} \times C_u^{np}\} - S_{\alpha \beta}$ and $\forall A_a \in C_{1 \times m} - S_{A_a}$, with $S_{\alpha \beta}$ and $S_{A_a}$ both having

measure zero; i.e. $\hat{x}(0)$ are geometrically isolated with probability-1.

Finally, it is reminded that the homotopy paths $\mu$ are $C^1$ (continuously differentiable).

Now, since $\hat{H}_{\alpha \beta}$ is complex analytic by assumption, and $\hat{H}_{\alpha \beta}(\hat{x}(\varepsilon), \varepsilon) = 0$ we can write

$$\hat{H}_{\alpha \beta}(\hat{x}(\varepsilon), \varepsilon) = \hat{u}_{\alpha \beta}(\text{Re}(\hat{x}(\varepsilon)), \text{Im}(\hat{x}(\varepsilon)), \varepsilon) + \hat{v}_{\alpha \beta}(\text{Re}(\hat{x}(\varepsilon)), \text{Im}(\hat{x}(\varepsilon)), \varepsilon)i = 0 + 0i$$

where

$$\hat{u}_{\alpha \beta}(\hat{x}(\varepsilon), \varepsilon) := \text{Re}(\hat{H}_{\alpha \beta}(\hat{x}(\varepsilon), \varepsilon))$$

$$\hat{v}_{\alpha \beta}(\hat{x}(\varepsilon), \varepsilon) := \text{Im}(\hat{H}_{\alpha \beta}(\hat{x}(\varepsilon), \varepsilon)).$$

From complex analysis [2], [158] it is known that (3.103), in view of (3.104) and (3.105), can be equivalently considered as the real system
\[
\hat{P}_{\alpha\beta}(\hat{x}_r(\epsilon), \epsilon) := \begin{bmatrix}
\hat{u}_{\alpha\beta}(\text{Re}(\hat{x}(\epsilon)), \text{Im}(\hat{x}(\epsilon)), \epsilon) \\
\hat{v}_{\alpha\beta}(\text{Re}(\hat{x}(\epsilon)), \text{Im}(\hat{x}(\epsilon)), \epsilon)
\end{bmatrix} = 0
\] (3.106)

where \( \hat{x}_r(\epsilon) := \begin{bmatrix} \text{Re}(\hat{x}(\epsilon)) \\ \text{Im}(\hat{x}(\epsilon)) \end{bmatrix} \) and \( \hat{x}_r \in \mathbb{R}^{2m} \).

Now, in order to consider a pseudo-arc length \( s \) parameterization (see chapter 2) of the homotopies (3.95) and (3.98) in theorems 3.5 and 3.6, the function \( s = f(\epsilon) \) similar to (2.43) is evaluated as

\[
s = \int_0^1 \frac{1}{\left\| \frac{d[\hat{x}_r(\epsilon(s))]}{d[\epsilon(s)]} \right\|_2^2 + 1} \frac{1}{2} d[\epsilon(s)].
\] (3.107)

Since \( \hat{x} = \hat{x}(\epsilon(s)) = \hat{x}(s) \) and \( \epsilon = \epsilon(s) \), the respective homotopy mappings are parameterized identically in \( s \) as \( \hat{P}_{\alpha\beta}(\hat{x}_r(\epsilon(s)), \epsilon(s)) = 0 \). Also, note that \( \hat{P}_{\alpha\beta} \) is \( C^2 \), transversal to zero, and the mapping

\[
\hat{x}_r : \mathbb{R}_+ \cup \{0\} \rightarrow \mathbb{R}^{2m}, \quad \hat{P}_{\alpha\beta}(\hat{x}_r(\epsilon(s)), \epsilon(s)) = 0,
\]

\[
\hat{x}_r = \hat{x}_r(\epsilon(s)) := \begin{bmatrix} \text{Re}(\hat{x}(\epsilon(s))) \\ \text{Im}(\hat{x}(\epsilon(s))) \end{bmatrix} = \hat{x}_r(s) = \begin{bmatrix} \text{Re}(\hat{x}(s)) \\ \text{Im}(\hat{x}(s)) \end{bmatrix},
\]

is \( C^1 \).

Hence, the IVP similar to (2.47) is defined by

\[
\begin{cases}
J_{\hat{\alpha}\hat{\beta}}(\hat{P}_{\alpha\beta}(\hat{x}_r(\epsilon(s)), \epsilon(s))) \begin{bmatrix}
\frac{d[\hat{x}_r(\epsilon(s))]}{d[\epsilon(s)]} \\
\frac{d[\hat{x}_r(\epsilon(s))]}{d[\epsilon(s)]}
\end{bmatrix} = 0, \\
\left\| \frac{d[\hat{x}_r(\epsilon(s))]}{d[\epsilon(s)]} \right\|_2 = 1
\end{cases}
\] (3.108)

\[
\epsilon(0) = 0, \quad \hat{x}_r(0) = \begin{bmatrix} \text{Re}(\hat{x}(0)) \\ \text{Im}(\hat{x}(0)) \end{bmatrix} \in \mathbb{R}^{2m}, \quad \hat{P}_{\alpha\beta}(\hat{x}(0)) = \hat{P}_{\alpha\beta}(\hat{x}(0), \alpha, \beta) = 0
\]

where

\[
J_{\hat{\alpha}\hat{\beta}}(\hat{P}_{\alpha\beta}(\hat{x}_r(\epsilon(s)), \epsilon(s))) = \begin{bmatrix}
\frac{\partial[\hat{P}_{\alpha\beta}(\hat{x}_r(\epsilon(s)), \epsilon(s))]}{\partial[\hat{x}_r(\epsilon(s))]} \\
\frac{\partial[\hat{P}_{\alpha\beta}(\hat{x}_r(\epsilon(s)), \epsilon(s))]}{\partial[\epsilon(s)]}
\end{bmatrix} = \begin{bmatrix} \frac{\partial[\hat{P}_{\alpha\beta}(\hat{x}_r(\epsilon(s)), \epsilon(s))]}{\partial[\hat{x}_r(\epsilon(s))]} \\
\frac{\partial[\hat{P}_{\alpha\beta}(\hat{x}_r(\epsilon(s)), \epsilon(s))]}{\partial[\epsilon(s)]} \end{bmatrix}
\] (3.109)
Chapter 3 Scaling & Homogeneous Projective Transformations 107

It is obvious that the parameterized homotopy paths $\mu$ in this case can be viewed as trajectories of the IVP (3.108), emanating from $s_0 = 0$ and terminating at $s = s_1$, where $e(s_1) = 1$. For a given initial condition $\hat{x}(0)$ (i.e. easy problem solution), the IVP (3.108) has a unique solution.

Concluding this section, it should be pointed out once again that all the results and discussions in sections 2.3.2 and 2.3.3, follow for the present analysis by setting $F(x) \leftarrow \hat{F}(\hat{x})$, $F_0(x) \leftarrow \hat{F}_0(\hat{x})$, $H_{\alpha \beta} \leftarrow \hat{H}_{\alpha \beta}$, $P_{\alpha \beta} \leftarrow \hat{P}_{\alpha \beta}$, $u_{\alpha \beta} \leftarrow \hat{u}_{\alpha \beta}$, $v_{\alpha \beta} \leftarrow \hat{v}_{\alpha \beta}$, $x \leftarrow \hat{x}$. For this reason, these results and discussions are not repeated here.

3.4.3 Numerical solution of (3.90)

It is apparent from sections 3.3-3.4.2 that the problem of the numerical solution of (3.90), is equivalent to the numerical solution of (3.87), since $\hat{X} = \text{vec}^{-1}(\hat{x})$. Using the probability-1 homotopy theorems 3.5 and 3.6, presented above, the solutions of (3.87) can be obtained from the solution of the IVP (3.101) and (3.108).

All that remains to be shown are the formulas of partial derivatives and Jacobians present in (3.101) and (3.108). These formulas are derived straightforwardly using the formulas (B.3.3)-(B.3.8), (B.2.3),(B.2.19) and (B.3.11) in appendix B.
For the IVP (3.101), the following formulas hold:

$$
\frac{d}{d\hat{x}(e)} \left[ \hat{R}(\hat{x}(e), \epsilon) \right] = \frac{d}{d\hat{x}(e)} \left[ \hat{R}(\hat{x}(e)) \right] - \gamma(1 - \epsilon) \frac{d}{d\hat{x}(e)} \left[ \hat{R}(\hat{x}(e), \alpha, \beta) \right]
$$
\begin{equation}
(3.113)
\end{equation}

$$
\frac{d}{d\hat{x}(e)} \left[ \hat{T}(\hat{x}(e), \epsilon) \right] = \hat{T}(\hat{x}(e)) + \gamma \hat{T}(\hat{x}(e), \alpha, \beta)
$$
\begin{equation}
(3.114)
\end{equation}

$$
\frac{d}{d\hat{x}(e)} \left[ \hat{F}(\hat{x}(e)) \right] = \frac{d}{d\hat{x}(e)} \left[ \text{vec}(E_{10}(u_C)) \right]
$$
\begin{equation}
(3.115)
\end{equation}

\begin{align*}
&= \left( E_{10}(u_C)U_{1 \times np} \right)^\circ \left( \frac{d}{d\hat{x}(e)} \left[ L(\hat{x}(e)) \right] \right) \left( \sum_{i=1}^{\varphi} \left( B_i^T \otimes A_i \right) \left( \left( E_{10}(u_X) \circ \text{vec}^{-1}(\hat{x}(e)) \right) D_i \right) \right) + \left( E_{10}(u_C)U_{1 \times np} \right)^\circ \left( \frac{d}{d\hat{x}(e)} \left[ L(\hat{x}(e)) \right] \right) \left( \text{vec}(G) \right) \\
&+ \left( E_{10}(u_C)U_{1 \times np} \right)^\circ \left( \frac{d}{d\hat{x}(e)} \left[ L(\hat{x}(e)) \right] \right) \left( \text{vec}(G) \right)
\end{align*}

\begin{equation}
(3.116)
\end{equation}

\begin{align*}
&= \left( U_{np \times 1}E_{10}(u_X)^T \right)^\circ \left( \sum_{i=1}^{\varphi} \left( B_i^T \otimes A_i \right) \right) \frac{d}{d\hat{x}(e)} \left[ L(\hat{x}(e)) \right] \left( \hat{x}(e) L(\hat{x}(e)) \right) \\
&= \left( U_{np \times 1}E_{10}(u_X)^T \right)^\circ \left( \sum_{i=1}^{\varphi} \left( B_i^T \otimes A_i \right) \right) \left( I_{np} L(\hat{x}(e)) + \hat{x}(e) A \right)
\end{align*}

\begin{equation}
(3.116)
\end{equation}

\begin{align*}
&= \frac{d}{d\hat{x}(e)} \left[ \sum_{i=1}^{\varphi} \left( E_i^T \otimes C_i \right) \left( I_p \circ \left( \left( E_{10}(u_X) \circ \text{vec}^{-1}(\hat{x}(e)) \right) D_i \right) \right) \right] \left( E_{10}(u_X) \circ \hat{x}(e) \right) \\
&= \frac{d}{d\hat{x}(e)} \left[ \sum_{i=1}^{\varphi} \left( E_i^T \otimes C_i \right) \left( I_p \circ \left( \left( E_{10}(u_X) \circ \text{vec}^{-1}(\hat{x}(e)) \right) D_i \right) \right) \right] \left( E_{10}(u_X) \circ \hat{x}(e) \right)
\end{align*}

\begin{equation}
(3.116)
\end{equation}
\[
\sum_{i=1}^{\xi} \left( (E_i^T \otimes C_i) \right) \\
\left[ \frac{d}{d[x_{11}(\varepsilon)]} \left( (E_{10}(U_X) \circ X(\varepsilon)) D_i \right) \right] E_{10}(u_X) \circ x(\varepsilon) + (I_p \otimes ((E_{10}(U_X) \circ X(\varepsilon)) D_i) \left( \frac{d}{d[x_{11}(\varepsilon)]} \left( (E_{10}(u_X) \circ x(\varepsilon)) \right) \right) \\
\left[ \frac{d}{d[x_{m1}(\varepsilon)]} \left( (E_{10}(u_X) \circ x(\varepsilon)) D_i \right) \right] E_{10}(u_X) \circ x(\varepsilon) + (I_p \otimes ((E_{10}(u_X) \circ X(\varepsilon)) D_i) \left( \frac{d}{d[x_{m1}(\varepsilon)]} \left( E_{10}(u_X) \right) \right) \\
= \sum_{i=1}^{\xi} \left( (E_i^T \otimes C_i) \right) \\
\left[ \frac{d}{d[x_{11}(\varepsilon)]} \left( (E_{10}(U_X) \circ X(\varepsilon)) D_i \right) \right] E_{10}(u_X) \circ x(\varepsilon) + (I_p \otimes ((E_{10}(u_X) \circ X(\varepsilon)) D_i) \left( \frac{d}{d[x_{11}(\varepsilon)]} \left( E_{10}(u_X) \circ x(\varepsilon) \right) \right) \\
\left[ \frac{d}{d[x_{m1}(\varepsilon)]} \left( (E_{10}(u_X) \circ x(\varepsilon)) D_i \right) \right] E_{10}(u_X) \circ x(\varepsilon) + (I_p \otimes ((E_{10}(u_X) \circ X(\varepsilon)) D_i) \left( \frac{d}{d[x_{m1}(\varepsilon)]} \left( E_{10}(u_X) \right) \right) \\
(3.117)
\]

Also for (3.117), subject to (B.3.12) in appendix B, a more compact alternative formula is (3.118), shown below.

\[
\sum_{i=1}^{\xi} \left( E_i^T \otimes C_i \right) \left[ \frac{d}{d[x_{11}(\varepsilon)]} \left( (E_{10}(U_X) \circ vec^{-1}(\hat{x}(\varepsilon)) D_i \right) \right] E_{10}(u_X) \circ \hat{x}(\varepsilon) \\
= \sum_{i=1}^{\xi} \left( E_i^T \otimes C_i \right) \left[ \frac{d}{d[x_{11}(\varepsilon)]} \left( (E_{10}(u_X) \circ vec^{-1}(\hat{x}(\varepsilon)) D_i \right) \right] E_{10}(u_X) \circ \hat{x}(\varepsilon) \\
= \sum_{i=1}^{\xi} \left( U_{np \times 1} E_{10}(u_X)^T \right) \left( E_i^T \otimes C_i \right) \left[ \frac{d}{d[x_{11}(\varepsilon)]} \left( (E_{10}(u_X) \circ X(\varepsilon)) D_i \right) \right] \\
= \sum_{i=1}^{\xi} \left( U_{np \times 1} E_{10}(u_X)^T \right) \left( E_i^T \otimes C_i \right) \left[ \frac{d}{d[x_{11}(\varepsilon)]} \left( (E_{10}(u_X) \circ X(\varepsilon)) D_i \right) \right] + \left( (E_{10}(u_X) \circ X(\varepsilon)) D_i \right) \left( E_i^T \otimes C_i \right) \left[ \frac{d}{d[x_{11}(\varepsilon)]} \right] I_n
(3.118)
\]

Furthermore, from (B.3.7)

\[
\frac{d}{d[x_{11}(\varepsilon)]} \left( \left( L(\hat{x}(\varepsilon)) \right)^2 vec(G) \right) = 2L(\hat{x}(\varepsilon)) vec(G) A_a \\
(3.119)
\]

Now, \( \left( \frac{d}{d[x_{11}(\varepsilon)]} \left( \hat{F}_0(\hat{x}(\varepsilon), \alpha, \beta) \right) \right) \) subject to (3.96) (under the homotopy mapping (3.95)) and subject to (3.99), (under the homotopy mapping (3.98)) is given by (2.65) and (2.66) in chapter 2
respectively. \[
\frac{d\hat{F}_0(\hat{x}(\varepsilon), \alpha, \beta)}{d\hat{x}^T(\varepsilon)} \text{ subject to } (3.97) \text{ (under the homotopy mapping (3.95))}, \text{ and subject to } (3.100) \text{ (under the homotopy mapping (3.98))}, \text{ is calculated as equations (3.120) and (3.121) respectively. For these calculations, the results (B.3.6) and (B.3.4) in appendix B were used. Hence,}
\[
\frac{d\hat{F}_0(\hat{x}(\varepsilon), \alpha, \beta)}{d\hat{x}^T(\varepsilon)} = \frac{d\alpha \circ L(\hat{x}(\varepsilon))\hat{x}(\varepsilon) - (L(\hat{x}(\varepsilon)))^2 \beta}{d\hat{x}^T(\varepsilon)} = \text{diag}(\alpha)(I - L(\hat{x}(\varepsilon)))\hat{x}(\varepsilon) + \hat{x}(\varepsilon)\alpha_a) - 2L(\hat{x}(\varepsilon))\beta_a. \quad (3.120)
\]

From the above it is obvious that under (3.115)-(3.121), (2.65), and (2.66), the IVP (3.101) is well defined under the assumptions of probability-1 theorems 3.5 and 3.6.

For the IVP (3.108), the corresponding formulas hold with respect to (3.109)-(3.112). The analytic expressions for (3.110), (3.111) are obtained under the substitution \( \varepsilon \rightarrow \varepsilon(s) \) in (3.113) and (3.114). since under such a substitution the formulas are structurally identical. Hence, (3.108) is well defined.

From all the above, it is obvious that probability-1 homotopy algorithms can be implemented via the tracking of the homotopy paths methods of section 3.4.2. Now, since (3.87) is solved then (3.90) is solved and under (3.92), (2.1) is solved too.

Finally, the discussions in section 2.4 about the advantages and disadvantages and relevant issues about the previous formulations follow and hence they are not repeated here.

3.5 Synopsis

This chapter has presented the concept of scaling for the general quadratic matrix equation (3.1) and the special case (2.1). The concept of homogeneous projective transformation has also been addressed for the general algebraic vector polynomial equation (3.7) and as a consequence the scaled version of (2.1). Since the nature of these subjects is quite mathematical, the presentation has necessarily been mathematical with great care taken over the clarity of statements. Therefore, in order to make the material more understandable for the non expert reader, simple numerical examples have been used to illustrate the methods. The chapter has also been written
to stand alone as a separate study, although there are some obvious connections with chapter 2 and appendices A and B, mainly in section 3.4.

For the homogeneous projective transformation the majority of the analysis was focused on vector polynomial equations like (3.7). Hence, the results hold generally for n-order vector polynomial systems. The homogeneous projective transformation can also be straightforwardly applied to n-order matrix polynomial equations, as soon as they have been transformed into the equivalent vector equation (3.7). This is one of the general characteristics, in terms of applications, of chapter 3.

Both the scaling and the homogeneous projective transformation were used along with the probability-1 homotopy methods presented in chapter 2. New probability-1 homotopy theorems were presented, which are characterized by their ability to calculate solutions at infinity. The only assumption required for the new probability-1 homotopy theorems to hold in a probabilistic sense, is the finiteness of the number of solutions of the original and the homogeneous projective transformed equations. Hence, the strong condition of (2.1) having no solutions at infinity, has been relaxed. This was the main goal of sections 3.3 and 3.4.

Although similar results have been presented in [121], [182], [183], the results here are actually different and the proofs vary considerably. Useful comments and comparisons have been made, in order to declare similarities and differences as well as advantages and disadvantages with other methods. Conclusions on each method are presented in each relevant section and hence are not repeated in this final section. Finally, the application of the methods will take place in later chapters through various numerical problems in science and engineering.
Chapter 4

Numerical Analysis

4.1 Introduction

In this chapter the accuracy and the stability of the probability-1 homotopy algorithms, presented in chapters 2 and 3 will be analyzed. Generally there are three major types of errors that can take place in a numerical computation: rounding errors, data uncertainty errors, and truncation errors.

Rounding errors occur almost always in a numerical computation and they are attributed to the finite precision arithmetic of the computational machine. For floating point arithmetic, rounding errors are identified via the roundoff unit $u_r$, which is defined as the minimum number that can be added to 1 without any effect. The rounding error analysis on each intermediate step of a numerical algorithm depends on the particular numerical method and on the particular data of the problem. This is what it makes it hard. Nevertheless, one can always analyze the rounding error to the computed solution of a given problem, by implementing rounding error analysis to the residual of the solution. This makes sense epistemologically, since all computational rounding errors are accumulative during a numerical process. Hence, their effect on a problem’s solution can be detected at the end of the numerical process with respect to the equation’s residual.

So far the majority of computer programs, including state-of-the-art commercial software, avoid providing roundoff error analysis. Some exceptions are the numerical algorithms in [184], [80], and [135]. For homotopy type methods, there is no relevant reference as far as the author is aware. Moreover, for virtually all algorithms, the rounding error analysis is available for real and not complex arithmetic. The reason for this is that rounding error analysis for complex arithmetic operations is more complicated than it is for real arithmetic. Fortunately, the results that apply for real arithmetic also hold for complex arithmetic when
scaled by a small constant multiplier (usually not more than 4) ([78]). In practice, the effect of this multiplier is null, and hence the results for complex arithmetic practically hold for real arithmetic and vice versa. Later in this chapter, it will be shown why this happens.

A decade ago, complex arithmetic in computer software (e.g. programming languages) was implemented implicitly via real arithmetic and almost doubled the computational effort. Present software codes include complex arithmetic as a built in feature. Moreover, some modern processors feature complex arithmetic as part of their hardware structure. All the above have contributed to a substantial reduction of computation times for complex arithmetic operations. The present work provides a rigorous rounding error analysis in complex arithmetic for the numerical solution of generalized algebraic quadratic matrix equations.

Data uncertainty errors are very likely to occur from rounding errors when storing the data on a computational machine, or when these data are solutions to previous numerical problems and hence carry errors from earlier computations. Also, for practical (physical) problems, data uncertainties can be introduced from error measurements. To understand data uncertainty errors the analysis is done using perturbation theory [162], [159]. This task is generally easier to the rounding error analysis.

Truncation errors arise when mathematical functions or formulas are approximated by series of finite terms (e.g. derivative approximations via truncated Taylor series etc). Fortunately, these kind of errors are not present in this work since, as shown in chapters 2 and 3, all the associated computational quantities are provided explicitly via analytic formulas.

It is important to note at this point that the whole analysis and the results provided in this chapter are independent of the numerical method used for the solution of the respective equations, and it is focused on the solutions. Therefore, the results of this chapter can be used along with any numerical algorithm. This allows chapter 4 to stand alone as a separate study.

By studying the effects of rounding and data uncertainty errors within a numerical algorithm, one can establish rigorous and sometimes empirical estimates about the accuracy, the precision and the stability of the numerical algorithm. Now, the accuracy refers to the absolute or relative error of an approximated quantity (where the relative error is the absolute error of the approximated quantity normalized by that quantity). The precision refers to the accuracy with which the floating point arithmetic operations are performed. Hence, accuracy is not necessarily limited by the precision [78]. The numerical stability (by means of error analysis and not convergence) of an algorithm is determined from two (both combining rounding and data uncertainty errors) types of errors: the backward error and the forward error. These errors and how they are associated with the numerical stability of algorithms, will be described in the respective sections of this chapter subject to the presented problems. A more general and detailed treatment can be found in [78].
The structure of this chapter is as follows. Section 4.2 presents the rounding error analysis for the homotopy equations (3.95) and (3.98). Section 4.3 determines condition number estimates for (3.95) and (3.98), which give information about the numerical conditioning of specific problems and their solutions [78], [134], [98]-[100]. Section 4.4 introduce the backward and forward errors for (3.95) and (3.98), and reports the respective analysis and results. Next in section 4.5, the results from the previous sections are combined in order to derive criteria about the numerical stability and behaviour of the homotopy equations (3.95) and (3.98). Note that studying (3.95) and (3.98) covers all the deformation process from the initial condition to the original problem solution. Section 4.6 includes further discussion and conclusions are given in the final section 4.7.

4.2 Rounding Error Analysis

The rounding error analysis aims to derive an a priori measure bound (usually norm-wise), in order to measure the effects of the rounding errors on an algorithm [78]. Establishing the existence of such a bound is the primary problem, followed by the variation of the magnitude of this bound with respect to different problem data. The best case scenario is when for all choices of problem data the bound is small. If the contrary is true, then characteristics about any potential numerical instability emerge and hence possible ways of cure can be proposed accordingly. Note that this is not the case for all algorithms (e.g. unstable algorithms).

The constant terms in a rounding error bound depend on the dimensionality of the problem. These are not so important because they often cause the bound to overestimate the actual error by a constant multiple. Hence, it is preferable to put the error bounds in a form that is easily interpreted. This is the reason why usually sharp error bounds (e.g. first order) are obtained in a way that the occurring actual rounding errors can be taken into account. Note however that sharp error bounds [98], can sometimes underestimate the actual rounding errors. But computational experience [100], [136] shows that this is not the case, since rounding errors can tend to cancel each other out.

As was stated in section 4.1, the rounding error analysis at any potential step of a numerical algorithm depends on the particular numerical method and on the particular data of the problem. Since this is a hard job to do in terms of analysis, posteriori rounding error bounds are most often obtained for the residual of the solution.

In the sequel, norm-wise sharp (first order) upper bound estimates of rounding errors to the computed solution are derived following the philosophy of [78]. More specifically, in subsection 4.2.1 the floating point number system and the model of arithmetic used in this study are presented. Subsection 4.2.2 presents some results in the form of lemmas considering estimates of rounding error bounds for matrix addition and multiplication in real and complex
arithmetic. In subsection 4.2.3, the residual of a solution to (3.95) and (3.98) is analyzed via perturbations of the data of (3.95) and (3.98) subject to rounding errors. Hence, rigorous norm-wise upper bound estimates of rounding errors to the computed solution are derived. Finally, at the end of the present section some relevant further discussion is given.

4.2.1 The Floating Point Number System and Model of Arithmetic

A floating point number system $F \subseteq \mathbb{R}$ is a subset of the real numbers whose elements have the form $\pm m_a \times b_a^{e-t}$ [78], [36]. A floating point number is characterized by four integer parameters:

1) The base $b_a$ (or radix).

2) The precision $t$.

3) The exponent range $e_{\min} \leq e \leq e_{\max}$.

4) The mantissa $m_a$: an integer such that $0 < m_a < b_a^{t-1}$.

For the sake of unique representation of every floating point number, it is assumed that $m_a \geq b_a^{t-1}$ for each nonzero number [78].

Now let $G = \{x \in \mathbb{R} | x = \pm m_a \times b_a^{e-t}, e \in \mathbb{R} \}$. If $x \in \mathbb{R}$ then $fl(x) \in G$ denotes an element of $G$ nearest to $x$, and the transformation (mapping) $x \rightarrow fl(x)$ is called rounding. It is said that $fl(x)$ overflows if $|fl(x)| > \max\{|y| y \in F\}$ and underflows if $|fl(x)| \in \left(0, \min\{|y| y \in F - \{0\}\}\right)$ respectively.

Another important parameter in the rounding error analysis is the roundoff unit $u_r$ (the minimum number that can be added to 1 without any effect). This varies for different machines and different standards on floating point arithmetic [36]. Among this variety, often used value for $u_r$ is the IEEE single (precision) arithmetic ($b_a = 2$, $t = 24$, $e_{\min} = -125$, $e_{\max} = 128$, $u_r = 2^{-24}$) and more often the IEEE double (precision) arithmetic ($b_a = 2$, $t = 53$, $e_{\min} = -1021$, $e_{\max} = 1024$, $u_r = 2^{-53}$). In terms of computer storage, the IEEE single uses a 32 bits word size from which 23 bits are given for the mantissa, 1 bit is given for the number’s sign and 8 bits are given for the exponent. The IEEE double uses a 64 bits word size from which 52 bits are given for the mantissa, 1 bit is given for the number’s sign and 11 bits are given for the exponent. Note that, for all numerical experiments in this work, the IEEE double float point arithmetic is used.
To implement a rounding error analysis of any algorithm, the following standard floating point arithmetic model, subject to the accuracy of basic arithmetic operators, is commonly used [78].

**Standard Floating Point Arithmetic Model:** \( \text{fl}(x \cdot y) := (x \cdot y)(1 + \delta), \quad \delta \in \mathbb{R}, \quad |\delta| \leq u_r, \quad \bullet = +, -, \times, \div. \) \( \bullet \) denotes one of the basic four operators (addition, subtraction, multiplication, division) and \( (x \cdot y) \in \mathbb{F} \).

In the above model, \( \text{fl}(\cdot) \) is an arithmetic expression denoting the computed value of that expression. The model states that the computed value of \( (x \cdot y) \) (i.e. \( \text{fl}(x \cdot y) \)) is "as good as" the rounded exact answer, in the sense that the relative error bound \( |\delta| \) (note that \( \delta = \frac{\text{fl}(x \cdot y) - (x \cdot y)}{(x \cdot y)} \) ) is the same in both cases. Note that \( \delta = 0 \) is the ideal case of the computed answer being actually equal to the exact answer.

From the above it is apparent that the **absolute** and **relative rounding errors** can be defined as \( |\text{fl}(x \cdot y) - (x \cdot y)| \) and \( |\delta| = \frac{|\text{fl}(x \cdot y) - (x \cdot y)|}{(x \cdot y)} \) respectively.

### 4.2.2 Some Results for Matrix Addition and Multiplication

Initially assume real arithmetic computations and let \( x, y \in \mathbb{R} \). Then according to the standard floating point arithmetic model in subsection 4.2.1, the arithmetic operators for addition, subtraction and multiplication are as:

\[
\text{fl}(x \pm y) := (x \pm y)(1 + \delta), \quad |\delta| \leq u_r \tag{4.1}
\]
\[
\text{fl}(xy) := xy(1 + \delta), \quad |\delta| \leq u_r. \tag{4.2}
\]

Note that both (4.1) and (4.2) can be viewed as perturbations of \( x \pm y \) with respect to \( x \) and \( y \) for (4.1), and \( xy \) with respect to either \( x \) or \( y \) for (4.2). Note also that (4.2) can be viewed also, as a first order approximation to the perturbation of \( xy \) with respect \( x \) and \( y \) assuming that \( u_r \) is sufficiently small such that \( 1 + 2\delta \approx 1 + \delta \) and \( \delta + \delta^2 \approx \delta \). More specifically, assuming \( \delta x := \delta \times x, \quad \delta y := \delta \times y \), it follows that

\[
(x + \delta x) \pm (y + \delta y) = (x \pm y) + (\delta x \pm \delta y) = (x \pm y) + (x \pm y)\delta = (x \pm y)(1 + \delta)
\]
\[
(x + \delta x)y = xy + \delta xy = xy(1 + \delta) \quad \text{or} \quad x(y + \delta y) = xy + x\delta y = xy(1 + \delta).
\]

Finally, if \( u_r \) is sufficiently small such that \( 1 + 2\delta \approx 1 + \delta \) and \( \delta + \delta^2 \approx \delta \), then

\[
(x + \delta x)(x + \delta x) = xy + 2\delta xy + \delta^2 xy = xy(1 + \delta).
\]
Lemma 4.1 ([78]):

If $|\delta_i| \leq u_r$ and $\rho_i = \pm 1$ for $i = 1, \ldots, k$, and $ku_r < 1$, then

$$\prod_{i=1}^{k} (1 + \delta_i)^{\rho_i} = 1 + \theta_k,$$

where $|\theta_k| \leq \gamma_k$.

$$\gamma_k := \frac{ku_r}{1 - ku_r}. \tag{4.3}$$

**Proof.**

The proof can be found in [78].

The extension of the above scalar operators to matrices is straightforward. Since, matrix addition and multiplication consist of a number of elementary scalar operators as above, using lemma 4.1 the following lemmas can easily be proven ([78]).

**Lemma 4.2:**

Let $K, L \in \mathbb{R}^{k \times l}$, $k, l \in \mathbb{Z}^+$. Then $fl(K \pm L) = (K \pm L)(1 + \delta)$, where $|\delta| \leq u_r$, which implies the following error bounds

$$\left| fl(K \pm L) - (K \pm L) \right| \leq |\delta||K \pm L| \leq |\delta||K| + |\delta||L|. \tag{4.4}$$

$$\left\| fl(K \pm L) - (K \pm L) \right\|_i \leq |\delta||K \pm L|_i \leq |\delta||K|_i + |\delta||L|_i, \quad i = 1, 2, F. \tag{4.5}$$

**Proof.**

The proof is immediate from lemma 4.1 and the standard 1, 2 and $F$ (Frobenius) norms inequalities. See also [184].

**Lemma 4.3 ([78]):** Let $K \in \mathbb{R}^{l \times k}$ and $x \in \mathbb{R}^k$, $l, k \in \mathbb{Z}^+$. Then $fl(Kx) = (K + \delta K)x$, $|\delta K| \leq \gamma_k |K|$, which implies the following error bounds

$$\left| fl(Kx) - Kx \right| = |\delta Kx| \leq |\delta K||x| \leq \gamma_k |K||x|. \tag{4.6}$$

$$\left\| fl(Kx) - Kx \right\|_i \leq \gamma_k \|K||x||_i, \quad i = 1, 2, F. \tag{4.7}$$

where $\gamma_k$ is given from (4.3).

**Proof.**

The proof can be found in [78].

From lemma 4.3 above, it is apparent that the rounding error analysis is done by means of problem data perturbations (e.g. $\delta K$). Moreover, the bounds obtained are posteriori bounds.
and hold no matter how the inner product computations, which determine each element of $Kx$, were performed.

So far all the previous results hold for real arithmetic. These can easily be extended to complex arithmetic, since complex arithmetic can be implemented using real arithmetic. In other words the corresponding Complex Floating Point Arithmetic Model is a consequence of the Standard Floating Point Arithmetic Model having in mind that for $x, y \in \mathbb{C}$, the arithmetic operators for addition, subtraction and multiplication are defined as:

$$fl(x \pm y) = (\text{Re}(x) + \text{Re}(y))(1 + \delta_1) \pm i(\text{Im}(x) + \text{Im}(y))(1 + \delta_2)$$

$$= x + y + (\text{Re}(x) + \text{Re}(y))\delta_1 \pm i(\text{Im}(x) + \text{Im}(y))\delta_2. \tag{4.8}$$

$$fl(xy) = (\text{Re}(x)\text{Re}(y))(1 + \delta_1) - \text{Im}(x)\text{Im}(y)(1 + \delta_2)(1 + \delta_3) + i(\text{Re}(x)\text{Im}(y)(1 + \delta_4) - \text{Im}(x)\text{Re}(y)(1 + \delta_5)(1 + \delta_6). \tag{4.9}$$

where $|\delta_i| \leq u_r$ for $i = 1, \ldots, 6$.

From (4.8) and (4.9), we have

$$fl(x \pm y) = (x \pm y)(1 + \delta), \quad |\delta| \leq u_r \tag{4.10}$$

$$fl(xy) = xy(1 + \delta), \quad |\delta| \leq \sqrt{2}u_2 \tag{4.11}$$

where $\gamma_2$, is given by (4.3) with $k = 2$.

Equations (4.10) and (4.11) are proven in [78] as lemma 3.5. This proof is obvious from (4.8) and (4.9).

Now for $u_r$ sufficiently small (more precisely for $u_r \leq \frac{w - \sqrt{2}}{2w}$, where $w \in \mathbb{R}$, $w > \sqrt{2}$) (4.11) can hold with $|\delta| \leq cu_r$, where $c = 2w$. Hence, both (4.10) and (4.11) can be written as

$$fl(x \pm y) = (x \pm y)(1 + \delta), \quad |\delta| \leq cu_r \tag{4.12}$$

$$fl(xy) = xy(1 + \delta), \quad |\delta| \leq cu_r \tag{4.13}$$

where $c = 2w$.

Hence, with respect to complex arithmetic, lemmas 4.2 and 4.3, can be restated as lemmas 4.4 and 4.5 respectively.

Lemma 4.4:

Let $K, L \in \mathbb{C}^{k \times l}$, $k, l \in \mathbb{Z}_+$. Then $fl(K \pm L) = (K \pm L)(1 + \delta)$, where $|\delta| \leq cu_r$, $c = 2w$, $u_r \leq \frac{w - \sqrt{2}}{2w}$, $w \in \mathbb{R}$, which implies the following error bounds.
\begin{equation}
\| f\ell(K \pm L) - (K \pm L) \|_i \leq \| \delta K \| + \| \delta L \|, \quad i = 1, 2, F.
\end{equation}

Proof.
The proof follows from Lemma 4.2 and (4.12).

Lemma 4.5:
Let $K \in \mathbb{C}^{l \times k}$ and $x \in \mathbb{C}^k$. Then $f\ell(Kx) = (K + \delta K)x$, $|\delta K| \leq \gamma_{c_k} |K|$, which implies the following error bounds
\begin{equation}
| f\ell(Kx) - Kx | = | \delta Kx | \leq | \delta K | |x| \leq \gamma_{c_k} |K||x|
\end{equation}
\begin{equation}
\| f\ell(Kx) - Kx \|_i \leq \gamma_{c_k} \| K \|_i |x|_i, \quad i = 1, 2, F,
\end{equation}
where
\begin{equation}
\gamma_{c_k} := \frac{k\epsilon r}{1 - k\epsilon r}.
\end{equation}

Proof.
The proof follows from Lemma 4.2 and (4.12), (4.13).

As pointed out in [78], bounds for the rounding errors in basic complex arithmetic operations are rarely given in the literature and virtually all published error analyses in matrix computations are for real arithmetic. Nevertheless, because of Lemmas 4.4 and 4.5 at least for matrix addition and multiplication the results for real arithmetic are valid for complex arithmetic with some constants in the formulas increased appropriately. This feature, will in the sequel of this chapter allow us to work without any further consideration in exactly the same way for both real and complex arithmetic operations.

Before continuing with the evaluation of rounding error bounds for the homotopy equations (3.95) and (3.98), Lemmas 4.6, 4.7, 4.8 and Theorems 4.1, 4.2, 4.3 are necessary for the establishment of the forthcoming analysis.

Lemma 4.6:
Let $y \in \mathbb{C}$, $P, R, S \in \mathbb{C}^{n \times p}$, $Q \in \mathbb{C}^{p \times n}$, $T \in \mathbb{C}^{n \times p}$, $V \in \mathbb{C}^{1 \times np}$, $W \in \mathbb{C}^{np}$, $n, p \in \mathbb{Z}_+$. Then the following inequalities hold:
\begin{equation}
\| y \otimes (P \circ ((Q(R \circ S)T) \otimes (VVW))) \|_F \leq \| y \|_F \| P \|_F \| Q(R \circ S) \|_2 \| T \|_F \| V \|_2 \| W \|_F
\end{equation}
Chapter 4 Numerical Analysis 120

\[ \left\| y \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) \right\|_F \leq \left\| y \right\|_F \left\| P \right\|_F \left\| Q(R \circ S)T \right\|_2 \left\| V \right\|_F \left\| W \right\|_F \]  

(4.19)

\[ \left\| y \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) \right\|_F \leq \left\| y \right\|_F \left\| P \right\|_F \left\| Q \right\|_F \left\| R \otimes S \right\|_2 \left\| T \right\|_F \left\| V \right\|_2 \left\| W \right\|_F \]  

(4.20)

\[ \left\| y \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) \right\|_F \leq \left\| y \right\|_F \left\| P \right\|_F \left\| Q \right\|_F \left\| R \circ S \right\|_2 \left\| T \right\|_F \left\| V \right\|_2 \left\| W \right\|_F \]  

(4.21)

Proof.

First note that in general for matrices \( K \in C^{k \times l} \), \( L \in C^{l \times p} \), \( M \in C^{l \times p} \), \( k, l, p \in \mathbb{Z}_+ \)

\[ \left\| K \otimes L \right\|_F = \left( \sum_{i=1}^{k} \sum_{j=1}^{l} K_{ij}^2 L_{ij}^2 \right)^{\frac{1}{2}} \]  

(4.22)

Furthermore,

\[ \left\| L \circ M \right\|_F^2 = \left( \sum_{i=1}^{p} \sum_{j=1}^{l} L_{ij}^2 M_{ij}^2 \right)^{\frac{1}{2}} \]  

and \( \left\| L \right\|_F^2 \left\| M \right\|_F^2 = \left( \sum_{i=1}^{p} \sum_{j=1}^{l} L_{ij}^2 \right) \left( \sum_{i=1}^{p} \sum_{j=1}^{l} M_{ij}^2 \right) \) therefore

\[ \left\| K \circ L \right\|_F \leq \left\| K \right\|_F \left\| L \right\|_F \]  

(4.23)

In view of (4.22), (4.23) and of \( \left\| KL \right\|_F \leq \left\| K \right\|_F \left\| L \right\|_F \), \( \left\| KL \right\|_F \leq \left\| K \right\|_2 \left\| L \right\|_F \), the inequalities/equalities below follow from \( \left\| y \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) \right\|_F \).

\[ \left\| y \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) \right\|_F \leq \left\| y \right\|_F \left\| P \right\|_F \left\| Q(R \circ S)T \right\|_2 \left\| V \right\|_F \left\| W \right\|_F \]  

(4.19)

\[ \left\| y \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) \right\|_F \leq \left\| y \right\|_F \left\| P \right\|_F \left\| Q \right\|_F \left\| R \otimes S \right\|_2 \left\| T \right\|_F \left\| V \right\|_2 \left\| W \right\|_F \]  

(4.20)

\[ \left\| y \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) \right\|_F \leq \left\| y \right\|_F \left\| P \right\|_F \left\| Q \right\|_F \left\| R \circ S \right\|_2 \left\| T \right\|_F \left\| V \right\|_2 \left\| W \right\|_F \]  

(4.21)

Hence, (4.19) has been proven.

Also,

\[ \left\| y \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) \right\|_F \leq \left\| y \right\|_F \left\| P \right\|_F \left\| Q(R \circ S)T \right\|_2 \left\| V \right\|_F \left\| W \right\|_F \]  

(4.19)

\[ \left\| y \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) \right\|_F \leq \left\| y \right\|_F \left\| P \right\|_F \left\| Q \right\|_F \left\| R \otimes S \right\|_2 \left\| T \right\|_F \left\| V \right\|_2 \left\| W \right\|_F \]  

(4.20)

\[ \left\| y \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) \right\|_F \leq \left\| y \right\|_F \left\| P \right\|_F \left\| Q \right\|_F \left\| R \circ S \right\|_2 \left\| T \right\|_F \left\| V \right\|_2 \left\| W \right\|_F \]  

(4.21)

Hence, (4.20) has been proven.
Finally,
\[
\left\| y \otimes P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right\|_F \leq \left\| y \right\|_F \left\| P \right\|_F \left\| (Q(R \circ S)T) \otimes (VW) \right\|_F
\]
\[
= \left\| y \right\|_F \left\| P \right\|_F \left\| Q(R \circ S)T \right\|_F \left\| VW \right\|_F \leq \left\| y \right\|_F \left\| P \right\|_F \left\| Q \right\|_F \left\| (R \circ S)T \right\|_F \left\| V \right\|_2 \left\| W \right\|_F
\]
\[
\leq \left\| y \right\|_F \left\| P \right\|_F \left\| Q \right\|_F \left\| R \circ S \right\|_F \left\| T \right\|_F \left\| V \right\|_2 \left\| W \right\|_F .
\]
Hence, (4.21) has been proven.

**Theorem 4.1:**

Let \( y_i \in C, \ P, R, S \in C^{n \times p}, \ Q \in C^{n \times n}, \ T \in C^{p \times p}, \ V \in C^{1 \times np}, \ W \in C^{np}, \)
\[ i, n, p, v \in \mathbb{Z}_+ \] and their respective perturbations \( \delta y_i \in C, \ \delta P, \delta R, \delta S \in C^{n \times n}, \ \delta Q \in C^{n \times n}, \)
\( \delta T \in C^{p \times p}, \ \delta V \in C^{1 \times np}, \ \delta W \in C^{np \times 1}, \) such that
\[
|\delta y_i| \leq \gamma_{c_1} |y_i|, \ \forall i \in \mathbb{Z}_+, \ |\delta P| \leq \gamma_{c_1} |P|, \)
\[
|\delta R| \leq \gamma_{c_p} |R|, \ |\delta S| \leq \gamma_{c_p} |S|, \ |\delta Q| \leq \gamma_{c_p} |Q|, \ |\delta T| \leq \gamma_{c_p} |T|, \ |\delta V| \leq \gamma_{c_1} |V|, \ |\delta W| \leq \gamma_{c_1} |W|,
\]
where \( \gamma_{c_k} \) is defined by (4.18) for \( k = np, n, p, 1 \). Then for sufficient small \( u_r \), the following rounding error upper bound exists.

\[
\left\| f \left( \prod_{i=1}^{v} y_i \otimes P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) - \left( \prod_{i=1}^{v} y_i \right) \otimes P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right\|_F
\]
\[
\leq \gamma_{c_1} \left( \sum_{i=1}^{v} \text{sign}(\|\delta y_i\|_F) \left\| y_i \prod_{j=1, j \neq i}^{v} y_j \right\|_F \right)
\]
\[
+ \left( \text{sign}(\|\delta P\|_F) \gamma_{c_p} + \text{sign}(\|\delta Q\|_F) \gamma_{c_p} + \text{sign}(\|\delta W\|_F) \gamma_{c_1} \right) \left( \prod_{i=1}^{v} y_i \right) \left\| P \right\|_F \left\| Q \right\|_F \left\| R \circ S \right\|_F \left\| T \right\|_F \left\| V \right\|_2 \left\| W \right\|_F
\]
\[
+ \left( \text{sign}(\|\delta R\|_F) + \text{sign}(\|\delta S\|_F) \right) \gamma_{c_p} \left( \prod_{i=1}^{v} y_i \right) \left\| P \right\|_F \left\| Q \right\|_F \left\| R \right\|_F \left\| S \right\|_F \left\| T \right\|_F \left\| V \right\|_2 \left\| W \right\|_F
\]
\[
+ \text{sign}(\|\delta V\|_F) \gamma_{c_1} \left( \prod_{i=1}^{v} y_i \right) \left\| P \right\|_F \left\| Q \right\|_F \left\| R \right\|_F \left\| S \right\|_F \left\| T \right\|_F \left\| V \right\|_2 \left\| W \right\|_F .
\]

Furthermore,
\[
\left\| f \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ ((Q(R \circ S)T) \otimes (VW)) \right) \right\|_F \\
\leq \gamma_{c_1} \left( \sum_{i=1}^{v} \text{sign}(\|\delta y_i\|_F) \left\| y_i \prod_{j=1 \atop j \neq i}^{v} y_j \right\|_F \right) \\
+ \left( \text{sign}(\|\delta q\|_F) \gamma_{c_p} + \text{sign}(\|\delta r\|_F) \gamma_{c_p} + \text{sign}(\|\delta w\|_F) \gamma_{c_1} \right) \left\| \prod_{i=1}^{v} y_i \right\|_F \\
\times \left\| P \right\|_F \left| Q(R \circ S) \right|_2 \left| T \right|_F \left| V \right|_2 \left| W \right|_F \\
+ \text{sign}(\|\delta q\|_F) \gamma_{c_p} \left\| \prod_{i=1}^{v} y_i \right\|_F \left\| P \right\|_F \left| Q \right|_2 \left| R \circ S \right|_F \left| S \right|_F \left| T \right|_F \left| V \right|_2 \left| W \right|_F \\
+ \text{sign}(\|\delta r\|_F) \gamma_{c_p} \left\| \prod_{i=1}^{v} y_i \right\|_F \left\| P \right\|_F \left| Q(R \circ S) \right|_2 \left| T \right|_F \left| V \right|_2 \left| W \right|_F \\
+ \left\| \prod_{i=1}^{v} y_i \right\|_F \otimes \left( P \circ ((Q(R \circ S)T) \otimes (VW)) \right) \right\|_F. \tag{4.25}
\]

**Proof.**

From the previous sections it is apparent that in order to consider rounding errors in a computed quantity, this quantity is perturbed assuming the perturbation distorts the actual result. Now, if a first order approximation of the perturbed quantity is assumed then the obtained bounds are first order (sharp) bounds. This is the most modest way to perform such an analysis and is the most commonly used tactic. Indeed, this is the case in lemma 4.3, as shown in [78].

Hence, considering the perturbation of \( \prod_{i=1}^{v} y_i \otimes \left( P \circ ((Q(R \circ S)T) \otimes (VW)) \right) \) and dropping the resulting terms of second and higher orders ([78], [99]), we have

\[
f \left( \prod_{i=1}^{v} y_i \otimes \left( P \circ ((Q(R \circ S)T) \otimes (VW)) \right) \right)
\]
For \( u_r \) sufficiently small, second order perturbed terms set to zero, we have

\[
\prod_{i=1}^{v} y_i \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right)
\]

\[
= \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) + \sum_{i=1}^{v} \left( \delta y_i \prod_{j \neq i}^{v} y_j \right) \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right)
\]

\[
+ \prod_{i=1}^{v} y_i \otimes \left( \delta P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) + \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ \left( (\delta Q(R \circ S)T) \otimes (VW) \right) \right)
\]

\[
+ \prod_{i=1}^{v} y_i \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) + \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (\delta VW) \right) \right)
\]

\[
+ \prod_{i=1}^{v} y_i \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (V\delta W) \right) \right)
\]

\[
\Rightarrow \prod_{i=1}^{v} y_i \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) - \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right)
\]

\[
= \sum_{i=1}^{v} \left( \delta y_i \prod_{j \neq i}^{v} y_j \right) \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) + \left( \prod_{i=1}^{v} y_i \right) \otimes \left( \delta P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right)
\]

\[
+ \prod_{i=1}^{v} y_i \otimes \left( P \circ \left( (\delta Q(R \circ S)T) \otimes (VW) \right) \right) + \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right)
\]

\[
+ \prod_{i=1}^{v} y_i \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) + \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (V\delta W) \right) \right)
\]
\[ \left\| \prod_{i=1}^{V} y_i \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) - \prod_{i=1}^{V} y_i \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) \right\|_F \]

\[ \leq \left\| \sum_{i=1}^{V} \left( \delta_{ij} \prod_{j=1}^{V} y_j \right) \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) \right\|_F 

+ \left\| \prod_{i=1}^{V} y_i \otimes \left( \partial P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) \right\|_F \]

+ \left\| \prod_{i=1}^{V} y_i \otimes \left( P \circ \left( (Q(\partial R \circ S)T) \otimes (VW) \right) \right) \right\|_F 

+ \left\| \prod_{i=1}^{V} y_i \otimes \left( P \circ \left( (Q(R \circ \partial S)T) \otimes (VW) \right) \right) \right\|_F 

+ \left\| \prod_{i=1}^{V} y_i \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (V\partial W) \right) \right) \right\|_F 

+ \left\| \prod_{i=1}^{V} y_i \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (V\partial W) \right) \right) \right\|_F \] (4.26)

Now, since \( |\delta_{ij}| \leq \gamma_{c1} |y|, \forall i \in \mathbb{Z}_+, \) \( |\partial P| \leq \gamma_{c_p} |P|, \) \( |\partial R| \leq \gamma_{c_p} |R|, \) \( |\partial S| \leq \gamma_{c_p} |S|, \) \( |\partial Q| \leq \gamma_{c_p} |Q|, \)

\( |\partial T| \leq \gamma_{c_p} |T|, \) \( |\partial V| \leq \gamma_{c_p} |V|, \) \( |\partial W| \leq \gamma_{c_1} |W|, \) it follows that \( \|\delta_{ij}\|_F \leq \gamma_{c_1} \|y\|_F, \forall i \in \mathbb{Z}_+, \)

\( \|\partial P\|_F \leq \gamma_{c_p} \|P\|_F, \) \( \|\partial R\|_F \leq \gamma_{c_p} \|R\|_F, \) \( \|\partial S\|_F \leq \gamma_{c_p} \|S\|_F, \) \( \|\partial Q\|_F \leq \gamma_{c_p} \|Q\|_F, \)

\( \|\partial T\|_F \leq \gamma_{c_p} \|T\|_F, \) \( \|\partial V\|_F \leq \gamma_{c_p} \|V\|_F, \) \( \|\partial W\|_F \leq \gamma_{c_1} \|W\|_F. \) Taking into account the last norm-wise inequalities and (4.19)-(4.21) from lemma 4.6, it follows from (4.26) that

\[ \left\| \prod_{i=1}^{V} y_i \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) - \prod_{i=1}^{V} y_i \otimes \left( P \circ \left( (Q(R \circ S)T) \otimes (VW) \right) \right) \right\|_F \]
\[
\begin{align*}
&\leq \left\| \sum_{i=1}^{V} \frac{\delta y_i}{\gamma} \prod_{j=1}^{V} y_j \right\| F \left\| P \right\| F \left\| Q(R \circ S) \right\|_2 \left\| T \right\| F \left\| V \right\|_2 \left\| W \right\| F \\
&+ \left\| \prod_{i=1}^{V} y_i \right\| F \left\| \delta P \right\| F \left\| Q(R \circ S) \right\|_2 \left\| T \right\| F \left\| V \right\|_2 \left\| W \right\| F \\
&+ \left\| \prod_{i=1}^{V} y_i \right\| F \left\| P \right\| F \left\| \delta Q \right\| F \left\| R \circ S \right\|_2 \left\| T \right\| F \left\| V \right\|_2 \left\| W \right\| F \\
&+ \left\| \prod_{i=1}^{V} y_i \right\| F \left\| P \right\| F \left\| \delta R \right\| F \left\| S \right\|_2 \left\| T \right\| F \left\| V \right\|_2 \left\| W \right\| F \\
&+ \left\| \prod_{i=1}^{V} y_i \right\| F \left\| P \right\| F \left\| Q \right\|_2 \left\| \delta S \right\| F \left\| T \right\| F \left\| V \right\|_2 \left\| W \right\| F \\
&+ \left\| \prod_{i=1}^{V} y_i \right\| F \left\| P \right\| F \left\| Q(R \circ S) \right\|_2 \left\| \delta T \right\| F \left\| V \right\|_2 \left\| W \right\| F \\
&+ \left\| \prod_{i=1}^{V} y_i \right\| F \left\| P \right\| F \left\| Q(R \circ S) \right\|_2 \left\| T \right\| F \left\| \delta V \right\| F \left\| W \right\| F \\
&+ \left\| \prod_{i=1}^{V} y_i \right\| F \left\| P \right\| F \left\| Q(R \circ S) \right\|_2 \left\| T \right\| F \left\| V \right\|_2 \left\| \delta W \right\| F \\
&= \left\| f_l \left( \prod_{i=1}^{V} y_i \right) \otimes \left( P \circ ((Q(R \circ S)T) \otimes (V W)) \right) - \left( \prod_{i=1}^{V} y_i \right) \otimes \left( P \circ ((Q(R \circ S)T) \otimes (V W)) \right) \right\| F \\
&\leq \gamma \varepsilon_1 \left( \sum_{i=1}^{V} \text{sign} \left( \left\| \delta y_i \right\|_F \right) \prod_{j=1}^{V} y_j \right) \\
&+ \text{sign} \left( \left\| \delta P \right\| F \right) \gamma \varepsilon_2 \left( \sum_{i=1}^{V} \text{sign} \left( \left\| \delta T \right\| F \right) \gamma \varepsilon_2 \left( \sum_{i=1}^{V} \text{sign} \left( \left\| \delta W \right\| F \right) \gamma \varepsilon_2 \left( \prod_{i=1}^{V} y_i \right) \right) \right) \\
&\times \left\| P \right\| F \left\| Q(R \circ S) \right\|_2 \left\| T \right\| F \left\| V \right\|_2 \left\| W \right\| F \\
&+ \text{sign} \left( \left\| \delta Q \right\| F \right) \gamma \varepsilon_2 \left( \prod_{i=1}^{V} y_i \right) \left\| P \right\| F \left\| Q \right\|_2 \left\| R \circ S \right\|_2 \left\| T \right\| F \left\| V \right\|_2 \left\| W \right\| F \\
&+ \text{sign} \left( \left\| \delta R \right\| F \right) \gamma \varepsilon_2 \left( \prod_{i=1}^{V} y_i \right) \left\| P \right\| F \left\| Q \right\|_2 \left\| \delta S \right\| F \left\| T \right\| F \left\| V \right\|_2 \left\| W \right\| F \\
&+ \text{sign} \left( \left\| \delta S \right\| F \right) \gamma \varepsilon_2 \left( \prod_{i=1}^{V} y_i \right) \left\| P \right\| F \left\| \delta R \right\| F \left\| S \right\|_2 \left\| T \right\| F \left\| V \right\|_2 \left\| W \right\| F \\
&+ \text{sign} \left( \left\| \delta T \right\| F \right) \gamma \varepsilon_2 \left( \prod_{i=1}^{V} y_i \right) \left\| P \right\| F \left\| \delta Q \right\| F \left\| R \circ S \right\|_2 \left\| T \right\| F \left\| V \right\|_2 \left\| W \right\| F \\
&+ \text{sign} \left( \left\| \delta V \right\| F \right) \gamma \varepsilon_2 \left( \prod_{i=1}^{V} y_i \right) \left\| P \right\| F \left\| \delta R \right\| F \left\| S \right\|_2 \left\| \delta T \right\| F \left\| V \right\|_2 \left\| W \right\| F \\
&+ \text{sign} \left( \left\| \delta W \right\| F \right) \gamma \varepsilon_2 \left( \prod_{i=1}^{V} y_i \right) \left\| P \right\| F \left\| \delta S \right\| F \left\| \delta T \right\| F \left\| V \right\|_2 \left\| W \right\| F \
\end{align*}
\]
Hence, (4.24) has been proven.

Finally,

\[
+ \left( \sum_{i=1}^{v} y_i \right) \left( P \circ \left( Q \left( R \circ S \right) T \right) \otimes (VW) \right) = \left( \sum_{i=1}^{v} y_i \right) \left( P \circ \left( Q \left( R \circ S \right) T \right) \otimes (VW) \right)
\]

The last inequality in view of (4.24), implies (4.25).
Lemma 4.7:

Let $\gamma \in \mathcal{C}$, $P \in \mathcal{C}^{n \times p}$, $Q \in \mathcal{C}^{n \times n}$, $R, S, V, W \in \mathcal{C}^{n \times p}$, $T \in \mathcal{C}^{p \times n}$, $Z \in \mathcal{C}^{p \times p}$, $n, p \in \mathbb{Z}_+$. Then the following inequalities hold:

\[
\|y \otimes (P \circ (Q \circ S)T(V \circ W)Z)\|_F \leq \|y\|_F \|P\|_F \|Q\|_F \|R \circ S\|T(V \circ W)\|_2 \|Z\|_F
\]  
(4.27)

\[
\|y \otimes (P \circ (Q \circ S)T(V \circ W)Z)\|_F \leq \|y\|_F \|P\|_F \|Q\|_F \|R \circ S\| \|T(V \circ W)\|_2 \|Z\|_F
\]  
(4.28)

\[
\|y \otimes (P \circ (Q \circ S)T(V \circ W)Z)\|_F \leq \|y\|_F \|P\|_F \|Q\|_F \|R \circ S\| \|T(V \circ W)\|_2 \|Z\|_F
\]  
(4.29)

\[
\|y \otimes (P \circ (Q \circ S)T(V \circ W)Z)\|_F \leq \|y\|_F \|P\|_F \|Q\|_F \|R \circ S\| \|T(V \circ W)\|_2 \|Z\|_F
\]  
(4.30)

\[
\|y \otimes (P \circ (Q \circ S)T(V \circ W)Z)\|_F \leq \|y\|_F \|P\|_F \|Q\|_F \|R \circ S\| \|T(V \circ W)\|_2 \|Z\|_F
\]  
(4.31)

Proof.

In view of (4.23) and of $\|KL\|_F \leq \|K\|_F \|L\|_F$ and $\|KL\|_F \leq \|K\|_2 \|L\|_F$, for any $K \in \mathcal{C}^{k \times l}$, $L \in \mathcal{C}^{l \times p}$, $k, l, p \in \mathbb{Z}_+$ [83], the inequalities below follow from

\[
\|P \circ (Q \circ S)T(V \circ W)Z\|_F.
\]

\[
\|y \otimes (P \circ (Q \circ S)T(V \circ W)Z)\|_F \leq \|y\|_F \|P\|_F \|Q\|_F \|R \circ S\| \|T(V \circ W)\|_2 \|Z\|_F.
\]

Hence, (4.27) has been proven.

\[
\|y \otimes (P \circ (Q \circ S)T(V \circ W)Z)\|_F \leq \|y\|_F \|P\|_F \|Q\|_F \|R \circ S\| \|T(V \circ W)\|_2 \|Z\|_F.
\]

Hence, (4.28) has been proven.

\[
\|y \otimes (P \circ (Q \circ S)T(V \circ W)Z)\|_F \leq \|y\|_F \|P\|_F \|Q\|_F \|R \circ S\| \|T(V \circ W)\|_2 \|Z\|_F.
\]

Hence, (4.29) has been proven.

\[
\|y \otimes (P \circ (Q \circ S)T(V \circ W)Z)\|_F \leq \|y\|_F \|P\|_F \|Q\|_F \|R \circ S\| \|T(V \circ W)\|_2 \|Z\|_F.
\]

Hence, (4.30) has been proven.
Hence, (4.31) has been proven.

\[ \mathcal{H} \]

\[ \text{Theorem 4.2:} \]

Let \( y_i \in \mathbb{C} \), \( P \in \mathbb{C}^{n \times p} \), \( Q \in \mathbb{C}^{n \times n} \), \( R, S, V, W \in \mathbb{C}^{n \times p} \), \( T \in \mathbb{C}^{p \times n} \), \( Z \in \mathbb{C}^{p \times p} \), \( i, n, p, v \in \mathbb{Z}_+ \) and let their respective perturbations be \( \delta y \in \mathbb{C} \), \( \delta P \in \mathbb{C}^{n \times p} \), \( \delta Q \in \mathbb{C}^{n \times n} \), \( \delta R, \delta S, \delta V, \delta W \in \mathbb{C}^{n \times p} \), \( \delta T \in \mathbb{C}^{p \times n} \), \( \delta Z \in \mathbb{C}^{p \times p} \), such that \( |\delta y_i| \leq \gamma_{c_1}|y_i| \), \( \forall i \in \mathbb{Z}_+ \),

\[ |\delta P| \leq \gamma_{c_p}|P|, \quad |\delta Q| \leq \gamma_{c_n}|Q|, \quad |\delta R| \leq \gamma_{c_p}|R|, \quad |\delta S| \leq \gamma_{c_p}|S|, \quad |\delta V| \leq \gamma_{c_p}|V|, \quad |\delta W| \leq \gamma_{c_p}|W|. \]

\[ |\delta T| \leq \gamma_{c_n}|T|, \quad |\delta Z| \leq \gamma_{c_p}|Z|, \] where \( \gamma_{c_k} \) is defined by (4.18) for \( k = n, p, 1 \). Then for sufficient small \( u_r \), the following upper bound on the rounding error exists:

\[ \left\| f \left( \prod_{i=1}^{v} y_i \right) \otimes (P \circ (Q \circ S)T(V \circ W)Z) \right\|_F - \left\| \prod_{i=1}^{v} y_i \right\|_F \]

\[ \leq \left[ \gamma_{c_1} \left( \sum_{i=1}^{v} \text{sign}(\|\delta y_i\|_F) \right) \prod_{j=1}^{v} y_j \right] \]

\[ + \left( \text{sign}(\|\delta P\|_F) \gamma_{c_p} + \text{sign}(\|\delta Q\|_F) \gamma_{c_n} + \text{sign}(\|\delta Z\|_F) \gamma_{c_p} \right) \prod_{i=1}^{v} y_i \]

\[ \times \|P\|_F \left\| Q(R \circ S)T(V \circ W) \right\|_2 \|Z\|_F \]

\[ + \left( \text{sign}(\|\delta R\|_F) + \text{sign}(\|\delta S\|_F) \right) \gamma_{c_p} \prod_{i=1}^{v} y_i \]

\[ \|P\|_F \left\| Q(R \circ S)T \right\|_2 \|V \circ W\|_2 \|Z\|_F \]

\[ + \left( \text{sign}(\|\delta V\|_F) + \text{sign}(\|\delta W\|_F) \right) \gamma_{c_p} \prod_{i=1}^{v} y_i \]

\[ \|P\|_F \left\| Q(R \circ S)T \right\|_2 \|V \|_2 \|W\|_F \|Z\|_F \].

(4.32)

Furthermore,
\[
\left\| f^\prime \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ \left( Q (R \circ S) T (V \circ W) Z \right) \right) \right\|_F \\
\leq \gamma_c \left( \sum_{i=1}^{v} \text{sign}\left( \| \delta y_i \|_F \right) \left\| y_i \prod_{j=1, j \neq i}^{v} y_j \right\|_F \right) \\
+ \text{sign}\left( \| \delta \|_F \right) \gamma_c \left\| \prod_{i=1}^{v} y_i \right\|_F \left\| P \right\|_F \left\| Q \right\|_F \left\| R \right\|_F \left\| S \right\|_F \left\| T \right\|_F \left\| V \circ W \right\|_2 \left\| Z \right\|_F \\
+ \text{sign}\left( \| \delta T \|_F \right) \gamma_c \left\| \prod_{i=1}^{v} y_i \right\|_F \left\| P \right\|_F \left\| Q (R \circ S) \right\|_2 \left\| T \right\|_F \left\| V \circ W \right\|_2 \left\| Z \right\|_F \\
+ \text{sign}\left( \| \delta V \|_F \right) + \text{sign}\left( \| \delta W \|_F \right) \gamma_c \gamma_c \left\| \prod_{i=1}^{v} y_i \right\|_F \left\| P \right\|_F \left\| Q (R \circ S) \right\|_2 \left\| T \right\|_F \left\| V \circ W \right\|_2 \left\| Z \right\|_F \\
+ \left\| \prod_{i=1}^{v} y_i \right\|_F \left\| P \circ \left( Q (R \circ S) T (V \circ W) Z \right) \right\|_F .
\] (4.33)

\textbf{Proof.}

As in the proof of theorem 4.1, considering the perturbation of
\[
\prod_{i=1}^{v} y_i \otimes \left( P \circ \left( Q (R \circ S) T (V \circ W) Z \right) \right)
\] and drooping the resulting terms of second and higher order ([78], [99]), we have
\[
\left\| f^\prime \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ \left( Q (R \circ S) T (V \circ W) Z \right) \right) \right\|_F
\]
\[
\begin{align*}
&= \left( \prod_{i=1}^{v} (y_i + \delta y_i) \right) \otimes \left( (P + \delta P) \circ (Q + \delta Q)(R + \delta R)(S + \delta S)(T + \delta T)(V + \delta V)(W + \delta W)+(Z + \delta Z) \right) \\
&= \left( \prod_{i=1}^{v} y_i \right) \otimes \left( (P \circ (Q \circ S)T(V \circ W)Z) + \sum_{j=1}^{v} \delta y_i \prod_{j \neq i}^{v} y_j \right) \otimes \left( (P \circ (Q \circ S)T(V \circ W)Z) \right) \\
&+ \left( \prod_{i=1}^{v} y_i \right) \otimes \left( \delta P \circ (Q \circ S)T(V \circ W)Z \right) + \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ (Q \circ S)T\delta(V \circ W)Z \right) \\
&+ \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ (Q \circ S)\delta(V \circ W)Z \right) + \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ (Q \circ S)T(V \circ W)\delta Z \right) \\
&+ \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ (Q \circ S)T(V \circ W)\delta Z \right) + \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ (Q \circ S)T(V \circ W)\delta Z \right) \\
&\Rightarrow f \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ (Q \circ S)T(V \circ W)Z \right) - \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ (Q \circ S)T(V \circ W)Z \right) \\
&= \left( \sum_{i=1}^{v} \delta y_i \prod_{j \neq i}^{v} y_j \right) \otimes \left( (P \circ (Q \circ S)T(V \circ W)Z) \right) + \left( \prod_{i=1}^{v} y_i \right) \otimes \left( \delta P \circ (Q \circ S)T(V \circ W)Z \right) \\
&+ \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ (Q \circ S)\delta(V \circ W)Z \right) + \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ (Q \circ S)T(V \circ W)\delta Z \right) \\
&+ \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ (Q \circ S)T(V \circ W)\delta Z \right) + \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ (Q \circ S)T(V \circ W)\delta Z \right) \\
&\Rightarrow \left\| f \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ (Q \circ S)T(V \circ W)Z \right) - \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ (Q \circ S)T(V \circ W)Z \right) \right\|_{F}
\end{align*}
\]
\begin{align*}
\leq & \left\| \delta_y \otimes \left( P \circ (Q(R \circ S)T(V \circ W)Z) \right) \right\|_F + \left\| \prod_{i=1}^{v} y_i \otimes \left( \delta P \circ (Q(R \circ S)T(V \circ W)Z) \right) \right\|_F \\
& + \left\| \prod_{i=1}^{v} y_i \otimes \left( P \circ (\delta Q(R \circ S)T(V \circ W)Z) \right) \right\|_F + \left\| \prod_{i=1}^{v} y_i \otimes \left( P \circ (Q(R \circ S)\delta T(V \circ W)Z) \right) \right\|_F \\
& + \left\| \prod_{i=1}^{v} y_i \otimes \left( P \circ (Q(R \circ S)T(\delta V \circ W)Z) \right) \right\|_F + \left\| \prod_{i=1}^{v} y_i \otimes \left( P \circ (Q(R \circ S)T(V \circ \delta W)Z) \right) \right\|_F \\
& + \left\| \prod_{i=1}^{v} y_i \otimes \left( P \circ (Q(R \circ S)T(V \circ W)\delta Z) \right) \right\|_F.
\end{align*}

(4.34)

Now, since $|\delta y_i| \leq \gamma_{c_1} y_i$, \(i \in Z_+\), $|\delta P| \leq \gamma_{c_2} |P|$, $|\delta Q| \leq \gamma_{c_1} |Q|$, $|\delta R| \leq \gamma_{c_2} |R|$, $|\delta S| \leq \gamma_{c_2} |S|$, $|\delta V| \leq \gamma_{c_2} |V|$, $|\delta W| \leq \gamma_{c_2} |W|$, $|\delta T| \leq \gamma_{c_1} |T|$, $|\delta Z| \leq \gamma_{c_2} |Z|$, it follows that

$\left\| \delta y_i \right\|_F \leq \gamma_{c_1} \left\| y_i \right\|_F$, \(i \in Z_+\), $\left\| \delta P \right\|_F \leq \gamma_{c_2} \left\| P \right\|_F$, $\left\| \delta Q \right\|_F \leq \gamma_{c_1} \left\| Q \right\|_F$, $\left\| \delta R \right\|_F \leq \gamma_{c_2} \left\| R \right\|_F$, $\left\| \delta S \right\|_F \leq \gamma_{c_2} \left\| S \right\|_F$, $\left\| \delta V \right\|_F \leq \gamma_{c_2} \left\| V \right\|_F$, $\left\| \delta W \right\|_F \leq \gamma_{c_2} \left\| W \right\|_F$, $\left\| \delta T \right\|_F \leq \gamma_{c_1} \left\| T \right\|_F$, $\left\| \delta Z \right\|_F \leq \gamma_{c_2} \left\| Z \right\|_F$. Taking into account the last norm-wise inequalities and (4.27)-(4.31) from lemma 4.7, it follows from (4.34) that

$\left\| \prod_{i=1}^{v} y_i \otimes \left( P \circ (Q(R \circ S)T(V \circ W)Z) \right) \right\|_F - \left\| \prod_{i=1}^{v} y_i \otimes \left( P \circ (Q(R \circ S)T(V \circ W)Z) \right) \right\|_F$

$\leq \sum_{j=1}^{v} \left( \delta y_j \prod_{j=1}^{v} y_j \right) \left\| P \otimes \left( Q(R \circ S)T(V \circ W) \right) \right\|_F$
\[ + \prod_{i=1}^{v} y_i \left\| P \right\|_F \left\| Q \right\|_F \left\| R \right\|_F \left\| \mathbf{\delta S} \right\|_F \left\| T(V \circ W) \right\|_2 \left\| Z \right\|_F \]

\[ + \prod_{i=1}^{v} y_i \left\| P \right\|_F \left\| Q(R \circ S) \right\|_2 \left\| \mathbf{\delta T} \right\|_F \left\| (V \circ W) \right\|_2 \left\| Z \right\|_F \]

\[ + \prod_{i=1}^{v} y_i \left\| P \right\|_F \left\| Q(R \circ S)T \right\|_2 \left\| \mathbf{\delta V} \right\|_F \left\| W \right\|_F \left\| Z \right\|_F \]

\[ + \prod_{i=1}^{v} y_i \left\| P \right\|_F \left\| Q(R \circ S)T(V \circ W) \right\|_2 \left\| \mathbf{\delta Z} \right\|_F \]

\[ \Rightarrow \left\| f \left( \prod_{i=1}^{v} y_i \right) \otimes \left( P \circ (Q(R \circ S)T(V \circ W)Z) \right) \right\|_F \leq \gamma c_1 \left( \sum_{i=1}^{v} \text{sign} \left( \left\| \delta y_i \right\|_F \right) \right) \prod_{i=1}^{v} y_i \]

\[ \left( \text{sign} \left( \left\| \mathbf{\delta P} \right\|_F \right) \gamma c_p + \text{sign} \left( \left\| \mathbf{\delta Q} \right\|_F \right) \gamma c_n + \text{sign} \left( \left\| \mathbf{\delta Z} \right\|_F \right) \gamma c_p \right) \prod_{i=1}^{v} y_i \]

\[ \times \left\| P \right\|_F \left\| Q(R \circ S)T(V \circ W) \right\|_2 \left\| Z \right\|_F \]

\[ + \left( \text{sign} \left( \left\| \mathbf{\delta R} \right\|_F \right) + \text{sign} \left( \left\| \mathbf{\delta S} \right\|_F \right) \right) \gamma c_p \prod_{i=1}^{v} y_i \left\| P \right\|_F \left\| Q \right\|_F \left\| R \right\|_F \left\| S \right\|_F \left\| T \right\|_F \left\| W \right\|_F \left\| Z \right\|_F \]

\[ + \left( \text{sign} \left( \left\| \mathbf{\delta R} \right\|_F \right) \right) \gamma c_n \prod_{i=1}^{v} y_i \left\| P \right\|_F \left\| Q(R \circ S) \right\|_2 \left\| T \right\|_F \left\| V \right\|_F \left\| W \right\|_F \left\| Z \right\|_F \]

\[ + \left( \text{sign} \left( \left\| \mathbf{\delta V} \right\|_F \right) + \text{sign} \left( \left\| \mathbf{\delta W} \right\|_F \right) \right) \gamma c_p \prod_{i=1}^{v} y_i \left\| P \right\|_F \left\| Q(R \circ S)T \right\|_2 \left\| V \right\|_F \left\| W \right\|_F \left\| Z \right\|_F \]

Hence, (4.32) has been proven.

Finally,
The last inequality in view of (4.32), implies (4.33).

**Theorem 4.3**:

Let \( y_i \in C \), \( P, V \in C^{n \times p} \), \( Q, S \in C^{1 \times n} \), \( R, T \in C^n \), \( i, n, p, v \in \mathbb{Z}_+ \) and let their respective perturbations be \( \delta y \in C \), \( \delta P, \delta V \in C^{n \times p} \), \( \delta Q, \delta S \in C^{1 \times n} \), \( \delta R, \delta T \in C^n \), such that

\[
|\delta y_i| \leq \gamma_{c_1} |y_i|, \quad \forall i \in \mathbb{Z}_+, \quad |\delta P| \leq \gamma_{c_p} |P|, \quad |\delta V| \leq \gamma_{c_p} |V|, \quad |\delta Q| \leq \gamma_{c_p} |Q|, \quad |\delta S| \leq \gamma_{c_p} |S|,
\]

\[
|\delta R| \leq \gamma_{c_p} |R|, \quad |\delta T| \leq \gamma_{c_p} |T|, \quad \text{where } \gamma_{c_k} \text{ is defined by (4.18) for } k = n, p, 1. \] Then for sufficient small \( u_{\tau} \), the following upper bound on the rounding error exists:

\[
\left\| f^L \left( \prod_{i=1}^{v} y_i \otimes (P \circ (Q \circ (R \circ S) T (V \circ W)Z)) \right) - \left( \prod_{i=1}^{v} y_i \right) \otimes (P \circ ((QRT) \otimes V)) \right\|_F.
\]
\[
\leq \left( \gamma_{c_1} \sum_{i=1}^{v} \left( \text{sign}(\|\delta_i\|_F)\gamma_1 \prod_{j=1}^{v} y_j \right) \right) + (\text{sign}(\|\delta\|_F) + \text{sign}(\|\delta V\|_F)) \gamma_p + \text{sign}(\|\delta T\|_F) \gamma_{c_1} \prod_{i=1}^{v} y_i \|P\|_F \|QR\|_2 \|T\|_F \|V\|_F \\
+ \text{sign}(\|\delta Q\|_F) \gamma_{c_n} \prod_{i=1}^{v} y_i \|P\|_F \|Q\|_2 \|R\|_F \|ST\|_2 \|V\|_F \\
+ \text{sign}(\|\delta S\|_F) \gamma_{c_n} \prod_{i=1}^{v} y_i \|P\|_F \|QR\|_2 \|S\|_F \|T\|_F \|V\|_F . \tag{4.35}
\]

Furthermore,

\[
\|f\left( \prod_{i=1}^{v} y_i \right) \otimes (P \circ ((QRST) \otimes V)) \|_F
\]

\[
\leq \left( \gamma_{c_1} \sum_{i=1}^{v} \left( \text{sign}(\|\delta_i\|_F)\gamma_1 \prod_{j=1}^{v} y_j \right) \right) + (\text{sign}(\|\delta\|_F) + \text{sign}(\|\delta V\|_F)) \gamma_p + \text{sign}(\|\delta T\|_F) \gamma_{c_1} \prod_{i=1}^{v} y_i \|P\|_F \|QR\|_2 \|T\|_F \|V\|_F \\
+ \text{sign}(\|\delta Q\|_F) \gamma_{c_n} \prod_{i=1}^{v} y_i \|P\|_F \|Q\|_2 \|R\|_F \|ST\|_2 \|V\|_F \\
+ \text{sign}(\|\delta S\|_F) \gamma_{c_n} \prod_{i=1}^{v} y_i \|P\|_F \|QR\|_2 \|S\|_F \|T\|_F \|V\|_F \\
+ \left( \prod_{i=1}^{v} y_i \right) \otimes (P \circ ((QRST) \otimes V)) \|_F . \tag{4.36}
\]
Proof.

As in the proof of theorem 4.1, considering the perturbation of
\[
\left( \prod_{i=1}^{V} y_i \right) \otimes \left( P \circ ((Q \ast R \ast S \ast T) \otimes V) \right)
\]
and drooping the resulting terms of second and higher order
([78], [99]), we have

\[
f_\ell \left( \left[ \prod_{i=1}^{V} y_i \right] \otimes (P \circ ((Q \ast R \ast S \ast T) \otimes V)) \right) \\
= \left[ \prod_{i=1}^{V} (y_i + \delta y_i) \right] \otimes \left( (P + \delta P \circ ((Q + \delta Q)(R + \delta R)(S + \delta S)(T + \delta T)) \otimes (V + \delta V)) \right) \\
= \left[ \prod_{i=1}^{V} y_i \right] \otimes (P \circ ((Q \ast R \ast S \ast T) \otimes V)) + \sum_{i=1}^{V} \delta y_i \prod_{j=1}^{V} y_j \otimes (P \circ ((Q \ast R \ast S \ast T) \otimes V)) + \left[ \prod_{i=1}^{V} y_i \right] \otimes (\delta P \circ ((Q \ast R \ast S \ast T) \otimes V)) + \left[ \prod_{i=1}^{V} y_i \right] \otimes (P \circ ((Q \ast R \ast S \ast T) \otimes \delta V)) \\
= \left[ \prod_{i=1}^{V} y_i \right] \otimes (P \circ ((Q \ast R \ast S \ast T) \otimes V)) + \sum_{i=1}^{V} \delta y_i \prod_{j=1}^{V} y_j \otimes (P \circ ((Q \ast R \ast S \ast T) \otimes V)) + \left[ \prod_{i=1}^{V} y_i \right] \otimes (\delta P \circ ((Q \ast R \ast S \ast T) \otimes V)) + \left[ \prod_{i=1}^{V} y_i \right] \otimes (P \circ ((Q \ast R \ast S \ast T) \otimes \delta V)) \\
\Rightarrow f_\ell \left( \left[ \prod_{i=1}^{V} y_i \right] \otimes (P \circ ((Q \ast R \ast S \ast T) \otimes V)) \right) = \left[ \prod_{i=1}^{V} y_i \right] \otimes (P \circ ((Q \ast R \ast S \ast T) \otimes V)) + \sum_{i=1}^{V} \delta y_i \prod_{j=1}^{V} y_j \otimes (P \circ ((Q \ast R \ast S \ast T) \otimes V)) + \left[ \prod_{i=1}^{V} y_i \right] \otimes (\delta P \circ ((Q \ast R \ast S \ast T) \otimes V)) + \left[ \prod_{i=1}^{V} y_i \right] \otimes (P \circ ((Q \ast R \ast S \ast T) \otimes \delta V)) \\
= \sum_{i=1}^{V} \left( \delta y_i \prod_{j=1}^{V} y_j \right) \otimes (P \circ ((Q \ast R \ast S \ast T) \otimes V)) + \left[ \prod_{i=1}^{V} y_i \right] \otimes (\delta P \circ ((Q \ast R \ast S \ast T) \otimes V)) + \left[ \prod_{i=1}^{V} y_i \right] \otimes (P \circ ((Q \ast R \ast S \ast T) \otimes \delta V)) + \left[ \prod_{i=1}^{V} y_i \right] \otimes (P \circ ((Q \ast R \ast S \ast T) \otimes \delta V))
\]
\[
\Rightarrow \left\| f \left( \prod_{i=1}^{V} y_i \right) \otimes \left( P \circ ((QRST) \otimes V) \right) - \left( \prod_{i=1}^{V} y_i \right) \otimes \left( P \circ ((QRST) \otimes V) \right) \right\|_F \\
\leq \left\| \sum_{i=1}^{V} \left( \delta y_i \prod_{j=1}^{V} y_j \right) \otimes \left( P \circ ((QRST) \otimes V) \right) \right\|_F + \left\| \prod_{i=1}^{V} y_i \right\|_F \otimes \left( \delta Q \circ ((QRST) \otimes V) \right) \right\|_F \\
+ \left\| \prod_{i=1}^{V} y_i \right\|_F \otimes \left( P \circ ((Q\delta ST) \otimes V) \right) \right\|_F + \left\| \prod_{i=1}^{V} y_i \right\|_F \otimes \left( P \circ ((QRST) \otimes V) \right) \right\|_F \\
+ \left\| \prod_{i=1}^{V} y_i \right\|_F \otimes \left( P \circ ((QRST) \otimes \delta V) \right) \right\|_F .
\]

Now, since \(|\delta y_i| \leq \gamma c_1 |y_i|, \forall i \in \mathbb{Z}_+, |\delta \mathcal{P}| \leq \gamma c_p |P|, |\delta V| \leq \gamma c_p |V|, |\delta Q| \leq \gamma c_n |Q|,
|\delta S| \leq \gamma c_p |S|, |\delta R| \leq \gamma c_p |R|, |\delta T| \leq \gamma c_n |T|,\) it follows that \(\|\delta y_i\|_F \leq \gamma c_1 \|y_i\|_F, \forall i \in \mathbb{Z}_+,\)
\([4.37]\) from lemma 4.7, (4.27) - (4.31) from lemma 4.7, it follows from (4.37) that
\[
\left\| f \left( \prod_{i=1}^{V} y_i \right) \otimes \left( P \circ ((QRST) \otimes V) \right) - \left( \prod_{i=1}^{V} y_i \right) \otimes \left( P \circ ((QRST) \otimes V) \right) \right\|_F \\
\leq \left\| \sum_{i=1}^{V} \left( \delta y_i \prod_{j=1}^{V} y_j \right) \right\|_F \|P\|_F \|QRS\|_2 \|T\|_F \|V\|_F + \left\| \prod_{i=1}^{V} y_i \right\|_F \|\delta \mathcal{P}\|_F \|QRS\|_2 \|T\|_F \|V\|_F \\
+ \left\| \prod_{i=1}^{V} y_i \right\|_F \|P\|_F \|\delta Q\|_F \|RS\|_2 \|T\|_F \|V\|_F + \left\| \prod_{i=1}^{V} y_i \right\|_F \|P\|_F \|Q\|_2 \|\delta R\|_F \|ST\|_2 \|V\|_F
\[+ \prod_{i=1}^{\nu} y_i \left\| P \right\|_F \left\| QR \right\|_2 \left\| \delta S \right\|_F \left\| T \right\|_F \left\| V \right\|_F + \prod_{i=1}^{\nu} y_i \left\| P \right\|_F \left\| QRS \right\|_2 \left\| \delta T \right\|_2 \left\| V \right\|_F \]

\[+ \prod_{i=1}^{\nu} y_i \left\| P \right\|_F \left\| QRS \right\|_2 \left\| T \right\|_F \left\| V \right\|_F. \]

\[\Rightarrow \left\| \prod_{i=1}^{\nu} y_i \left( P \circ (QRST) \otimes V \right) - \prod_{i=1}^{\nu} y_i \left( P \circ (QRST) \otimes V \right) \right\|_F \]

\[\leq \gamma_{c_1} \sum_{i=1}^{\nu} \left( \left\| \delta y_i \right\|_F \right) \left\| y_i \prod_{j=1, j \neq i}^{\nu} y_j \right\|_F \]

\[\leq \left( \left\| \delta y \right\|_F \right) \gamma_{c_1} \left\| \prod_{i=1}^{\nu} y_i \right\|_F \left\| P \right\|_F \left\| QR \right\|_2 \left\| R \right\|_F \left\| V \right\|_F \]

\[+ \left\| \delta y \right\|_F \gamma_{c_1} \left\| \prod_{i=1}^{\nu} y_i \right\|_F \left\| P \right\|_F \left\| QRS \right\|_2 \left\| S \right\|_F \left\| T \right\|_F \left\| V \right\|_F. \]

Hence, (4.35) has been proven.

Finally,

\[f_l \left( \prod_{i=1}^{\nu} y_i \left( P \circ (QRST) \otimes V \right) \right) = f_l \left( \prod_{i=1}^{\nu} y_i \left( P \circ (QRST) \otimes V \right) \right) \]

\[- \left( \prod_{i=1}^{\nu} y_i \left( P \circ (QRST) \otimes V \right) \right) + \left( \prod_{i=1}^{\nu} y_i \left( P \circ (QRST) \otimes V \right) \right) \]

\[\Rightarrow \left\| f_l \left( \prod_{i=1}^{\nu} y_i \left( P \circ (QRST) \otimes V \right) \right) \right\|_F = \left\| \prod_{i=1}^{\nu} y_i \left( P \circ (QRST) \otimes V \right) \right\|_F \]

\[+ f_l \left( \prod_{i=1}^{\nu} y_i \left( P \circ (QRST) \otimes V \right) \right) - \left( \prod_{i=1}^{\nu} y_i \left( P \circ (QRST) \otimes V \right) \right) \left\| \right. \]
The last inequality in view of (4.35), implies (4.36).

Lemma 4.8:

Let $K_i \in \mathbb{C}^{k \times l}$, $\forall k, l \in \mathbb{Z}_+$, $i \in \mathbb{Z}_+$. Then for sufficiently small $u_r$ and $|\delta| \leq u_r$, $\forall n \in \mathbb{Z}_+-\{1\}$,

$$
\left\| \| \left( \prod_{i=1}^{n} \delta K_i \right) - \sum_{i=1}^{n} K_i \right\|_F \leq \sum_{i=1}^{n} \left\| \delta^2 K_i \right\|_F.
$$

Proof.

Denoting by $\delta K_i$ and $\delta^2 K_i$, the first and second order perturbed terms of the perturbation of $K_i$, we have by construction

$$
\left\| \| \left( \prod_{i=1}^{n} \delta K_i \right) - \sum_{i=1}^{n} K_i \right\|_F = \left\| \| \left( \prod_{i=1}^{n} (K_i + \delta K_i) \right) - \sum_{i=1}^{n} K_i \right\|_F
$$

$$
= \sum_{i=1}^{n} (K_i + \delta K_i) + \sum_{i=1}^{n} \delta^2 K_i = \sum_{i=1}^{n} K_i + \sum_{i=1}^{n} \delta K_i + \sum_{i=1}^{n} \delta^2 K_i
$$

$$
\Rightarrow \left\| \| \left( \prod_{i=1}^{n} \delta K_i \right) - \sum_{i=1}^{n} K_i \right\|_F = \sum_{i=1}^{n} \delta K_i + \sum_{i=1}^{n} \delta^2 K_i
$$

$$
\Rightarrow \left\| \| \left( \prod_{i=1}^{n} \delta K_i \right) - \sum_{i=1}^{n} K_i \right\|_F = \sum_{i=1}^{n} \delta K_i + \sum_{i=1}^{n} \delta^2 K_i \leq \sum_{i=1}^{n} \delta K_i + \sum_{i=1}^{n} \delta^2 K_i.
$$
Now, for sufficiently small $u_r$ and since $|\varphi| \leq u_r$, we have $\delta^2 K_i \approx 0$ for all $i$. Also, by convention $f_l(K_i) = K_i + \delta K_i \Rightarrow \delta K_i = f_l(K_i) - K_i$ and therefore the last inequality becomes equivalently

$$
\left\| f_l \left( \sum_{i=1}^{n} f_l(K_i) \right) - \sum_{i=1}^{n} K_i \right\|_F \leq \left\| \sum_{i=1}^{n} (f_l(K_i) - K_i) \right\|_F \leq \sum_{i=1}^{n} \|f_l(K_i) - K_i\|_F.
$$

**Remark 4.1:**

It is obvious that in all the previous formulations for the rounding error bounds, if a matrix is absent from a formula, then the formula holds by removing any resulted terms subject to absent matrices. However, in order to avoid pessimism, it is proper to evaluate the above bounds for each special problem separately, following similar methodology as in the previous sections.

**Remark 4.2:**

(4.38) in lemma 4.8 reveals the accumulating feature of rounding errors in summation operations. This is something that makes sense and is expected, because summation is a linear computational process.

The evaluation of rounding error bounds for the homotopy equations (3.95) and (3.98) follows in the next subsection.

### 4.2.3 Rounding Error Estimates for the Homotopy Equations (3.95) and (3.98)

Consider the homotopy equations (3.95) and (3.98), as defined in chapter 3. Note that both equations have the general form $H_{\alpha \beta}(\hat{x}, \varepsilon) = e\hat{F}(\hat{x}) - \gamma(1 - \varepsilon)\hat{F}_0(\hat{x})$ and that they are vector equations. In order to make the analysis consistent with the results of the previous subsections, instead of (3.95) and (3.98), the matrix forms of these equations will be used. At this point note that these results hold for the vector forms of the respective matrix quantities. This is obvious since $\text{vec} H_{\alpha \beta}(\hat{x}, \varepsilon) = H_{\alpha \beta}(\hat{x}, \varepsilon)$ and $\|\text{vec}()\|_F = \|\text{vec}^{-1}()\|_F$. This makes the analysis more convenient in terms of presentation as well.

According to the above and in view of (3.88), consider the equivalent matrix form of (3.95):

$$
\hat{H}_{\alpha \beta}(\hat{x}, \varepsilon) = \sum_{i=1}^{\omega} \varepsilon \otimes \left( E_{10}(U_C) \circ \left( A_i | E_{10}(U_X) \circ \hat{X} \right) B_i \right) \otimes \left( A_2 \text{vec}(\hat{x}) \right).
$$
\[ + \sum_{i=1}^{\omega} \left( \varepsilon \otimes \left( E_{10}(U_C) \circ \left( A_i \left( E_{10}(U_X) \circ \hat{X} \right) B_i \right) \otimes a_j(m+1) \right) \right) \]
\[ + \sum_{i=1}^{\xi} \left( \varepsilon \otimes \left( E_{10}(U_C) \circ \left( C_i \left( E_{10}(U_X) \circ \hat{X} \right) D_i \left( E_{10}(U_X) \circ \hat{X} \right) E_i \right) \right) \right) \]
\[ + \varepsilon \otimes \left( E_{10}(U_C) \circ \left( A_{a \vec{v} e c}(\hat{X}) A_{a \vec{v} e c}(\hat{X}) \right) \otimes G \right) \]
\[ + \varepsilon \otimes \left( E_{10}(U_C) \circ \left( G \otimes \left( a_{j(m+1)} a_{j(m+1)} \right) \right) \right) \]
\[ + \left( 2 \varepsilon a_{j(m+1)} \right) \otimes \left( E_{10}(U_C) \circ \left( G \otimes \left( A_{a \vec{v} e c}(\hat{X}) \right) \right) \right) \]
\[ - \gamma \otimes \left( A \circ \hat{X} \right) \otimes \left( A_{a \vec{v} e c}(\hat{X}) \right) + (\gamma \varepsilon) \otimes \left( A \circ \hat{X} \right) \otimes \left( A_{a \vec{v} e c}(\hat{X}) \right) \]
\[ + \gamma \otimes \left( \left( A_{a \vec{v} e c}(\hat{X}) A_{a \vec{v} e c}(\hat{X}) \right) \otimes B \right) \]
\[ + \gamma \otimes \left( B \otimes \left( a_{j(m+1)} a_{j(m+1)} \right) \right) + \left( 2 \gamma a_{j(m+1)} \right) \otimes \left( B \otimes \left( A_{a \vec{v} e c}(\hat{X}) \right) \right) \]
\[ - (\gamma \varepsilon) \otimes \left( \left( A_{a \vec{v} e c}(\hat{X}) A_{a \vec{v} e c}(\hat{X}) \right) \otimes B \right) \]
\[ + (\gamma \varepsilon) \otimes \left( B \otimes \left( a_{j(m+1)} a_{j(m+1)} \right) \right) + \left( 2 \gamma a_{j(m+1)} \right) \otimes \left( B \otimes \left( A_{a \vec{v} e c}(\hat{X}) \right) \right), \quad (4.39) \]

where \( \hat{X} = vec^{-1}(\hat{x}) \), \( A := vec^{-1}(\alpha) \), \( B := vec^{-1}(\beta) \), \( \hat{X}, A, B \in \mathbb{C}^{n \times p} \), \( m = np \).

Applying theorems 4.1-4.3 accordingly to each single term of (4.39) and unifying the results under lemma 4.8, the following upper bound on the rounding error is obtained:

\[ \| \hat{H}_{\alpha \beta} (\hat{X}, \varepsilon) - \hat{H}_{\alpha \beta} (\hat{X}, \varepsilon) \|_F \]
\[ \leq \sum_{i=1}^{\omega} \left( \left( \text{sign}(\| e \|_F) + \text{sign}(\| \vec{e} \|_F) \right) \gamma_{c_1} + \left( \text{sign}(\| \vec{e} \|_F) + \text{sign}(\| \vec{e} \|_F) \right) \gamma_{c_p} \right) \]
\[ \times \| e \|_F \| E_{10}(U_C) \|_F \| A_i \|_F \| E_{10}(U_X) \circ \hat{X} \|_2 \| B_i \|_F \| A_{a \vec{v} e c} \|_F \| vec(\hat{X}) \|_F \]
\[ + \| e \|_F \| E_{10}(U_C) \|_F \| A_i \|_F \| E_{10}(U_X) \circ \hat{X} \|_2 \| B_i \|_F \| A_{a \vec{v} e c} \|_F \| vec(\hat{X}) \|_F \]
\[ + \left( \text{sign}(\| \vec{e} \|_F) \right) \gamma_{c_p} \]
\[ \times \| e \|_F \| E_{10}(U_C) \|_F \| A_i \|_F \| E_{10}(U_X) \circ \hat{X} \|_2 \| B_i \|_F \| A_{a \vec{v} e c} \|_F \| vec(\hat{X}) \|_F \]
\[ + \text{sign}(\|\vec{A}_a\|_F)\gamma_{cn_p} \|\vec{e}\|_F \times \|\vec{E}_{10}(U:\mathcal{C})\|_F \|A_i(\mathcal{E}_{10}(U:\mathcal{X}) \circ \vec{X})\|_2 \|B_i\|_F \|\vec{A}_a\|_F \|\vec{A}_a\|_2 \|\vec{G}\|_F \]

\[ + \left( \text{sign}(\|\vec{e}\|_F) + \text{sign}(\|\vec{E}_{10}(U:\mathcal{C})\|_F) \right) \gamma_{c_1} + \left( \text{sign}(\|\vec{A}_a\|_F) + \text{sign}(\|\vec{B}_i\|_F) \right) \gamma_{c_p} \]

\[ \times \|\vec{e}\|_F \|\vec{E}_{10}(U:\mathcal{C})\|_F \|A_i(\mathcal{E}_{10}(U:\mathcal{X}) \circ \vec{X})\|_2 \|B_i\|_F \|\vec{A}_a\|_F \|\vec{A}_a\|_2 \|\vec{G}\|_F \]

\[ + \sum_{i=1}^{n} \left( \text{sign}(\|\vec{e}\|_F) \gamma_{c_1} + \text{sign}(\|\vec{E}_{10}(U:\mathcal{C})\|_F) \gamma_{c_p} \right) \]

\[ \left( \text{sign}(\|\vec{A}_a\|_F) + \text{sign}(\|\vec{B}_i\|_F) \right) \gamma_{c_1} + \left( \text{sign}(\|\vec{A}_a\|_F) + \text{sign}(\|\vec{B}_i\|_F) \right) \gamma_{c_p} \]

\[ \times \|\vec{e}\|_F \|\vec{E}_{10}(U:\mathcal{C})\|_F \|C_i(\mathcal{E}_{10}(U:\mathcal{X}) \circ \vec{X})\|_2 \|D_i\|_F \|\vec{E}_{10}(U:\mathcal{X}) \circ \vec{X}\|_2 \|E_i\|_F \]

\[ + \left( \text{sign}(\|\vec{A}_a\|_F) + \text{sign}(\|\vec{B}_i\|_F) \right) \gamma_{c_p} \]

\[ \times \|\vec{e}\|_F \|\vec{E}_{10}(U:\mathcal{C})\|_F \|C_i(\mathcal{E}_{10}(U:\mathcal{X}) \circ \vec{X})\|_2 \|D_i\|_F \|\vec{E}_{10}(U:\mathcal{X}) \circ \vec{X}\|_2 \|E_i\|_F \]

\[ + \left( \text{sign}(\|\vec{A}_a\|_F) + \text{sign}(\|\vec{B}_i\|_F) \right) \gamma_{c_1} + \left( \text{sign}(\|\vec{A}_a\|_F) + \text{sign}(\|\vec{B}_i\|_F) \right) \gamma_{c_p} \]

\[ \times \|\vec{e}\|_F \|\vec{E}_{10}(U:\mathcal{C})\|_F \|A_i \vec{a} \text{vec}(\vec{X}) \vec{A}_a\|_2 \|\vec{A}_a \text{vec}(\vec{X})\|_F \|\vec{G}\|_F \]

\[ + \text{sign}(\|\vec{A}_a\|_F)\gamma_{cn_p} \|\vec{e}\|_F \|\vec{E}_{10}(U:\mathcal{C})\|_F \|A_i \vec{a} \text{vec}(\vec{X})\|_2 \|\vec{A}_a \text{vec}(\vec{X})\|_F \|\vec{G}\|_F \]

\[ + \text{sign}(\|\vec{B}_i\|_F) \gamma_{c_1} \|\vec{e}\|_F \|\vec{E}_{10}(U:\mathcal{C})\|_F \|A_i \vec{a} \text{vec}(\vec{X})\|_2 \|\vec{A}_a \text{vec}(\vec{X})\|_F \|\vec{G}\|_F \]

\[ + \left( \text{sign}(\|\vec{A}_a\|_F) + 2 \text{sign}(\|\vec{A}_a\|_F) \right) \gamma_{c_1} + \left( \text{sign}(\|\vec{A}_a\|_F) + \text{sign}(\|\vec{B}_i\|_F) \right) \gamma_{c_p} \]
\[ x \| \epsilon \|_F \| a_{l(m+1)} \|_F^2 \| E_{10}(UC) \|_F \| G \|_F + \left( \text{sign}(\| \delta \|_F) + \text{sign}(\| \delta a_{l(m+1)} \|_F) \text{vec}(\hat{X}) \right) \gamma_{c_1} \]

\[ + \left( \text{sign}(\| \delta \|_F) + \text{sign}(\| \delta \|_F) \text{vec}(\hat{X}) \right) \gamma_{c_p} + \text{sign}(\| \delta \|_F) \gamma_{c_{np}} \]

\[ \times \| \| \epsilon \|_F \| a_{l(m+1)} \|_F \| E_{10}(UC) \|_F \| G \|_F \| a \|_F \| \text{vec}(\hat{X}) \|_F \]

\[ + \left( \text{sign}(\| \delta \|_F) \gamma_{c_1} + \left( \text{sign}(\| \delta \|_F) + \text{sign}(\| \delta \|_F) \text{vec}(\hat{X}) \right) \gamma_{c_p} + \text{sign}(\| \delta \|_F) \gamma_{c_{np}} \right) \]

\[ \times \| \| \epsilon \|_F \| a \|_F \| \text{vec}(\hat{X}) \|_F \]

\[ + \left( \text{sign}(\| \delta \|_F) + \text{sign}(\| \delta \|_F) + \text{sign}(\| \delta \|_F \text{vec}(\hat{X}) \right) \gamma_{c_1} \]

\[ + \left( \text{sign}(\| \delta \|_F) + \text{sign}(\| \delta \|_F) \text{vec}(\hat{X}) \right) \gamma_{c_p} + \text{sign}(\| \delta \|_F) \gamma_{c_{np}} \right) \]

\[ \times \| \| \epsilon \|_F \| a \|_F \| \text{vec}(\hat{X}) \|_F \]

\[ + \left( \text{sign}(\| \delta \|_F) + \text{sign}(\| \delta \|_F) \text{vec}(\hat{X}) \right) \gamma_{c_1} + \left( \text{sign}(\| \delta \|_F) + \text{sign}(\| \delta \|_F) \text{vec}(\hat{X}) \right) \gamma_{c_p} \]

\[ \times \| \| \epsilon \|_F \| a \|_F \| \text{vec}(\hat{X}) \|_F \]

\[ + \left( \text{sign}(\| \delta \|_F) + \text{sign}(\| \delta \|_F) \text{vec}(\hat{X}) \right) \gamma_{c_1} + \left( \text{sign}(\| \delta \|_F) + \text{sign}(\| \delta \|_F) \text{vec}(\hat{X}) \right) \gamma_{c_p} \]

\[ \times \| \| \epsilon \|_F \| a \|_F \| \text{vec}(\hat{X}) \|_F \]

\[ + \left( \text{sign}(\| \delta \|_F) + \text{sign}(\| \delta \|_F) \text{vec}(\hat{X}) \right) \gamma_{c_1} + \left( \text{sign}(\| \delta \|_F) + \text{sign}(\| \delta \|_F) \text{vec}(\hat{X}) \right) \gamma_{c_p} \]

\[ \times \| \| \epsilon \|_F \| a \|_F \| \text{vec}(\hat{X}) \|_F \]

\[ + \text{sign}(\| \delta \|_F) \gamma_{c_{np}} \| \| \epsilon \|_F \| a \|_F \| \text{vec}(\hat{X}) \|_F \| a \|_F \| \text{vec}(\hat{X}) \|_F \| B \|_F \]

\[ + \text{sign}(\| \delta \|_F) \gamma_{c_{np}} \| \| \epsilon \|_F \| a \|_F \| \text{vec}(\hat{X}) \|_F \| a \|_F \| \text{vec}(\hat{X}) \|_F \| B \|_F \]
Similarly, in view of (3.88) consider the equivalent matrix form of (3.98)

\[ H_{\alpha \beta}(\vec{X}, \epsilon) = \sum_{i=1}^{\alpha} \epsilon \odot \left( E_{10}(U_C) \circ \left( A_i \circ E_{10}(U_X) \circ \vec{X}_i \right) \circ A_{a_{\text{vec}}(\vec{X})} \right) \]

+ \sum_{i=1}^{\alpha} \epsilon \odot \left( E_{10}(U_C) \circ \left( A_i \circ E_{10}(U_X) \circ \vec{X}_i \right) \circ a_{l(m+1)} \right)
\[ + \sum_{i=1}^{\varepsilon} \left( \varepsilon \otimes \left( E_{10}(U_C) \circ \left( C_i \circ \left( E_{10}(U_X) \circ \hat{X} \right) D_i \circ \left( E_{10}(U_X) \circ \hat{X} \right) E_i \right) \right) \right) \]
\[ + \varepsilon \otimes \left( E_{10}(U_C) \circ \left( A_\vec{a} \vec{\text{vec}}(\hat{X}) A_\vec{a} \vec{\text{vec}}(\hat{X}) \otimes G \right) \right) \]
\[ + \varepsilon \otimes \left( E_{10}(U_C) \circ \left( G \otimes (a_i(m+1) a_i(m+1)) \right) \right) \]
\[ + (2\varepsilon a_i(m+1)) \otimes \left( E_{10}(U_C) \circ \left( G \otimes (A_\vec{a} \vec{\text{vec}}(\hat{X})) \otimes B \right) \right) \]
\[ - \gamma \otimes (A \circ \hat{X} \circ \hat{X}) + (\varepsilon) \otimes (A \circ \hat{X} \circ \hat{X}) + \gamma \otimes \left( A_\vec{a} \vec{\text{vec}}(\hat{X}) A_\vec{a} \vec{\text{vec}}(\hat{X}) \otimes B \right) \]
\[ + \gamma \otimes \left( B \otimes (a_i(m+1) a_i(m+1)) \right) + \left( 2\gamma a_i(m+1) \right) \otimes \left( B \otimes (A_\vec{a} \vec{\text{vec}}(\hat{X})) \right) \]
\[ - (\varepsilon) \otimes \left( \left( A_\vec{a} \vec{\text{vec}}(\hat{X}) A_\vec{a} \vec{\text{vec}}(\hat{X}) \otimes B \right) \right) \]
\[ + (\varepsilon) \otimes \left( B \otimes (a_i(m+1) a_i(m+1)) \right) + \left( 2\gamma a_i(m+1) \right) \otimes \left( B \otimes (A_\vec{a} \vec{\text{vec}}(\hat{X})) \right), \tag{4.41} \]

where \( \hat{X} = \text{vec}^{-1}(\hat{x}) \), \( A = \text{vec}^{-1}(\alpha) \), \( B = \text{vec}^{-1}(\beta) \), \( \hat{X}, A, B \in \mathbb{C}^{n \times p} \), \( m = np \).

Again, applying theorems 4.1-4.3 to each single term of (4.41) and unifying the results under lemma 4.8, the following upper bound on the rounding error is obtained:

\[
\| f(\hat{H}_{\alpha \beta}(\hat{X}, \varepsilon)) - \hat{H}_{\alpha \beta}(\hat{X}, \varepsilon) \|_F \leq \sum_{i=1}^{\varepsilon} \left( \left( \| \text{sign}(\| \delta \vec{a} \|_F) \| \vec{\text{vec}}(\hat{X}) \|_F \right) \gamma_c_1 + \left( \| \text{sign}(\| \delta \vec{E}_{10}(U_C) \|_F) \| \vec{\text{vec}}(\hat{X}) \|_F \right) \gamma_c_2 \right) \]
\[ \times \| \varepsilon \|_F \| \vec{E}_{10}(U_C) \|_F \| A_i \left( \vec{E}_{10}(U_X) \circ \hat{X} \right) \|_2 \| B_i \|_F \| A_\vec{a} \|_2 \| \vec{\text{vec}}(\hat{X}) \|_F \]
\[ + \| \text{sign}(\| \delta \vec{A} \|_F) \| \varepsilon \|_F \| \vec{E}_{10}(U_C) \|_F \| A_i \|_F \| E_{10}(U_X) \circ \hat{X} \|_2 \| B_i \|_F \| A_\vec{a} \|_2 \| \vec{\text{vec}}(\hat{X}) \|_F \]
\[ + \left( \| \text{sign}(\| \delta \vec{E}_{10}(U_X) \|_F) \| \hat{X} \|_F \| A_i \|_F \| E_{10}(U_X) \circ \hat{X} \|_2 \| B_i \|_F \| A_\vec{a} \|_2 \| \vec{\text{vec}}(\hat{X}) \|_F \right) \gamma_c_3 \]
\[ \times \| \varepsilon \|_F \| \vec{E}_{10}(U_C) \|_F \| A_i \|_F \| E_{10}(U_X) \circ \hat{X} \|_2 \| B_i \|_F \| A_\vec{a} \|_2 \| \vec{\text{vec}}(\hat{X}) \|_F \]
\[ + \| \text{sign}(\| \delta \vec{A} \|_F) \| \varepsilon \|_F \times \| \vec{E}_{10}(U_C) \|_F \| A_i \left( \vec{E}_{10}(U_X) \circ \hat{X} \right) \|_2 \| B_i \|_F \| A_\vec{a} \|_2 \| \vec{\text{vec}}(\hat{X}) \|_F \]
\[
\begin{align*}
&+ \left( \text{sign}(\|\Delta e\|_F) + \text{sign}(\|\vec{\delta} \mathbf{A}_{l(m+1)}\|_F) \right) \gamma_{c_1} + \left( \text{sign}(\|\mathbf{E}_{10}(\mathbf{U}_C)\|_F) + \text{sign}(\|\mathbf{B}_{l}\|_F) \right) \gamma_{c_p} \\
&\quad \times \|\varepsilon\|_F \|\mathbf{E}_{10}(\mathbf{U}_C)\|_F \|\mathbf{A}_{l}\|_F \|\mathbf{E}_{10}(\mathbf{U}_X) \triangleright \hat{\mathbf{X}}\|_2 \|\mathbf{B}_{l}\|_F \|\mathbf{A}_{l(m+1)}\|_F \\
&+ \text{sign}(\|\mathbf{B}_{l}\|_F) \gamma_{c_n} \|\varepsilon\|_F \|\mathbf{E}_{10}(\mathbf{U}_C)\|_F \|\mathbf{A}_{l}\|_F \|\mathbf{E}_{10}(\mathbf{U}_X) \triangleright \hat{\mathbf{X}}\|_2 \|\mathbf{B}_{l}\|_F \|\mathbf{A}_{l(m+1)}\|_F \\
&+ \left( \text{sign}(\|\Delta e\|_F) + \text{sign}(\|\delta \mathbf{G}\|_F) \right) \gamma_{c_p} \\
&\quad \times \|\varepsilon\|_F \|\mathbf{E}_{10}(\mathbf{U}_C)\|_F \|\mathbf{C}_l(\mathbf{E}_{10}(\mathbf{U}_X) \triangleright \hat{\mathbf{X}})\|_2 \|\mathbf{D}_l\|_F \|\mathbf{E}_{10}(\mathbf{U}_X) \triangleright \hat{\mathbf{X}}\|_2 \|\mathbf{E}_l\|_F \\
&+ \left( \text{sign}(\|\Delta e\|_F) + \text{sign}(\|\delta \mathbf{e}\|_F) \right) \gamma_{c_1} + \left( \text{sign}(\|\mathbf{E}_{10}(\mathbf{U}_C)\|_F) + \text{sign}(\|\mathbf{A}_{l}\|_F) \right) \gamma_{c_p} \\
&\quad \times \|\varepsilon\|_F \|\mathbf{E}_{10}(\mathbf{U}_C)\|_F \|\mathbf{A}_{l}\|_2 \|\mathbf{A}_{l}\|_2 \|\mathbf{A}_{l}\|_F \|\mathbf{G}\|_F \\
&+ \text{sign}(\|\delta \mathbf{a}_{l}\|_F) \gamma_{c_{np}} \|\varepsilon\|_F \|\mathbf{E}_{10}(\mathbf{U}_C)\|_F \|\mathbf{A}_{l}\|_F \|\mathbf{A}_{l}\|_F \|\mathbf{A}_{l}\|_F \|\mathbf{G}\|_F \\
&+ \text{sign}(\|\mathbf{a}_{l}\|_F) \gamma_{c_{np}} \|\varepsilon\|_F \|\mathbf{E}_{10}(\mathbf{U}_C)\|_F \|\mathbf{A}_{l}\|_F \|\mathbf{A}_{l}\|_F \|\mathbf{A}_{l}\|_F \|\mathbf{G}\|_F \\
&+ \left( \text{sign}(\|\Delta e\|_F) + 2\text{sign}(\|\Delta \mathbf{a}_{l(m+1)}\|_F) \right) \gamma_{c_1} \\
&+ \left( \text{sign}(\|\mathbf{E}_{10}(\mathbf{U}_C)\|_F) + \text{sign}(\|\delta \mathbf{G}\|_F) \right) \gamma_{c_p} \\
&\quad \times \|\varepsilon\|_F \|\mathbf{a}_{l(m+1)}\|_2^2 \|\mathbf{E}_{10}(\mathbf{U}_C)\|_F \|\mathbf{G}\|_F
\end{align*}
\]
\[ \left( \text{sign}(\parallel \delta \mathbf{e} \parallel_F) + \text{sign}(\parallel \delta \mathbf{a}(m+1) \parallel_F) + \text{sign}(\parallel \text{vec}(\hat{X}) \parallel_F) \right) \gamma_{c_1} + \left( \text{sign}(\parallel \delta \mathbf{r} \parallel_F) \gamma_{c_1} + \left( \text{sign}(\parallel \delta \mathbf{r} \parallel_F) + \text{sign}(\parallel \delta \mathbf{A} \parallel_F) \right) \gamma_{c_p} \right) \times \parallel \mathbf{r} \parallel_F \parallel \mathbf{a}(m+1) \parallel_F \parallel \mathbf{e}_{10}(\mathbf{U} \mathbf{C}) \parallel_F \parallel G \parallel_F \parallel \mathbf{A} \parallel_F \parallel \text{vec}(\hat{X}) \parallel_F \]

\[ + \left( \text{sign}(\parallel \delta \mathbf{r} \parallel_F) \gamma_{c_1} + \left( \text{sign}(\parallel \delta \mathbf{r} \parallel_F) + \text{sign}(\parallel \delta \mathbf{A} \parallel_F) \right) \gamma_{c_p} \right) \times \parallel \gamma \parallel_F \parallel \mathbf{r} \parallel_F \parallel \mathbf{A} \parallel_F \parallel \mathbf{f} \parallel_F \]

\[ + \left( \text{sign}(\parallel \delta \mathbf{r} \parallel_F) \right) \gamma_{c_1} + \left( \text{sign}(\parallel \delta \mathbf{r} \parallel_F) + \text{sign}(\parallel \delta \mathbf{A} \parallel_F) \right) \gamma_{c_p} \times \parallel \mathbf{r} \parallel_F \parallel \mathbf{a} \parallel_F \parallel \text{vec}(\hat{X}) \parallel_F \parallel \mathbf{f} \parallel_F \]

\[ + \left( \text{sign}(\parallel \delta \mathbf{r} \parallel_F) \right) \gamma_{c_1} + \left( \text{sign}(\parallel \delta \mathbf{r} \parallel_F) + \text{sign}(\parallel \delta \mathbf{A} \parallel_F) \right) \gamma_{c_p} \times \parallel \gamma \parallel_F \parallel \mathbf{r} \parallel_F \parallel \mathbf{A} \parallel_F \parallel \mathbf{f} \parallel_F \]

\[ + \left( \text{sign}(\parallel \delta \mathbf{r} \parallel_F) \right) \gamma_{c_1} + \left( \text{sign}(\parallel \delta \mathbf{r} \parallel_F) + \text{sign}(\parallel \delta \mathbf{A} \parallel_F) \right) \gamma_{c_p} \times \parallel \gamma \parallel_F \parallel \mathbf{r} \parallel_F \parallel \mathbf{A} \parallel_F \parallel \mathbf{f} \parallel_F \]
4.2.4 Further Discussions

Equations (4.40) and (4.42) in subsection 4.2.3 determine first order rounding error upper bounds for the fixed point and polynomial probability-1 homotopy equations (3.95) and (3.98). It should be emphasized that the whole analysis is made under first order approximations of the perturbations of (3.95) and (3.98), and this is why the characterization ‘first order’ is used. One characteristic of this type of analysis is that the error bounds obtained so far hold asymptotically for reasonable small roundoff unit $u_r$ and therefore are called sharp or local bounds.

Alternative to the first order approximation study above, one can perform a full order analysis taking into account all perturbation orders (e.g. first and second order for the present study) [99]. This will lead to the determination of complete (by means of containing all the orders of perturbed terms) error bounds. At this point it should be said that usually in practice, first order bounds and not full order bounds are used. The reason for this is that the later ones very often produce pessimistic results in terms of computations, since errors tend to cancel in numerical stable algorithms during computation [78]. Moreover, the final computed answer can be much more accurate than the intermediate steps of the algorithm. As pointed out in [78], this phenomenon is not universally appreciated, perhaps because there is the tension to look at the intermediate numbers in an algorithm only when something is wrong and not when the computed answer is satisfactory.

During the analysis, the scaling factors $E_{10}(U_C)$ and $E_{10}(UX)$ were assumed to be already given. In other words, the numerical process of computing them was not considered in terms of rounding error analysis. On the other hand, the computation of the homogeneous
projective transformation (3.88) has been taken into account. This is because (3.88) provides the equation’s solutions at every particular point of the deformation process (see chapters 2 and 3).

Finally, calculating norm inequalities in order to determine upper error bounds it is preferable to use as much as is possible products within the 2-norms instead of splitting the 2-norm in two 2-norm products. This is in order to avoid strong pessimism due to cancellation. To explain this, consider for example $A = B = \begin{bmatrix} 0 & 10^7 \\ 0 & 0 \end{bmatrix}$. Then $\|AB\|_2 = 0$, while $\|A\|_2 \|B\|_2 = 10^{14}$. Hence, $\|AB\|_2 = 0$ is more convenient for an error bound instead of the pessimistic $\|A\|_2 \|B\|_2 = 10^{14}$. This is why indeed this strategy has been used so far in the present chapter.

4.3 Conditioning

The conditioning of an algebraic equation is described by the equation’s condition number. The condition number is a measure of the maximum sensitivity of the equation’s solution to the perturbations in the equation’s data (i.e. $\max \left( \frac{\|\delta X\|_F}{\|\hat{X}\|_F} \right)$). If the condition number is very large then probably floating point errors can affect the solution independently of the numerical method that is used for the solution of the equation [136].

Since (3.95) and (3.98) have the same data, there is one definition for both condition numbers of these equations. This definition is given next.

**Definition 4.1 (Condition Number of (3.95), (3.98));**

The condition number of either (3.95) or (3.98) is defined as

$$
C_n(\hat{X}) = \lim_{\delta_c \to 0} \sup \left\{ \frac{\|\delta X\|_F}{\|\hat{X}\|_F}, \frac{\|\delta \xi F}{\|\xi F\|_F}, \frac{\|\delta E_{10}(U C)\|_F}{\|E_{10}(U C)\|_F}, \frac{\|\delta E_{10}(U X)\|_F}{\|E_{10}(U X)\|_F}, \right. \\
\left. \frac{\|\delta A\|_F}{\|A\|_F}, \frac{\|\delta B\|_F}{\|B\|_F}, \frac{\|\delta C\|_F}{\|C\|_F}, \frac{\|\delta D\|_F}{\|D\|_F}, \frac{\|\delta G\|_F}{\|G\|_F}, \right. \\
\left. \frac{\|\delta \alpha F}{\|\alpha F\|_F}, \frac{\|\delta \beta F}{\|\beta F\|_F} \right\} \leq \delta_c, \delta_c \in \mathbb{R}_+, \forall i \in \mathbb{Z}_+ 
$$

(4.43)
It is apparent from (4.43) that for sufficiently small $\delta_c$, 
$$\frac{\|\hat{X}\|_F}{\|\hat{X}\|_F} \leq C_n(\hat{X}) \delta_c.$$ 
Subject to (4.43), small and large $C_n(\hat{X})$ indicate well-conditioned and ill-conditioned numerical problems respectively. Now, how large or small $C_n(\hat{X})$ is, is measured with respect to 1. More specifically, when $C_n(\hat{X})$ is close to 1 then the problem is well-conditioned and when it is much bigger than 1 the problem is ill-conditioned. The derivation of $C_n(\hat{X})$ according to (4.43) for (3.95) and (3.98) follows.

For the matrix form of (3.95), $\hat{H}_{\alpha\beta}(\hat{X},\epsilon) = \text{vec}^{-1}(\hat{H}_{\alpha\beta}(\hat{X},\epsilon))$, consider the respective perturbed equation

$$\Delta \hat{H}_{\alpha\beta}(\hat{X},\epsilon) = (\epsilon + \delta\epsilon)(\mathbf{E}_{10}(U_C) + \mathbf{E}_{10}(U_X))^{-1} \sum_{i=1}^{\omega} \left((A_i + \delta A_i)((\mathbf{E}_{10}(U_X) + \mathbf{E}_{10}(U_X))^{-1}) \delta X_i \right)$$

$$+ \sum_{i=1}^{\omega} \left((C_i + \delta C_i)((\mathbf{E}_{10}(U_X) + \mathbf{E}_{10}(U_X))^{-1}) \delta X_i \right)$$

$$\times \left((\mathbf{E}_{10}(U_X) + \mathbf{E}_{10}(U_X))^{-1} \delta X_i \right)$$

$$+ \left((A_a + \delta A_a) \text{vec}(\hat{X} + \delta\hat{X}) + a_{l(m+1)} + \delta a_{l(m+1)} \right)^2 (\mathbf{G} + \delta\mathbf{G})$$

$$-(\gamma + \delta\gamma)(1 - (\epsilon + \delta\epsilon))\left(\text{vec}^{-1}(\alpha) + \text{vec}^{-1}(\delta\alpha)\right)$$

$$+ \left((A_a + \delta A_a) \text{vec}(\hat{X} + \delta\hat{X}) + a_{l(m+1)} + \delta a_{l(m+1)} \right)^2 \left(\text{vec}^{-1}(\beta) + \text{vec}^{-1}(\delta\beta)\right) = 0. \quad (4.44)$$

The perturbations in (4.44) are assumed to be measured norm-wise by

$$\eta_c := \|\eta\|_F.$$ 

where

$$\eta := \left[\begin{array}{cccc}(A_a + \delta A_a) \text{vec}(\mathbf{E}_{10}(U_C)) & (\mathbf{E}_{10}(U_C))^T & (A_a + \delta A_a) \text{vec}(\mathbf{E}_{10}(U_X)) & (\mathbf{E}_{10}(U_X))^T \end{array}\right]^T.$$
\[ ... \alpha^+ \delta_{\text{vec}(\Delta a)} \quad \sigma^+ \delta a_{j(m+1)} \quad \delta a_{j(m+1)} \quad + \quad \alpha_{\delta \bar{a}} \circ \text{vec}(\bar{\Delta a})^T \quad + \quad \alpha_{\delta \bar{a}} \circ \text{vec}(\bar{\Delta a})^T \quad ... \]

\[ ... + \quad \alpha_{\delta \bar{a}} \circ \text{vec}(\delta C)^T \quad + \quad \alpha_{\delta \bar{a}} \circ \text{vec}(\delta D)^T \quad + \quad \alpha_{\delta \bar{a}} \circ \text{vec}(\delta E)^T \quad + \quad \alpha_{\delta \bar{a}} \circ \text{vec}(\delta a)^T \quad ... \]

\[ ... \alpha^+ \delta \alpha^T \quad \alpha^+ \delta \beta^T \quad ... \]

(4.46)

\[ \forall \alpha_j \in \mathbb{R}, \quad \forall j = \delta e, \delta y, \delta E_{10}(U_C), \delta E_{10}(U_X), \delta a_{j(m+1)}, \delta a, \delta \beta, \text{ where } j \text{ is a symbol and not a number, and} \]

\[ a^+ := \begin{cases} a^{-1} & a > 0, \forall a \in \mathbb{R} \\ 0 & a = 0 \end{cases} \quad (4.47) \]

(i.e. the pseudo-inverse of \( a \)). Note that, a zero value for an \( \alpha_j \), forces the corresponding perturbation \( \delta \alpha_j \) to be zero.

Moreover, \( \forall i = \delta a, \delta B, \delta C, \delta D, \delta E, \quad \forall j = \omega, \xi, \text{ where } i \text{ is a symbol and not a number,} \]

\[ i \in C^{q \times r}, \quad q, r \in \mathbb{Z}_+ , \]

\[ \alpha_i := \left[ \begin{array}{cccc} \alpha_{i1} & \alpha_{i2} & \cdots & \alpha_{ij} \end{array} \right] \otimes U_{q \times r} \quad (4.48) \]

and \( \forall i = A, B, C, D, E, \quad \forall j = \omega, \xi, \text{ where } i \text{ is a symbol and not a number,} \]

\[ \delta i := \left[ \begin{array}{cccc} \delta i_1 & \delta i_2 & \cdots & \delta i_j \end{array} \right] , \quad (4.49) \]

\[ \quad + \alpha_i := \left[ \begin{array}{cccc} \alpha^+_{i1} & \alpha^+_{i2} & \cdots & \alpha^+_{ij} \end{array} \right] \otimes U_{q \times r} \quad (4.50) \]

Performing all the algebra in (4.44) and dropping the second order perturbed terms as well as considering (3.90), the first order approximation of the perturbation of (3.95) is obtained as

\[ \sum_{i=1}^\omega \delta i \left( E_{10}(U_C) \circ A_i \left( E_{10}(U_X) \circ \left( A_{\text{vec}(X)} + a_{j(m+1)} \right) \right) B_i \right) \]

\[ + \sum_{i=1}^\omega \epsilon \left( E_{10}(U_C) \circ \delta i \left( E_{10}(U_X) \circ \left( A_{\text{vec}(X)} + a_{j(m+1)} \right) \right) B_i \right) \]

\[ + \sum_{i=1}^\omega \epsilon \left( E_{10}(U_C) \circ \delta i \left( E_{10}(U_X) \circ \left( A_{\text{vec}(X)} + a_{j(m+1)} \right) \right) B_i \right) \]

\[ + \sum_{i=1}^\omega \epsilon \left( E_{10}(U_C) \circ A_i \left( E_{10}(U_X) \circ \left( \delta i_{\text{vec}(X)} \right) \right) \right) B_i \]

\[ + \sum_{i=1}^\omega \epsilon \left( E_{10}(U_C) \circ \delta i \left( E_{10}(U_X) \circ \left( \delta i_{\text{vec}(X)} \right) \right) \right) B_i \]
\[ + \sum_{i=1}^{\omega} \left( \text{E}_{i} \left( \text{U}_{C} \right) \circ A_{i} \left( \text{E}_{i} \left( \text{U}_{X} \right) \circ \left[ \left( \delta \text{a}_{i(m+l)} \right) \hat{X} \right] \right) \right) \]

\[ + \sum_{i=1}^{\omega} \left( \text{E}_{i} \left( \text{U}_{C} \right) \circ A_{i} \left( \text{E}_{i} \left( \text{U}_{X} \right) \circ \left[ \left( \text{a}_{i} \text{vec}(\hat{X}) \right) \hat{X} \right] \right) \right) \]

\[ + \sum_{i=1}^{\omega} \left( \text{E}_{i} \left( \text{U}_{C} \right) \circ A_{i} \left( \text{E}_{i} \left( \text{U}_{X} \right) \circ \left[ \left( \text{a}_{i} \text{vec}(\hat{X}) + \text{a}_{i(m+l)} \right) \hat{X} \right] \right) \right) \]

\[ + \sum_{i=1}^{\omega} \left( \text{E}_{i} \left( \text{U}_{C} \right) \circ A_{i} \left( \text{E}_{i} \left( \text{U}_{X} \right) \circ \left[ \left( \text{a}_{i} \text{vec}(\hat{X}) + \text{a}_{i(m+l)} \right) \hat{X} \right] \right) \right) \]

\[ + \sum_{i=1}^{\omega} \delta \text{E}_{i} \left( \text{E}_{i} \left( \text{U}_{C} \right) \circ C_{i} \left( \text{E}_{i} \left( \text{U}_{X} \right) \circ \hat{X} \right) \right) \]

\[ + \sum_{i=1}^{\omega} \delta \text{E}_{i} \left( \text{E}_{i} \left( \text{U}_{C} \right) \circ C_{i} \left( \text{E}_{i} \left( \text{U}_{X} \right) \circ \hat{X} \right) \right) \]

\[ + \sum_{i=1}^{\omega} \left( \left( \text{c}_{i} \text{vec}(\hat{X}) \right) \hat{X} \right) \]

\[ + \sum_{i=1}^{\omega} \left( \left( \text{c}_{i} \text{vec}(\hat{X}) \right) \hat{X} \right) \]

\[ + \sum_{i=1}^{\omega} \delta \text{E}_{i} \left( \text{E}_{i} \left( \text{U}_{C} \right) \circ C_{i} \left( \text{E}_{i} \left( \text{U}_{X} \right) \circ \hat{X} \right) \right) \]

\[ + \sum_{i=1}^{\omega} \left( \left( \text{c}_{i} \text{vec}(\hat{X}) \right) \hat{X} \right) \]

\[ + \sum_{i=1}^{\omega} \left( \left( \text{c}_{i} \text{vec}(\hat{X}) \right) \hat{X} \right) \]

\[ + \sum_{i=1}^{\omega} \delta \text{E}_{i} \left( \text{E}_{i} \left( \text{U}_{C} \right) \circ C_{i} \left( \text{E}_{i} \left( \text{U}_{X} \right) \circ \hat{X} \right) \right) \]

\[ + \sum_{i=1}^{\omega} \left( \left( \text{c}_{i} \text{vec}(\hat{X}) \right) \hat{X} \right) \]

\[ + \sum_{i=1}^{\omega} \left( \left( \text{c}_{i} \text{vec}(\hat{X}) \right) \hat{X} \right) \]

\[ + \sum_{i=1}^{\omega} \delta \text{E}_{i} \left( \text{E}_{i} \left( \text{U}_{C} \right) \circ C_{i} \left( \text{E}_{i} \left( \text{U}_{X} \right) \circ \hat{X} \right) \right) \]

\[ + \sum_{i=1}^{\omega} \left( \left( \text{c}_{i} \text{vec}(\hat{X}) \right) \hat{X} \right) \]

\[ + \sum_{i=1}^{\omega} \left( \left( \text{c}_{i} \text{vec}(\hat{X}) \right) \hat{X} \right) \]

\[ + \sum_{i=1}^{\omega} \delta \text{E}_{i} \left( \text{E}_{i} \left( \text{U}_{C} \right) \circ C_{i} \left( \text{E}_{i} \left( \text{U}_{X} \right) \circ \hat{X} \right) \right) \]

\[ + \sum_{i=1}^{\omega} \left( \left( \text{c}_{i} \text{vec}(\hat{X}) \right) \hat{X} \right) \]

\[ + \sum_{i=1}^{\omega} \left( \left( \text{c}_{i} \text{vec}(\hat{X}) \right) \hat{X} \right) \]

\[ + \sum_{i=1}^{\omega} \delta \text{E}_{i} \left( \text{E}_{i} \left( \text{U}_{C} \right) \circ C_{i} \left( \text{E}_{i} \left( \text{U}_{X} \right) \circ \hat{X} \right) \right) \]

\[ + \sum_{i=1}^{\omega} \left( \left( \text{c}_{i} \text{vec}(\hat{X}) \right) \hat{X} \right) \]

\[ + \sum_{i=1}^{\omega} \left( \left( \text{c}_{i} \text{vec}(\hat{X}) \right) \hat{X} \right) \]

\[ + \sum_{i=1}^{\omega} \delta \text{E}_{i} \left( \text{E}_{i} \left( \text{U}_{C} \right) \circ C_{i} \left( \text{E}_{i} \left( \text{U}_{X} \right) \circ \hat{X} \right) \right) \]

\[ + \sum_{i=1}^{\omega} \left( \left( \text{c}_{i} \text{vec}(\hat{X}) \right) \hat{X} \right) \]
\[ +2\varepsilon E_{10}(U_C) \circ \left( A_{\text{vec}}(\delta \tilde{X}) A_{\text{vec}}(\tilde{X}) \right) G + \varepsilon E_{10}(U_C) \circ \left( A_{\text{vec}}(\tilde{X}) \right)^2 \delta G \]

\[ +2\varepsilon E_{10}(U_C) \circ \left( a_{l(m+1)} A_{\text{vec}}(\tilde{X}) \right) G + 2\varepsilon E_{10}(U_C) \circ \left( a_{l(m+1)} A_{\text{vec}}(\tilde{X}) \right) G \]

\[ +2\varepsilon E_{10}(U_C) \circ \left( \delta a_{l(m+1)} A_{\text{vec}}(\tilde{X}) \right) G + 2\varepsilon E_{10}(U_C) \circ \left( \delta a_{l(m+1)} A_{\text{vec}}(\tilde{X}) \right) G \]

\[ +2\varepsilon E_{10}(U_C) \circ \left( a_{l(m+1)} A_{\text{vec}}(\delta \tilde{X}) \right) G + 2\varepsilon E_{10}(U_C) \circ \left( a_{l(m+1)} A_{\text{vec}}(\delta \tilde{X}) \right) G \]

\[ +\delta E \left( E_{10}(U_C) \circ \left( a_{l(m+1)}^2 G \right) \right) + \varepsilon \left( E_{10}(U_C) \circ \left( a_{l(m+1)}^2 \delta G \right) \right) \]

\[ -\gamma vec^{-1}(\alpha) \circ \left( A_{\text{vec}}(\tilde{X}) + a_{l(m+1)} \tilde{X} \right) \]

\[ -\gamma vec^{-1}(\delta \alpha) \circ \left( A_{\text{vec}}(\tilde{X}) + a_{l(m+1)} \tilde{X} \right) \]

\[ -\gamma vec^{-1}(\alpha) \circ \left( \delta A_{\text{vec}}(\tilde{X}) \tilde{X} \right) - \gamma vec^{-1}(\alpha) \circ \left( A_{\text{vec}}(\delta \tilde{X}) \tilde{X} \right) \]

\[ -\gamma vec^{-1}(\alpha) \circ \left( A_{\text{vec}}(\tilde{X}) + a_{l(m+1)} \delta \tilde{X} \right) - \gamma vec^{-1}(\alpha) \circ \left( \delta a_{l(m+1)} \tilde{X} \right) \]

\[ +\gamma vec^{-1}(\delta \alpha) \circ \left( A_{\text{vec}}(\tilde{X}) + a_{l(m+1)} \tilde{X} \right) \]

\[ +\gamma vec^{-1}(\alpha) \circ \left( \delta A_{\text{vec}}(\tilde{X}) \tilde{X} \right) + \gamma vec^{-1}(\alpha) \circ \left( A_{\text{vec}}(\delta \tilde{X}) \tilde{X} \right) \]

\[ +\gamma vec^{-1}(\alpha) \circ \left( A_{\text{vec}}(\tilde{X}) + a_{l(m+1)} \delta \tilde{X} \right) + \gamma vec^{-1}(\alpha) \circ \left( \delta a_{l(m+1)} \tilde{X} \right) \]

\[ +\delta \gamma \left( A_{\text{vec}}(\tilde{X}) \right)^2 vec^{-1}(\beta) + 2\gamma \left( \delta A_{\text{vec}}(\tilde{X}) A_{\text{vec}}(\tilde{X}) \right) vec^{-1}(\beta) \]

\[ +2\gamma \left( A_{\text{vec}}(\delta \tilde{X}) \right) vec^{-1}(\beta) + \gamma \left( A_{\text{vec}}(\tilde{X}) \right)^2 vec^{-1}(\delta \beta) \]

\[ +2\gamma \left( a_{l(m+1)} A_{\text{vec}}(\tilde{X}) \right) vec^{-1}(\beta) + 2\gamma \left( \delta a_{l(m+1)} A_{\text{vec}}(\tilde{X}) \right) vec^{-1}(\delta \beta) \]

\[ +2\gamma \left( a_{l(m+1)} \delta A_{\text{vec}}(\tilde{X}) \right) vec^{-1}(\beta) + 2\gamma \left( a_{l(m+1)} A_{\text{vec}}(\delta \tilde{X}) \right) vec^{-1}(\beta) \]
Using now the vec(·) operator along with (B.2.1)-(B.2.23) (see appendix B) in each term of (4.51) and grouping the terms with common factors, equation (4.52) follows equivalently.

\[ K_{\delta \hat{X}}(\hat{X}) \text{vec}(\delta \hat{X}) = \]

\[ - \left[ \alpha_{\delta \hat{X}}K_{\delta \hat{X}}(\hat{X}) \quad \alpha_{\delta \hat{X}}K_{\delta \hat{X}}(\hat{X}) \quad \alpha_{\mathcal{E}_{10}(U_X)}K_{\mathcal{E}_{10}(U_X)}(\hat{X}) \quad \alpha_{\mathcal{E}_{10}(U_X)}K_{\mathcal{E}_{10}(U_X)}(\hat{X}) \right] \cdots \]

\[ \cdots \alpha_{\delta a_{l+m+1}K_{\delta a_{l+m+1}}(\hat{X})} \cdots \alpha_{\delta a_{l+m+1}K_{\delta a_{l+m+1}}(\hat{X})} \cdots \]

\[ \cdots (U_{p \times n} \otimes \alpha_{\delta \hat{X}}) \circ K_{\delta \hat{X}}(\hat{X}) \circ (U_{n \times p} \otimes \alpha_{\delta \hat{X}}) \circ K_{\delta \hat{X}}(\hat{X}) \cdots \]

\[ \cdots (U_{p \times n} \otimes \alpha_{\delta \hat{X}}) \circ K_{\delta \hat{X}}(\hat{X}) \circ (U_{n \times p} \otimes \alpha_{\delta \hat{X}}) \circ K_{\delta \hat{X}}(\hat{X}) \cdots \]

\[ \cdots \alpha_{\delta \hat{X}}K_{\delta \hat{X}}(\hat{X}) \alpha_{\delta \hat{X}}K_{\delta \hat{X}}(\hat{X}) \alpha_{\delta \hat{X}}K_{\delta \hat{X}}(\hat{X}) \alpha_{\delta \hat{X}}K_{\delta \hat{X}}(\hat{X}) \right] \eta. \] (4.52)

where \( \eta \) is given by (4.46), and \( \forall i \in \mathbb{Z}_+ \)

\[ K_{\delta \hat{X}}(\hat{X}) := \text{vec} \left[ \varepsilon \left( \left( \mathcal{E}_{10}(U_X) \circ \left( \sum_{i=1}^{o} (A_i \left( \mathcal{E}_{10}(U_X) \circ \hat{X} \right) B_i ) \right) \right) \right] \right] A_a \]

\[ + \varepsilon \left( A_a \text{vec}(\hat{X}) + a_{l(m+1)} \text{diag}(\text{vec}(\mathcal{E}_{10}(U_X))) \left( \sum_{i=1}^{o} (B_i^T \otimes A_i) \right) \right) \text{diag}(\text{vec}(\mathcal{E}_{10}(U_X))) \]
\[ + \text{diag}(E_{10}(U_C)) \left( \sum_{i=1}^{\xi} \left( E_i^T \otimes C_i \left( \left( E_{10}(U_X)^T \cdot \hat{X}^T \right) D_i^T \right) \otimes I_n \right) \right) \text{diag}(E_{10}(U_X)) \]

\[ + \text{diag}(E_{10}(U_C)) \left( \sum_{i=1}^{\xi} \left( E_i^T \otimes C_i \left( I_p \otimes \left( E_{10}(U_X)^T \cdot \hat{X} \right) D_i \right) \right) \right) \text{diag}(E_{10}(U_X)) \]

\[ + 2\varepsilon \text{vec}(E_{10}(U_C)) \odot G \left( A_{a \text{vec}}(\hat{X}) + a_l(m+1) \right) A_a \]

\[ - \gamma(1 - \varepsilon) \left( \alpha \circ \text{vec}(\hat{X}) \right) A_a + \left( A_{a \text{vec}}(\hat{X}) + a_l(m+1) \right) \text{diag}(\alpha) \]

\[ + 2\gamma(1 - \varepsilon) \left( A_{a \text{vec}}(\hat{X}) + a_l(m+1) \right) \beta \]

\[ K_{\delta \varepsilon}(\hat{X}) = \text{vec} \left( \sum_{i=1}^{\omega} \left( E_{10}(U_C) \odot A_i \left( E_{10}(U_X) \circ \left( A_{a \text{vec}}(\hat{X}) + a_l(m+1) \right) \hat{X} \right) B_i \right) \right) \]

\[ + \text{vec} \left( E_{10}(U_C) \odot \sum_{i=1}^{\xi} \left( C_i \left( E_{10}(U_X) \circ \hat{X} \right) D_i \left( E_{10}(U_X) \circ \hat{X} \right) E_i \right) \right) \]

\[ + \text{vec} \left( E_{10}(U_C) \odot \left( A_{a \text{vec}}(\hat{X}) + a_l(m+1) \right)^2 G \right) \]

\[ + \gamma \left( \text{vec}^{-1}(\alpha) \odot \left( A_{a \text{vec}}(\hat{X}) + a_l(m+1) \right) \hat{X} \right) - \gamma \left( 2a_l(m+1) A_{a \text{vec}}(\hat{X}) + a_l^2(m+1) \right) \beta \]

\[ (4.53) \]

\[ K_{\delta \varepsilon}(\hat{X}) = -(1 - \varepsilon) \left( A_{a \text{vec}}(\hat{X}) + a_l(m+1) \right) \text{vec}(\hat{X}) \circ \alpha - \left( A_{a \text{vec}}(\hat{X}) + a_l(m+1) \right)^2 \beta \]

\[ (4.54) \]

\[ K_{\varepsilon 10(U_C)}(\hat{X}) = \varepsilon \left( A_{a \text{vec}}(\hat{X}) + a_l(m+1) \right) \text{diag} \left( \sum_{i=1}^{\omega} \text{vec} \left( A_i \left( E_{10}(U_X) \circ (\hat{X}) \right) B_i \right) \right) \]

\[ + \text{diag} \left( \varepsilon \sum_{i=1}^{\xi} \left( C_i \left( E_{10}(U_X) \circ \hat{X} \right) D_i \left( E_{10}(U_X) \circ \hat{X} \right) E_i \right) \right) \]

\[ + \varepsilon \left( A_{a \text{vec}}(\hat{X}) + a_l(m+1) \right)^2 \text{diag}(\text{vec}(G)) \]

\[ (4.56) \]

\[ K_{\varepsilon 10(U_X)}(\hat{X}) = \varepsilon \left( A_{a \text{vec}}(\hat{X}) + a_l(m+1) \right) \text{diag}(\text{vec}(E_{10}(U_C))) \left( \sum_{i=1}^{\omega} \left( B_i^T \otimes A_i \right) \text{diag}(\text{vec}(\hat{X})) \right) \]

\[ + \text{diag}(\text{vec}(E_{10}(U_C))) \left( \varepsilon \sum_{i=1}^{\xi} \left( E_i^T \otimes C_i \left( \left( E_{10}(U_X)^T \cdot \hat{X}^T \right) D_i^T \right) \otimes I_n \right) \right) \text{diag}(\text{vec}(\hat{X})) \]
\[ K_{\delta 1} (\hat{X}) := \text{vec} \left( E_{10}(U_C) \circ \left( \sum_{i=1}^{\omega} \alpha \left( A_i \left( E_{10}(U_X) \circ \hat{X} \right) B_i \right) \right) \right) \text{vec}(\hat{X})^T \]

\[ -\gamma (1 - \varepsilon) \left( \text{vec}(\hat{X}) \circ \alpha - (A_{\alpha} \text{vec}(\hat{X}) + a_{l(m+1)}) \right) \beta \left( \text{vec}(\hat{X}) \right)^T \]

\[ K_{\delta 3l(m+1)} (\hat{X}) := \text{vec} \left( E_{10}(U_C) \circ \left( \sum_{i=1}^{\omega} \alpha \left( A_i \left( E_{10}(U_X) \circ \hat{X} \right) B_i \right) \right) \right) \text{vec}(\hat{X})^T \]

\[ +2\varepsilon (A_{\alpha} \text{vec}(\hat{X}) + a_{l(m+1)}) \left( \text{vec}(E_{10}(U_C) \circ G) \right) - \gamma (1 - \varepsilon) (\alpha \circ \text{vec}(\hat{X})) \]

\[ +2\gamma (1 - \varepsilon) (A_{\alpha} \text{vec}(\hat{X}) + a_{l(m+1)}) \beta \]

\[ K_{\delta 1} (\hat{X}) := \left[ K_{\delta 1} (\hat{X}) \ K_{\delta 12} (\hat{X}) \ \ldots \ K_{\delta 1\omega} (\hat{X}) \right] . \]

where

\[ K_{\delta 1} (\hat{X}) := \varepsilon \left( A_{\alpha} \text{vec}(\hat{X}) + a_{l(m+1)} \right) \text{diag}(\text{vec}(E_{10}(U_C))) \left( \left( B_i^T \left( E_{10}(U_X^T) \circ \hat{X}^T \right) \right) \otimes I_n \right) \]

\[ K_{\delta 2} (\hat{X}) := \left[ K_{\delta 1} (\hat{X}) \ K_{\delta 12} (\hat{X}) \ \ldots \ K_{\delta 1\omega} (\hat{X}) \right] . \]

where

\[ K_{\delta 1} (\hat{X}) := \varepsilon \left( A_{\alpha} \text{vec}(\hat{X}) + a_{l(m+1)} \right) \text{diag}(\text{vec}(E_{10}(U_C))) \left( I_p \otimes \left( A_i \left( E_{10}(U_X) \circ \hat{X} \right) \right) \right) \]

\[ K_{\delta 2} (\hat{X}) := \left[ K_{\delta 1} (\hat{X}) \ K_{\delta 12} (\hat{X}) \ \ldots \ K_{\delta 1\omega} (\hat{X}) \right] . \]

where

\[ K_{\delta 1} (\hat{X}) := \text{diag}(\text{vec}(E_{10}(U_C))) \left( \varepsilon \left( E_i^T \left( E_{10}(U_X^T) \circ \hat{X}^T \right) D_i^T \left( E_{10}(U_X^T) \circ \hat{X}^T \right) \right) \otimes I_n \right) \]

\[ K_{\delta 2} (\hat{X}) := \left[ K_{\delta 1} (\hat{X}) \ K_{\delta 12} (\hat{X}) \ \ldots \ K_{\delta 1\omega} (\hat{X}) \right] . \]

where

\[ K_{\delta 1} (\hat{X}) := \text{diag}(\text{vec}(E_{10}(U_C))) \left( \varepsilon (E_i^T \otimes C_i) \left( E_{10}(U_X^T) \circ \hat{X}^T \right) \otimes \left( E_{10}(U_X^T) \circ \hat{X} \right) \right) \]

\[ K_{\delta 2} (\hat{X}) := \left[ K_{\delta 1} (\hat{X}) \ K_{\delta 12} (\hat{X}) \ \ldots \ K_{\delta 1\omega} (\hat{X}) \right] . \]
where

\[ K_{\delta \xi} \left( \hat{X} \right) := \text{diag} \left( \text{vec}(E_{10}(U_C)) \left( \varepsilon I_p \otimes \left( C_i \left( E_{10}(U_X) \circ \hat{X} \right) D_i \left( E_{10}(U_X) \circ \hat{X} \right) \right) \right) \right) \]  

\[ K_{\delta G} \left( \hat{X} \right) := \varepsilon \left( A_{\delta} \text{vec}(\hat{X}) + a_{l(m+1)} \right)^2 \text{diag} \left( \text{vec}(E_{10}(U_C)) \right) \]  

\[ K_{\delta a} \left( \hat{X} \right) := -\gamma (1-\varepsilon) \text{diag} \left( \text{vec}(\left( A_{\delta} \text{vec}(\hat{X}) + a_{l(m+1)} \right) \hat{X}) \right) \]  

\[ K_{\delta \beta} \left( \hat{X} \right) := \gamma (1-\varepsilon) \left( A_{\delta} \text{vec}(\hat{X}) + a_{l(m+1)} \right)^2 \otimes I_{np}. \]  

Similarly, for the matrix form of (3.98), consider the respective perturbed equation

\[ \Delta \hat{H}_{\alpha}(\hat{X}, \varepsilon) := \left( \varepsilon + \delta \varepsilon \right) \left( E_{10}(U_C) + \delta E_{10}(U_C) \right) \left( \sum_{i=1}^{\omega} \left( A_i + \delta A_i \right) \left( \left( E_{10}(U_X) + \delta E_{10}(U_X) \right) \circ \left( \hat{X} + \delta \hat{X} \right) \right) \right) B_i + \delta B_i \]

\[ + \left( \sum_{i=1}^{\omega} \left( C_i + \delta C_i \right) \left( \left( E_{10}(U_X) + \delta E_{10}(U_X) \right) \circ \left( \hat{X} + \delta \hat{X} \right) \right) D_i + \delta D_i \right) \]

\[ \times \left( \left( E_{10}(U_X) + \delta E_{10}(U_X) \right) \circ \left( \hat{X} + \delta \hat{X} \right) \right) E_i + \delta E_i \]

\[ + \left( \left( A_{\delta} + \delta A_{\delta} \right) \text{vec}(\hat{X} + \delta \hat{X}) + a_{l(m+1)} + \delta a_{l(m+1)} \right)^2 \left( G + \delta G \right) \]

\[ -\left( \gamma + \delta \gamma \right) (1 - (\varepsilon + \delta \varepsilon)) \left( \text{vec}^{-1}(\alpha) + \text{vec}^{-1}(\delta \alpha) \right) \left( \hat{X} + \delta \hat{X} \right) \]

\[ + \left( A_{\delta} + \delta A_{\delta} \right) \text{vec}(\hat{X} + \delta \hat{X}) + a_{l(m+1)} + \delta a_{l(m+1)} \right)^2 \left( \text{vec}^{-1}(\beta) + \text{vec}^{-1}(\delta \beta) \right) = 0. \]  

The perturbations in (4.44) are assumed to be measured norm-wise by (4.45).

Hence, performing all the algebra in (4.73) and dropping the second order perturbed terms as well as considering (3.90), the first order approximation of the perturbation of (3.98) is obtained as

\[ \sum_{i=1}^{\omega} \delta E \left( E_{10}(U_C) \circ A_i \left( E_{10}(U_X) \circ \left( A_{\delta} \text{vec}(\hat{X}) + a_{l(m+1)} \right) \hat{X} \right) \right) B_i \]

\[ + \sum_{i=1}^{\omega} \delta \left( \delta E_{10}(U_C) \circ A_i \left( E_{10}(U_X) \circ \left( A_{\delta} \text{vec}(\hat{X}) + a_{l(m+1)} \right) \hat{X} \right) \right) B_i \]

\[ + \sum_{i=1}^{\omega} \delta E \left( \delta A_i \left( E_{10}(U_X) \circ \left( A_{\delta} \text{vec}(\hat{X}) + a_{l(m+1)} \right) \hat{X} \right) \right) B_i \]
\[+
\sum_{i=1}^{\infty} \varepsilon \left( E_{10}(U_C) \circ \left( A_i \left( \partial E_{10}(U_X) \circ \left( A_{\text{vec}}(\hat{\xi}) + \hat{a}_{(m+1)} \right) \hat{X} \right) \right) B_i \right)\]

\[+
\sum_{i=1}^{\infty} \varepsilon \left( E_{10}(U_C) \circ \left( A_i \left( E_{10}(U_X) \circ \left( \partial A_{\text{vec}}(\hat{\xi}) \right) \hat{X} \right) \right) B_i \right)\]

\[+
\sum_{i=1}^{\infty} \varepsilon \left( E_{10}(U_C) \circ \left( A_i \left( E_{10}(U_X) \circ \left( \partial \hat{a}_{(m+1)} \right) \hat{X} \right) \right) B_i \right)\]

\[+
\sum_{i=1}^{\infty} \varepsilon \left( E_{10}(U_C) \circ \left( A_i \left( E_{10}(U_X) \circ \left( A_{\text{vec}}(\hat{X}) + \hat{a}_{(m+1)} \right) \delta \hat{X} \right) \right) B_i \right)\]

\[+
\sum_{i=1}^{\infty} \varepsilon \left( E_{10}(U_C) \circ \left( A_i \left( E_{10}(U_X) \circ \left( A_{\text{vec}}(\hat{X}) + \hat{a}_{(m+1)} \right) \right) \delta B_i \right)\]

\[+
\sum_{i=1}^{\infty} \varepsilon \left( E_{10}(U_C) \circ \left( C_i \left( E_{10}(U_X) \circ \hat{X} \right) D_i \left( E_{10}(U_X) \circ \hat{X} \right) E_i \right) \right)\]

\[+
\sum_{i=1}^{\infty} \varepsilon \left( \partial E_{10}(U_C) \circ \left( C_i \left( E_{10}(U_X) \circ \hat{X} \right) D_i \left( E_{10}(U_X) \circ \hat{X} \right) E_i \right) \right)\]

\[+
\sum_{i=1}^{\infty} \varepsilon \left( E_{10}(U_C) \circ \left( \partial C_i \left( E_{10}(U_X) \circ \hat{X} \right) D_i \left( E_{10}(U_X) \circ \hat{X} \right) E_i \right) \right)\]

\[+
\sum_{i=1}^{\infty} \varepsilon \left( E_{10}(U_C) \circ \left( C_i \left( \partial E_{10}(U_X) \circ \hat{X} \right) D_i \left( E_{10}(U_X) \circ \hat{X} \right) E_i \right) \right)\]

\[+
\sum_{i=1}^{\infty} \varepsilon \left( E_{10}(U_C) \circ \left( C_i \left( E_{10}(U_X) \circ \hat{X} \right) D_i \left( \partial E_{10}(U_X) \circ \hat{X} \right) E_i \right) \right)\]

\[+
\sum_{i=1}^{\infty} \varepsilon \left( E_{10}(U_C) \circ \left( C_i \left( E_{10}(U_X) \circ \hat{X} \right) D_i \left( E_{10}(U_X) \circ \hat{X} \right) E_i \right) \right)\]

\[+
\sum_{i=1}^{\infty} \varepsilon \left( E_{10}(U_C) \circ \left( C_i \left( E_{10}(U_X) \circ \hat{X} \right) D_i \left( E_{10}(U_X) \circ \hat{X} \right) E_i \right) \right)\]
\[ + \delta \varepsilon_{10}(U_C) \circ \left( A_a vec(\hat{X}) \right) + \varepsilon \varepsilon_{10}(U_C) \circ \left( \left( A_a vec(\hat{X}) \right)^2 \right) \]

\[ + 2 \varepsilon \varepsilon_{10}(U_C) \circ \left( \left( \delta A_a vec(\hat{X}) A_a vec(\hat{X}) \right) \right) \]

\[ + 2 \varepsilon \varepsilon_{10}(U_C) \circ \left( \left( A_a vec(\delta \hat{X}) A_a vec(\hat{X}) \right) \right) + \varepsilon \varepsilon_{10}(U_C) \circ \left( \left( A_a vec(\hat{X}) \right)^2 \right) \]

\[ + 2 \varepsilon \varepsilon_{10}(U_C) \circ \left( \left( a_{j(m+l)} A_a vec(\hat{X}) \right) \right) + \varepsilon \varepsilon_{10}(U_C) \circ \left( \left( A_a vec(\hat{X}) \right)^2 \right) \]

\[ + 2 \varepsilon \varepsilon_{10}(U_C) \circ \left( \left( a_{j(m+l)} \delta A_a vec(\hat{X}) \right) \right) + \varepsilon \varepsilon_{10}(U_C) \circ \left( \left( A_a vec(\hat{X}) \right)^2 \right) \]

\[ + \delta \varepsilon \varepsilon_{10}(U_C) \circ \left( a_{j(m+l)} A_a vec(\hat{X}) \right) + \varepsilon \varepsilon_{10}(U_C) \circ \left( a_{j(m+l)} A_a vec(\hat{X}) \right) \]

\[ - \gamma (1 - \varepsilon) \left( vec^{-1}(\alpha) \circ \hat{X} \circ \hat{X} \right) - \gamma (1 - \varepsilon) \left( vec^{-1}(\delta \alpha) \circ \hat{X} \circ \hat{X} \right) \]

\[ - 2 \gamma (1 - \varepsilon) \left( vec^{-1}(\alpha) \circ \hat{X} \circ \delta \hat{X} \right) - \gamma \delta \varepsilon \left( vec^{-1}(\alpha) \circ \hat{X} \circ \hat{X} \right) \]

\[ + \delta \varepsilon \left( A_a vec(\hat{X}) \right)^2 vec^{-1}(\beta) + 2 \gamma \left( \delta A_a vec(\hat{X}) A_a vec(\hat{X}) \right) vec^{-1}(\beta) \]

\[ + 2 \gamma \left( A_a vec(\delta \hat{X}) A_a vec(\hat{X}) \right) vec^{-1}(\beta) + \gamma \left( A_a vec(\hat{X}) \right)^2 vec^{-1}(\delta \beta) \]

\[ + 2 \gamma \left( a_{j(m+l)} A_a vec(\hat{X}) \right) vec^{-1}(\beta) + 2 \gamma \left( \delta a_{j(m+l)} A_a vec(\hat{X}) \right) vec^{-1}(\beta) \]

\[ + 2 \gamma \left( a_{j(m+l)} \delta A_a vec(\hat{X}) \right) vec^{-1}(\beta) + 2 \gamma \left( a_{j(m+l)} A_a vec(\delta \hat{X}) \right) vec^{-1}(\beta) \]

\[ + 2 \gamma \left( a_{j(m+l)} A_a vec(\hat{X}) \right) vec^{-1}(\delta \beta) + \gamma a_{j(m+l)}^2 vec^{-1}(\beta) \]

\[ + 2 \gamma a_{j(m+l)} \delta a_{j(m+l)} vec^{-1}(\beta) + 2 \gamma a_{j(m+l)}^2 vec^{-1}(\delta \beta) \]

\[ - \gamma \varepsilon \left( A_a vec(\hat{X}) \right)^2 vec^{-1}(\beta) - 2 \gamma \varepsilon \left( \delta A_a vec(\hat{X}) A_a vec(\hat{X}) \right) vec^{-1}(\beta) \]

\[ - 2 \gamma \varepsilon \left( A_a vec(\delta \hat{X}) A_a vec(\hat{X}) \right) vec^{-1}(\beta) - \gamma \varepsilon \left( A_a vec(\hat{X}) \right)^2 vec^{-1}(\delta \beta) \]

\[ - 2 \gamma \varepsilon \left( a_{j(m+l)} A_a vec(\hat{X}) \right) vec^{-1}(\beta) - 2 \gamma \varepsilon \left( a_{j(m+l)} A_a vec(\hat{X}) \right) vec^{-1}(\delta \beta) \]
\[-2\gamma e \left( \delta \tilde{a}_{j_{(m+1)}} A_{\mathbf{a}} \mathbf{vec}(\tilde{X}) \right) \mathbf{vec}^{-1}(\beta) - 2\gamma e \left( a_{j_{(m+1)}} \delta \mathbf{a} \mathbf{vec}(\tilde{X}) \right) \mathbf{vec}^{-1}(\beta)\]

\[-2\gamma e \left( a_{j_{(m+1)}} A_{\mathbf{a}} \mathbf{vec}(\tilde{X}) \right) \mathbf{vec}^{-1}(\beta) - 2\gamma e \left( a_{j_{(m+1)}} A_{\mathbf{a}} \mathbf{vec}(\tilde{X}) \right) \mathbf{vec}^{-1}(\delta \beta)\]

\[-\delta \gamma e a_{j_{(m+1)}}^2 \mathbf{vec}^{-1}(\beta) - \gamma \delta e a_{j_{(m+1)}}^2 \mathbf{vec}^{-1}(\beta)\]

\[-2\gamma e a_{j_{(m+1)}} \delta a_{j_{(m+1)}} \mathbf{vec}^{-1}(\beta) - \gamma e a_{j_{(m+1)}}^2 \mathbf{vec}^{-1}(\delta \beta) = 0.\]  

(4.74)

Using now the \( \mathbf{vec}(\cdot) \) operator along with (B.2.1)-(B.2.23) (see appendix B) in each term of (4.74) and grouping the terms with common factors, the structure of equation (4.52) follows as previously with data defined as shown next.

\[\forall i \in Z_+ ,\]

\[\mathbf{K}_{\delta \tilde{e}}(\tilde{X}) := \mathbf{vec} \left( e \left( \tilde{E}_{10}(U \mathbf{C}) \circ \sum_{i=1}^{o} \left( A_i \left( \tilde{E}_{10}(U \mathbf{X}) \circ \tilde{X} \right) \mathbf{B}_i \right) \right) \right) \mathbf{A}_{\mathbf{a}}\]

\[+ e \left( A_{\mathbf{a}} \mathbf{vec}(\tilde{X}) + a_{j_{(m+1)}} \right) \mathbf{diag}(\mathbf{vec}(\tilde{E}_{10}(U \mathbf{C}))) \left( \sum_{i=1}^{o} \left( \mathbf{B}_i^T \otimes \mathbf{A}_i \right) \right) \mathbf{diag}(\mathbf{vec}(\tilde{E}_{10}(U \mathbf{X})))\]

\[+ \mathbf{diag}(\tilde{E}_{10}(U \mathbf{C})) \left( e \sum_{i=1}^{e} \left( \left( \tilde{E}_i^T \otimes \mathbf{C}_i \right) \left( \left( \tilde{E}_{10}(U \mathbf{X}) \circ \tilde{X} \right)^T \mathbf{D}_i^T \right) \otimes \mathbf{I}_n \right) \right) \mathbf{diag}(\tilde{E}_{10}(U \mathbf{X}))\]

\[+ \mathbf{diag}(\tilde{E}_{10}(U \mathbf{C})) \left( e \sum_{i=1}^{e} \left( \left( \tilde{E}_i^T \otimes \mathbf{C}_i \right) \left( \mathbf{I}_p \otimes \left( \tilde{E}_{10}(U \mathbf{X}) \circ \tilde{X} \right) \mathbf{D}_i \right) \right) \right) \mathbf{diag}(\tilde{E}_{10}(U \mathbf{X}))\]

\[+ 2e \mathbf{vec}(\tilde{E}_{10}(U \mathbf{C}) \circ \mathbf{G}) \left( A_{\mathbf{a}} \mathbf{vec}(\tilde{X}) + a_{j_{(m+1)}} \right) \mathbf{A}_{\mathbf{a}}\]

\[-2\gamma (1 - e) \mathbf{diag} \left( \left( \alpha \circ \mathbf{vec}(\tilde{X}) \right) \right) + 2\gamma (1 - e) \left( A_{\mathbf{a}} \mathbf{vec}(\tilde{X}) + a_{j_{(m+1)}} \right) \beta \mathbf{A}_{\mathbf{a}}\]  

(4.75)

\[\mathbf{K}_{\delta e}(\tilde{X}) := \mathbf{vec} \left( \sum_{i=1}^{o} \left( \tilde{E}_{10}(U \mathbf{C}) \circ \left( A_i \left( \tilde{E}_{10}(U \mathbf{X}) \circ \tilde{X} \right) \mathbf{B}_i \right) \right) \right) \]

\[+ \mathbf{vec} \left( \tilde{E}_{10}(U \mathbf{C}) \circ \sum_{i=1}^{e} \left( \left( \mathbf{C}_i \left( \tilde{E}_{10}(U \mathbf{X}) \circ \tilde{X} \right) \mathbf{D}_i \left( \tilde{E}_{10}(U \mathbf{X}) \circ \tilde{X} \right) \mathbf{E}_i \right) \right) \right) \]

\[+ \mathbf{vec} \left( \tilde{E}_{10}(U \mathbf{C}) \circ \left( A_{\mathbf{a}} \mathbf{vec}(\tilde{X}) + \mathbf{a}_{j_{(m+1)}} \right)^2 \mathbf{G} \right) \]

\[+ \gamma \left( \mathbf{vec}^{-1}(\alpha) \circ \tilde{X} \circ \tilde{X} \right) - \gamma \left( 2 \mathbf{a}_{j_{(m+1)}} A_{\mathbf{a}} \mathbf{vec}(\tilde{X}) + a_{j_{(m+1)}}^2 \right) \beta \]  

(4.76)
\[ K_{\delta y}(\hat{X}) := -(1 - \varepsilon) \left( \text{vec}(\hat{X} \circ \hat{X}) \circ \alpha - \left( A_a \text{vec}(\hat{X}) + a_{l(m+1)} \right)^2 \beta \right) \]  

(4.77)

\[ K_{\delta E_{10}(U_C)}(\hat{X}), \ K_{\delta E_{10}(U_X)}(\hat{X}) \]  are defined exactly the same as in (4.56), (4.57) respectively.

\[ K_{\delta \alpha_l}(\hat{X}) := \text{vec} \left( E_{10}(U_C) \circ \left[ \sum_{i=1}^\omega \left( \varepsilon \left( A_l \left( E_{10}(U_X) \circ \hat{X} \right) B_i \right) \right) \right] \right) \text{vec}(\hat{X})^T \]

\[-(1 - \varepsilon) \left( \text{vec}(\hat{X} \circ \hat{X}) \circ \alpha - \left( A_a \text{vec}(\hat{X}) + a_{l(m+1)} \right) \beta \right) \text{vec}(\hat{X})^T \]

(4.78)

\[ K_{\delta a_{l(m+1)}}(\hat{X}) := \text{vec} \left( E_{10}(U_C) \circ \left[ \sum_{i=1}^\omega \left( A_l \left( E_{10}(U_X) \circ \hat{X} \right) B_i \right) \right] \right) \]

\[ + 2(1 - \varepsilon) \left( A_a \text{vec}(\hat{X}) + a_{l(m+1)} \right) \text{vec}(E_{10}(U_C) \circ G) - \gamma (1 - \varepsilon) \left( \alpha \circ \text{vec}(\hat{X} \circ \hat{X}) \right) \]

\[ + 2(1 - \varepsilon) \left( A_a \text{vec}(\hat{X}) + a_{l(m+1)} \right) \beta \]

(4.79)

\[ K_{\delta \alpha_l}(\hat{X}), \ K_{\delta \beta_l}(\hat{X}), \ K_{\delta E}(\hat{X}), \ K_{\delta \beta}(\hat{X}) \]  and \( K_{\delta \beta}(\hat{X}) \) are defined exactly the same as in (4.61)-(4.70).

\[ K_{\delta \alpha}(\hat{X}) := -(1 - \varepsilon) \text{diag} \left( \text{vec}(\hat{X} \circ \hat{X}) \right) \]

(4.80)

\[ K_{\delta \beta}(\hat{X}) \]  is defined exactly the same as (4.72).

At this point, the major assumption to make is that \( K^{-1}_{\delta y}(\hat{X}) \) exists. Hence multiplying (4.52) by \( K^{-1}_{\delta y}(\hat{X}) \), taking 2-norms, and using \( \|\text{vec}(\cdot)\|_2 = \|\cdot\|_F \), we obtain the bound

\[ \frac{\|\delta \hat{X}\|_F}{\|\hat{X}\|_F} = \frac{\|\delta \hat{X}\|_F}{\|\hat{X}\|_F} \leq C_n(\hat{X}) \eta_c, \]

(4.81)

where \( \hat{x} := \text{vec}(\hat{X}) \), \( \delta \hat{x} := \text{vec}(\delta \hat{X}) \), \( \eta_c \) is defined in (4.45) and

\[ C_n(\hat{X}) := \|\hat{X}\|_F K^{-1}_{\delta y}(\hat{X}) \]

\[ \times \alpha_{\delta E} K_{\delta E}(\hat{X}) \alpha_{\delta y} K_{\delta y}(\hat{X}) \alpha_{E_{10}(U_C)} K_{E_{10}(U_C)}(\hat{X}) \alpha_{E_{10}(U_X)} K_{E_{10}(U_X)}(\hat{X}) \ldots \]

\[ \ldots \alpha_{\delta a_l} K_{\delta a_l}(\hat{X}) \alpha_{\delta a_{l(m+1)}} K_{\delta a_{l(m+1)}}(\hat{X}) \ldots \]
\[
\cdots (U_{p \times n} \otimes \alpha_{\delta A}) \circ K_{\delta A}(\hat{X}) \ (U_{n \times p} \otimes \alpha_{\delta B}) \circ K_{\delta B}(\hat{X}) \ \cdots \\
\cdots (U_{p \times n} \otimes \alpha_{\delta C}) \circ K_{\delta C}(\hat{X}) \ (U_{n \times p} \otimes \alpha_{\delta D}) \circ K_{\delta D}(\hat{X}) \ (U_{n \times p} \otimes \alpha_{\delta E}) \circ K_{\delta E}(\hat{X}) \ \cdots \\
\cdots \alpha_{\delta \alpha} K_{\delta \alpha}(\hat{X}) \ \alpha_{\delta \alpha} K_{\delta \alpha}(\hat{X}) \ \alpha_{\delta \beta} K_{\delta \beta}(\hat{X}) \|_{2} .
\]

(4.82)

From the above, (4.82) is a sharp bound to first order in \( \eta_c \). Therefore, according to definition 4.1, (4.82) calculates the condition number for both (3.95) and (3.98).

Finally, when a matrix is absent from the formulas in the present section, these formulas hold by removing accordingly any resulted terms subject to absent matrices.

### 4.4 Relative Forward and Backward Errors

After computing a solution of (3.95) or (3.98) subject to an arithmetic precision \( u_r \), the question that follows is how accurate is the solution. Almost always the result is satisfactory when \( \| \hat{X}_c - \hat{X} \|_F \approx u_r \), where \( \hat{X}_c, \hat{X} \) denote the computed and the actual solution respectively. In other words when the relative forward error of \( \hat{X}_c \) is very close to \( u_r \). The relative forward error is defined as follows.

**Definition 4.2 (Relative Forward Error);**

Let \( \text{vec}^{-1}(\hat{X}_c), \text{vec}^{-1}(\hat{X}) \) be the computed and the actual solutions respectively of either (3.95) or (3.98). The relative forward error of \( \hat{X}_c \) is defined as

\[
\text{E}_{\text{rf}}(\hat{X}_c) = \frac{\| \hat{X}_c - \hat{X} \|_F}{\| \hat{X} \|_F} .
\]

(4.83)

Generally, the forward error of a solution of a matrix equation is the most important characteristic of the solution’s accuracy [134], [136]. From (4.83) is apparent that \( \text{E}_{\text{rf}}(\hat{X}_c) \geq 0 \).

Now, \( \text{E}_{\text{rf}}(\hat{X}_c) = 0 \), when the computed solution is exact and \( \text{E}_{\text{rf}}(\hat{X}_c) > 0 \), when the computed solution has no accurate digits.
It should be mentioned that for the majority of practical problems, the exact solution \( \hat{X} \) is unknown. Because of this instead of using (4.83), it is natural to use a modified error defined next.

**Definition 4.3 (Practical Relative Forward Error);**

Let \( \text{vec}^{-1}(\hat{X}_c) \), \( \text{vec}^{-1}(\hat{X}) \) be the computed and the actual solutions respectively of either (3.95) or (3.98). The practical relative forward error of \( \hat{X}_c \) is defined as

\[
E_{\text{prf}}(\hat{X}_c) = \frac{\| \hat{X}_c - \hat{X} \|_F}{\| \hat{X}_c \|_F}. \tag{4.84}
\]

If the relative forward error or the practical relative forward error (when the exact solution is unknown) is of the order of the machine’s precision, then the numerical method that computes the solution is characterized as **perfectly numerical stable from the forward viewpoint** or simply **forward stable**, [78].

As pointed out in [136], the disadvantage of using the relative forward error is that it depends on the numerical properties of the numerical method and furthermore it depends on the conditioning of the problem solved. Hence, the knowledge of this error does not reveal totally the numerical behaviour of the numerical method that is used. Moreover, the optimistic result \( E_{\text{rf}}(\hat{X}_c) = \nu_r \) is not always obtained in practice.

Therefore, instead of the relative forward error, the **relative backward error** is introduced. The definitions of this type of error for (3.95) and (3.98) are given next.

**Definition 4.4 (Relative Backward Error for (3.95));**

Let \( \text{vec}^{-1}(\hat{X}_c) \), \( \text{vec}^{-1}(\hat{X}) \) be a computed and the actual solutions respectively of (3.95). The relative backward error of \( \hat{X}_c \) is defined as

\[
E_{\text{rb}}(\hat{X}_c) = \min \left\{ \delta_c, ((\epsilon + \delta \epsilon)(E_{10}(U_C) + \delta E_{10}(U_C)) + \sum_{i=1}^{\omega}((A_i + \delta A_i)(E_{10}(U_X) + \delta E_{10}(U_X))) \right\}
\]

\[
q \left( (A \delta + \delta A \hat{X}) \text{vec}(\hat{X}_c) + A_l + \delta A_l \right) \left( B_i + \delta B_i \right)
\]
\[ + \sum_{i=1}^{\xi} \left( C_i + \delta C_i \right) \left( (E_{10}(U_X) + \delta E_{10}(U_X)) \cdot \hat{X}_c \right) (D_i + \delta D_i) \]
\[ \times \left( (E_{10}(U_X) + \delta E_{10}(U_X)) \cdot \hat{X}_c \right) (E_i + \delta E_i) \]
\[ + \left( (A_a + \delta A_a) \text{vec}(\hat{X}_c) + \alpha_l(m+l) + \delta \alpha_l(m+l) \right)^2 \left( G + \delta G \right) \]
\[ - (\gamma + \delta \gamma) (1 - (\epsilon + \delta \epsilon)) \left( \text{vec}^{-1}(\alpha) + \text{vec}^{-1}(\delta \alpha) \right) \]
\[ \times \left( (A_a + \delta A_a) \text{vec}(\hat{X}_c) + \alpha_l(m+l) + \delta \alpha_l(m+l) \right) \]
\[ \times \left\{ \text{vec}^{-1}(\beta) + \text{vec}^{-1}(\delta \beta), \eta_c \leq \delta_c \right\}. \quad (4.85) \]

**Definition 4.5 (Relative Backward Error for (3.98)):**

Let \( \text{vec}^{-1}(\hat{X}_c) \), be a computed solution of (3.98). The relative backward error of \( \hat{X}_c \) is defined as
\[ E_{rb}(\hat{X}_c) \defeq \min \left\{ \delta_c, (\epsilon + \delta \epsilon) (E_{10}(U_C) + \delta E_{10}(U_C)) \cdot \sum_{i=1}^{\omega} \left( (A_i + \delta A_i) \left( (E_{10}(U_X) + \delta E_{10}(U_X)) \cdot \hat{X}_c \right) (D_i + \delta D_i) \right) \times \left( (E_{10}(U_X) + \delta E_{10}(U_X)) \cdot \hat{X}_c \right) (E_i + \delta E_i) \right\} \]
\[ + \left( (A_a + \delta A_a) \text{vec}(\hat{X}_c) + \alpha_l(m+l) + \delta \alpha_l(m+l) \right)^2 \left( G + \delta G \right) \]
\[ - (\gamma + \delta \gamma) (1 - (\epsilon + \delta \epsilon)) \left( \text{vec}^{-1}(\alpha) + \text{vec}^{-1}(\delta \alpha) \right) \cdot \hat{X}_c \cdot \hat{X}_c \]
\[ + \left( (A_a + \delta A_a) \text{vec}(\hat{X}_c) + \alpha_l(m+l) + \delta \alpha_l(m+l) \right)^2 \]
\[ \times \left\{ \text{vec}^{-1}(\beta) + \text{vec}^{-1}(\delta \beta), \eta_c \leq \delta_c \right\}. \quad (4.86) \]
From definitions 4.4 and 4.5, it is apparent that the relative backward error is the minimum perturbation in the equation’s data such that the perturbed exact solution of the equation coincides with the computed solution. If the relative backward error is of the order of the machine’s precision, then the numerical method that computes the solution is characterized as perfectly numerical stable from the backward viewpoint or simple backward stable [136], [78].

In the next subsections, first order upper bounds for $E_{rb}({\hat{X}_c})$ for (3.95) and (3.98) are derived together with estimates of practical forward error bounds.

### 4.4.1 Relative Backward and Practical Forward Error Bounds Estimates

In the sequel, first order approximations to perturbed equations are considered only. Hence the results obtained are of first order and hold locally.

In order to determine the relative backward error for (3.95) and (3.98) according to definitions 4.4 and 4.5 respectively, consider the constraint equation in (4.85) and (4.86). Note that, by construction of (3.95) and (3.98), $\hat{H}_{def}({\hat{X}_c},\varepsilon) = vec^{-1}\left(\hat{H}_{def}\left(vec({\hat{X}_c}),\varepsilon\right)\right)$ is the computed residual subject to $\hat{X}_c$. By dropping the second order perturbed terms, the constraint equation in (4.85) and (4.86) can be written in vector format, using the $vec(\cdot)$ operator and (B.2.1)-(B.2.23) (see appendix B), as

$$K(\hat{X}_c)\eta = -vec\left(\hat{H}_{def}(\hat{X}_c,\varepsilon)\right),$$

where $\eta$ is given by (4.46) and

$$K(\hat{X}_c) =$$

$$\left[\alpha_{\delta e}K_{\delta e}(\hat{X}_c) \quad \alpha_{\delta y}K_{\delta y}(\hat{X}_c) \quad \alpha_{\delta_{10}(U_C)}K_{\delta_{10}(U_C)}(\hat{X}_c) \quad \alpha_{\delta_{10}(U_X)}K_{\delta_{10}(U_X)}(\hat{X}_c) \quad \cdots \right]$$

$$\cdots$$

$$\left(\alpha_{\delta a}K_{\delta a}(\hat{X}_c) \quad \alpha_{\delta a_{l(m+1)}}K_{\delta a_{l(m+1)}}(\hat{X}_c) \quad \cdots \right)$$

with (4.54)-(4.72) holding for (3.95) and (4.56), (4.57), (4.61)-(4.70), (4.72), (4.76)-(4.80) hold for (3.98), with respect to $\hat{X}_c$. 

Now, assuming that $K(\hat{X}_c)$ has full rank, then this implies that (4.87) has a solution, which in turn implies that the backward error is finite [79]. Now, the backward error is the minimum 2-norm solution to (4.88). Hence,

$$E_{rb}(\hat{X}_c) = \left\| -K^+(\hat{X}_c)\text{vec}(\widehat{H}_{\alpha\beta}(\hat{X}_c, \varepsilon)) \right\|_2 = \left\| K^+(\hat{X}_c)\text{vec}(\widehat{H}_{\alpha\beta}(\hat{X}_c, \varepsilon)) \right\|_2,$$

(4.89)

where $K^+(\hat{X}_c)$ denotes the pseudo-inverse of $K(\hat{X}_c)$. An upper bound for $E_{rb}(\hat{X}_c)$ is derived considering (4.89) as follows:

$$E_{rb}(\hat{X}_c) \leq \left\| K^+(\hat{X}_c) \right\|_2 \left\| \text{vec}(\widehat{H}_{\alpha\beta}(\hat{X}_c, \varepsilon)) \right\|_F = \frac{\left\| \widehat{H}_{\alpha\beta}(\hat{X}_c, \varepsilon) \right\|_F}{\sigma_{min}(K(\hat{X}_c))},$$

(4.90)

$\sigma_{min}(K(\hat{X}_c))$ denotes the minimum singular value of $K(\hat{X}_c)$. To this end, note that since $K(\hat{X}_c)$ is of full rank by assumption, $\sigma_{min}(K(\hat{X}_c)) \neq 0$ and therefore (4.90) is well defined.

At this point the rounding error analysis, of section 4.2 can come into play and expand bound (4.90). In order to take into account the rounding errors in forming $\widehat{H}_{\alpha\beta}(\hat{X}_c, \varepsilon)$, the latter in (4.90) is replaced by $\widehat{H}_{\alpha\beta}(\hat{X}_c, \varepsilon) + \phi(\widehat{H}_{\alpha\beta}(\hat{X}_c, \varepsilon)) - \widehat{H}_{\alpha\beta}(\hat{X}_c, \varepsilon)$. Hence, (4.90) gives

$$E_{rb}(\hat{X}_c) \leq \frac{\left\| \widehat{H}_{\alpha\beta}(\hat{X}_c, \varepsilon) \right\|_F + \left\| \phi(\widehat{H}_{\alpha\beta}(\hat{X}_c, \varepsilon)) - \widehat{H}_{\alpha\beta}(\hat{X}_c, \varepsilon) \right\|_F}{\sigma_{min}(K(\hat{X}_c))},$$

(4.91)

where $\left\| \phi(\widehat{H}_{\alpha\beta}(\hat{X}_c, \varepsilon)) - \widehat{H}_{\alpha\beta}(\hat{X}_c, \varepsilon) \right\|_F$ is determined by (4.40) for (3.95) and by (4.42) for (3.98). In other words, the bound is expanded by adding to the norm of the residual of the computed solution the norm of its rounding errors. The above way of incorporating rounding errors has been applied similarly in [135] and [78].

As was mentioned previously, $E_{rf}(\hat{X}_c)$ is the most important characteristic for the accuracy of a solution. If the exact solution is known then (4.83) can be used straightaway. Otherwise, the need for the determination of an upper bond for $E_{rf}(\hat{X}_c)$ in (4.84) for (3.95) and (3.98) is immediate. To do this for (3.95) and (3.98), consider (4.53) and (3.73) respectively subject to $\hat{X}_c$ (i.e. $\Delta\widehat{H}_{\alpha\beta}(\hat{X}_c, \varepsilon)$). Also, apart from $\hat{X}_c$ let the corresponding perturbations of the equation’s data be set to zero in.
Under the above assumptions, the equation below follows

\[ K_{\delta x_c} (\hat{X}_c) \text{vec}(\delta x_c) = -\text{vec}(\hat{H}_{\alpha \beta}(\hat{X}_c, \epsilon)), \] (4.92)

where \( K_{\delta x_c} (\hat{X}_c) \) is determined with respect to \( \hat{X}_c \) from (4.53) for (3.95) and from (4.75) for (3.98) respectively.

Now, assume that \( K_{\delta x_c} (\hat{X}_c) \) is of full rank. Then there is a finite solution to (4.92)

\[ \text{vec}(\delta x_c) = -K^{-1}_{\delta x_c} (\hat{X}_c) \text{vec}(\hat{H}_{\alpha \beta}(\hat{X}_c, \epsilon)), \] (4.93)

where \( K^{-1}_{\delta x_c} \) denotes the inverse of \( K_{\delta x_c} \). Hence from (4.93) it follows that

\[
\| \text{vec}(\delta x_c) \|_F \leq \frac{\| -K^{-1}_{\delta x_c} (\hat{X}_c) \|_F \| \text{vec}(\hat{H}_{\alpha \beta}(\hat{X}_c, \epsilon)) \|_F}{\sigma_{\min}(K_{\delta x_c} (\hat{X}_c)) \| \hat{X}_c \|_F}
\]

which implies

\[
\| \hat{X} - \hat{X}_c \|_F \leq \frac{\| \hat{H}_{\alpha \beta}(\hat{X}_c, \epsilon) \|_F}{\sigma_{\min}(K_{\delta x_c} (\hat{X}_c)) \| \hat{X}_c \|_F}
\]

\[
E_{\text{prf}}(\hat{X}_c) \leq \frac{\| \hat{H}_{\alpha \beta}(\hat{X}_c, \epsilon) \|_F}{\sigma_{\min}(K_{\delta x_c} (\hat{X}_c)) \| \hat{X}_c \|_F}, \tag{4.94}
\]

\( \sigma_{\min}(K_{\delta x_c} (\hat{X}_c)) \) denotes the minimum singular value of \( K_{\delta x_c} (\hat{X}_c) \), and it is non zero since \( K_{\delta x_c} (\hat{X}_c) \) is of full rank by assumption.

As done previously, in order to take into account the rounding errors in forming \( \hat{H}_{\alpha \beta}(\hat{X}_c, \epsilon) \), the latter in (4.94) is replaced by \( \hat{H}_{\alpha \beta}(\hat{X}_c, \epsilon) + \text{fl}(\hat{H}_{\alpha \beta}(\hat{X}_c, \epsilon)) - \hat{H}_{\alpha \beta}(\hat{X}, \epsilon) \).

Hence,

\[
E_{\text{prf}}(\hat{X}_c) \leq \frac{\| \hat{H}_{\alpha \beta}(\hat{X}_c, \epsilon) \|_F + \| \text{fl}(\hat{H}_{\alpha \beta}(\hat{X}_c, \epsilon)) - \hat{H}_{\alpha \beta}(\hat{X}, \epsilon) \|_F}{\sigma_{\min}(K_{\delta x_c} (\hat{X}_c)) \| \hat{X}_c \|_F}, \tag{4.95}
\]

where \( \| \text{fl}(\hat{H}_{\alpha \beta}(\hat{X}_c, \epsilon)) - \hat{H}_{\alpha \beta}(\hat{X}, \epsilon) \|_F \) is determined by (4.40) for (3.95) and by (4.42) for (3.98).
The next section reports how numerical stability and the behaviour of computations for (3.95) and (3.98) can be stated in terms of backward and forward errors.

### 4.5 Numerical Stability and the Behaviour of Computations

As stated in the previous section, two types of numerical stability for a computational method have been introduced. Namely, backward and forward stability. Hence, if either the backward error $E_{rb}(\hat{X}_e)$ or the forward error $E_{pf}(\hat{X}_e)$ is of the order of $\epsilon_r$, then the method is characterized as perfectly backward stable or perfectly forward stable respectively.

The relationship between the backward error $E_{rb}(\hat{X}_e)$ and forward error $E_{pf}(\hat{X}_e)$, hence between backward and forward stability, is governed by the conditioning of the numerical problem [78]. There are rules of thumb in numerical analysis ([134], [78]) that state that

\[
E_{pf}(\hat{X}_e) < C_n(\hat{X}_e) E_{rb}(\hat{X}_e),
\]

when the exact solution is known and

\[
E_{pf}(\hat{X}_e) < C_n(\hat{X}_e) E_{rb}(\hat{X}_e),
\]

when the exact solution is unknown.

In view of (4.83), (4.84), (4.89) and (4.91), the rules of thumb (4.96) and (4.97) become respectively

\[
E_{pf}(\hat{X}_e) \leq C_n(\hat{X}_e) \left\| K^+(\hat{X}_e) \vec{H}_{\alpha\beta}(\hat{X}_e, \epsilon) \right\|_2
\]

and

\[
E_{pf}(\hat{X}_e) \leq C_n(\hat{X}_e) \left\| K^+(\hat{X}_e) \vec{H}_{\alpha\beta}(\hat{X}_e, \epsilon) \right\|_2,
\]

considering no rounding errors, and

\[
E_{pf}(\hat{X}_e) \leq C_n(\hat{X}_e) \frac{\left\| \hat{H}_{\alpha\beta}(\hat{X}_e, \epsilon) \right\|_F + \left\| \hat{H}_{\alpha\beta}(\hat{X}_e, \epsilon) - \hat{H}_{\alpha\beta}(\hat{X}_e, \epsilon) \right\|_F}{\sigma_{\min}(K(\hat{X}_e))}
\]

and

\[
E_{pf}(\hat{X}_e) \leq C_n(\hat{X}_e) \frac{\left\| \hat{H}_{\alpha\beta}(\hat{X}_e, \epsilon) \right\|_F + \left\| \hat{H}_{\alpha\beta}(\hat{X}_e, \epsilon) - \hat{H}_{\alpha\beta}(\hat{X}_e, \epsilon) \right\|_F}{\sigma_{\min}(K(\hat{X}_e))},
\]

considering rounding errors.
According to (4.96) and (4.97), an ill-conditioned problem (i.e. when $C_n(\hat{X}_e)$ is sufficiently large) can have a large forward error even when it has small backward error, since the backward error is amplified by $C_n(\hat{X}_e)$. Also, it is apparent that backward stability implies forward stability but not vice versa. Hence, a computational method is characterized as perfectly numerical stable if it is perfectly backward stable.

From the above it is clear that the numerical stability for the probability-1 homotopy equations (3.95) and (3.98) can by mathematically identified via

**Proposition 4.1 (Numerical Stability of (3.95), (3.98));**

Any computational algorithm solving (3.95) or (3.98), is numerically stable if (4.98) holds, when the exact solution is known, or (4.99) holds, when the exact solution is unknown, for either (3.95) or (3.98).

**Proof:**
The proof follows from the previous discussion in this section.

From the above, all the behaviour of the probability-1 homotopy algorithms can by identified in terms of backward and forward errors, condition numbers, rules of thumb and numerical stability. Hence, the quality of a computed solution can be analysed to see whether it should be accepted or not.

### 4.6 Some General Remarks

So far, the numerical analysis was done for (3.95) and (3.98). It is reminded that these homotopies are subject to scaling and homogeneous projective transformations (see chapter 3) and that they are the most general equations of this thesis. Any other quadratic matrix equation is a specialization of (3.95), (3.98) and as was stated in section 4.2 by removing appropriate terms from (3.95), (3.98) the results can be adapted to special cases.

In this respect, note that additional care should be taken when dealing with symmetric matrix terms that are repetitive in the equations. Such examples are for example the algebraic matrix equations (see chapter 6). In these cases when for example the perturbation of

\[
\left( A_i^T + \delta A_i^T \right) X + X(A_i + \delta A_i), \quad A_i \in \mathcal{C}^{n \times n}, \quad X \in \mathcal{C}^{n \times n}
\]

is considered, then it should not analyzed as \( (X^T \otimes I_n) \text{vec}(\delta A_i^T) + (I_n \otimes X) \text{vec}(\delta A_i) \) since both \( \delta A_i \) and \( \delta A_i^T \) have the same norms, but it should analyzed as
\begin{align*}
(X^T \otimes I_n) \Pi \text{vec} (\delta A_i) + (I_n \otimes X) \text{vec} (\delta A_i) = & \left( (X^T \otimes I_n) \Pi + (I_n \otimes X) \right) \text{vec} (\delta A_i),
\end{align*}

where

\begin{equation}
\Pi := \sum_{i=1}^{n} \sum_{j=1}^{n} \left( e_{ui} e_{uj}^T \right) \otimes \left( e_{uj} e_{ui}^T \right), \tag{4.100}
\end{equation}

where, \( e_{ui} \in \mathbb{Z}^n_+ \), \( [e_{ui}]_{j1} := \begin{cases} 1, & j = i \\ 0, & j \neq i \end{cases} \).

\( \Pi \) is known as the \textit{vec-permutation} matrix [65], and has the properties that \( \text{vec} (M^T) = \Pi \text{vec} (M) \), \( (M \otimes N) \Pi = \Pi (N \otimes M) \), \( \forall M, N \in \mathbb{C}^{n \times n} \). Note that the use of the \textit{vec-permutation} matrix produces error bounds that are less pessimistic. This is evident in algebraic Riccati and Lyapunov matrix equations (see [134], [78]). Hence, for such cases symmetric matrices should taken as common factors using (4.100) and modifying appropriately the formulas of this chapter.

Once again it is stated that in order to avoid pessimism, it is proper to evaluate the norm-wise bounds in the previous analysis, for each special problem separately. In latter chapters, for special equations where numerical analysis results already exists in the literature, comments on how the results of the present chapter can be adapted to these equations will be made. Moreover, we will consider examples with known exact solutions, in order to justify the numerical stability and behaviour of the proposed algorithms.

As pointed out in chapter 3, the homogeneous projective transformation and scaling change the homotopy equations and might result in an ill-conditioned problem and vice versa. All depends on the numerical data of the specific problem. For example, it might be possible that a probability-1 homotopy algorithm for an equation of a specific form subject to two sets of different data, is numerically stable for the one set of data and unstable for the other. Now, how computed solutions during the whole deformation process of the homotopy algorithms behave subject to scaling and homogeneous projective transformations, can be studied a posteriori using the results of the present chapter.

For problems with large dimensions, the different types of norms included in the results of this chapter, can be calculated more economically using numerical routines for norm estimates. Such routines are good enough to approximate the true value of the norm and are available in commercial software (e.g. [8], [115]).

Including rounding errors in the analysis can produce pessimistic results when these errors cancel out during the computational process. Hence, one should pay attention to all possible variations of error bounds as presented in the previous sections of this chapter.
The local bound derived in the previous sections neglect second order perturbed terms and because of that they are valid only asymptotically as the perturbations tends to zero. As pointed out in [99], it is often impossible to say for a small finite perturbation, whether the neglected second order terms are really negligible. Furthermore, there can be no claim that the magnitude of the neglected terms is less than or equal to the order of the magnitude of the local bound. Also, for some data perturbations in which the solutions might not exist or tend to infinity, the local bounds will still be defined for a non existing or very large solution, which is nonsense.

The above disadvantages of the local bound can be, but not always, eliminated via full order non-local perturbation analysis. Many times this leads to results that can theoretically guaranteed the existence of solutions in a rigorous mathematical manner as shown in [99], [100]. The problem with non-local perturbation analysis is that the obtained bounds may be pessimistic or may not exist.

A final useful comment to make is that the error bounds considered in the present chapter are norm-wise. This means that they are maximally compressed, and they are not affected by the influence of a particular perturbation of an element in a matrix coefficient. Therefore, the norm-wise bound may not be a good measure for the solution’s sensitivity if there are large differences in the perturbations of different elements in the data and/or the solution. To this end, note that the scaling with magnitude variation (3.60) in chapter 3 may help this problem.

Anyway, to avoid the above problem component-wise bounds can be used in the analysis. These bounds estimate the influence of the perturbations in individual elements of the data on the perturbations in the elements of the solution. Component-wise bound analysis follows similar to the analysis in this chapter, with matrix quantities being measured as absolute values. This type of analysis lies outside the scope of this thesis, but it is trivial to perform. More details about component-wise error analysis can be found in [97]-[100], [78].

4.7 Synopsis

A numerical analysis of the probability-1 homotopy equations, presented in chapter 3 (and hence chapter 2) has been given. The presentation was very technical and detailed discussions and comments were reported in the relevant sections, along with useful references for further study. The analysis provided results that are independent of the numerical method used for the solution of the respective equations, and focused on the solutions. This makes chapter 4 a stand-alone chapter.

Numerical stability and behaviour criteria were presented in a general manner. For particular problems some adaptation of the general results should be made. This is why the
results in this chapter are not implemented in the software code (see chapter 5, and Appendix C). Nevertheless, the software is built in such a way that it provides all the necessary data for a posteriori analysis. Also, one can interfere with the code and add the equations and criteria of this chapter adapted to the specific problem that is processed. The above addition could be for example in the intermediate steps of the homotopy algorithm and act as a step corrector criterion when a solution for a specific homotopy variable is not computed accurately. More discussion about this will take place in chapter 5.

Finally, it should be emphasized that as far as the author is aware the analysis of this chapter and the results obtained are novel and not reported elsewhere. Chapter 4 can also viewed as a unified framework for the numerical analysis of general quadratic matrix equations. Hence, it can be a reference for further studies and research considering the specialization to specific types of equations.
Chapter 5
Software Design
and Implementation

5.1 Introduction

In this chapter, the algorithms, which theoretically have been developed in the previous chapters, are implemented in software. The programming of the algorithms is done using the ‘high level’ programming language of MATLAB v. 5.3.1.29215a (R11.1) [115]. MATLAB provides a user-friendly environment that allows those inexperienced of languages such as FORTRAN, C++, etc to perform numerical computations easily and reasonably fast for linear algebra, ordinary differential equations, numerical optimization, graphics, etc. MATLAB functions and programming are feasible to be implemented in an object-oriented manner [139], which is what was done in this thesis. This makes the programming more economical, flexible and robust.

The designed software is in the form of a MATLAB toolbox (a collection of separate functions (objects) and driver programs [115]), named The GQME-Toolbox (The General Quadratic Matrix Equation Toolbox). The generated code of GQME-Toolbox is given in appendix C. The toolbox structure and its special functions (subroutines) will be described in the sequel of this chapter.

Additional care has been taken to make the software environment as user friendly as possible. Help features guiding the user are included within both the toolbox’s main driver and the special functions. For example, building error indicators will inform the user if something is not setup appropriately (e.g. given matrices with inconsistent dimensions etc.) and will guide the user to setup the problem correctly. Hence, the program can run only after a problem has been posed appropriately and this has as a consequence the elimination of possible failures of
arithmetic operators due to data inconsistencies. Finally, additional care and extensive numerical tests have taken place in order to detect and remove runtime errors and bugs, before finalizing the code.

Now there is a tradeoff between software speed and user friendly software. User friendly software is often slower because it contains additional code for detecting the inappropriate user setups and is built in a high level programming environment (programming languages are slower as one moves from low to high level) as mentioned previously. In this respect, it has been judged by the author and software designer of this thesis that it is preferable to include software that is user friendly for two major reasons.

First, the contribution of this work may be directed to those engineers and scientists unfamiliar with computer programming and moreover those unfamiliar with the mathematical background of the algorithms. The second major reason is that computer power gets stronger day by day. Very soon, into the not too distant future, computers will be fast enough that there will be practically no difference between low and high level computations for reasonable problems. Hence, the provided software is written and organized in such a way that, it can be used by users who are less knowledgeable about the theoretical aspects of the algorithms. Users that are interested in the theory behind the algorithms should refer to chapters 2-4, which constitute in a way a full detailed ‘background theory manual’ for the provided software.

Finally, the code of the provided software is so detailed and commented that it is relatively routine to interpret and implement in a low-level computer language. In this respect, MATLAB provides a C++ source code transformation engine that can automatically interpret and compile the given MATLAB code (see appendix C) into a C++ code. To this respect, tests have been carried out with unfortunately not substantial (almost null) computational speed improvements. The reason for this is that the produced C++ code is not optimized. Moreover, it was necessary to remove break (stop) and graphical (e.g. plot) commands from the MATLAB code, since these commands produce runtime errors in C++ program compilation and execution. Hence, user friendly features of the code are destroyed as well.

However, a careful and independent C++ code implementation of the present software (see appendix C) will eventually provide faster computations. With respect to such an implementation, there is the need to either borrow from existing software libraries (e.g. LAPACK, ODEPACK, NAG) or to write separate routines of matrix arithmetic operators, linear algebra, ordinary differential equation solvers, numerical optimization, graphical outputs, etc. Note that all the above routines are available as built in commands (functions) in the MATLAB environment. In addition, the implementation of complex arithmetic (a built in feature in MATLAB) should be separately considered. This will make the software code remarkable large, and obviously is a difficult task, especially when the code is aimed to be user friendly and as solid as possible. This is out of the scope of this thesis and it is left for future practical research.
Relevant to the subject of this thesis, some examples of low level code software implementations (which are not very user friendly but are ok for experienced users) are the HOMPACK (written initially in FORTRAN 77 [182] and recently in FORTRAN 90 [183]) and its predecessor CONSOL [122] (written in FORTRAN 77).

As will become apparent, some of the notation in the code is slightly different from the notation in chapters 2-4. This is unavoidable since it is impossible to use within the software codes special graphic characters, which are used in the equations of chapters 2-4. To counteract this narrow inconsistency, the declaration for each variable is given within the code and in the help features, of each separate function (subroutine).

The structure of this chapter is as follows. Section 5.2 presents the program’s inputs and outputs (i.e. the equations’ solutions, errors, and other parameters described in chapters 2-3) and their significance in numerical computations. This is actually the description of the main driver of the Toolbox. Next, section 5.3 presents the general structure of the GQME-Toolbox, i.e. its subroutines and special functions. Section 5.4 includes useful discussions for possible (problem dependent) program failures, other relevant problems and possible future software additions and improvements. Finally, in section 5.5 general conclusions are given. The description in all sections is comprehensive, since all details are given via the software code help and comments in appendix C.

5.2 Program Inputs and Outputs

The program’s inputs and outputs are given to and provided respectively by the GQME-Toolbox main driver. This is the m-file, GQMEHOM1.m (see appendix C). Before proceeding, it should be declared that the whole software has been written with respect to \( \omega = \xi = 2 \) in the equation (2.1). The reason for this choice is that it covers all of the engineering and scientific problems in this thesis. If one wishes to increase more the values of \( \omega \) and \( \xi \) then it is necessary to change code, adding the respective quantities of the additional terms. Note that the theory is valid for any \( \omega, \xi \in \mathbb{N}, \omega + \xi \neq 0 \), as shown in chapters 2-3. First the program’s inputs are described, then the outputs.

5.2.1 Program Inputs

The program inputs are:

- \( A_1, B_1, A_2, B_2, C_1, D_1, E_1, C_2, D_2, E_2, G \):
These are the data of (2.1). Namely, \( A_1 \in \mathbb{C}^{n \times n} \), \( B_1 \in \mathbb{C}^{p \times p} \), \( A_2 \in \mathbb{C}^{n \times n} \), \( B_2 \in \mathbb{C}^{p \times p} \), \( C_1 \in \mathbb{C}^{n \times n} \), \( D_1 \in \mathbb{C}^{p \times p} \), \( E_1 \in \mathbb{C}^{p \times p} \), \( C_2 \in \mathbb{C}^{n \times n} \), \( D_2 \in \mathbb{C}^{p \times p} \), \( E_2 \in \mathbb{C}^{p \times p} \), \( G \in \mathbb{C}^{n \times p} \), \( n, p \in \mathbb{Z}_+ \).

- \( A_{10}, B_{10}, A_{20}, B_{20}, C_{10}, D_{10}, E_{10}, C_{20}, D_{20}, E_{20}, G_0 \): These matrices have the same dimensions respectively with \( A_1, B_1, A_2, B_2, C_1, D_1, E_1, C_2, D_2, E_2, G \), and are initial conditions to an outer loop discrete homotopy in the data of the equation (2.1). More specifically, the program under a fixed step given as input \( \text{step} \ (\text{step} \in [0,1]) \), solves a family of equations, which are defined under the data deformation

\[
\begin{align*}
A_i(d_{\text{step}}) &= A_{i0} - d_{\text{step}}(A_{i0} - A_i) , \\
B_i(d_{\text{step}}) &= B_{i0} - d_{\text{step}}(B_{i0} - B_i) , \\
C_i(d_{\text{step}}) &= C_{i0} - d_{\text{step}}(C_{i0} - C_i) , \\
D_i(d_{\text{step}}) &= D_{i0} - d_{\text{step}}(D_{i0} - D_i) , \\
E_i(d_{\text{step}}) &= E_{i0} - d_{\text{step}}(E_{i0} - E_i) , \\
G_i(d_{\text{step}}) &= G_{i0} - d_{\text{step}}(G_{i0} - G_i) ,
\end{align*}
\]

\(d_{\text{step}} \in [0,1], \ d_{\text{step}} = 0, \text{step}, 2\text{step}, ..., 1, \ i = 1,2\).

with respect to (2.1), via the continuous probability-1 homotopy algorithms of chapters 2-3. Thus in total there are two homotopies, a discrete homotopy in the data of (2.1) and a continuous homotopy as in chapters 2-3. Note that the probability-1 feature is obviously preserved \( \forall d_{\text{step}} \in [0,1] \). It is apparent that if \( \text{step} = 1 \), then the discrete homotopy above effectively is not taking place. At this point note that a similar technique is reported in [144].

In later chapters, all the numerical examples are taking place with \( \text{step} = 1 \) (i.e. no data deformation is used). However, the user can do experiments with various \( \text{step} \) values, or even try to use different deformation structures for the data as in [144]. For the latter choice, the appropriate data deformations should replace the last equations, shown above, in GQMEHOM1.m. To this respect, note that computational experience showed that the data deformations shown above perform excellently, although all the problems could be solved without them.

The reason of including the above outer loop data homotopy in the software is in order to relax problems, where the initial condition to the equation's solution is very far from the solution (see [144] for more details). Finally, another reason for including the data deformations is for completeness in order to provide the user with more freedom to investigate a specific problem.
• \textit{flagscale}: \\
This ‘flag’ informs the program whether or not a scaling method (see chapter 3) to (2.1) will be used. Hence, flagscale = 0 when scaling is to be used, and flagscale \neq 0 otherwise.

• \textit{flagv}: \\
This ‘flag’ informs the program whether or not any of the scaling methods with magnitude variation (3.60) or (3.71) will be used. Hence, whenever flagscale = 0, flagv = 0 then scaling magnitude variation (3.60) or (3.71) is to be used, and flagv \neq 0 otherwise. Note that if flagscale \neq 0, then scaling is not used anyway, regardless of the value of flagv.

• \textit{eps}: \\
This determines the precision to zero for all equation’s data, in a scaling problem computation. eps gives the user the freedom to perform trials and study how the solution of the scaling problem (and hence the solution of (2.1)) is affected by assuming small numbers equal to zero. In most cases and for all examples of this thesis eps = 0. It has been observed that sometimes eps \neq 0, produces instability, divergence from the ‘true’ solution and catastrophic failures to the numerical computations too. The value of eps is considered only when flagscale = 0.

• \textit{flagopt}: \\
This ‘flag’ informs the program whether or not the optimal scaling methods (3.31) or (3.60) or the sub-optimal scaling methods (3.63) or (3.71) will be used. Hence, whenever flagscale = 0, flagopt = 0 then the optimal scaling (3.31) or (3.60) is to be used, and flagopt \neq 0 when the sub-optimal scaling (3.63) is to be used. If flagscale \neq 0, then scaling is not used anyway, regardless of the value of flagopt. Note that, which of (3.31), (3.60) or which of (3.63), (3.71) is to be used, is determined by flagv as shown previously.

• \textit{flagsym}: \\
This ‘flag’ informs the program whether or not the equation’s solution and data scaling coefficients need to be symmetric matrices (in the cases of equations with square dimensions and which have square matrices as solutions). Hence, flagsym = 0 when symmetry is not required and flagsym \neq 0 otherwise. Note that, the symmetry is just a setup of the lower triangular part of the equation’s solution and data scaling coefficients to be equal to the upper triangular part. This setup is done after the computation of the respective scaling coefficients, without any symmetry consideration.
• sux, seo:
These inputs are the symbolic definition of the scaling coefficient of the equation’s solution $U_X$ and data $U_C$ respectively. Note that, for the solution of the scaling problems with magnitude variation, symbolic computations are taking place using the symbolic computation toolbox in MATLAB (see [116]). Hence, it is necessary for $U_X$ and $U_C$ to be initially defined as a symbolic matrices.

For the previous scaling flags, the hierarchy of dependence between them is according to the flow:

$$\text{flagscale} \rightarrow \text{eps} \rightarrow \begin{cases} \text{flagv} \\ \text{sux} \\ \text{seo} \end{cases} \rightarrow \text{flagopt} \rightarrow \text{flagvsym}.$$  

To this end, note that the scaling is independent of any possible projective transformations, and is subject to (3.1) (see chapter 3).

The scaling problem is computed by the separate function $SGQME.m$ (see section 5.3 and appendix C). All the above are explained and are more easy to follow by reading the help and comments of $SGQME.m$ (see appendix C).

• flagproj:
This ‘flag’ informs the program whether or not an optimal projective transformation is to be used within the numerical computation of the equations’ solution. flagproj = 0 when an homogeneous projective transformation is used and flagproj = 1 otherwise.

• Aa, anplus1:
These inputs determine the projective transformation, and more specifically are $A_a, a_{(np+1)}$ respectively in (3.74) or (3.88).

• gam:
This is the parameter $\gamma$ appearing in the homotopy equations (see chapters 2-4).

• pdef:
This parameter is used together with $\text{gam}$ when some directionality rules that are to be determined in chapter 6 are used in order to trace positive ($\text{pdef} = 1$ and $\text{gam} \in \mathbb{R}_+$) or not positive ($\text{pdef} = -1$ and $\text{gam} \in \mathbb{R}_-$) or negative ($\text{pdef} = 1$ and $\text{gam} \in \mathbb{R}_-$) or not negative
Chapter 5 Software Design and Implementation

(pdef $= -1$ and $\gamma \in \mathbb{R}_+$) define (semidefinite) solutions. These rules can be used as step correctors in the homotopic continuation of algebraic Riccati matrix equations. See chapter 6 for more details.

- **$X_0, Y_0$:**
  These are $\text{vec}^{-1}(\beta)$ and $\text{vec}^{-1}(\alpha)$ respectively, appearing in the easy problems of the homotopy equations (see chapters 2-4).

- **Sign$X_0 Y_0$:**
  This matrix has the same dimensions as $\text{vec}^{-1}(\alpha)$ and $\text{vec}^{-1}(\beta)$, and determines via the sign of its elements which one of the solutions of the easy problem will be used as initial condition. $\text{Sign}X_0 Y_0$ takes an active role in GQM EHOM l.m only when the easy problem is equivalent to a second order polynomial equation with the solutions appearing dual (i.e. $\pm$) in sign; i.e. whenever a polynomial homotopy (see chapters 2, 3) is used.

- **$\text{step}$:**
  $\text{step} \in (0,1]$ is the step in the discrete homotopy in the data of (2.1), as stated previously.

- **$e$:**
  This is the precision to zero in the computations. Used in ‘stop’ and ‘branch’ criteria inside the code (see appendix C).

- **$\text{efinal}$:**
  This is the precision to zero, with which the homotopy ‘end game’ (i.e. the computation of the solution at $\varepsilon = 1$) is computed. For this scope, the MATLAB’s optimization function $\text{fsolve.m}$ is used. Now, $\text{efinal}$ is the value of 'TolFun' in the optimset setup of $\text{fsolve.m}$. For more details see $\text{fsolve.m}$ in MATLAB 5.3, and $\text{FQMEHOM1.m}$, $\text{GQMEHOM1.m}$ in appendix C.

- **$\text{flh}$:**
  This ‘flag’ determines whether or not a fixed point or a polynomial homotopy (see chapters 2, 3) will be used. Hence, $\text{flh} = 0$ when a fixed point homotopy is used and $\text{flh} \neq 0$ otherwise.
• \texttt{flchom}:
This ‘flag’ determines the homotopy’s Euclidean space. More specifically, \texttt{flchom} = 0 if the homotopy is defined in the complex Euclidean space, and \texttt{flchom} \neq 0 if the homotopy is defined in the real Euclidean.

• \texttt{flcarith}:
This ‘flag’ determines whether or not a complex arithmetic will be used in the computations. Hence, \texttt{flcarith} = 0 when complex arithmetic is used and \texttt{flcarith} \neq 0 when real arithmetic is used.

• \texttt{flpseudos}:
This ‘flag’ determines whether or not a pseudo arc-length parameterization will be used in the computations (see sections 2.3.2, 2.4, 3.4.2, and 3.4.3). Hence, \texttt{flpseudos} = 0 when pseudo arc-length parameterization will not be used and \texttt{flpseudos} \neq 0 otherwise.

• \texttt{pseudosfin}:
This is the estimated value of the pseudo arc-length, when such a parameterization will take place. Hence, \texttt{pseudosfin} is active in \texttt{GQMEHOM1.m} only when \texttt{flpseudos} \neq 0. If the shape of the homotopy paths are somehow known \texttt{pseudosfin} helps in the integration process by the means of accurate minimum integration step determination. For more detailed similar discussions, see chapter 2. If the ‘true’ pseudo arc-length is bigger than \texttt{pseudosfin}, then the program continues the tacking of the homotopy paths with the message that the estimated length has been exceeded. Usually \texttt{pseudosfin} = 2 is a good choice for normal problems.

• \texttt{maxnsteps}:
This is the maximum number of integration steps allowed for the tracking of the homotopy paths.

• \texttt{minstep}:
This is the minimum allowable integration step. It is used in combination with \texttt{maxnsteps}, and for the same purpose.

Both \texttt{maxnsteps} and \texttt{minstep}, are helpful for the detection of the divergence of a path to infinity, when for example a homogeneous projective transformation is not used or when there are are
infinite number of solutions at infinity and hence the homogeneous projective transformation may not be capable of computing such solutions. The last case could be assumed if the steps of integration exceed a reasonably large $\text{maxnsteps}$ and are smaller than $\text{minstep}$. However, note that any such assumptions may stop the process incorrectly, because of a large path arc-length. Hence, additional care should be paid in these cases. See chapters 2, 3 for more details.

- **flagcor:**
  This 'flag' determines whether or not a correction to each intermediate solution obtained along the homotopy paths will take place. The criterion for correction is when the Frobenious norm of the homotopy equation's residual over the homotopy equation's computed solution is bigger than $\varepsilon$ (see above). The correction is done via MATLAB's optimization function `fsolve.m` for the homotopy equation with respect to the specific homotopy variable $\varepsilon$. Hence, $\text{flagcor} = 0$ when no correction will be posed and $\text{flagcor} \neq 0$ otherwise. For more details see the integration function `RK45GQMEHOM1.m` and `GQMEHOM1.m` in appendix C.

- **largescale, jacobiangqme, linesearch, displaysolver:**
  These are the settings in 'LargeScale', 'Jacobian', 'LineSearchType', and 'Display' respectively in the `optimset` setup of `fsolve.m`. When $\text{largescale}='\text{on}'$ or $\text{largescale}='\text{off}'$, then a large or a medium scale respectively optimization algorithm is used in `fsolve.m`. When $\text{jacobiangqme}='\text{on}'$, then the Jacobian of the homotopy equation is computed explicitly from the m-files `FGQMEODE.m` and `FGQMEHOM1.m` analytically. When $\text{jacobiangqme}='\text{off}'$, then the Jacobian of the homotopy equation is computed via numerical approximation. For more details see `fsolve.m` in MATLAB 5.3, and also `GQMEHOM1.m` in appendix C.

  The medium optimization algorithm of `fsolve.m` is a Gauss-Newton optimization with line search. There are two types of line search, which are specified as `linesearch='cubicspoly'` and `linesearch='quadcubic'`. 'cubicspoly' is a safeguarded mixed quadratic and cubic polynomial interpolation and extrapolation method. A safeguarded cubic polynomial method can be selected by setting `linesearch='quadcubic'`. This method generally requires fewer function evaluations but more gradient evaluations. Thus, if gradients are being supplied and can be calculated inexpensively, the cubic polynomial line search method is preferable.

  The large scale optimization algorithm of `fsolve.m` is a subspace trust region method, based on an interior-reflective Newton method [117]. Note that, in the case of large scale optimization algorithm, line search is not used and therefore `linesearch` is redundant. For more details see [117].
Now \texttt{displaysolver='iter'}, when output information (optimization step, norm of residual, Jacobian evaluations, etc) for each step in \texttt{fsolve.m} is required to be printed on the screen during the computational process, and \texttt{displaysolver='off'} otherwise.

- \texttt{displayode}:
  \texttt{displayode='on'}, when integration step and homotopy variable and pseudo arc-length (when \texttt{flpsusos\neq0}) are required to be printed on the screen during the computational process, and \texttt{displayode='off'} otherwise.

- \texttt{plotdisplay}:
  \texttt{plotdisplay = 'on'}, when a graph of the homotopy paths is required to be printed on screen after the computational process, and \texttt{plotdisplay = 'off'} otherwise.

So far, all the necessary inputs for the program to run have been described. Sets of the above inputs are inputs to the separate functions (subroutines) of the main driver \texttt{GQMEHOM1.m}. It is worth remembering that if something is given incorrectly by the user, then the program will respond accordingly with error messages informing the user about the correct setup. More details about the above program inputs can be found in \texttt{GQMEHOM1.m} and the rest of functions (m-files) in appendix C, as well as in [115]-[117]. Later on, it will be shown how to setup and run specific problems. It is recommended that the reader should have a general view of chapters 2-3, before getting familiar with the GQME-Toolbox. Finally, all the above inputs are explained and specified in the help menu of \texttt{GQMEHOM1.m}. The user can view this by setting the MATLAB editor in the directory where the \texttt{GQMEHOM1.m} and the rest of the m-files of GQME-Toolbox are, and by typing the command ‘help GQMEHOM1’.

### 5.2.2 Program Outputs

The program outputs are:

- \texttt{error}:
  This is the absolute error of the equations’ (2.1) residual with respect to the computed solution.

- \texttt{X}:
  This is the computed solution of (2.1).
• JacobianX:

This is the Jacobian of the equation’s computed solution, with respect to this solution. This is useful, in order to scrutinize the singularity of the computed solution by checking its rank.

• LX:

This is the value of the homogeneous projective transformation with respect to the computed solution. This is useful, in order to determine whether or not the computed solution is at infinity (e.g. when \( LX \to 0 \)).

• \(\text{ux, eo}\):

These are the computed values for the equation’s scaling factors \( U_X \) and \( U_C \) respectively, when scaling is used. Note that the scaling problem is computed via the function SGQME.m, in which \(\text{ux, eo}\) are not the only outputs that can be provided (see SGQME.m).

• polyXa:

This is a matrix \(\text{polyXa} = \begin{bmatrix} \text{repolyXa} \\ \text{impolyXa} \end{bmatrix}\), containing on each row the coefficients of a fifth order polynomial (subject to variable \( \epsilon \)) approximation of \( X(\epsilon) \). This polynomial is computed via a polynomial interpolation to the computed points \( X(\epsilon) \) along the homotopy paths, using MATLAB’s functions \text{polyfit.m} and \text{polyval.m}. The first \( np \) rows of \(\text{polyXa} \) (i.e. matrix \(\text{repolyXa}\)) are with respect to the real part of the homotopy equation, while the last \( np \) rows (i.e. matrix \(\text{impolyXa}\)) are with respect to the imaginary part of the homotopy equation. Hence, the software approximates the homotopy paths defined from \( X(\epsilon) \) as polynomial functions of \( \epsilon \).

• \(\text{plX, plepsilon, plpseudos}\):

These are the \( X(\epsilon), \epsilon, \) and \( s \) respectively of \( (X(\epsilon),\epsilon,s) \) triplets of the computed points of the homotopy paths.

• \(\text{nsteps, stepmin}\):

These are the values of the number of integration steps and the minimum integration step respectively, used in the integration subroutine (ODE solver, see chapters 2, 3) \text{RK45GQMEHOM1.m}. 

• \texttt{tCPU, nFLOPS:}  
These are the values of the computer CPU time and number of flops, required for tracing the homotopy paths when \texttt{step=1}.

• \texttt{tCPU\,scale, nFLOPS\,scale:}  
These are the values of the computer CPU time and number of flops, required for solving the scaling problem.

At this point the presentation of all program inputs and outputs via the main driver \texttt{GQM\,EHOM\,l.m} is finished. Additional technical details can be found in the help menus and comments of the GQME-Toolbox's functions. Next, the general structure of GQME-Toolbox and each one of its functions are described.

### 5.3 The GQME-Toolbox

The GQME-Toolbox consists of one major driver the \texttt{GQM\,EHOM\,l.m}, as stated previously, and fifteen additional m-files that are functions (subroutines). The general structure of the toolbox and the relation between its m-files and their functionality are given in figure 5.1. These functions are described next.

• \texttt{GQMEHOM1.m}  
This is the main driver of GQME-Toolbox. In this file, all the necessary inputs are specified. Then error checking subject to these inputs is made, in order to avoid inconsistent setups, code failures and runtime errors. This m-file includes (used) all the remaining elements of the GQME-Toolbox. It is written with detailed comments and help features and is very easy to follow, along with the background theory in chapters 2 and 3 (see appendix \texttt{GQM\,EHOM\,l.m} in C). The inputs and outputs of this file have been described in section 5.2.

• \texttt{SGQME.m}  
This function computes solutions to the scaling problems in chapter 3. \texttt{SGQME.m} considers the general dimensional case \( t \neq n \neq p \neq l \) as stated in chapter 3, although when used for the solution of (2.1) it is used with inputs \( t = n \neq p = l \). Thus, \texttt{SGQME.m} is developed in such a way that it can be used alone and independent of the main driver \texttt{GQM\,EHOM1.m}. Hence, \texttt{SGQME.m} can determine scaling coefficients independent of the probability-1 homotopy algorithms and therefore, the scaling equation (and hence the original equation (2.1)) can be solved with different numerical solvers. For more details, see chapter 3.
Now, the inputs to SGQME.m are: A1, B1, A2, B2, C1, D1, E1, C2, D2, E2, G, flagscale, flagv, eps, flagopt, flagsym, sux and seo. These inputs and their role have been described in subsection 5.2.1, and are inputs primarily to the main driver GQMEHOM1.m.
Chapter 5  

The outputs of SGQME.m are: Abar, Bbar, Cbar, ux, eo, S, Sm, A1hat, B1hat, A2hat, B2hat, C1hat, D1hat, E1hat, C2hat, D2hat, E2hat, Ghat, tCPUscale and nFLOPSscale.

From them, ux, eo, tCPUscale and nFLOPSscale, have been described in subsection 5.2.2 and are outputs of the main driver GQM EHOM1.m too. Abar, Bbar, Cbar, are the values of \( Z (S_{B,A},oE_0) \) and \( SG \circ E_0 \) respectively in (3.29) with respect to \( \omega = \xi = 2 \). S and Sm are the values of the objective functions \( S \) or \( S_i \) that are to be minimized in the scaling problems of chapter 3. S corresponds to the objective function’s value before scaling and Sm corresponds to the objective function’s value after scaling.

A1hat, B1hat, A2hat, B2hat, C1hat, D1hat, E1hat, C2hat, D2hat, E2hat and Ghat, are the scaled equation’s data \( \hat{A}_1, \hat{B}_1, \hat{A}_2, \hat{B}_2, \hat{C}_1, \hat{D}_1, \hat{E}_1, \hat{C}_2, \hat{D}_2, \hat{E}_2 \) and \( \hat{G} \), after sub-optimal scaling which correspond to \( A_1, B_1, A_2, B_2, C_1, D_1, E_1, C_2, D_2, E_2 \) and \( G \) respectively. Note that since the scaling factors are scalars in sub-optimal scaling, the equation (2.1) after scaling can be written as

\[
f(\hat{X}) = \sum_{i=1}^{2} \hat{A}_i \hat{X} \hat{B}_i + \sum_{i=1}^{2} \hat{C}_i \hat{X} \hat{D}_i \hat{X} \hat{E}_i + \hat{G} = 0,
\]

where \( \hat{X} := E_0(uX) \hat{X} \) and \( \hat{X} \) is the scaled equation’s unknown\(^*\).

For more technical details of SGQME.m, the reader should look at the respective m-file in appendix C.

• RK45GQMEHOM1.m

This function (subroutine) constitutes the ordinary differential equation (ODE) solver, with which the homotopy paths are traced via continuous homotopy methods (see chapters 2, 3). The integration method implements the Runge-Kutta 45 order method [75], [76]. RK45GQMEHOM1.m is a substantial modification of ode45.m m-file of MATLAB 4.1 (an old version of MATLAB).

This function also uses the functions INGQMEHOM1.m and FGQMEODE.m, which will be presented in the sequel. The inputs to RK45GQMEHOM1.m are: ypfun, t0,

\(^*\) Note the inconsistency in the notation in chapter 3, where for scaling the notation \( \overline{X} \) is used instead of \( \hat{X} \).
tfinal, y0, tol, n, p, gam, pdef, flh, flpseudos, maxnsteps, minstep, flagcor, and displayode.

From the above inputs, gam, pdef, flh, flpseudos, maxnsteps, minstep, flagcor, and displayode, were presented in subsection 5.2.1 and are inputs to the main driver GQM EHOM l.m as well. n and p are the dimensions $n \times p$ of the equation’s solution $X$. tol is the desired accuracy in the ODE and in GQM EHOM l.m it is set to be tol = e. ypfun is a string containing the name of the problem description (i.e. the analytic formulas of derivatives). In GQM EHOM l.m, ypfun = 'INGQMEHOM1'. t0, tfinal and y0, are the initial integration point, the final integration point and the initial value respectively. Intuitively in GQM EHOM l.m, t0 = 0, tfinal = 1 or tfinal = s(1) (if pseudo arc-length parameterization is used) and y0 = vec($\hat{X}(0)$). Now, $\hat{X}(0)$ is determined as the solution of the easy problem (see GQM EHOM l.m).

The outputs of RK45GQM EHOM l.m are: tout, yout, nsteps and stepmin. nsteps and stepmin, described in section 5.2.2 and are outputs in GQM EHOM l.m too. tout and yout are vectors containing the ODE computed points. In GQM EHOM l.m, these are set as pepsilon and plX respectively.

For more details see the comments and help sections of the code in INGQM EHOM l.m, RK45GQM EHOM l.m and GQM EHOM l.m.

**INGQMEHOM l.m**
This m-file, defines the IVPs (2.40), (2.47), (3.101) and (3.108) for the ODE solver RK45GQM EHOM l.m. The inputs of this function are the homotopy variable epsilon, the unknown variable X and the ‘flag’ flh. There is only one output, the derivative of X with respect to epsilon. For details see INGQMEHOM l.m in appendix C.

**FGQM EODE.m**
This m-file defines the fixed point and polynomial homotopy equations of chapters 2 and 3, and is used with the optimization solver fsolve.m, in order to compute the solution and the Jacobian of this solution with respect to this solution. Note that fsolve.m under FGQM EODE.m is acting as solution corrector at each integration step of the RK45GQM EHOM l.m (i.e. for every computed point along the homotopy paths). FGQM EODE.m activated, whenever flagcor $\neq$ 0 and the Frobenious norm of the homotopy equation’s residual over the homotopy equation’s computed solution is bigger than e (see section 5.2.1).
• **FGQMEHOM1.m**

This m-file defines the homotopy equation of chapters 2, 3, at the endpoint $\epsilon = 1$, and is used with the optimization solver `fsolve.m`, in order to compute the solution and the Jacobian of this solution with respect to this solution. Note that `fsolve.m` under `FGQMEHOM1.m` constitutes the 'end game' of the homotopic continuation, which calculates the solution at $\epsilon = 1$. `FGQMEHOM1.m` activated, whenever the `flagcor ≠ 0` and the Frobenious norm of the homotopy equation's residual over the homotopy equation's computed solution is bigger than $\epsilon_{\text{final}}$ (see section 5.2.1).

All the previous functions (subroutines) constitute the hardcore of the GQME-Toolbox. For their technical details subject to code programming, the reader is referred to appendix C. The theoretical background can be found in chapters 2 and 3.

Apart from the above functions there are some additional functions for implementing matrix operators that MATLAB cannot do. These are the $\text{vec}(\cdot)$, $\text{vec}^{-1}(\cdot)$ and $\otimes$ operators (see appendix B). The respective m-files are straightforward and can be found in appendix C. These functions are reported comprehensively below.

• **vecr.m**

Forms a column vector from the rows of a general matrix.

• **vecc.m**

Forms a column vector from the columns of a general matrix (i.e. the $\text{vec}(\cdot)$ operator).

• **invecr.m**

This is the inverse of `vecr.m` and forms back the original matrix from the column vector, constructed by the rows of the matrix.

• **invecc.m**

This is the inverse of `vecc.m` and forms back the original matrix from the column vector, constructed by the rows of the matrix.

• **kronsum.m**

Forms the Kronecker sum of two general matrices.
• **svecr.m**
Forms a column vector from the rows of a general matrix using symbolic computation.

• **svecc.m**
Forms a column vector from the columns of a general matrix (i.e. the vec(·) operator) using symbolic computation.

• **insvecr.m**
This is the inverse of *svecr.m* and forms back the original matrix from the column vector, constructed by the rows of the matrix.

• **insvecc.m**
This is the inverse of *svecc.m* and forms back the original matrix from the column vector, constructed by the rows of the matrix.

• **skronsum.m**
Forms the Kronecker sum of two general matrices using symbolic computation.

So far all the presented software covers chapters 2 and 3 but not chapter 4. For the last case, as it has been pointed out in chapter 4, every problem should be tackled separately. However, for the case of (2.1) with $\omega = \xi = 2$, and assuming that all matrices are different from each other and are nonzero, the m-file *NAGQME.m* is provided by the GQME-Toolbox (see appendix C). This function is reported next.

• **NAGQME.m**
NAGQME.m performs a numerical analysis to the computed solution of (2.1). In NAGQME.m all the results of chapter 4 have been adapted for (2.1) with $\omega = \xi = 2$, under the above assumption that all matrices are different from each other and nonzero.

At this point it should be said, that a big range of quadratic matrix equations and also symmetric equations can be adapted in NAGQME.m. To illustrate this for the case of symmetric equations, consider the case of the algebraic matrix Riccati equation

$$A^T X + XA + CXC + D = 0, \quad A, C, D \in \mathbb{C}^{n \times n}, \quad n, p \in \mathbb{Z}_+ \text{ (see chapter 6)}$$

and observe that,

$$A^T X + XA + CXC + D = (0.1A)^T X(10I_n) + (0.2I_n)X(5A) + (0.1C)X(3C) + (0.35C)X(2C) + D = 0$$

In the left hand side, $A$ and $B$ are repeated and there are matrices subject to (2.1) with $\omega = \xi = 2$ that are zero. As pointed out in chapter 4, working with the left hand side the
permutation matrix in (4.100) is wise to be used in order to avoid pessimism. In the right hand side of above equality, \(0.1A, 10I_n, 0.2I_n, 5A, 0.1C, 3C, 0.35C, 2C\) and \(D\) are different from each other and different than zero. Hence, by writing the equation as on the right hand side the modification of the results in chapter 4 with respect to (4.100) are not necessary. Note that, in order to avoid any inconsistencies the equation’s data must be setup in GQMEHOM1.m exactly as in NAGQME.m and vice versa. Hence, the user can compare the results of the NAGQME.m with the results of other software (e.g. [80], [79], [135], etc).

The inputs of this function are the equation’s solution, data matrices and their respective perturbation indices (see (4.47) in chapter 4): \(X, A_1, a_{A_1}, A_2, a_{A_2}, B_1, a_{B_1}, B_2, a_{B_2}, C_1, a_{C_1}, C_2, a_{C_2}, D_1, a_{D_1}, D_2, a_{D_2}, E_1, a_{E_1}, E_2, a_{E_2}, G, a_G\). Again, this m-file contains error facilities that guide the user to giving the appropriate data for a correct setup.

The outputs of NAGQME.m are:

\[CnX]\: the condition number.
\[E_{prf_{max}}]\: is the upper bound of the practical relative forward error of the solution.
\[E_{prf_{max}fl}]\: is the upper bound of the practical relative forward error of the solution, including rounding errors.
\[E_{rb}\]: is the actual relative backward error.
\[E_{rb_{max}}]\: is the upper bound of the relative backward error of the solution.
\[E_{rb_{max}fl}\]: is the upper bound of the relative backward error of the solution, including rounding errors.
\[NS\]: is an index for the numerical stability according to proposition 4.1 (see chapter 4). If \(NS = 1\) then the method is numerically stable, whereas if \(NS = 0\) then the method is numerically unstable.
\[NS_{fl}\]: is an index for numerical stability including rounding errors (see chapter 4). If \(NS_{fl} = 1\) then the method is numerically stable, whereas if \(NS_{fl} = 0\) then the method is numerically unstable.

The above outputs are mathematically defined in sections 4.2-4.5 in chapter 4.

Now, apart from NAQME.m, GQME-Toolbox includes the driver \textit{NATEST.m} for NAQME.m.

\textbullet \hspace{1em} \textit{NATEST.m}

\textit{NATEST.m} is just a collection of a set of MATLAB editor commands that are executed by typing in the editor the command \textit{natest}. This should take place after the run of GQMEHOM1.m. By typing \textit{natest} and then pressing enter, the numerical analysis test to the computed solution that is provided by GQMEHOM1.m will take place. \textit{NATEST.m} can be found in appendix C. In \textit{NATEST.m}, the perturbation indices of the matrices are set to be equal.
to their respective Frobenious norms. For different values, the user should interfere with the code in NATEST.m and set its desired values.

Closing this section, it should pointed out that both the elements of the GQME-Toolbox and any typing commands must take place in the same directory in the MATLAB workspace.

5.4 Possible Failures and Criticism of Computed Results

In this section, possible (problem dependent) program failures, other relevant problems such as how to criticize computed results and also possible future software additions and improvements will be presented.

Runtime errors and catastrophic failures because of data inconsistency are less likely to occur in GQME-Toolbox, since the software enforces the user to setup a particular problem appropriately before it begins the major computations. Another positive feature of the software is its robustness subject to the globally convergence algorithms with probability-1. This is something that can be guaranteed from the mathematical perspective. However, algorithmic failures and breakdowns are still possible.

The only failure that has been observed so far from the author’s computational experience with GQME-Toolbox, is breakdowns of the RK45GQMEHOM1.m function due to singular solutions at the end point ($\epsilon = 1$) of the homotopy paths. These solutions could be either at infinity or finite. The actual cause of such breakdowns is that the Jacobian of the homotopy equation with respect to the computed solution point sometimes became ill conditioned, in terms of matrix inverse computations in INGQMEHOM1.m, as $\epsilon \to 1$. Nevertheless, the breakdowns occur at a point on the homotopy path very close to the end point (e.g. for $\epsilon = 0.9999999999$). Hence, the tracking of homotopy path stops very close to the end point, and the end game solver takes this last computed point as initial condition in order to compute the solution at the end point. In other words, after occurring breakdowns of RK45GQMEHOM1.m the flow of computations in GQMEHOM1.m do not stop. Hence, a solution will always be provided from the software.

Another possible hazard case is the already mentioned case of tracking a path which diverges to infinity as the homotopy variable $\epsilon \to 1$. In this case, the bad thing is that one must wait forever without getting any answer. Note that although many of such cases can be exterminated using the homogeneous projective transformations (see chapter 3), this extermination is guaranteed only in a probabilistic sense and only under certain assumptions. The epistemological weakness of not knowing the geometry of solutions in many cases makes the above failure a realistic scenario.
Hence, the program should be in a position to scrutinize any path divergence to infinity and to stop the numerical process. A logical criterion is when the number of integration steps in RK45GQMEHOM1.m and the number of minimum steps are less than some preset values (i.e. \texttt{nsteps}, \texttt{stepmin}).

So far it should be said that the whole spectrum of hazard failures has focused on ill conditioning operations on Jacobians through the computational task of tracking the homotopy paths. Many experiments and trials have to be performed still in order to reveal any other possible failures. Also, various comparisons with problems with known solutions and numerical behaviour should take place. GQME-Toolbox is a relatively young software and it is logical to say that it must be tested extensively. Many, tests will generate possible ways of improvements and extensions of the software. All these constitute a new ‘experimental’ research topic, and are out of the scope of this thesis. However, the GQME-Toolbox seems to perform very well so far for the majority of the problems that have been tested. Many of these numerical problems are engineering and scientific problems and will be reported in the next chapters of this thesis. Furthermore, examples that have been solved with another homotopy software [182], [121], will be solved in this thesis as well, and comparisons made.

Another important issue is runtime errors. These can occur when trying to compute all the solutions of an equation, via polynomial homotopy, without causing any problem to the computational process. There are different types of runtime errors and criteria for analysing them. Hence, a computation can be characterized as ‘successful’, if the number of different solutions obtained (when using polynomial homotopy) equals the total degree and no singular solutions exist. A computation can be characterized as ‘almost successful’, if the number of computed solutions equals the total degree and the singular solutions have only multiplicity 2. A computation is characterized as ‘unknown successful’, if there is no knowledge about the geometry and number of solutions. A computation can be characterized as ‘runtime failure’, if two paths converged to a nonsingular solution, or if only one of a pair of complex conjugate solutions was found when the equation has real coefficients, or if a path was incorrectly stopped because of unrealistic \texttt{nsteps}, \texttt{stepmin} values, or if a solution that is known to exist was not found.

Due to the above runtime errors, possible measures to eliminate them are to increase the precision along the computations and make any decision criteria more tight. Unfortunately, the absence of the information about the geometry and number of solutions in many problems makes any rigorous criticism of the computation impossible.

Another problem that has been revealed with GQME-Toolbox, is the dilemma of deciding when a homogeneous projective transformation should be assumed zero. Here again there is no definite answer. Usually, a common sense criterion is when the recovered solution produces a big equation residual and analysis indicates numerical stability.
It is important to mention that, the present software performs reasonably well in terms of computation times in personal computers for medium size matrix dimensions. For solutions containing more than 900 elements, the computations start to become slow. In this respect, in the software, norm estimators instead of actual norm computations are used via the MATLAB function \textit{normest.m}. However, note that computation power is getting stronger from month to month. The first programs of GQME-Toolbox were executed on Pentium-I 150 MHz processors and now the market is full of Pentium-IV 1.5G MHz processors with upgrades on the way. This seems optimistic with respect to reduced computation times. Also, parallel computing for the present software is another research issue.

5.5 Synopsis

In this chapter the GQME-Toolbox, has been presented. The software code is provided in the appendix C. Technical details and conclusions were reported through this chapter and they will not repeated in this section.

An important feature of GQME-Toolbox is its capability of solving a wide range of numerical solution and analysis problems of algebraic second order polynomial equations.

Once more, the reader is strongly recommended to read the code along with the present chapter and with chapters 3 and 4 as well. This will help him or her to understand better and be able to criticize any computation. Numerical examples will be given as case studies in the next chapters of this thesis. In these chapters, the code of each problem setup will also be provided in order to practically prove the robustness and how user friendly the GQME-Toolbox is.
Chapter 6

Generalized Algebraic Riccati Equations

6.1 Introduction

In this chapter, the specialization of (2.1) into the generalized algebraic Riccati equation (ARE) is studied. Equation (2.1) with $\omega = \xi = 2$ can be viewed as a generalized ARE. AREs play very important roles in stability analysis, in systems theory, and particularly in modern control systems design. There is a vast literature on AREs.

The practical significance of AREs in control systems design has initiated two main branches of research. The first is the derivation of necessary and/or sufficient conditions for the existence of special symmetric solutions (e.g. positive, negative definite semi-definite). The second is their numerical solution.

Dealing with the existence and the geometry of solutions of such equations is a very difficult theoretical task. So far, there are only a few isolated results for some specific forms of AREs [151], [152]. The majority of these results have been derived in terms of algebraic [147] and differential geometry [67]. The study of existence and geometry of solutions of AREs has become a separate field of research in abstract mathematics.

The numerical solution of some special symmetric forms of AREs has dominated the majority of the literature. There is state-of-the-art software, which can very efficiently compute some special kind of solutions for real symmetric AREs. The solutions of AREs that are of practical interest in control systems design and analysis are the positive and the negative definite/semidefinite solutions. These solutions are often a main part of feedback control laws.

The majority of the literature on the numerical solution of AREs is dedicated to the design of iterative pure algebraic matrix algorithms. There is only a very small part of research...
dedicated to the theoretical study and the numerical solution of these equations from the dynamical systems viewpoint.

To be more specific, the majority of the existing algorithms for the numerical solution of AREs, focus on the symmetric ARE

\[ A^T X + XA - XDX + C = 0 \]  

(6.1)

and its dual

\[ AX + XAT - XDX + C = 0, \]  

(6.2)

where \( A, C, D, X \in \mathbb{R}^{n \times n}, n \in \mathbb{Z}^+, C > 0 \) or \( C \geq 0, D > 0 \) or \( D \geq 0 \).

For (6.1) and (6.2), the majority of existing algorithms (and their variations) are categorized as:

b) Schur methods [107].
c) Sign function methods [57].
d) Newton iterative methods [106], [188].
e) Deflation correction iterative methods [118].
f) Optimization methods: Davidov method [112], gradient method with and without exact line search [19], global Newton method [137], [53].
g) Initial value problem methods [23], [103], [141].
h) Linear matrix inequality methods (ellipsoid and interior point polynomial algorithms) [63], [24], [154].
i) Neural networks methods [150].

Although different, the numerical processes of the algorithms for most of the above methods can be modelled as dynamic matrix flows [35], [92]. Hence, in the end all methods can be viewed from a dynamical systems viewpoint. So far studies of this nature are almost absent from the literature. The main reason is that these methods require large computational power, which fortunately is becoming more readily available.

(6.1) and (6.2) often appear in the design of continuous time control systems, where usually a positive definite (semidefinite) solution is required. (6.1) and (6.2) are also known as continuous time classical AREs. Now, the numerical analysis of (6.1) and (6.2) can be found in [133]-[136], [94].

In descriptor continuous time control systems design [17] the ARE

\[ A^T XE + E^TXA - E^T XDXE + C = 0 \]  

(6.3)

and its dual

\[ AXE^T + EXA^T - EXDXE^T + C = 0, \]  

(6.4)

is often met, where \( A, C, D, E, X \in \mathbb{R}^{n \times n}, n \in \mathbb{Z}^+, C > 0 \) or \( C \geq 0, D > 0 \) or \( D \geq 0 \).
For (6.2) and (6.3), the majority of existing algorithms (and their variations) are focused on Newton iterative methods [19]. Note that (6.3) and (6.4), are also called generalized AREs, although in this thesis, the term 'generalized' is concerned with more general forms than (6.3) and (6.4); namely

\[ A_1XB_1 + A_2XB_2 + C_1XD_1XE_1 + C_2XD_2XE_2 + G = 0. \]  

(6.5)

Now, a postmodern control system design very frequently considers the minimization of multi-objective cost functions. These problems usually end up with design equations as (6.5) (i.e. equations more complex than the AREs (6.1)-(6.4)). Therefore, (6.5) might be useful for future complex designs. Apart from its practical usefulness, equation (6.5) is mathematically attractive in the sense that standard existing methods stated above for the numerical solution of classical and descriptor continuous time AREs do not apply to its solution.

Recent examples of generalized AREs in multi-objective postmodern control systems are the optimal projection equations (OPEs) appearing in [21], [85], [68], and [172]. An OPE, usually has the form

\[ A^TXA + XDA - XDZ + tDXZt + C = 0 \]  

(6.6)

and its dual is given by

\[ AX + XAT - XDX + tDXZtT + C = 0, \]  

(6.7)

where \( A, C, D, \tau, X \in \mathbb{R}^{n \times n}, n \in \mathbb{Z}^+, C > 0 \) or \( C \geq 0 \), \( D > 0 \) or \( D \geq 0 \), and \( \tau \) is idempotent\(^\dagger\).

For (6.1)-(6.4), (6.6) and (6.7), theoretical results consider only sufficient and/or necessary conditions for the existence of positive definite (semidefinite) solutions \( X \). As far as the author is aware, there is no theory developed for the existence and geometry of solutions of (6.5). This is an open research problem.

It is obvious that all the previous equations, can be considered as special cases of (2.1), and hence their numerical solution and analysis can be carried out via the probability-1 homotopy algorithms of chapters 2-3 and the analysis of chapter 4. At this point it should be noted that some heuristic iteration and continuation algorithms for (6.6) and (6.7) were presented in [21], [143], [31] and [173]. Exceptions to these heuristic algorithms are [178] and [60].

As was stated previously the subject of AREs is very big and practically impossible to review comprehensively in this chapter. Hence, this chapter deals with the numerical solution of equations (6.1)-(6.7) only. The concept of the existence of solutions and their geometry is not considered here. Details of these subjects can be found in [106], [23], [90], [151], [152], [188], [21], [143] and their references. In addition, a very good survey for the numerical solution of AREs can be found in [18].

\(^\dagger\) A square matrix said to be idempotent if \( \tau^2 = \tau \) [140].
Now, apart from their applications in control systems theory, AREs in a more restricted form arise in the study of quadratic dynamical systems. In other words, the steady state part of such systems, being set equal to zero, can be equivalently written as generalized AREs. Hence, the solutions to such AREs can determine the dynamical systems equilibrium points and therefore can be useful in bifurcation analysis.

The structure of this chapter is as follows. Section 6.2 demonstrates how to extract via the GQME-Toolbox (see chapter 5), solutions of special kind (e.g. positive definite solutions) for (6.1)-(6.4), (6.6) and (6.7). Next, in section 6.3, examples with known solutions will be solved and comparisons with existing methods given in terms of numerical stability, behaviour and accuracy of computations. Finally, in section 6.4 general conclusions are given.

### 6.2 Computation of Special Kind Solutions of AREs

A major problem when dealing with OPEs and AREs, is to compute, when they exist, solutions of a specific definiteness. Of particular interest are the positive definite or semidefinite solutions or negative definite or semidefinite solutions. The rest of this section is given to establishing a way to compute such solutions as well as indefinite solutions. First the problem with respect to a symmetric OPE is considered, which can specialised to a classical ARE.

Consider (6.6). Since only one solution is desirable to compute, define the fixed point homotopy (matrix) equation

\[
H(X(\varepsilon), \varepsilon) := \varepsilon(A^T X(\varepsilon) + X(\varepsilon)A - X(\varepsilon)DX(\varepsilon) + \tau^T X(\varepsilon)DX(\varepsilon)\tau + C) - \gamma(1 - \varepsilon)(X(\varepsilon) - X(0))
\] (6.8)

Assuming that (6.6) has a positive definite (or semidefinite) solution, it will be shown that with a specific choice of \(X(0) > 0\) and a \(\gamma \in \mathbb{R}_+\), there is a path consisting of positive definite and/or positive semidefinite solutions, connecting \(X(0) > 0\) with the positive definite (or semidefinite) solution of (6.8). In other words, the traced homotopy path consists of positive definite and/or positive semidefinite solutions. Hence the algorithms proposed in chapter 2 will be able to trace this path and give the desired solution. Similarly, a negative definite or semidefinite solution is computed with \(X(0) < 0\) and a \(\gamma \in \mathbb{R}_-\). In this case, the traced homotopy path is well defined and consists of negative definite and/or negative semidefinite solutions.

Suppose that for a specific \(\varepsilon\), (6.8) has a real symmetric solution \(X(\varepsilon) = X^T(\varepsilon)\). There exists (see [113]) a \(V_\varepsilon := V(\varepsilon)\), \(V_\varepsilon \in \mathbb{R}^{n \times n}\) such that

\[
V_\varepsilon^TV_\varepsilon = V_\varepsilon V_\varepsilon^T = I_n
\] (6.9)
and
\[ X(\epsilon) = V_\epsilon X_d(\epsilon)V_\epsilon^T, \]  
(6.10)

where \( X_d(\epsilon) \in \mathbb{R}^{n \times n} \), \( X_d(\epsilon) := \text{diag}\left( [\lambda_1(\epsilon) \ \lambda_2(\epsilon) \ \ldots \ \lambda_n(\epsilon)] \right) \) is the diagonal matrix consisting of the real eigenvalues of \( X(\epsilon) \). Substituting (6.10) into (6.9) we have
\[ H(X(\epsilon),\epsilon) = 0 \iff H(X_d(\epsilon),\epsilon) = 0, \]  
(6.11)

\[ H(X_d(\epsilon),\epsilon) := \epsilon\left( A^TV_\epsilon X_d(\epsilon)V_\epsilon^T + V_\epsilon X_d(\epsilon)V_\epsilon^T A - V_\epsilon X_d(\epsilon)V_\epsilon^T D V_\epsilon X_d(\epsilon)V_\epsilon^T + \tau^TV_\epsilon X_d(\epsilon)V_\epsilon^T D V_\epsilon X_d(\epsilon)V_\epsilon^T \tau + C - \gamma(1-\epsilon)V_\epsilon X_d(\epsilon)V_\epsilon^T - X(0) \right). \]  
(6.12)

Now define the square matrices:
\[ \tilde{A} := V_\epsilon^T AV_\epsilon, \quad \tilde{D} := V_\epsilon^T DV_\epsilon, \quad \tilde{\tau} := V_\epsilon^T \tau V_\epsilon, \quad \tilde{C} := V_\epsilon^T CV_\epsilon, \quad \tilde{X}(0) := V_\epsilon^T X(0)V_\epsilon. \]

Since \( C = C^T > 0 \) (or \( C = C^T \geq 0 \)) we have, as for \( X(\epsilon) \), that \( C = V_C C_d V_C^T \), \( V_C \in \mathbb{R}^{n \times n} \), such that \( V_C^T V_C = V_C V_C^T = I_n \) and \( C_d \in \mathbb{R}^{n \times n} \), \( C_d := \text{diag}\left( [\lambda_{C_1} \ \lambda_{C_2} \ \ldots \ \lambda_{C_n}] \right) \), and \( C_d > 0 \) (or \( C_d \geq 0 \)). Using (6.9) we have,
\[ \tilde{C} := V_\epsilon^T CV_\epsilon = V_\epsilon^T V_C C_d V_C^T V_\epsilon = (V_\epsilon^T V_C) C_d (V_\epsilon^T V_C)^T. \]

From (6.9) and \( V_C^T V_C = V_C V_C^T = I_n \), we have
\[ \left( V_\epsilon^T V_C \right)^T V_\epsilon^T V_C = V_\epsilon^T V_C^T V_C V_\epsilon = (V_C V_\epsilon)^T (V_C V_\epsilon). \]

Therefore \( \tilde{C} > 0 \) (or \( \tilde{C} \geq 0 \)). Similarly, it can be proven that \( \tilde{D} > 0 \) (or \( \tilde{D} \geq 0 \)) if \( D < 0 \) respectively; and \( \tilde{X}(0) > 0 \) or \( \tilde{X}(0) < 0 \) if \( X(0) > 0 \) or \( X(0) < 0 \), respectively.

Setting (6.12) equal to zero because of (6.11), multiplying (6.12) = 0 from the left and right with \( V_\epsilon^T \) and \( V_\epsilon \) respectively, and using (6.10), (6.12) = 0 becomes equivalently
\[ H(X_d(\epsilon),\epsilon) = \epsilon\left( \tilde{A}^T X_d(\epsilon) + X_d(\epsilon)\tilde{A} - X_d(\epsilon)\tilde{D} X_d(\epsilon) + \tilde{\tau}^T X_d(\epsilon)\tilde{D} X_d(\epsilon)\tilde{\tau} + \tilde{C} - \gamma(1-\epsilon)X_d(\epsilon)\right) = 0. \]  
(6.13)

By doing all the algebra in (6.13) it follows that the \( (i,j) \) element of \( H(X_d(\epsilon),\epsilon) \) is given by
\[ H_{ij}(X_d(\epsilon),\epsilon) := \epsilon \tilde{A}_{ij} \lambda_i(\epsilon) + \epsilon \tilde{A}_{ji} \lambda_j(\epsilon) + \epsilon \sum_{k=1}^{n} \left[ \sum_{k=1}^{n} \left( \tilde{\tau}_{ki} \tilde{D}_{kj} \lambda_k(\epsilon) \lambda_i(\epsilon) \right) \tilde{\tau}_{kj} \right] - \epsilon \tilde{D}_{ji} \lambda_j(\epsilon) \lambda_i(\epsilon) - \gamma(1-\epsilon) X_d_{ij}(\epsilon) + \epsilon \tilde{C}_{ij} + \gamma(1-\epsilon) \tilde{X}_{ij}(0), \]  
(6.14)
where \((\cdot)_{ij}\) denotes the \((i,j)\) element of a matrix.

To study the definiteness of \(X(\epsilon)\), is equivalent to studying the definiteness of \(X_d(\epsilon)\) and hence the signs of the diagonal elements of \(X_d(\epsilon)\). This is feasible by examining the signs of the diagonal elements of (6.13). Hence we have to analyse \(n\) scalar second order polynomials.

From (6.14), considering the diagonal elements \(H_{ii}(X_d(\epsilon),\epsilon)\ \forall i = 1,\ldots,n\), we have

\[
H_{ii}(X_d(\epsilon),\epsilon) = 2\tilde{\epsilon} \lambda_i(\epsilon) + \epsilon \sum_{k=1}^{n} \left( \sum_{k=1}^{n} (\tilde{\tau}_{ki} \tilde{D}_{ki} \lambda_i(\epsilon) \lambda_i(\epsilon)) \tilde{\tau}_{ki} \right) - \epsilon \tilde{D}_{ii} \lambda_i^2(\epsilon) - \gamma(1-\epsilon) \lambda_i(\epsilon) + \epsilon \tilde{C}_{ii} + \gamma(1-\epsilon) \tilde{X}_{ii}(0) = 0
\]

\[
\Leftrightarrow 2\tilde{\epsilon} \lambda_i(\epsilon) + \epsilon \left( \sum_{k=1}^{n} (\tilde{\tau}_{ki} \tilde{D}_{ki} \lambda_i^2(\epsilon) + \epsilon \sum_{k=1}^{n} \left( \sum_{k=1}^{n} (\tilde{\tau}_{ki} \tilde{D}_{ki} \lambda_i(\epsilon) \lambda_i(\epsilon)) \tilde{\tau}_{ki} \right) \right) \lambda_i(\epsilon)
\]

\[
- \epsilon \tilde{D}_{ii} \lambda_i^2(\epsilon) - \gamma(1-\epsilon) \lambda_i(\epsilon) + \epsilon \tilde{C}_{ii} + \gamma(1-\epsilon) \tilde{X}_{ii}(0) = 0
\]

\[
\Leftrightarrow \epsilon \tilde{D}_{ii} \left( \tilde{\tau}_{ii} \sum_{k=1}^{n} (\tilde{\tau}_{ki}) - 1 \right) \lambda_i^2(\epsilon)
\]

\[
+ 2\tilde{\epsilon} \lambda_i + \epsilon \left( \sum_{k=1}^{n} \left( \sum_{k=1}^{n} (\tilde{\tau}_{ki} \tilde{D}_{ki} \lambda_i(\epsilon) \lambda_i(\epsilon)) \tilde{\tau}_{ki} \right) \right) - \gamma(1-\epsilon) \lambda_i(\epsilon)
\]

\[
+ \epsilon \tilde{C}_{ii} + \gamma(1-\epsilon) \tilde{X}_{ii}(0) = 0.
\]  

(6.14)

Define the quantities below:

\[
a_i := \epsilon \tilde{D}_{ii} \left( \tilde{\tau}_{ii} \sum_{k=1}^{n} (\tilde{\tau}_{ki}) - 1 \right)
\]

(6.15)

\[
d_i := 2\tilde{\epsilon} \lambda_i + \epsilon \left( \sum_{k=1}^{n} \left( \sum_{k=1}^{n} (\tilde{\tau}_{ki} \tilde{D}_{ki} \lambda_i(\epsilon) \lambda_i(\epsilon)) \tilde{\tau}_{ki} \right) \right) - \gamma(1-\epsilon)
\]

(6.16)

\[
c_i := \epsilon \tilde{C}_{ii} + \gamma(1-\epsilon) \tilde{X}_{ii}(0)
\]

(6.17)

Using (6.15)-(6.17), (6.14) can be written as

\[
H_{ii}(X_d(\epsilon),\epsilon) = a_i \lambda_i^2(\epsilon) + d_i \lambda_i(\epsilon) + c_i = 0.
\]

(6.18)

So the problem is now to study the signs of (6.18) \(\forall i = 1,\ldots,n\).
Next, with lemma 6.1, the topology of solutions of (6.18) is determined.

**Lemma 6.1:** Let \( c_i \geq 0 \), and \( \Delta := d_i^2 - 4a_i c_i \). The topology of solutions of (6.18) is determined from the following:

(i) If \( a_i < 0 \) and \( d_i \leq 0 \), then the positive and negative solutions of (6.18) belong to the sets

\[
S^+_{a_i d_i^-1} \left( \lambda_i \right) = \left\{ \lambda_i \in \mathbb{R} : i = 1, \ldots, n \left| 0 \leq \lambda_i \left( \varepsilon \right) \leq \frac{1}{2} \left| a_i^{-1} \sqrt{\Delta} \right| \right. \}
\]

and

\[
S^-_{a_i d_i^-1} \left( \lambda_i \right) = \left\{ \lambda_i \in \mathbb{R} : i = 1, \ldots, n \left| \lambda_i \left( \varepsilon \right) \leq -\frac{1}{2} \left| a_i^{-1} \sqrt{\Delta} \right| \right. \}
\]

respectively.

(ii) If \( a_i < 0 \) and \( d_i \geq 0 \), then the positive and negative solutions of (6.18) belong to the sets

\[
S^+_{a_i d_i^+} \left( \lambda_i \right) = \left\{ \lambda_i \in \mathbb{R} : i = 1, \ldots, n \left| \lambda_i \left( \varepsilon \right) \geq \frac{1}{2} \left| a_i^{-1} \sqrt{\Delta} \right| \right. \}
\]

and

\[
S^-_{a_i d_i^+} \left( \lambda_i \right) = \left\{ \lambda_i \in \mathbb{R} : i = 1, \ldots, n \left| \lambda_i \left( \varepsilon \right) \geq -\frac{1}{2} \left| a_i^{-1} \sqrt{\Delta} \right| \right. \}
\]

respectively.

(iii) If \( a_i > 0 \), \( d_i \leq 0 \) and \( \Delta > 0 \), then the positive and negative solutions of (6.18) belong to the sets

\[
S^+_{a_i d_i^-} \left( \lambda_i \right) = \left\{ \lambda_i \in \mathbb{R} : i = 1, \ldots, n \left| \lambda_i \left( \varepsilon \right) \geq 0 \right. \}
\]

and

\[
S^-_{a_i d_i^-} \left( \lambda_i \right) = \emptyset
\]

respectively.

(iv) If \( a_i > 0 \), \( d_i \geq 0 \) and \( \Delta > 0 \), then the positive and negative solutions of (6.18) belong to the sets

\[
S^+_{a_i d_i^+} \left( \lambda_i \right) = \emptyset
\]

and

\[
S^-_{a_i d_i^-} \left( \lambda_i \right) = \left\{ \lambda_i \in \mathbb{R} : i = 1, \ldots, n \left| \lambda_i \left( \varepsilon \right) \leq 0 \right. \}
\]

respectively.

(v) If \( a_i = 0 \), then (6.18) has the unique solution \( \lambda_i \left( \varepsilon \right) = -c_i d_i^{-1} \).

**Proof.**

For (i), \( \Delta > 0 \), hence there are two real solutions:

\[
\lambda_i_1 \left( \varepsilon \right) = \frac{1}{2} \left( -d_i + \sqrt{\Delta} \right) a_i^{-1}, \quad \lambda_i_2 \left( \varepsilon \right) = \frac{1}{2} \left( -d_i - \sqrt{\Delta} \right) a_i^{-1}.
\]

Now
Therefore $\lambda_i^1(\varepsilon) \leq 0$. Now suppose that $-d_i - \sqrt{\Delta} \geq 0$. Then

$$-d_i \geq \sqrt{\Delta} \iff |d_i| \geq |d_i|^2 \geq d_i^2 - 4a_i c_i \iff a_i \geq 0,$$

which is a contradiction. Therefore

$$-d_i - \sqrt{\Delta} \leq 0,$$

and hence $-d_i - \sqrt{\Delta} \leq 0 \iff \frac{1}{2}(-d_i - \sqrt{\Delta})a_i^{-1} \geq 0 \iff \lambda_i^2(\varepsilon) \geq 0$. Also

$$-d_i \geq 0 \iff -d_i - \Delta^2 \geq -\sqrt{\Delta} \iff \frac{1}{2}(-d_i - \sqrt{\Delta})a_i^{-1} \leq \frac{1}{2} |\sqrt{\Delta}a_i^{-1}| \iff \lambda_i^2(\varepsilon) \leq \frac{1}{2} |\sqrt{\Delta}a_i^{-1}|.$$
From the above it is clear that the positive and negative solutions of (6.18) belong to the sets

\[ S^+_{a_i^+d_i^-}(\varepsilon) = \{ \lambda_i(\varepsilon) \in \mathbb{R}, i = 1, \ldots, n | \lambda_i(\varepsilon) \geq 0 \} \quad \text{and} \quad S^-_{a_i^-d_i^+}(\varepsilon) = \emptyset, \]

respectively. For (iv) again by the hypothesis of \( \Delta > 0 \), there are two real roots

\[ \lambda_{i_1}(\varepsilon) = \frac{1}{2} (-d_i + \sqrt{\Delta}) a_i^{-1}, \quad \lambda_{i_2}(\varepsilon) = \frac{1}{2} (-d_i - \sqrt{\Delta}) a_i^{-1}. \]

It is obvious that \( \lambda_{i_2}(\varepsilon) \leq 0 \). Also

\[ -d_i \leq 0 \iff -d_i - \sqrt{\Delta} \leq -d_i + \sqrt{\Delta} \iff \frac{1}{2} (-d_i - \sqrt{\Delta}) a_i^{-1} \leq - \frac{1}{2} \sqrt{\Delta} a_i^{-1} \iff \lambda_{i_2}(\varepsilon) \leq - \frac{1}{2} \sqrt{\Delta} a_i^{-1}. \]

Now suppose that, \( -d_i + \sqrt{\Delta} \geq 0 \). Then \( |d_i| \leq \sqrt{\Delta} \iff |d_i| \leq d_i^2 + 4a_i a_i \iff a_i \leq 0 \), which is a contradiction. Hence \( -d_i + \sqrt{\Delta} \leq 0 \), and we have

\[ -d_i + \sqrt{\Delta} \leq 0 \iff \frac{1}{2} (-d_i + \sqrt{\Delta}) a_i^{-1} \leq 0 \iff \lambda_{i_1}(\varepsilon) \leq 0. \]

Therefore the positive and negative solutions of (6.18) belong to the sets

\[ S^+_{a_i^+d_i^-}(\varepsilon) = \emptyset \quad \text{and} \quad S^-_{a_i^-d_i^+}(\varepsilon) = \{ \lambda_i(\varepsilon) \in \mathbb{R}, i = 1, \ldots, n | \lambda_i(\varepsilon) \leq 0 \}. \]

The proof of (v) is straightforward given \( \lambda_i(\varepsilon) = -c_i d_i^{-1} \) when \( a_i = 0 \).}

A schematic representation of the topology of solutions of (6.18) for an \( \varepsilon \in \{0,1]\) is shown in figure 6.1.

![Figure 6.1. Topology of solutions of (6.18).](image-url)
By observing figure 6.1 it is clear that the real solutions of (6.11) are governed by two kinds of ordering for \( \forall \epsilon \in (0,1] \). We refer to these as type-1 when \( a_i < 0 \) and type-2 when \( a_i > 0 \). This ordering, however, collapses at \( \epsilon = 0 \), where we have the unique solution \( \tilde{X}(0) \).

Also note that
\[
S^+_{a_i} b_i^+ (\epsilon) \cup S^-_{a_i} b_i^- (\epsilon) = \mathbb{R}^+ \cup \{0\},
\]
\[
S^-_{a_i} d_i^+ (\epsilon) \cup S^+_{a_i} d_i^- (\epsilon) = \mathbb{R}^- \cup \{0\},
\]
\[
S^+_{a_i} d_i^+ (\epsilon) \cup S^+_{a_i} d_i^- (\epsilon) \cup S^-_{a_i} d_i^+ (\epsilon) \cup S^-_{a_i} d_i^- (\epsilon) = \mathbb{R}.
\]

Now with respect to different kinds of real solutions of (6.6) we distinguish the following cases:

1. **Positive and negative definite/semidefinite solutions for AREs;** The OPE (6.6) with \( \sigma = 0 \), is the classical ARE (6.1). Suppose that \( D > 0 \) (or \( D \geq 0 \)), \( C > 0 \) (or \( C \geq 0 \)). Also, assume that a positive definite/semidefinite solution exists. Note that, a unique maximal positive definite/semidefinite solution exists if and only if the pair \( (A,D) \) is controllable [106]. Under the controllability assumption on \( (A,D) \) and \( \forall X(0) > 0 \) (or \( \forall X(0) < 0 \)), \( \forall y \in \mathbb{R}_+ \) (or \( \forall y \in \mathbb{R}_- \)), it will be shown next that a unique maximal positive (minimal negative) definite/semidefinite solution is well defined as a solution to the homotopy equation (6.8) \( \forall \epsilon \in \overline{U} \).

**Lemma 6.2;** Let (6.8) = 0 and assume that \( \sigma = 0 \), \( D > 0 \) or \( D \geq 0 \), \( C > 0 \) or \( C \geq 0 \), \( (A,D) \) is controllable. Then almost always \( \forall X(0) > 0 \), \( \forall y \in \mathbb{R}_+ \) and \( \forall \epsilon \in \overline{U} \), (6.8) possesses a unique maximal positive definite/semidefinite solution and \( \forall X(0) < 0 \), \( \forall y \in \mathbb{R}_- \) and \( \forall \epsilon \in \overline{U} \), (6.8) possesses a unique minimal negative definite/semidefinite solution.

**Proof.**

First note that (6.8) = 0 under the assumptions of lemma 6.2 can be equivalently written as
\[
\epsilon \left( A^T X(\epsilon) + X(\epsilon)A - X(\epsilon)DX(\epsilon) + C \right) - \gamma(1 - \epsilon)(X(\epsilon) - X(0)) = 0
\]
\( \varepsilon \left( A - \frac{1}{2} \gamma (1 - \varepsilon)I_n \right)^T X(\varepsilon) + X(\varepsilon) \varepsilon \left( A - \frac{1}{2} \gamma (1 - \varepsilon)I_n \right) - X(\varepsilon)(\varepsilon D)X(\varepsilon) + (\varepsilon C + \gamma (1 - \varepsilon)X(0)) = 0 \)  

(6.19)

For \( \varepsilon = 0 \), (6.19) has the unique positive (negative) definite solution \( X(0) \). For \( \varepsilon = 1 \), (6.19) is equivalent to ARE (6.1) and has a unique maximal positive (minimal negative) definite/semidefinite solution if and only if \( (A, D) \) is controllable (see [106], [103]), which is true in the present case. Since \( \forall \varepsilon \in U, \varepsilon D > 0 \) or \( \varepsilon D \geq 0 \) and \( \varepsilon C + \gamma (1 - \varepsilon)X(0) > 0 \) whether \( X(0) > 0 \) and \( \gamma \in \mathbb{R}_+ \) or \( X(0) < 0 \) and \( \gamma \in \mathbb{R}_- \), (6.19) will possess a unique maximal positive (minimal negative) definite/semidefinite solution, if and only if \( \left( \varepsilon \left( A - \frac{1}{2} \gamma (1 - \varepsilon)I_n \right), (\varepsilon D) \right) \) is controllable (see [106], [103]). Now, \( \left( \varepsilon \left( A - \frac{1}{2} \gamma (1 - \varepsilon)I_n \right), (\varepsilon D) \right) \) is controllable and this is a generic result (which hold with probability-1) the proof of which can be found in theorem 3 of [187].

Hence, it is apparent that the probability-1 fixed point homotopy algorithms in chapter 2 will be able with the appropriate setup (according to lemma 6.2) to obtain positive (negative) definite/semidefinite solutions for the ARE (6.1) when they exist. Note that under the assumptions of lemma 6.2 the homotopy paths are well defined under the same kind of definiteness of solutions. Hence, the convergence is guaranteed with probability 1 without taking into account the homogeneous projective transformations in chapter 3.

Next, the behavior of a homotopy path tracking process under lemma 6.2 is analyzed in geometric terms in view of the topology of solutions of (6.18) in figure 6.1.

Assume first the case of \( X(0) > 0 \), \( \gamma \in \mathbb{R}_+ \). From (6.15) and (6.17) and because \( \tau = 0 \), \( a_i < 0 \) and \( c_i > 0 \) respectively, for every \( \varepsilon \in (0,1] \). Therefore \( \forall \varepsilon \in (0,1] \), we have solutions of type-1 only. For very small values of \( \varepsilon \) (i.e. \( \varepsilon \to 0^+ \)), we have \( d_i \to -\gamma < 0 \), and the solutions belong to \( S^+_{a_i, d_i}(\varepsilon) \), \( S^-_{a_i, d_i}(\varepsilon) \). For the case of \( \varepsilon = 0 \), the unique solution is positive. On the other hand there are no negative solutions for \( \varepsilon = 0 \). Hence, a continuous path from \( \varepsilon = 0 \) to \( \varepsilon = 1 \) consisting of positive definite/semidefinite solutions is defined. Hence, the homotopy (6.8) will be able to provide the desired solution. Now when tracing the path in practice (e.g. as an IVP), as \( \varepsilon \) increases from \( \varepsilon = \varepsilon_1 \) to \( \varepsilon = \varepsilon_2 \), with correspondingly small steps, if \( \varepsilon_1, \varepsilon_2 \), correspond to a \( d_i < 0 \), then according to the continuity and differentiability of (6.11) we have
\[ \lambda_1(\varepsilon_1), \lambda_2(\varepsilon_2) \in \left\{ S_+^{a_i^{-1} d_i^{-1}}(\varepsilon_1) \cap S_+^{a_i^{-1} d_i^{-1}}(\varepsilon_2) \right\} . \] Similarly when \( \varepsilon_1 \) corresponds to a \( d_i < 0 \) and \( \varepsilon_2 \) corresponds to a \( d_i > 0 \), or vice versa, \( \lambda_1(\varepsilon_1), \lambda_2(\varepsilon_2) \in \left\{ S_-^{a_i^{-1} d_i^{-1}}(\varepsilon_1) \cap S_-^{a_i^{-1} d_i^{-1}}(\varepsilon_2) \right\} . \] Note for that for \( \varepsilon \in (0,1] \), moving from the set \( S_+^{a_i^{-1} d_i^{-1}}(\varepsilon_1) \) to the set \( S_-^{a_i^{-1} d_i^{-1}}(\varepsilon_1) \), will result in a non-smooth (i.e. not differentiable) path and is therefore unlikely.

The above all hold, under the assumption that a positive definite/semidefinite solution exists.

Similarly to the previous case above, a negative definite/semidefinite solution, can be evaluated \( \forall X(0) < 0 \) and \( \forall \gamma \in \mathbb{R}_+ \).

2. **Positive and negative definite/semidefinite solutions for OPEs:** The computation of positive and negative definite/semidefinite solutions for OPEs, are obtained with the same choices of \( X(0) \) and \( \gamma \) as in the ARE case 1 above. The analysis is exactly the same as before, assuming for example a diagonal structure on \( \tau \) and also that \( \tau \) is idempotent. When \( \tau \) is not diagonal then solutions of both types 1 and 2 are considered. In any case, the existence of a specific kind of definiteness along the whole homotopy paths is a matter of geometry of solutions, and should be proved according to derived results similar to lemma 6.2. For the OPEs this task is out of the scope of this thesis.

3. **Positive and negative definite solutions for algebraic Lyapunov matrix equations (ALE) \[^77\];** This case has to do with (v) of lemma 6.1, from where it is obvious that homotopy (6.8) consists only of unique positive definite/semidefinite solutions, when \( A \) has eigenvalues with negative real part and \( C > 0 / C \geq 0 \) or negative definite/semidefinite solutions when \( A \) has eigenvalues with positive real part and \( C > 0 / C \geq 0 \).

4. **Indefinite solutions for the AREs, OPEs:** In the computation of indefinite solutions for AREs and OPEs, the signs of the eigenvalues of the solutions no longer play a role. In order to get such a symmetric solution we choose \( X(0) \) indefinite and any \( \gamma \).

5. **Non-symmetric solutions for the AREs, OPEs, algebraic Sylvester matrix equations (ASE) \[^58\], \[^59\], \[^34\] and other forms of (6.5);** For non-symmetric solutions of AREs, OPEs and Sylvester algebraic matrix equations the previous analysis of this section does not apply. In this case in order to break the symmetry of the problem formulation we initialise the problem with a non-symmetric \( X(0) \) and any \( \gamma \).
Note it is obvious that the results of this section apply similarly for the dual equations (6.2), (6.7). Also for (6.3) and (6.4), it is routine to show that similar results can be obtained.

If a solution path defined by (6.11) is very stiff, then additional attention should be given to the integration algorithm when using a continuous tracking method, or to the discrete tracking step when using a discrete tracking method (see chapter 2). In the case of a continuous tracking method, it is better to use state-of-the-art algorithms for stiff systems [75], [76], [62].

In the case of AREs and OPEs, when specific kinds of solutions are traced along the homotopy paths, some ‘directionality rules’ can be included within the IVP ODE algorithm. These directionality rules can give the right direction and act as step correctors to the tracking of homotopy paths. For stiff problems, if no special consideration is given to the integration algorithm, or to the discrete tracking step, then using the directionality rules, it is still possible to achieve the desired solution at $\varepsilon = 1$. Another advantage of these rules can be for example in situations where jumps from one path to another occur during the tracking process. Four such directionality rules are presented next.

### 6.2.1 Directionality Rules

The directionality rules which will be developed in this section consist of propositions 6.1-6.4, as stated below. In the sequel, $A \succeq 0$ ($A \preceq 0$) means that $A \in \mathbb{R}^{n \times n}$ is not negative (not positive) definite and $\lambda_{\min}(A)$, $\lambda_{\max}(A)$ denote the minimum and the maximum eigenvalues of $A$ respectively.

**Theorem 6.1 (sufficient condition):** Let $A, B \in \mathbb{R}^{n \times n}$, $A = A^T$, $A \succeq 0$, $B = B^T$, $k \in \mathbb{R}_+$. Then

$$\left( \lambda_{\min}(B) < 0 \text{ and } k \leq -\frac{\lambda_{\max}(A)}{\lambda_{\min}(B)} \right) \text{ or } \left( k > 0 \text{ and } \lambda_{\min}(B) \geq 0 \right) \Rightarrow A + kB \succeq 0.$$  

**Proof.**

Let the eigenvalues of $A$, $B$ and $A + kB$, namely $\lambda_i(A)$, $\lambda_i(B)$, and $\lambda_i(A + kB)$ respectively, be arranged in increasing order (i.e. $\lambda_i(\cdot) \leq \lambda_{i+1}(\cdot)$, $\forall i = 1, 2, \ldots, n$).

From the Weyl inequality (B.2.28) in appendix B we have:

$$\lambda_i(A) + \lambda_{\min}(kB) \leq \lambda_i(A + kB) \leq \lambda_i(A) + \lambda_{\max}(kB).$$

Note that $A + kB \succeq 0$ is equivalent to $\lambda_i(A + kB) \geq 0$. From the Weyl inequality with $i = \max$ we have

$$\lambda_{\max}(A) + \lambda_{\min}(kB) \leq \lambda_{\max}(A + kB) \leq \lambda_{\max}(A) + \lambda_{\max}(kB).$$

Now for $\lambda_{\max}(A + kB) \geq 0$ it is sufficient that
\[ \lambda_{\max}(A) + \lambda_{\min}(kB) \geq 0 \] or equivalently \[ \lambda_{\max}(A) + k\lambda_{\min}(B) \geq 0. \]

From the last inequality we have \[ k \leq -\frac{\lambda_{\max}(A)}{\lambda_{\min}(B)} \] when \[ \lambda_{\min}(B) < 0 \] (i.e. \( B \leq 0 \)) and \[ k \geq -\frac{\lambda_{\max}(A)}{\lambda_{\min}(B)} \] when \[ \lambda_{\min}(B) > 0 \] (i.e. \( B > 0 \)). Now because \( A \geq 0 \) we have \( \lambda_{\max}(A) > 0 \).

Hence \( \lambda_{\min}(B) < 0 \Leftrightarrow -\frac{\lambda_{\max}(A)}{\lambda_{\min}(B)} \geq 0 \) and \( \lambda_{\min}(B) > 0 \Leftrightarrow -\frac{\lambda_{\max}(A)}{\lambda_{\min}(B)} \leq 0 \). Now since, \( k > 0 \) the result of theorem 6.1 follows.

Lemma 6.3 (sufficient condition): Let \( A, B \in \mathbb{R}^{n \times n} \), \( A = A^T \), \( A \geq 0 \). \( B = B^T \).

\[ 0 < k \leq h , \ h \in \mathbb{R}_+ \] Then

\[ \left( \lambda_{\min}(B) < 0 \text{ and } h \leq -\frac{\lambda_{\max}(A)}{\lambda_{\min}(B)} \right) \text{ or } \left( h > 0 \text{ and } \lambda_{\min}(B) \geq 0 \right) \Rightarrow A + kB \geq 0. \]

Proof.

Summing up \( h \leq -\frac{\lambda_{\max}(A)}{\lambda_{\min}(B)} \) and \( 0 < k \leq h \) we have \( k + h \leq -\frac{\lambda_{\max}(A)}{\lambda_{\min}(B)} + h \), hence

\[ k \leq -\frac{\lambda_{\max}(A)}{\lambda_{\min}(B)} \]. Now, from theorem 6.1 the result follows.

As a straightforward result of lemma 6.3, we have the following:

Proposition 6.1 (not negative directionality rule): Let \( A_j \geq 0 \) and \( k_j \) be obtained from lemma 6.3. Then a sequence of not negative matrices \( \{A_i + k_iB_i\}_{i=j}^{\infty} \geq 0 \) is generated.

Proof.

The proof is an immediate consequence of lemma 6.3.

Theorem 6.2 (sufficient condition): Let \( A, B \in \mathbb{R}^{n \times n} \), \( A = A^T \), \( A \leq 0 \). \( B = B^T \).

\[ k \in \mathbb{R}_+ \]. Then

\[ \left( \lambda_{\max}(B) > 0 \text{ and } k \leq -\frac{\lambda_{\min}(A)}{\lambda_{\max}(B)} \right) \text{ or } \left( k > 0 \text{ and } \lambda_{\max}(B) \geq 0 \right) \Rightarrow A + kB \leq 0. \]
**Proof.**

Working, as in the proof of theorem 6.1, with the Weyl inequality (B.2.28) in appendix B we have: \( \lambda_i(A) + \lambda_{\min}(kB) \leq \lambda_i(A + kB) \leq \lambda_i(A) + \lambda_{\max}(kB) \). Note that \( A + kB \preceq 0 \) is equivalent to \( \lambda_i(A + kB) \leq 0 \). From the Weyl inequality with \( i = \min \) we have
\[
\lambda_{\min}(A) + \lambda_{\min}(kB) \leq \lambda_{\min}(A + kB) \leq \lambda_{\min}(A) + \lambda_{\max}(kB).
\]
Now for \( \lambda_{\min}(A + kB) \leq 0 \) it is sufficient that
\[
\lambda_{\max}(A) + \lambda_{\min}(kB) \leq 0 \quad \text{or equivalently} \quad \lambda_{\min}(A) + k \lambda_{\max}(B) \leq 0.
\]

From the last inequality we have \( k \leq \frac{-\lambda_{\min}(A)}{\lambda_{\max}(B)} \) when \( \lambda_{\max}(B) > 0 \) (i.e. \( B > 0 \)) and
\[
k \geq \frac{-\lambda_{\min}(A)}{\lambda_{\max}(B)} \quad \text{when} \quad \lambda_{\max}(B) < 0 \quad \text{(i.e.} \quad B < 0 \text{)}. \]

Hence \( \lambda_{\max}(B) > 0 \iff \frac{-\lambda_{\min}(A)}{\lambda_{\max}(B)} \geq 0 \) and \( \lambda_{\max}(B) < 0 \iff \frac{-\lambda_{\min}(A)}{\lambda_{\max}(B)} \leq 0 \). Now since, \( k > 0 \) the result of theorem 6.2 follows.

**Lemma 6.4 (sufficient condition):** Let \( A, B \in \mathbb{R}^{n \times n} \), \( A = A^T \), \( A \leq 0 \), \( B = B^T \), \( 0 < k \leq h \), \( h \in \mathbb{R}_+ \). Then
\[
\left( \lambda_{\max}(B) > 0 \quad \text{and} \quad h \leq \frac{-\lambda_{\min}(A)}{\lambda_{\max}(B)} \right) \quad \text{or} \quad \left( h > 0 \quad \text{and} \quad \lambda_{\max}(B) \geq 0 \right) \Rightarrow A + kB \preceq 0.
\]

**Proof.**

Summing up \( h \leq \frac{-\lambda_{\min}(A)}{\lambda_{\max}(B)} \) and \( 0 < k \leq h \) we have \( k + h \leq \frac{-\lambda_{\min}(A)}{\lambda_{\max}(B)} + h \), hence
\[
k \leq \frac{-\lambda_{\min}(A)}{\lambda_{\max}(B)}. \] Now, from theorem 6.1 the result follows.

As a straightforward result of lemma 6.4, we have the following:

**Proposition 6.2 (not positive directionality rule):** Let \( A_j \preceq 0 \) and \( k_j \) be obtained from lemma 6.4. Then a sequence of not negative matrices \( \{A_i + k_i B_i\}_{i=j}^{\infty} \leq 0 \) is generated.

**Proof.**

The proof is an immediate consequence of lemma 6.4.
Chapter 6 Generalized Algebraic Riccati Equations 208

Having developed direction rules for the generation of not negative and not positive sequences, we now give similar theorems and propositions for the generation of positive definite (semidefinite) and negative definite (semidefinite) sequences. The proofs of these results will be based on the Rayleigh-Ritz theorem B.2.1 in appendix B.

Theorem 6.3 (sufficient condition): Let $A, B \in \mathbb{R}^{n \times n}$, $A = A^T$, $B = B^T$, $k \in \mathbb{R}_+$. Then

If $A > 0$:

$$\left( \lambda_{\min}(B) \geq 0 \text{ or } \left( k < \frac{-\lambda_{\min}(A)}{\lambda_{\min}(B)} \text{ and } \lambda_{\min}(B) < 0 \right) \right) \Rightarrow A + kB > 0$$

(6.20)

If $A \geq 0$:

$$\lambda_{\min}(B) > 0 \Rightarrow A + kB > 0$$

(6.22)

$$\lambda_{\min}(B) = 0 \Rightarrow A + kB \geq 0$$

(6.23)

Proof.

Let the eigenvalues of $A$, $B$ and $A + kB$, be $\lambda_i(A)$, $\lambda_i(B)$, and $\lambda_i(A + kB)$ respectively $\forall i = 1, 2, ..., n$. From (B.2.27) in the Rayleigh-Ritz theorem B.2.2 in appendix B we have $\forall x \in \mathbb{R}_+^n$

$$x^T(A + kB)x = x^T A x + x^T(kB)x \geq \lambda_{\min}(A)x^T x + \lambda_{\min}(kB)x^T x$$

(6.20)

$$= \lambda_{\min}(A)x^T x + k \lambda_{\min}(B)x^T x = (\lambda_{\min}(A) + k \lambda_{\min}(B))x^T x .$$

From $\lambda_{\min}(B) \geq 0$ we have $B > 0$ if $\lambda_{\min}(B) > 0$ or $B \geq 0$ if $\lambda_{\min}(B) = 0$. Since $k > 0$ and $A > 0$ (i.e. $\lambda_{\min}(A) > 0$) we have $\forall x \in \mathbb{R}_+^n$, $x^T x > 0$ hence $(\lambda_{\min}(A) + k \lambda_{\min}(B))x^T x > 0$ and so $A + kB > 0$. Now since we want $A + kB > 0$ it is sufficient $\forall x \in \mathbb{R}_+^n$, that

$$(\lambda_{\min}(A) + k \lambda_{\min}(B))x^T x > 0 \Leftrightarrow \lambda_{\min}(A) + k \lambda_{\min}(B) > 0,$$

and since $x^T x > 0$ (6.20) follows.

Now, for $A + kB \geq 0$ it is sufficient $\forall x \in \mathbb{R}_+^n$, that

$$(\lambda_{\min}(A) + k \lambda_{\min}(B))x^T x \geq 0 \Leftrightarrow \lambda_{\min}(A) + k \lambda_{\min}(B) \geq 0,$$

and since $x^T x > 0$ (6.21) follows.
For \( A \geq 0 \) we have \( (\lambda_{\min}(A) + k\lambda_{\min}(B))x^T x = k\lambda_{\min}(B)x^T x \) and (6.22) and (6.23) follow.

**Lemma 6.5** (sufficient condition): Let \( A, B \in \mathbb{R}^{n \times n} \), \( A = A^T \), \( A \leq 0 \), \( B = B^T \), \( 0 < k \leq h \), \( h \in \mathbb{R}_+ \) and let the following logic hold:

If \( A > 0 \) then

\[
\begin{align*}
0 & < \frac{-\lambda_{\min}(A)}{\lambda_{\min}(B)} < h \quad \text{and} \quad \lambda_{\min}(B) > 0 \\
\text{then} \quad k & = \frac{-\lambda_{\min}(A)}{\lambda_{\min}(B)} \\
\text{else} \quad k & = h
\end{align*}
\]

If \( A \geq 0 \) then

\[
\begin{align*}
\text{If} \quad \lambda_{\min}(B) & \geq 0 \\
\text{then} \quad k & = h
\end{align*}
\]

Then under the validity of the above logic we have \( A + kB > 0 \) or \( A + kB \geq 0 \).

**Proof.**

Consider theorem 6.3 with \( 0 < k \leq h \). We will now determine \( k \) with respect to \( A \) and \( B \), in order to make \( A + kB > 0 \). We distinguish the following cases.

(i) If \( A > 0 \) then:

\[
\begin{align*}
\text{If} \quad B & \geq 0 \quad \text{or} \quad B > 0 \quad \text{then} \quad k = h \\
\text{If} \quad B & \leq 0 \quad \text{or} \quad B < 0 \quad \text{or} \quad B \quad \text{has both real positive and negative eigenvalues} \quad \text{then if} \quad \left(k < \frac{-\lambda_{\min}(A)}{\lambda_{\min}(B)} = a > 0 \quad \text{and} \quad k \leq h \right) \quad \text{then} \quad k = h \quad \text{if} \quad a \geq h \quad \text{or} \quad k = a \quad \text{if} \quad a < h.
\end{align*}
\]

(ii) If \( A \geq 0 \) then:

\[
\begin{align*}
\text{If} \quad B & \geq 0 \quad \text{or} \quad B > 0 \quad \text{then} \quad k = h.
\end{align*}
\]

In order to make \( A + kB \geq 0 \) we similarly distinguish the following cases.

(i) If \( A > 0 \) then:
If \( B \leq 0 \) or \( B < 0 \) or \( B \) has both real positive and negative eigenvalues then
\[
\frac{-\lambda_{\min}(A)}{\lambda_{\min}(B)} = a > 0 \text{ and } k \leq h
\]
then \( k = h \) if \( a \geq h \) or \( k = a \) if \( a < h \).

(ii) If \( A \geq 0 \) then if \( B \geq 0 \) then \( k = h \).

As a straightforward result of lemma 6.5, we have the following:

**Proposition 6.3 (positive definite (semidefinite) directionality rule):** Let \( A_j \) and \( k_j \) be obtained from lemma 6.5. Then a sequence consisting of positive definite and/or positive semidefinite matrices \( \{A_j + k_j B_i\}_{i=j}^{\infty} > 0 \) and/or \( \{A_j + k_j B_i\}_{i=j}^{\infty} \geq 0 \) is generated.

**Proof.**
The proof is an immediate consequence of lemma 6.5.

**Theorem 6.4 (sufficient condition):** Let \( A, B \in \mathbb{R}^{n \times n} \), \( A = A^T \), \( B = B^T \), \( k \in \mathbb{R}_+ \).

Then

If \( A > 0 \):
\[
\left( \lambda_{\max}(B) \leq 0 \right) \text{ or } \left( k < \frac{-\lambda_{\max}(A)}{\lambda_{\max}(B)} \text{ and } \lambda_{\max}(B) > 0 \right) \Rightarrow A + kB < 0 \tag{6.24}
\]
\[
\left( k \leq \frac{-\lambda_{\max}(A)}{\lambda_{\max}(B)} \text{ and } \lambda_{\max}(B) > 0 \right) \Rightarrow A + kB \leq 0 \tag{6.25}
\]

If \( A \geq 0 \):
\[
\lambda_{\max}(B) < 0 \Rightarrow A + kB < 0 \tag{6.26}
\]
\[
\lambda_{\max}(B) = 0 \Rightarrow A + kB \leq 0 \tag{6.27}
\]

**Proof.**
Let the eigenvalues of \( A \), \( B \) and \( A + kB \), be \( \lambda_i(A) \), \( \lambda_i(B) \), and \( \lambda_i(A + kB) \) respectively \( \forall i = 1, 2, \ldots, n \). From (B.2.27) in the Rayleigh-Ritz theorem B.2.1 in appendix B we have \( \forall x \in \mathbb{R}^n \)
\[
x^T(A + kB)x = x^T Ax + x^T (kB)x \leq \lambda_{\max}(A)x^Tx + \lambda_{\max}(kB)x^Tx
\]
\[
= \lambda_{\max}(A)x^Tx + k\lambda_{\max}(B)x^Tx = (\lambda_{\max}(A) + k\lambda_{\max}(B))x^T x.
\]
From $\lambda_{\max}(B) \leq 0$ we have $B < 0$ if $\lambda_{\max}(B) < 0$ or $B \leq 0$ if $\lambda_{\max}(B) = 0$. Since $k > 0$ and $A < 0$ (i.e. $\lambda_{\max}(A) < 0$) we have $\forall x \in \mathbb{R}^n_+$, $x^T x > 0$ hence

$$(\lambda_{\max}(A) + k\lambda_{\max}(B))x^T x < 0$$

and so $A + kB < 0$. Now since we want $A + kB < 0$ it is sufficient $\forall x \in \mathbb{R}^n_+$, that

$$(\lambda_{\max}(A) + k\lambda_{\max}(B))x^T x < 0 \iff \lambda_{\max}(A) + k\lambda_{\max}(B) < 0,$$

and since $x^T x > 0$ (6.24) follows.

Now, for $A + kB \leq 0$ it is sufficient $\forall x \in \mathbb{R}^n_+$, that

$$(\lambda_{\max}(A) + k\lambda_{\max}(B))x^T x \leq 0 \iff \lambda_{\max}(A) + k\lambda_{\max}(B) \leq 0,$$

and since $x^T x > 0$, (6.25) follows.

For $A \leq 0$ we have

$$(\lambda_{\max}(A) + k\lambda_{\max}(B))x^T x = k\lambda_{\max}(B)x^T x$$

and (6.26) and (6.27) follow. □

**Lemma 6.6 (sufficient condition):** Let $A, B \in \mathbb{R}^{n \times n}$, $A = A^T$, $A \leq 0$, $B = B^T$, $0 < k \leq h$, $h \in \mathbb{R}_+$ and let the following logic hold:

If $A < 0$ then

If $0 < \frac{-\lambda_{\max}(A)}{\lambda_{\max}(B)} < h$ and $\lambda_{\max}(B) > 0$

then

$$k = \frac{-\lambda_{\max}(A)}{\lambda_{\max}(B)}$$

else

$$k = h$$

end

end

If $A \leq 0$ then

If $\lambda_{\min}(B) \leq 0$ then

$$k = h$$

end

end

Then under the validity of the above logic we have $A + kB < 0$ or $A + kB \leq 0$.

*Proof.*
Consider theorem 6.4 with $0 < k \leq h$. We will now determine $k$ with respect to $A$ and $B$, in order to make $A + kB < 0$. We distinguish the following cases.

(i) If $A < 0$ then:
If $B \leq 0$ or $B < 0$ then $k = h$. If $B \geq 0$ or $B = 0$ or $B$ has both real positive and negative eigenvalues then if $k < \frac{\lambda_{\max}(A)}{\lambda_{\max}(B)} = a > 0$ and $k \leq h$ then $k = h$ if $a \geq h$ or $k = a$ if $a < h$.

(ii) If $A \leq 0$ then:
If $B \leq 0$ or $B < 0$ then $k = h$.

In order to make $A + kB \leq 0$ we similarly distinguish the following cases.

(i) If $A < 0$ then:
If $B \geq 0$ or $B > 0$ or $B$ has both real positive and negative eigenvalues then if $k \leq \frac{\lambda_{\max}(A)}{\lambda_{\max}(B)} = a > 0$ and $k \leq h$ then $k = h$ if $a \geq h$ or $k = a$ if $a < h$.

(ii) If $A \leq 0$ then if $B \leq 0$ then $k = h$.

As a straightforward result of lemma 6.5, we have the following:

**Proposition 6.4 (negative definite (semidefinite) directionality rule):** Let $A_j$ and $k_j$ be obtained from lemma 6.6. Then a sequence consisting of negative definite and/or negative semidefinite matrices, $\{A_i + k_i B_i\}_{i=j}^{\infty} < 0$ and/or $\{A_i + k_i B_i\}_{i=j}^{\infty} \leq 0$ is generated.

**Proof.**
The proof is an immediate consequence of lemma 6.5.

At this point note that an ODE numerical routine, for tracing the homotopy paths, can be described generally as $\text{vec}(Y_{i+1}) = \text{vec}(Y_i) + h \frac{d(f(\text{vec}(Y_i)))}{d(\text{vec}(Y_i))^T}$. Now, propositions 6.1-6.4 take place with $A_i := Y_i$, $B_i := \frac{d(f(\text{vec}(Y_i)))}{d(\text{vec}(Y_i))^T}$, $B^{(i)} := f'(Y_i)$, and finally the actual computation is $\text{vec}(Y_{i+1}) = \text{vec}(Y_i) + k_i \frac{d(f(\text{vec}(Y_i)))}{d(\text{vec}(Y_i))^T}$. Since all propositions 6.1-6.4 are sufficient conditions when the step length $k_i$ does not satisfy the appropriate definiteness, then
This so far takes place only in cases where a specific definiteness of a solution does not exist.

For the case of a discrete tracking method of the homotopy paths, when one of propositions 6.1-6.4 is used it determines the discrete tracking homotopy step of the process. Note that when using one of propositions 6.1-6.4 within a stiff problem, the path that is tracked is an approximation to the actual one. This is because the step provided from the proposition can still be bigger than the step required for an accurate tracking of the real path. Nevertheless, if one is interested in the solution point \( X(1) \), and not the intermediate points \( X(\epsilon) \) \( \forall \epsilon \in U \), then propositions 6.1-6.4 are useful, in not increasing the computational time as a very small integration of discrete tracking homotopy step would do. Note that depending on the kind of solution that is to be computed, an appropriate proposition, among propositions 6.1-6.4, is posed within the numerical process. This is implemented in RK45GQMEHOM1.m in GQME-Toolbox (see appendix C). In RK45GQMEHOM1.m the appropriate sections subject to a positive (negative) definite/semidefinite solution or subject to a not positive (not negative) definite/semidefinite solution, are determined by the choice of p\( \text{def} \). Hence, for positive (negative) definite/semidefinite solutions, p\( \text{def}=1 \) (p\( \text{def}=-1 \)). For not positive (not negative) definite/semidefinite solution, p\( \text{def}=2 \) (p\( \text{def}=-2 \)).

Finally propositions 6.1-6.4, can be used as step corrector criteria in the ODE step, even when the problem is not stiff. For example, if both a positive and a negative solution are very close to zero, then it may be a case where the ODE algorithm, or the discrete tracking process, produces a step which results in a jump from a point on the path of one solution to a point on the path of the other solution. In such a situation the appropriate proposition 6.1-6.4 will correct the step in order to track the right path.

### 6.3 Numerical Examples

In this section three well known and one modified numerical example from the literature are examined via the GQME-Toolbox. Useful discussions and remarks are given in the relevant sections for each example. The MATLAB driver m-files for each example are given in appendix D. The computations implemented in a two parallel PIII 550 MHz personal computer using MATLAB v. 5.3.1.29215a (R11.1) (Unix version) with unit roundoff \( u_r = 2^{-53} \) under the operating system RedHat Linux v. 6.2.

#### 6.3.1 Numerical Example 6.1

Consider the ARE (6.1) with data,
This particular equation has the exact positive definite solution
\[
\begin{bmatrix}
\frac{1}{D_{11}}(A_{11} + \sqrt{A_{11}^2 + C_{11}D_{11}}) & 0 & 0 \\
0 & \frac{1}{D_{22}}(A_{22} + \sqrt{A_{22}^2 + C_{22}D_{22}}) & 0 \\
0 & 0 & \frac{1}{D_{33}}(A_{33} + \sqrt{A_{33}^2 + C_{33}D_{33}})
\end{bmatrix}
\]

It is known from [135] and it will also be experimentally verified in the sequel that (6.1) with the above data becomes ill-conditioned with an increase of \( k \) due to an increase of \( \|X\|_2 \).

There are several ways to write the above ARE in the form of (2.1) with \( \omega = \xi = 2 \). The most apparent is by setting \( A_1 := A^T \), \( B_2 := A \), \( A_2 := B_1 := C_1 := E_1 := I_3 \), \( D_1 := D \), \( G := C \), \( C_2 := D_2 := E_2 := 0 \). With the last formulation and for the reasons mentioned in chapter 4, the posteriori numerical analysis of the computed solutions can be pessimistic since some matrices are repeated. Nevertheless, the above problem can be skipped with the formulation \( A_1 := 0.1A^T \), \( B_2 := 10I_3 \), \( A_2 := 5I_3 \), \( B_2 := 0.2A \), \( C_1 := 0.4I_3 \), \( D_1 := 2D \), \( E_1 := 0.5I_3 \), \( C_2 := 0.6I_3 \), \( D_2 := 0.5D \), \( E_2 := 2I_3 \), \( G := C \). At this point note that it is obvious that all the general results that have been developed in section 6.2, hold regardless of the way that (6.1) is written as (2.1).

For example 6.1, the positive definite solutions are computed for \( k = 1, \ldots, 6 \) via the fixed point homotopy algorithm of GQME-Toolbox with \( \gamma = 1 \). The initial condition for all cases where such that the easy problem solution was \( X(0) = I_3 \). Solutions have been computed considering no homogeneous projective transformations, since the existence of the positive definite homotopy path is guaranteed from lemma 6.2 in section 6.2. Also, the solutions of the homotopy equations have been computed as \( X_c(e) \) considering no scaling and as \( \bar{X}_c(e) \) considering the scaling (3.31). Note that, by construction the scaling (3.31) is preserving the symmetry properties of (6.1) and the general results of section 6.2. In all cases a posteriori numerical analysis of the computed solutions have been conducted.

The driver to run example 6.2 in GQME-Toolbox is the m-file Example61.m. The user has only to type Example61 and to press Enter in the MATLAB workspace where the GQME-
Toolbox is installed. Example61.m is given in appendix D and it can cover problem cases with larger dimensions (see appendix D).

The computational results are presented in figure 6.2 and in tables 6.1.1, 6.1.2. In these tables, \( n.p \) stands for a computation with no homogeneous projective transformation and \( n.s. \), \( s \) stands for a computation with no scaling and scaling (3.31) respectively. The computation of the scaling problem (3.31) (see chapter 3) for \( k = 1, \ldots, 6 \) is as follows.

All cases took 0.05 (sec) CPU time and 140408 number of FLOPS.

(i) \( k = 1 \): \[ S(u_X, u_C^0) = 0.0) = 6.0451e+002, \quad \min \{ S(u_X, u_C^0) \} = 4.1183e+002, \]

\[ u_X = vec \begin{bmatrix} 5.2893e-001 \\ 3.5963e-001 \\ 5.0720e-001 \end{bmatrix}, \]

\[ u_C = -vec \begin{bmatrix} 4.3981e-001 \\ 4.5584e-001 \\ 4.7534e-001 \end{bmatrix}. \]

(ii) \( k = 2 \): \[ S(u_X, u_C^0) = 0.0) = 1.7872e+003, \quad \min \{ S(u_X, u_C^0) \} = 1.2042e+003, \]

\[ u_X = vec \begin{bmatrix} 8.7396e-001 \\ 5.3799e-001 \\ 8.3469e-001 \end{bmatrix}, \]

\[ u_C = -vec \begin{bmatrix} 6.3318e-001 \\ 6.6026e-001 \\ 6.9633e-001 \end{bmatrix}. \]

(iii) \( k = 3 \): \[ S(u_X, u_C^0) = 0.0) = 3.7019e+003, \quad \min \{ S(u_X, u_C^0) \} = 2.5138e+003, \]

\[ u_X = vec \begin{bmatrix} 1.2190e+000 \\ 7.1635e-001 \\ 1.1622e+000 \end{bmatrix}, \]

\[ u_C = -vec \begin{bmatrix} 8.2655e-001 \\ 8.6468e-001 \\ 9.1733e-001 \end{bmatrix}. \]
(iv) $k = 4$: $S(u_X, u_C)_{|(u_X, u_C) = (0, 0)} = 6.3487e + 003$, $\min\{S(u_X, u_C)\} = 4.3406e + 003$

$$u_X = \begin{bmatrix} 1.5640e + 000 \\ 8.9471e - 001 \\ 1.4897e + 000 \end{bmatrix}$$

$$u_C = \begin{bmatrix} 1.0199e + 000 \\ 1.0691e + 000 \\ 1.1383e + 000 \end{bmatrix}$$

(v) $k = 5$: $S(u_X, u_C)_{|(u_X, u_C) = (0, 0)} = 9.7274e + 003$, $\min\{S(u_X, u_C)\} = 6.6848e + 003$

$$u_X = \begin{bmatrix} 1.9091e + 000 \\ 1.0731e + 000 \\ 1.8171e + 000 \end{bmatrix}$$

$$u_C = \begin{bmatrix} 1.2133e + 000 \\ 1.2735e + 000 \\ 1.3593e + 000 \end{bmatrix}$$

(vi) $k = 6$: $S(u_X, u_C)_{|(u_X, u_C) = (0, 0)} = 1.3838e + 004$, $\min\{S(u_X, u_C)\} = 9.5462e + 003$

$$u_X = \begin{bmatrix} 2.2541e + 000 \\ 1.2514e + 000 \\ 2.1446e + 000 \end{bmatrix}$$

$$u_C = \begin{bmatrix} 1.4067e + 000 \\ 1.4779e + 000 \\ 1.5803e + 000 \end{bmatrix}$$

Now, in tables 6.1.1 and 6.1.2, $\text{det}(J_{X_c})$, $\frac{\|\text{Residual}(X_c)\|_F}{\|X_c\|_F}$, $\frac{\|X - X_c\|_F}{\|X\|_F}$ and $C_n(X_c)$, denote the determinant of the jacobian of the, the normalized residual error, the exact backward error and the condition number of the computed solution (4.83) respectively. $\text{E}_{\text{prf}_{\text{max}}}(X_c)$, $\text{E}_{\text{prf}_{\text{min}}}(X_c)$, $\text{E}_{\text{rb}}(X_c)$, $\text{E}_{\text{prb}_{\text{max}}}(X_c)$ and $\text{E}_{\text{prb}_{\text{min}}}(X_c)$, denote the left hand parts of (4.95), (4.96), (4.90), (4.91) and (4.92) respectively. $NS_{\text{fl}}$, $NS$ are the numerical stability indexes considering and not considering floating point rounding errors respectively.
Figure 6.2. Homotopy paths for example 6.1.
### Table 6.1.1

<table>
<thead>
<tr>
<th>$k$</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\det(JX_c)$</td>
<td>n.s.n.p</td>
<td>-4.924e+015</td>
<td>-3.2012e+023</td>
</tr>
<tr>
<td>$\left| Residual(X_c) \right|_F$</td>
<td>s.n.p</td>
<td>-4.924e+015</td>
<td>-3.2012e+023</td>
</tr>
<tr>
<td>$\left| X_c \right|_F$</td>
<td>n.s.n.p</td>
<td>1.0904e-014</td>
<td>9.4373e-014</td>
</tr>
<tr>
<td>$\left| X \right|_F$</td>
<td>s.n.p</td>
<td>1.0904e-014</td>
<td>9.4373e-014</td>
</tr>
<tr>
<td>$\left| X - X_c \right|_F$</td>
<td>n.s.n.p</td>
<td>0</td>
<td>2.3685e-019</td>
</tr>
<tr>
<td>$\left| X - X_c \right|_F$</td>
<td>s.n.p</td>
<td>0</td>
<td>2.3685e-019</td>
</tr>
<tr>
<td>$C_n(X_c)$</td>
<td>n.s.n.p</td>
<td>9.8386</td>
<td>7.2496e+001</td>
</tr>
<tr>
<td>$s.n.p$</td>
<td>9.8386</td>
<td>7.2496e+001</td>
<td>7.2204e+002</td>
</tr>
<tr>
<td>$E_{pr_{\text{max}}}(X_c)$</td>
<td>n.s.n.p</td>
<td>9.5439e-017</td>
<td>8.3348e-017</td>
</tr>
<tr>
<td>$s.n.p$</td>
<td>9.5439e-017</td>
<td>8.3348e-017</td>
<td>4.7650e-017</td>
</tr>
<tr>
<td>$E_{pr_{\text{Ilmax}}}(X_c)$</td>
<td>n.s.n.p</td>
<td>4.4616e-014</td>
<td>3.5278e-013</td>
</tr>
<tr>
<td>$s.n.p$</td>
<td>3.8037e-016</td>
<td>2.3406e-015</td>
<td>3.4368e-012</td>
</tr>
<tr>
<td>$E_{\text{rd}}(X_c)$</td>
<td>n.s.n.p</td>
<td>5.1003e-017</td>
<td>3.2088e-018</td>
</tr>
<tr>
<td>$s.n.p$</td>
<td>5.1003e-017</td>
<td>3.2088e-018</td>
<td>6.1374e-017</td>
</tr>
<tr>
<td>$E_{pr_{\text{bmax}}}(X_c)$</td>
<td>n.s.n.p</td>
<td>2.6720e-014</td>
<td>2.8652e-013</td>
</tr>
<tr>
<td>$s.n.p$</td>
<td>2.6720e-014</td>
<td>2.8652e-013</td>
<td>1.6052e-012</td>
</tr>
<tr>
<td>$E_{pr_{\text{bIlmax}}}(X_c)$</td>
<td>n.s.n.p</td>
<td>1.2491e-011</td>
<td>1.2127e-009</td>
</tr>
<tr>
<td>$s.n.p$</td>
<td>1.0649e-013</td>
<td>8.0462e-012</td>
<td>1.1576e-007</td>
</tr>
<tr>
<td>$NS$</td>
<td>n.s.n.p</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$s.n.p$</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$NS_{fl}$</td>
<td>n.s.n.p</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$s.n.p$</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$e_{tol_{\text{IVP}}}$</td>
<td>n.s.n.p</td>
<td>1.0000e-012</td>
<td>1.0000e-012</td>
</tr>
<tr>
<td>$s.n.p$</td>
<td>1.0000e-012</td>
<td>1.0000e-012</td>
<td>1.0000e-012</td>
</tr>
<tr>
<td>$n_{\text{steps}_{\text{IVP}}}$</td>
<td>n.s.n.p</td>
<td>502</td>
<td>637</td>
</tr>
<tr>
<td>$s.n.p$</td>
<td>458</td>
<td>605</td>
<td>670</td>
</tr>
<tr>
<td>$min_{\text{step}_{\text{IVP}}}$</td>
<td>n.s.n.p</td>
<td>1.912300e-005</td>
<td>1.937339e-007</td>
</tr>
<tr>
<td>$s.n.p$</td>
<td>3.59082e-005</td>
<td>4.516728e-007</td>
<td>5.605260e-009</td>
</tr>
<tr>
<td>$CPU_{\text{IVP}}$ (sec)</td>
<td>n.s.n.p</td>
<td>2.6390e+001</td>
<td>3.3250e+001</td>
</tr>
<tr>
<td>$s.n.p$</td>
<td>2.4440e+001</td>
<td>3.2010e+001</td>
<td>3.5350e+001</td>
</tr>
<tr>
<td>$n_{\text{FLOPS}_{\text{IVP}}}$</td>
<td>n.s.n.p</td>
<td>43987308</td>
<td>55856807</td>
</tr>
<tr>
<td>$s.n.p$</td>
<td>40147120</td>
<td>52976839</td>
<td>58736948</td>
</tr>
</tbody>
</table>

Table 6.1.1. Computation results for example 6.1.
<table>
<thead>
<tr>
<th>$k$</th>
<th>$det(J_{X_c})$</th>
<th>$|\text{Residual}(X_c)|_F$</th>
<th>$|X_c|_F$</th>
<th>$|X-X_c|_F$</th>
<th>$C_n(X_c)$</th>
<th>$E_{prf_{\text{max}}}(X_c)$</th>
<th>$E_{prf_{\text{max}}}(X_c)$</th>
<th>$E_{\text{rb}}(X_c)$</th>
<th>$E_{prb_{\text{max}}}(X_c)$</th>
<th>$E_{prb_{\text{flmax}}}(X_c)$</th>
<th>$NS$</th>
<th>$NS_{fl}$</th>
<th>$e_{tol_{IVP}}$</th>
<th>$n_{\text{steps}_{IVP}}$</th>
<th>$\text{min}<em>{\text{step}</em>{IVP}}$</th>
<th>$t_{CPU_{IVP}}(sec)$</th>
<th>$n_{FLOPS_{IVP}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$n.s.n.p$</td>
<td>$s.n.p.$</td>
<td>$n.s.n.p$</td>
<td>$s.n.p.$</td>
<td>$n.s.n.p$</td>
<td>$s.n.p.$</td>
<td>$n.s.n.p$</td>
<td>$s.n.p.$</td>
<td>$n.s.n.p$</td>
<td>$s.n.p.$</td>
<td>$n.s.n.p$</td>
<td>$s.n.p.$</td>
<td>$n.s.n.p$</td>
<td>$s.n.p.$</td>
<td>$n.s.n.p$</td>
<td>$s.n.p.$</td>
<td>$n.s.n.p$</td>
</tr>
</tbody>
</table>

Table 6.1.2. Computation results for example 6.1.
If the method is numerically stable according to proposition 4.1, then the index takes the value of 1 and the value of 0 otherwise. For more details see NAGQME.m in appendix C. $e_{tol_{IVP}}$ denotes the IVP relative tolerance (i.e. $e$ in GQMEHOM1.m, (see chapter 5)). $n_{steps_{IVP}}$ denotes the IVP total number of steps (i.e. $n_{steps}$ in GQMEHOM1.m, (see chapter 5)). $min_{step_{IVP}}$ denotes the IVP minimum step (i.e. $step_{min}$ in GQMEHOM1.m, (see chapter 5)). $t_{CPU_{IVP}}(sec)$, $n_{FLOPS_{IVP}}$ denote the CPU time in seconds and the number of FLOPS of IVP respectively (i.e. $t_{CPU}$ and $n_{FLOPS}$ in GQMEHOM1.m, (see chapter 5)). Details of the above quantities in tables 6.1.1 and 6.1.2 can be found in chapters 2, 3, 4 and 5.

Now, the same ARE for $k = 1, \ldots, 6$, has been computed with already existing state-of-the-art routines. More specifically, these routines are the MATLAB v. 5.3.1.29215a (R11.1) (Unix version) are.m, aresolv.m (with respect to eigenspace decomposition, referred to as the eigen method), aresolv.m (with respect to Schur decomposition, referred to as the schur method), care.m, the sign function method of [135], [57] (not included but implemented by the author in MATLAB and referred to as sfcare method). Also the initial value problem method [23], [103] (referred to as the ode15s method) was performed using the MATLAB v. 5.3.1.29215a (R11.1) (Unix version) IVP numerical routine for stiff problems ode15s.m. In addition, computations considering the scaling method of [135] (see also [94], [132]) were implemented via are.m, aresolv.m (with respect to both eigenspace and Schur decompositions, referred to as the eigens and schurs methods respectively), and the sign function method mentioned previously (referred to as the sfcares method).

Hence, in total twelve methods were used for the computation of example 6.1. For every method a posteriori numerical analysis was conducted via NAGQME.m and the results are compared graphically in figures 6.3 and 6.4.

At this point, it should be said that the arithmetic used in the present example is the MATLAB fixed complex arithmetic (i.e. $flih=0$), but effectively computations are in real arithmetic because the homotopy equation is defined in real Euclidean vector space. Now, no homotopy step corrector was posed (i.e. $flag_{cor}=1$) and the 'end game' tolerance was set to $efinal=1e^{-20}$ with the jacobian formula provided in the 'end game' solver under a large scale optimization method $largescale=\text{'on'}$. The maximum allowed number of steps and the minimum allowed integration step of the IVP routine RK45GQMEHOM1.m were set to the MATLAB real maximum and minimum number values respectively (i.e. $maxn_{steps}=\text{realmax}$, $min_{step}=\text{realmin}$). Also, no discrete homotopy in the data was considered, thus $step=1$. 
Figure 6.3. Comparison graphs for example 6.1.
Figure 6.4. Comparison graphs for example 6.1.
Now, considering the directionality rules in subsection 6.2.1, it has been found that positive definite solutions for the case with and without scaling, can be obtained without the positive definite directionality proposition 6.3 (i.e. $p_{def} = 0$) if $e_{tol_{IVP}} \leq 10^{-009}$ and $e_{tol_{IVP}} \leq 10^{-011}$ respectively. When the positive definite directionality proposition 6.3 is posed (i.e. $p_{def} = 1$), it has been found that, positive definite solutions for the case with and without scaling, can be obtained if $e_{tol_{IVP}} \leq 10^{-009}$ and $e_{tol_{IVP}} \leq 10^{-008}$ respectively. Obviously by reducing $e_{tol_{IVP}}$ we gain in computational time, but this is in conflict with numerical accuracy. Hence, an appropriate setup subject to $e_{tol_{IVP}}$ and $p_{def}$ is user depended.

From the above results, the main conclusion is that the GQME-Toolbox results generally in a numerical stable algorithm for the numerical solution of a fairly difficult numerical problem. Since the exact solutions are known, tables 6.1.1, 6.1.2 reveal the conclusion that the error estimates are by no means pessimistic. Specifically, the backward error estimates are very close to the exact backward errors with respect to the unit roundoff accuracy $u_r = 2^{-53}$. For all cases, the forward and backward error estimates are less or about equal in magnitude of order $u_r = 2^{-53}$.

From the error estimation viewpoint, improvements can be observed when scaling (3.31) is used in the computations. Also, there is a slight improvement in the number of steps of the IVP routine and hence in the computation times when scaling is used. Moreover, scaling (3.31) is affecting the length of homotopy paths as it can be observed from figure 6.2, and this vindicates the theoretical results and discussions of chapter 3. Finally, from figure 6.2 it is apparent that the increase of $k$ results in a stiff IVP because of the big difference between $|X_c(0)|$ and $|X_c(1)| = |X_c|$, something that was pointed out in chapters 2 and 3. Now, scaling (3.31) removes some but not all the stiffness, as can be seen in figure 6.2. Note that, similar stiff problems were found to be present when the ode15s method was used.

Now, there is a decrease in the values of estimated backward error maximum bounds as $k$ increases (see tables 6.1.1 and 6.1.2). This is in accordance with the theory of chapter 4, since these bound are simply speaking of the same order as the respective forward error bounds amplified by the respective condition numbers. Moreover, as far as it concerns the forward errors, these are of magnitudes of order less than $u_r = 2^{-53}$.

Note that, there is pessimism in the numerical stability of the computed solution via the GQME-Toolbox with respect to the results obtained considering floating point rounding errors. This may be attributed to the fact that because of the diagonal structure of the data, floating
point rounding errors are cancelling each other, since there are several zero number multiplications resulting in exact zero quantities in the floating point operations.

Finally, viewing the GQME-Toolbox algorithm GQMEHOM1.m together with other state-of-the-art algorithms in figures 6.3 and 6.4, it is apparent that although GQMEHOM1.m is a general quadratic equation solver it is a very good competitor when it is specialized to ARE. Note that the MATLAB routines that where used in comparison include tools that recognize the diagonal structure of a problem and hence perform the computations accordingly in such a way that the accuracy is improved. Now, reducing more $e_{tol_{IVP}}$ will generally result in better accuracy for GQMEHOM1.m. More importantly, it will be shown in the next subsection that when the recognition of the diagonal structure of the problem is not feasible, then for ill-conditioned problems GQMEHOM1.m is one of the best competitors for the solution of AREs.

6.3.2 Numerical Example 6.2

This example considers the ARE (6.1) as in example 6.1 but with $A$, $D$ and $C$ replaced by $ZAZ^{-1}$, $ZDZ^T$ and $(Z^{-1})^TCZ^{-1}$ respectively. $Z := I - \frac{2}{||U_3||_2^2}U_3^T U_1 U_3$ and the exact positive definite solution is $(Z^{-1})^T X Z^{-1}$, where $X$ is the same as in example 6.1. In the sequel of this section, by $X$ we will actually mean the exact solution of the present example $(Z^{-1})^T X Z^{-1}$.

The new feature of this example is that the diagonal structure of the ARE data is not present as in example 6.1. The setup of the problem is exactly the same as in example 6.1 and the computation is performed via the driver Example62.m given in appendix D.

The computational results are presented in figure 6.5 and in tables 6.2.1, 6.2.2. In these tables, n.s, s.s stands for a computation without scaling and with sub-optimal scaling (3.63) respectively. The rest of notation is the same as in example 6.1. Note that positive definite solutions could not be computed with the scaling (3.31) due to singularities in the IVP routine as the homotopy variable $\varepsilon \to 1$. As was pointed out in chapter 3, scaling may not always improve things and may have a negative effect. The computation of the sub-optimal scaling problem (3.63) (see chapter 3) for $k = 1, \ldots, 6$ is as follows.

All cases took 0.06 (sec) CPU time and 67091 number of FLOPS.

(i) $k = 1$: $S(u_X,u_C) = 8.5512e + 002, \min\left\{S(u_X,u_C)\right\} = 2.8532e + 003$,
\[ u_X = -1.0745e + 000 U_{3 \times 1}, \quad u_C = 1.4986e + 000 U_{3 \times 1} \].

(ii) \( k = 2 \): \( S(u_X, u_C)_{u_X, u_C} = 1.2668e + 003 \), \( \min \{ S(u_X, u_C) \} = 4.4683e + 003 \),
\[ u_X = -2.0775e + 000 U_{3 \times 1}, \quad u_C = 3.4975e + 000 U_{3 \times 1} \].

(iii) \( k = 3 \): \( S(u_X, u_C)_{u_X, u_C} = 2.0569e + 003 \), \( \min \{ S(u_X, u_C) \} = 7.6190e + 003 \),
\[ u_X = -3.0786e + 000 U_{3 \times 1}, \quad u_C = 5.4979e + 000 U_{3 \times 1} \].

(iv) \( k = 4 \): \( S(u_X, u_C)_{u_X, u_C} = 3.2373e + 003 \), \( \min \{ S(u_X, u_C) \} = 1.2340e + 004 \),
\[ u_X = -4.0787e + 000 U_{3 \times 1}, \quad u_C = 7.4979e + 000 U_{3 \times 1} \].

(v) \( k = 5 \): \( S(u_X, u_C)_{u_X, u_C} = 4.8131e + 003 \), \( \min \{ S(u_X, u_C) \} = 1.8643e + 004 \),
\[ u_X = -5.0787e + 000 U_{3 \times 1}, \quad u_C = 9.4979e + 000 U_{3 \times 1} \].

(vi) \( k = 6 \): \( S(u_X, u_C)_{u_X, u_C} = 4.8131e + 003 \), \( \min \{ S(u_X, u_C) \} = 1.8643e + 004 \),
\[ u_X = -5.0787e + 000 U_{3 \times 1}, \quad u_C = 9.4979e + 000 U_{3 \times 1} \].

As can be seen from the above, no minimum is obtained in all cases for the sub-optimal scaling problem (3.63). This degrades the performance slightly, as logically expected.

In total twelve methods were used for the computation of example 6.2 as in example 6.1 and the results are compared graphically in figures 6.6 and 6.7.

Considering the directionality rules in subsection 6.2.1, it has been found that positive definite solutions for the case without scaling, can be obtained without the positive definite directionality proposition 6.3 (i.e. \( \text{pdef} = 0 \)) if \( e_{tol, IVP} \leq 1e - 012 \). Now, for the case of sub-optimal scaling, positive definite solutions can be obtained without the positive definite directionality proposition 6.3 (i.e. \( \text{pdef} = 0 \)) if \( e_{tol, IVP} \leq 1e - 012 \) for \( k = 1, 2, 3 \) and if \( e_{tol, IVP} \leq 1e - 013 \) for \( k = 4 \). When the positive definite directionality proposition 6.3 is posed (i.e. \( \text{pdef} = 1 \)), it has been found that, positive definite solutions for the case with and without scaling, can be obtained if \( e_{tol, IVP} \leq 1e - 009 \) and \( e_{tol, IVP} \leq 1e - 008 \) respectively.
Figure 6.5. Homotopy paths for example 6.2.
<table>
<thead>
<tr>
<th>$k$</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{det}(J_{X_c})$</td>
<td>n.s.n.p</td>
<td>$-4.9243e + 015$</td>
<td>$-3.2012e + 023$</td>
</tr>
<tr>
<td></td>
<td>ss.n.p</td>
<td>$-4.9243e + 015$</td>
<td>$-3.2012e + 023$</td>
</tr>
<tr>
<td>$|\text{Residual}(X_c)|_F$</td>
<td>n.s.n.p</td>
<td>$8.9233e - 013$</td>
<td>$7.3762e - 013$</td>
</tr>
<tr>
<td></td>
<td>ss.n.p</td>
<td>$1.5569e - 012$</td>
<td>$1.7182e - 012$</td>
</tr>
<tr>
<td>$|X - X_c|_F$</td>
<td>n.s.n.p</td>
<td>$3.5607e - 013$</td>
<td>$3.4488e - 013$</td>
</tr>
<tr>
<td>$C_n(X_c)$</td>
<td>n.s.n.p</td>
<td>$2.9941e + 001$</td>
<td>$2.8179e + 002$</td>
</tr>
<tr>
<td></td>
<td>ss.n.p</td>
<td>$2.9941e + 001$</td>
<td>$2.8179e + 002$</td>
</tr>
<tr>
<td>$E_{prf_{\text{max}}}(X_c)$</td>
<td>n.s.n.p</td>
<td>$4.1912e + 015$</td>
<td>$3.9325e + 016$</td>
</tr>
<tr>
<td>$E_{prf_{\text{llin}}}(X_c)$</td>
<td>n.s.n.p</td>
<td>$6.9830e + 014$</td>
<td>$5.7685e + 013$</td>
</tr>
<tr>
<td></td>
<td>ss.n.p</td>
<td>$7.3800e + 014$</td>
<td>$5.7737e + 013$</td>
</tr>
<tr>
<td>$E_{rb}(X_c)$</td>
<td>n.s.n.p</td>
<td>$3.0519e + 012$</td>
<td>$4.3537e + 012$</td>
</tr>
<tr>
<td></td>
<td>ss.n.p</td>
<td>$5.7069e + 012$</td>
<td>$1.0016e + 011$</td>
</tr>
<tr>
<td>$E_{prb_{\text{max}}}(X_c)$</td>
<td>n.s.n.p</td>
<td>$1.4114e + 011$</td>
<td>$1.8868e + 011$</td>
</tr>
<tr>
<td></td>
<td>ss.n.p</td>
<td>$2.6905e + 011$</td>
<td>$4.2043e + 011$</td>
</tr>
<tr>
<td>$E_{prb_{\text{llin}}}(X_c)$</td>
<td>n.s.n.p</td>
<td>$1.4261e + 010$</td>
<td>$1.6064e + 008$</td>
</tr>
<tr>
<td></td>
<td>ss.n.p</td>
<td>$1.5553e + 010$</td>
<td>$1.6090e + 008$</td>
</tr>
<tr>
<td>$NS$</td>
<td>n.s.n.p</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ss.n.p</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$NS_{fl}$</td>
<td>n.s.n.p</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ss.n.p</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$e_{\text{tol}_{\text{IVP}}}$</td>
<td>n.s.n.p</td>
<td>$1.0000e + 012$</td>
<td>$1.0000e + 012$</td>
</tr>
<tr>
<td></td>
<td>ss.n.p</td>
<td>$1.0000e + 012$</td>
<td>$1.0000e + 012$</td>
</tr>
<tr>
<td>$n_{\text{steps}_{\text{IVP}}}$</td>
<td>n.s.n.p</td>
<td>481</td>
<td>619</td>
</tr>
<tr>
<td></td>
<td>ss.n.p</td>
<td>544</td>
<td>627</td>
</tr>
<tr>
<td>$\text{min}<em>{\text{step}</em>{\text{IVP}}}$</td>
<td>n.s.n.p</td>
<td>$2.063733e - 005$</td>
<td>$2.066334e - 007$</td>
</tr>
<tr>
<td></td>
<td>ss.n.p</td>
<td>$2.666891e - 006$</td>
<td>$9.378135e - 010$</td>
</tr>
<tr>
<td>$t_{\text{CPU}_{\text{IVP}}}(\text{sec})$</td>
<td>n.s.n.p</td>
<td>$2.5980e + 001$</td>
<td>$3.3340e + 001$</td>
</tr>
<tr>
<td></td>
<td>ss.n.p</td>
<td>$2.9710e + 001$</td>
<td>$3.4480e + 001$</td>
</tr>
<tr>
<td>$n_{\text{FLOPS}_{\text{IVP}}}$</td>
<td>n.s.n.p</td>
<td>$72394935$</td>
<td>$92985774$</td>
</tr>
<tr>
<td></td>
<td>ss.n.p</td>
<td>$81804285$</td>
<td>$95457805$</td>
</tr>
</tbody>
</table>

Table 6.2.1. Computation results for example 6.2.
## Table 6.2.2

<table>
<thead>
<tr>
<th>$k$</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\det(JX_c)$</td>
<td>$n.s.n.p$</td>
<td>-3.0583e + 039</td>
<td>-3.0570e + 047</td>
</tr>
<tr>
<td></td>
<td>$ss.n.p$</td>
<td>-3.0583e + 039</td>
<td>-</td>
</tr>
<tr>
<td>$|Residual(X_c)|_F$</td>
<td>$n.s.n.p$</td>
<td>2.6256e - 008</td>
<td>8.9168e - 008</td>
</tr>
<tr>
<td></td>
<td>$ss.n.p$</td>
<td>8.8350e - 012</td>
<td>-</td>
</tr>
<tr>
<td>$|X - X_c|_F$</td>
<td>$n.s.n.p$</td>
<td>5.7805e - 013</td>
<td>1.3503e - 013</td>
</tr>
<tr>
<td></td>
<td>$ss.n.p$</td>
<td>2.5657e - 013</td>
<td>-</td>
</tr>
<tr>
<td>$C_n(X_c)$</td>
<td>$n.s.n.p$</td>
<td>2.8168e + 004</td>
<td>2.8168e + 005</td>
</tr>
<tr>
<td></td>
<td>$ss.n.p$</td>
<td>2.8168e + 004</td>
<td>-</td>
</tr>
<tr>
<td>$E_{\text{prf}_\text{max}}(X_c)$</td>
<td>$n.s.n.p$</td>
<td>1.3080e - 013</td>
<td>4.7031e - 014</td>
</tr>
<tr>
<td></td>
<td>$ss.n.p$</td>
<td>4.3487e - 017</td>
<td>-</td>
</tr>
<tr>
<td>$E_{\text{prf}<em>\text{ll}</em>\text{max}}(X_c)$</td>
<td>$n.s.n.p$</td>
<td>5.6850e - 011</td>
<td>5.6824e - 010</td>
</tr>
<tr>
<td></td>
<td>$ss.n.p$</td>
<td>5.6788e - 011</td>
<td>-</td>
</tr>
<tr>
<td>$E_{\text{rb}}(X_c)$</td>
<td>$n.s.n.p$</td>
<td>1.4674e - 008</td>
<td>1.0223e - 010</td>
</tr>
<tr>
<td></td>
<td>$ss.n.p$</td>
<td>1.1056e - 011</td>
<td>-</td>
</tr>
<tr>
<td>$E_{\text{prb}_\text{max}}(X_c)$</td>
<td>$n.s.n.p$</td>
<td>5.8121e - 007</td>
<td>2.1174e - 006</td>
</tr>
<tr>
<td></td>
<td>$ss.n.p$</td>
<td>2.0276e - 010</td>
<td>-</td>
</tr>
<tr>
<td>$E_{\text{prb}<em>\text{ll}</em>\text{max}}(X_c)$</td>
<td>$n.s.n.p$</td>
<td>1.5381e - 004</td>
<td>1.5365e - 002</td>
</tr>
<tr>
<td></td>
<td>$ss.n.p$</td>
<td>1.5354e - 004</td>
<td>-</td>
</tr>
<tr>
<td>$NS$</td>
<td>$n.s.n.p$</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>$ss.n.p$</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>$NS_{fl}$</td>
<td>$n.s.n.p$</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>$ss.n.p$</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>$\epsilon_{\text{tol}_\text{IVP}}$</td>
<td>$n.s.n.p$</td>
<td>1.0000e - 012</td>
<td>1.0000e - 012</td>
</tr>
<tr>
<td></td>
<td>$ss.n.p$</td>
<td>1.0000e - 013</td>
<td>1.0000e - 016</td>
</tr>
<tr>
<td>$n_{\text{steps}_\text{IVP}}$</td>
<td>$n.s.n.p$</td>
<td>707</td>
<td>729</td>
</tr>
<tr>
<td></td>
<td>$ss.n.p$</td>
<td>937</td>
<td>-</td>
</tr>
<tr>
<td>$\min_{\text{step}_\text{IVP}}$</td>
<td>$n.s.n.p$</td>
<td>2.066937e - 011</td>
<td>2.064668e - 013</td>
</tr>
<tr>
<td></td>
<td>$ss.n.p$</td>
<td>8.67904e - 017</td>
<td>-</td>
</tr>
<tr>
<td>$t_{\text{CPU}_\text{IVP}}$ (sec)</td>
<td>$n.s.n.p$</td>
<td>3.8620e + 001</td>
<td>3.9740e + 001</td>
</tr>
<tr>
<td></td>
<td>$ss.n.p$</td>
<td>5.3070e + 001</td>
<td>-</td>
</tr>
<tr>
<td>$n_{\text{FLOPS}_\text{IVP}}$</td>
<td>$n.s.n.p$</td>
<td>108114428</td>
<td>111995390</td>
</tr>
<tr>
<td></td>
<td>$ss.n.p$</td>
<td>143812804</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 6.2.2. Computation results for example 6.2.
Figure 6.6. Comparison graphs for example 6.2.
$\log_{10}(E_{rb}(X_c))$

$k$

$\log_{10}(E_{prob_{max}}(X_c))$

$k$

$\log_{10}(E_{prob_{max}}(X_c))$

$k$

Figure 6.7. Comparison graphs for example 6.2.
Hence the results are similar to example 6.1. The fundamental difference is that for 
k = 5, 6 and $e_{\text{tol}}_{IVP} = 10^{-016}$ the GQMEHOM1.m was unable to obtain positive definite
solutions and only indefinite solutions were obtained with the occurrence of a jump from a
positive definite homotopy path to an indefinite homotopy path. This case may attributed to the
hard stiffness of the problem, so the IVP routine RK45GQMEHOM1.m can not be accurate
enough. One solution, for example, might be the usage of PI step control methods [76] within
the Runge-Kutta based algorithm RK45GQMEHOM1.m. This will constitute a problem of
future research and it is out of the scope of the present thesis.

From tables 6.2.1 and 6.2.2, the main conclusion is that the GQME-Toolbox always
results in a numerical stable algorithm for the numerical solution of a fairly difficult numerical
problem. Since the exact solutions are known, it is apparent from tables 6.2.1, 6.2.2 that the
error estimates are by no means pessimistic. The backward error estimates are very close to the
exact backward errors with respect to the unit roundoff accuracy $u_p = 2^{-53}$, and the forward
error estimates are less than the backward error estimates.

From the error estimation viewpoint, improvements can be observed when sub-optimal
scaling (3.63) is used in the computations but not in all cases. Similar observations hold for the
number of steps of the IVP routine and hence in the computation times when sub-optimal
scaling is used. Moreover, sub-optimal scaling (3.63) is affecting the length of homotopy paths
(enlargement for $k = 1, \ldots, 4$ and reduction for $k = 5, 6$), as can be observed from figure 6.5, and
this vindicates the theoretical results and discussions of chapter 3. Finally, as in example 6.1,
from figure 6.2 it is apparent that the increase of $k$ results in a stiff IVP because of the big
difference between $|X_c(0)|$ and $|X_c(1)|$. Now, the sub-optimal scaling (3.63) adds more
stiffness in cases $k = 1, \ldots, 4$, and removes some stiffness in the cases $k = 5, 6$, as can be seen in
figure 6.5. Note that stiffness was found when the ode15s method was used.

As in example 6.1, there is a decrease to the values of estimated backward error
maximum bounds as $k$ increases (see tables 6.2.1 and 6.2.2) something that is in accordance
with the theory of chapter 4. In contrast to example 6.1 there is no pessimism in the numerical
stability of the computed solution via the GQME-Toolbox with respect to the results obtained
considering floating point rounding errors. This may be attributed to the fact that the diagonal
structure in the data is absent in example 6.2 and no floating point rounding error cancellations
occur.

Now, viewing the GQMEHOM1.m together with other state-of-the-art algorithms in
figures 6.6 and 6.7, it is apparent that GQMEHOM1.m is one of the best and in some cases the
best, when specialized to the ARE.
In figure 6.8, the conditioning of both examples 6.1 and 6.2, for all the twelve methods that were used is shown. From figure 6.8, the linear relation of the condition number of the computed solution with $k$ is apparent.
6.3.3 Numerical Example 6.3

Consider the ARE in [19] (6.1) with data 
\[ A = \begin{bmatrix} 0 & 1 & 0 & 1 \\ -1 & 0 & -1 & 0 \\ 0 & 1 & 0 & 1 \\ -1 & 0 & -1 & 0 \end{bmatrix}, \quad D = U_{4 \times 4}, \quad C = U_{4 \times 4}. \]

The special feature of this example is that since \((A,D)\) is not controllable, most state-of-the-art methods can not be applied for the computation of solutions. To be more precise, all MATLAB special routines presented in examples 6.1 and 6.2 including the sign function based routines sfcare and sfcares can not be used, they fail by construction for uncontrollable pairs \((A,D)\). On the contrast, the ode 15s method and the gradient method with and without exact line search of [19] can be applied. In addition to these last two methods, the GQME-Toolbox successfully computes the solutions.

There are several ways to write the above ARE in the form of (2.1) with \(\omega = \xi = 2\), as was shown in example 6.1. In order to avoid pessimism in the posteriori numerical analysis of the computed solutions, due to repeated matrices, the formulation that is used is as in example 6.1. \(A_1 : = 0.1A^T, \quad B_1 : = 10I_4, \quad A_2 : = 5I_4, \quad B_2 : = 0.2A, \quad C_1 : = 0.4I_4, \quad D_1 : = 2D, \quad E_1 : = 0.5I_4, \quad C_2 : = 0.6I_4, \quad D_2 : = 0.5D, \quad E_2 : = 2I_4, \quad G : = C. \) All the general results that have been developed in section 6.2, hold regardless of the way that (6.1) is written as (2.1).

Example 6.3, happens to possess a manifold of positive definite solutions. Four such solutions are computed via the fixed point homotopy algorithm of GQME-Toolbox with \(\gamma = 1\). The initial conditions for the two computed solutions were such that the easy problem solutions were \(X(0) : = X_a = I_4\) and \(X(0) : = X_b = 1e^{-0.15I_4}. \) While the other two solutions were obtained with the same initial conditions as before and with the usage of scaling (3.31). Note that, since there is not a unique positive definite solution but there are infinite positive definite solutions, the scaling method will generally lead to a neighbouring solution of the respective solution that is obtained without the usage of scaling. This is due to rounding errors. Homogeneous projective transformations are not considered in the present example.

The notation used is the same as in the example 6.1 before. In all cases a posteriori numerical analysis of the computed solutions has been conducted. The driver to run example 6.3 in GQME-Toolbox is the m-file Example63.m and is given in appendix D.

The arithmetic used in the present example is the MATLAB fixed complex arithmetic (i.e. flag = 0), but effectively computations are in real arithmetic because the homotopy equation is defined in real Euclidean vector space). Now, no homotopy step corrector was posed (i.e. flagcor = 1) and the ‘end game’ tolerance was set to efinal = 1e−20 with the jacobian formula provided in the ‘end game’ solver under a large scale optimization method.
The maximum allowed number of steps and the minimum allowed integration step of the IVP routine RK45GQMEHOM1.m were set to \texttt{maxnsteps=realmax} (i.e. the MATLAB real maximum number) and \texttt{minstep=1e-60} respectively. Since no discrete homotopy in the data was considered, \texttt{step=1}. Finally, no directionality rules were posed (i.e. \texttt{pdef = 0}).

The computational results are presented in figure 6.9 and in tables 6.3.1, 6.3.2. In these tables, \texttt{n.p} stands for a computation with no homogenous projective transformation and \texttt{n.s}, \texttt{s} stands for a computation with no scaling and scaling (3.31) respectively.

The computation of the scaling problem (3.31) (see chapter 3) took 0.13 (sec) CPU time and 674811 number of FLOPS. It has been found that
\[
S(u_X, u_C)(u_X, u_C) = 5.8366e+003, \quad \min_{u_X, u_C} S(u_X, u_C) = 5.8403e+003,
\]
\[
u_X = 1.3510e-002U_{4 \times 1}, \quad u_C = -1.2716e-002U_{4 \times 1}.
\]

Hence, no minimum obtained for the scaling problem. This is expected since all the elements of the matrix data of example 6.3 are ±1, and the equation is already perfectly scaled. Because of this the scaling will be expected to damage generally the overall performance of the computations and this is indeed what is happening.

The computed solutions obtained with respect to \texttt{e_{tol}^{l_{IVP}} = 1e-012} are as follows.

(i) \(X(0) = X_a\) and \texttt{n.s.n.p}:

\[
X_c = \begin{bmatrix}
1.6250e+000 & 2.6212e-012 & -3.7500e-001 & 3.0085e-012 \\
3.0085e-012 & 1.6250e+000 & 3.0085e-012 & -3.7500e-001 \\
-3.7500e-001 & 3.0085e-012 & 1.6250e+000 & 3.0085e-012 \\
2.6212e-012 & -3.7500e-001 & 3.0085e-012 & 1.6250e+000 \\
\end{bmatrix}
\]

(ii) \(X(0) = X_a\) and \texttt{s.n.p}:

\[
X_c = \begin{bmatrix}
1.6566e+000 & 2.8101e-012 & -4.0660e-001 & 2.9604e-012 \\
2.9604e-012 & 1.6566e+000 & 2.8101e-012 & -4.0660e-001 \\
-4.0660e-001 & 2.9604e-012 & 1.6566e+000 & 2.8101e-012 \\
2.8101e-012 & -4.0660e-001 & 2.9604e-012 & 1.6566e+000 \\
\end{bmatrix}
\]

(iii) \(X(0) = X_b\) and \texttt{n.s.n.p}:

\[
X_c = \begin{bmatrix}
\end{bmatrix}
\]
(iv) $X(0) = X_b$ and s.n.p:

$$X_c = \begin{bmatrix}
6.2500e-001 & 1.4268e-012 & 6.2500e-001 & 1.3849e-012 \\
1.3850e-012 & 6.2500e-001 & 1.4266e-012 & 6.2500e-001 \\
6.2500e-001 & 1.3849e-012 & 6.2500e-001 & 1.4265e-012 \\
1.4266e-012 & 6.2500e-001 & 1.3849e-012 & 6.2500e-001
\end{bmatrix}.$$
Table 6.3.1. Computation results for example 6.3.

<table>
<thead>
<tr>
<th></th>
<th>( X(0) )</th>
<th>( X_a )</th>
<th>( X_b )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{det}(J_{X_c}) )</td>
<td>\text{n.s.n.p}</td>
<td>2.7522 e - 027</td>
<td>-4.2308 e - 043</td>
</tr>
<tr>
<td></td>
<td>\text{s.n.p}</td>
<td>2.4700 e - 028</td>
<td>2.0465 e - 027</td>
</tr>
<tr>
<td>( |\text{Residual}(X_c)|_F )</td>
<td>\text{n.s.n.p}</td>
<td>1.9405 e - 012</td>
<td>1.7622 e - 012</td>
</tr>
<tr>
<td></td>
<td>\text{s.n.p}</td>
<td>1.9447 e - 012</td>
<td>1.8280 e - 012</td>
</tr>
<tr>
<td>( C_n(X_c) )</td>
<td>\text{n.s.n.p}</td>
<td>2.4758 e + 019</td>
<td>7.7090 e + 016</td>
</tr>
<tr>
<td></td>
<td>\text{s.n.p}</td>
<td>3.3651 e + 018</td>
<td>2.2343 e + 018</td>
</tr>
<tr>
<td>( E_{prf_{\text{max}}}(X_c) )</td>
<td>\text{n.s.n.p}</td>
<td>1.1366 e - 013</td>
<td>9.4830 e - 014</td>
</tr>
<tr>
<td></td>
<td>\text{s.n.p}</td>
<td>1.1293 e - 013</td>
<td>9.0815 e - 014</td>
</tr>
<tr>
<td>( E_{prf_{\text{if max}}}(X_c) )</td>
<td>\text{n.s.n.p}</td>
<td>1.2183 e - 013</td>
<td>8.7595 e - 014</td>
</tr>
<tr>
<td></td>
<td>\text{s.n.p}</td>
<td>1.2120 e - 013</td>
<td>9.8094 e - 014</td>
</tr>
<tr>
<td>( E_{rb}(X_c) )</td>
<td>\text{n.s.n.p}</td>
<td>2.2033 e - 013</td>
<td>1.0606 e - 013</td>
</tr>
<tr>
<td></td>
<td>\text{s.n.p}</td>
<td>2.2584 e - 013</td>
<td>1.1002 e - 013</td>
</tr>
<tr>
<td>( E_{prb_{\text{max}}}(X_c) )</td>
<td>\text{n.s.n.p}</td>
<td>9.9797 e - 013</td>
<td>2.5623 e - 011</td>
</tr>
<tr>
<td></td>
<td>\text{s.n.p}</td>
<td>1.0254 e - 012</td>
<td>2.6580 e - 011</td>
</tr>
<tr>
<td>( E_{prb_{\text{if max}}}(X_c) )</td>
<td>\text{n.s.n.p}</td>
<td>1.0350 e - 012</td>
<td>2.6722 e - 011</td>
</tr>
<tr>
<td></td>
<td>\text{s.n.p}</td>
<td>1.0641 e - 012</td>
<td>2.7679 e - 011</td>
</tr>
<tr>
<td>( NS )</td>
<td>\text{n.s.n.p}</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>\text{s.n.p}</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( NS_{\beta} )</td>
<td>\text{n.s.n.p}</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>\text{s.n.p}</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( \epsilon_{\text{tol}_{IVP}} )</td>
<td>\text{n.s.n.p}</td>
<td>1.0000 e - 012</td>
<td>1.0000 e - 012</td>
</tr>
<tr>
<td></td>
<td>\text{s.n.p}</td>
<td>1.0000 e - 012</td>
<td>1.0000 e - 012</td>
</tr>
<tr>
<td>( n_{\text{steps}_{IVP}} )</td>
<td>\text{n.s.n.p}</td>
<td>116</td>
<td>160</td>
</tr>
<tr>
<td></td>
<td>\text{s.n.p}</td>
<td>119</td>
<td>159</td>
</tr>
<tr>
<td>( \text{min}<em>{\text{step}</em>{IVP}} )</td>
<td>\text{n.s.n.p}</td>
<td>1.743570 e - 003</td>
<td>3.753845 e - 004</td>
</tr>
<tr>
<td></td>
<td>\text{s.n.p}</td>
<td>1.659816 e - 003</td>
<td>2.224304 e - 003</td>
</tr>
<tr>
<td>( CPU_{IVP} (sec) )</td>
<td>\text{n.s.n.p}</td>
<td>7.6700 e + 000</td>
<td>1.0160 e + 001</td>
</tr>
<tr>
<td></td>
<td>\text{s.n.p}</td>
<td>8.0600 e + 000</td>
<td>1.0760 e + 001</td>
</tr>
<tr>
<td>( n_{\text{FLOPS}_{IVP}} )</td>
<td>\text{n.s.n.p}</td>
<td>75086857</td>
<td>103808558</td>
</tr>
<tr>
<td></td>
<td>\text{s.n.p}</td>
<td>76975258</td>
<td>103179879</td>
</tr>
</tbody>
</table>
Table 6.3.2: Computation results for example 6.3.
From figure 6.9 and the tables 6.3.1, 6.3.2, it is obvious that GQMEHOM1.m constitutes a numerically stable method. The computed solutions can be characterized as realistic and moreover as singular, subject to \( \text{det}(JX_e) \) in tables 6.3.1 and 6.3.2. There is a conflict between the accuracy and computational times. Finally, as was expected, scaling (3.31) degrade the overall computation performance.

### 6.3.4 Numerical Example 6.4

This example considers (6.6) with data,

\[
A = \begin{bmatrix}
A_{11} & A_{12} \\
A_{21} & A_{22}
\end{bmatrix}, \quad D = \begin{bmatrix}
D_{11} & D_{12} \\
D_{21} & D_{22}
\end{bmatrix}, \quad C = \begin{bmatrix}
0_{4 \times 4} & 0_{4 \times 4} \\
0_{4 \times 4} & C_{22}
\end{bmatrix}, \quad \tau = \begin{bmatrix}
\tau_{11} & \tau_{12} \\
\tau_{21} & \tau_{22}
\end{bmatrix}.
\]

where

\[
A_{11} = \begin{bmatrix}
-1.6100 \times 10^{-1} & 1 & 0 & 0 \\
-6.0040 & 0 & 0 & 0 \\
-5.8822 & 0 & 0 & 0 \\
-9.9835 & 0 & 0 & 0
\end{bmatrix}, \quad A_{12} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0
\end{bmatrix}.
\]

\[
A_{21} = \begin{bmatrix}
-4.0730 \times 10^{-1} & 0 & 0 & 0 \\
-3.9820 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}, \quad A_{22} = \begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0
\end{bmatrix}.
\]

\[
D_{11} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}, \quad D_{12} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}.
\]

\[
D_{21} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}, \quad D_{22} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}.
\]

\[
C_{22} = \begin{bmatrix}
0.4096 \times 10^{-5} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0.4563 \times 10^{-4} & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}.
\]

\[
7.1314 \times 10^{-2} & 7.4509 \times 10^{-3} & 7.0979 \times 10^{-2} \\
\end{bmatrix}.
\]
Apart from $\tau$, the rest of the data are similar to those in [21]. Furthermore, $\text{rank}(\tau) = 5$ and $\tau$ is not actually an idempotent matrix and hence (6.6) cannot be characterized as an OPE. OPEs will be separately considered in chapter 8. Before presenting the computation results using the GQME-Toolbox, it is worthwhile mentioning the weakness of other methods for computing positive definite solutions for the present example.

For the example 6.4, it is clear that MATLAB AREs solvers and the scfarc and scfares methods are by nature infeasible for the numerical solution of (6.6).

The ode15s method is also incapable of obtaining a positive definite solution. The reason for this is that since the method considers the numerical solution of the differential system

\[
\frac{d(\text{vec}(X(t)))}{dt} = \text{vec}\left( A^T X(t) + X(t)A - X(t)DX(t) + \tau^T X(t)DX(t)\tau + C \right),
\]

\[X(t_o) \in \mathbb{R}^{n \times n}, \quad t \in [t_o, +\infty)\]

($n = 8$ in the present case), convergence to an equilibrium point corresponding to a positive definite solution can be feasible only if this equilibrium is an attractor point. Now, such an attractor is defined if all eigenvalues of

\[
\frac{d}{d(t)} \left( \text{vec}(X) \right)^T \left( \frac{d(\text{vec}(X))}{dt} \right) X = X(\infty)
\]

have negative real parts. Note that, $X(\infty)$ is actually the desired positive definite solution and that
where $J_{vec(X)}^{T}$ is the jacobian of (6.6) with respect to $vec(X)^{T}$.

Unfortunately, in the present case $\frac{d}{d(vec(X))^T} \left( \begin{array}{c} d(vec(X)) \\ dt \end{array} \right) \bigg|_{X=X(\infty)}$ does not fulfill the characteristic of having all its eigenvalues with negative real part. The reader can vindicate that by computing the eigenvalues of $\frac{d}{d(vec(X))^T} \left( \begin{array}{c} d(vec(X)) \\ dt \end{array} \right) \bigg|_{X=X(\infty)}$ with respect to the computed solution (also denoted as $X(\infty)$) given in the sequel of this section. Subject to the computed positive definite solution in the sequel, $\frac{d}{d(vec(X))^T} \left( \begin{array}{c} d(vec(X)) \\ dt \end{array} \right) \bigg|_{X=X(\infty)}$ has two pairs of complex conjugate eigenvalues with positive real part; namely $3.4941e \cdot 001 \pm 1.4120e \cdot 001i$ and $2.0112e \cdot 002 \pm 3.4588e \cdot 001i$.

Hence, whatever the initial condition $X(t_0)$, the trajectory of the above differential system will not rest to the equilibrium $X(\infty)$ because it is not stable. More about the ode15s method and its features will be mentioned in the next section, since the analysis of the method is related to deep theoretical results about the geometry of solutions.

Apart from ode15s method, the Davidon gradient flow method, [112] adapted for the case, implemented via the global Newton algorithm $fsolve.m$ of MATLAB, was also used for the computation of a positive definite solution. This method was capable of computing a positive definite solution subject to positive definite initial conditions, but with large errors since the optimization process has been trapped in local minimum points, indicating the non convexity of the problem subject to the norm of the residual of (6.6).

From the above it is clear that, a wide variety of methods can not computed solutions of a specific kind. Hence the advantage, the superiority and the usefulness of the GQME-Toolbox and more specifically of GQMEHOM1.m.

There are several ways to write the above equation in the form of (2.1) with $\omega = \xi = 2$. The most apparent is one is by setting $A_1 := A^T$, $B_2 := A$, $A_2 := B_1 := C_1 := E_1 := I_3$, $D_1 := D$, $C_2 := \tau^T$, $D_2 := -D$, $E_2 := \tau$, $G := C$. In order to avoid pessimism in the posteriori numerical analysis of the computed solutions, due to repeated matrices, the formulation that is
Chapter 6

Generalized Algebraic Riccati Equations

241

used is, \( A_1 := 0.4 A^T \), \( B_1 := 2.5 I_8 \), \( A_2 := 5 I_8 \), \( B_2 := 0.2 A \), \( C_1 := 0.2 I_8 \), \( D_1 := D \), \( E_1 := 5 I_8 \), \( C_2 := \tau^T \), \( D_2 := 0.2 D \), \( E_2 := 5 \tau \), \( G := C \).

The setup of the example 6.4 in terms of the GQME-Toolbox is implemented via the driver Example64.m, given in appendix D. For this setup, a positive definite solution is computed via the fixed point homotopy algorithm of GQME-Toolbox with \( \gamma = 1 \). Note that, the existence of more than one positive definite solution is not known since theoretical results for the existence and the geometry of solutions for such equations are not available with respect to the author’s knowledge. The initial condition was such that the easy problem solution was \( X(0) = 10^{-16} I_8 \). Scaling and Homogeneous projective transformations were not considered.

The notation used is the same as in the previous examples of this chapter. A posteriori numerical analysis of the computed solution have been conducted.

The arithmetic used in the present example is the MATLAB fixed complex arithmetic (i.e. flh = 0), but effectively computations are in real arithmetic because the homotopy equation is defined in real Euclidean vector space. No homotopy step corrector was posed (i.e. flagcor = 1) and the ‘end game’ tolerance was set to efinal = 1e-20 with the jacobian formula provided in the ‘end game’ solver under a large scale optimization method largescale = ‘on’. The maximum allowed number of steps and the minimum allowed integration step of the IVP routine RK45GQMEHOM1.m were set to maxnsteps = realmax and minstep = realmin (i.e. the MATLAB real maximum and real minimum number respectively) respectively. No discrete homotopy in the data was considered (i.e. step = 1) and no directionality rules posed (i.e. pdef = 0). The computation was implemented twice subject to etoljvp = 1e-012 and etoljvp = 1e-006 respectively.

The computational results are presented in figure 6.10 and in table 6.4. From table 6.4, the robustness and the numerical stability of the algorithm is apparent. Also, there is an indication that accuracy is sacrificed for speed and vice versa.

6.4 Synopsis

In this chapter the solution of the specialization of (2.1) into ARE-type symmetric matrix equations was discussed and implemented with four numerical examples. Conclusions and discussions about the numerical examples took place in the relevant sections and they will not repeated here. The computations have revealed the efficiency of the GQME-Toolbox in computing solutions of a specific kind.
Figure 6.10. Homotopy paths for example 6.4.

| TABLE 6.4 |
|------------------|------------------|------------------|
| \( \det(J_{X_c}) \) | 1.3877e+01 | 1.3877e+01 |
| \( \|Residual(X_c)\|_F \) | 2.1881e-13 | 3.4988e-08 |
| \( \|X_c\|_F \) | 2.1881e-13 | 3.4988e-08 |
| \( \text{C}_n(X_c) \) | 1.0766e+04 | 1.0766e+04 |
| \( E_{\text{prf}_{\text{max}}}(X_c) \) | 3.4956e-015 | 5.7402e-010 |
| \( E_{\text{prf}_{\text{max}}}(X_c) \) | 1.8846e-014 | 5.7403e-010 |
| \( E_{\text{rb}}(X_c) \) | 2.0202e-012 | 9.4233e-007 |
| \( E_{\text{prb}_{\text{max}}}(X_c) \) | 3.7160e-010 | 6.9177e-005 |
| \( E_{\text{prb}_{\text{max}}}(X_c) \) | 1.1364e-009 | 6.9177e-005 |
| \( \text{NS} \) | 1 | 1 |
| \( \text{NS flat} \) | 1 | 1 |
| \( \epsilon_{\text{tol}_{\text{IVP}}} \) | 1.0000e-012 | 1.0000e-006 |
| \( n_{\text{steps}_{\text{IVP}}} \) | 352 | 36 |
| \( \text{minstep}_{\text{IVP}} \) | 2.9717e-005 | 5.1891e-004 |
| \( t_{\text{CPU}_{\text{IVP}}}(\text{sec}) \) | 1.6066e+002 | 2.0560e+001 |
| \( n_{\text{FLOPS}_{\text{IVP}}} \) | 15617235159 | 1983464311 |

Table 6.4. Computation results for example 6.4.
The numerical examples of this chapter dealt only with the computation of positive
definite solutions. Other kinds of solution can also be computed with appropriate setups
according to the theory presented in this chapter. The reader can perform experiments using the
GQME-Toolbox with different setups in the drivers provided in appendix D.

Noted that, apart from the numerical examples of this chapter other examples have also
been successfully tested with the GQME-Toolbox.

At this point it should be said that, since the GQME-Toolbox is a general algebraic
quadratic matrix equation solver, the jacobian and derivative computations of symmetric ARE-
type equations are done with respect to the general formulations of chapters 2 and 3.

When dealing with ARE-type symmetric equations, jacobian and derivative
computations can be done more economically, considering only the symmetric parts of the
respective equations. In addition, for the AREs the jacobians with respect to the equation’s
variable can be computed as solutions to symmetric ALEs or ASEs. There are state-of-the-art
algorithms that proving the economy in the computations regarding the total size dimensions of
the problem [58], [59], [34]. For example, for the fixed point homotopy equation subject to
(6.1), the vectorization of the Frechet derivative (see appendix B) of this equation defines

\[
\frac{d\left(\text{vec}(X(\varepsilon))\right)}{d\varepsilon} \text{ as the solution of an ALE. This is shown below.}
\]

\[
d(X(\varepsilon),\varepsilon)\left[H(X(\varepsilon),\varepsilon)\right] = \varepsilon(A^T d(X(\varepsilon)) + d(X(\varepsilon))A - d(X(\varepsilon))D X(\varepsilon) - X(\varepsilon)Dd(X(\varepsilon)))
\]

\[
-\gamma(1 - \varepsilon)d(X(\varepsilon)) + d\varepsilon\left(A^T X(\varepsilon) + X(\varepsilon)A - X(\varepsilon)DX(\varepsilon) - C + \gamma X(\varepsilon) - \gamma X(0)\right) = 0
\]

\[
\Leftrightarrow \left(\varepsilon A - \frac{1}{2} \varepsilon^2 (1 - \varepsilon)I_n - \varepsilon D X(\varepsilon)\right)^T \frac{d(X(\varepsilon))}{d\varepsilon} + \frac{d(X(\varepsilon))}{d\varepsilon} \left(\varepsilon A - \frac{1}{2} \varepsilon^2 (1 - \varepsilon)I_n - \varepsilon D X(\varepsilon)\right)
\]

\[
+ \left(A^TX(\varepsilon) + X(\varepsilon)A - X(\varepsilon)DX(\varepsilon) - C + \gamma X(\varepsilon) - \gamma X(0)\right) = 0,
\]

where \(d(X(\varepsilon),\varepsilon)\left[H(X(\varepsilon),\varepsilon)\right] \in \mathbb{R}^{n \times n}\) denotes the Frechet derivative of \(H(X(\varepsilon),\varepsilon)\) with
respect to \(X(\varepsilon) \in \mathbb{R}^{n \times n}\) and \(\varepsilon \in \mathbb{U}\).

Now, \(\frac{d(X(\varepsilon))}{d\varepsilon}\) can be computed as the solution to the last algebraic matrix equation
above, which is an ALE with respect to \(\frac{d(X(\varepsilon))}{d\varepsilon}\). Finally,

\[
\frac{d\left(\text{vec}(X(\varepsilon))\right)}{d\varepsilon} = \text{vec}\left(\frac{d(X(\varepsilon))}{d\varepsilon}\right).
\]

Hence, when specialized to symmetric AREs computation times can be shorted by
computing the derivatives as ALEs. Note however, that for stiff problems like those in examples
6.1 and 6.2 the ALE and ASE solvers [58], [59], [34] can sometimes fail due to ill-conditioning.
This is what can be observed for example 6.1, when using the ALE MATLAB solver lyap.m.
So far the GQME-Toolbox software code has not been specialized, for the symmetric AREs-type, by computing derivatives via ALE or ASE solvers. This is a subject for future research and it is out of the scope of this thesis.

The homotopy algorithms by their nature view every algebraic problem as the dynamical behaviour of a dynamical system over a finite horizon. A similar, but not the same, philosophy is applied to the ode15s method which was used previously in this chapter. In this latter case, a generalized ARE is viewed as an infinite horizon problem (see [103], [141], [23], [151], [152]) as discussed in example 6.4. In this case according to [151] and [152] the solutions are traced through the trajectory of the differential system

\[
\frac{d(\text{vec}(X(t)))}{dt} = \text{vec}(A^TX(t) + X(t)A - X(t)DX(t) + \tau^TX(t)DX(t)\tau + C),
\]

\[X(t_o) \in \mathfrak{S}^{n \times n}, \quad t \in [t_o, +\infty],\]

on a manifold defined in $\mathfrak{S}^{2n \times 1}$ Euclidean space.

Note that, for the case of symmetric AREs where $\tau = 0$, the trajectory of the above system converges to a desired positive definite solution as $t \to \infty$ if $(A,D)$ is controllable, otherwise convergence can not be sufficiently guaranteed (see [103], [141], [23], [151], [152]). Now in practice, instead of performing an infinite time integration (which is impossible), $[t_o, +\infty]$ is replaced by $[t_o, t_f]$, where $t_f > 0$ has sufficiently large distance from $t_o$. Note that, $t_f$ need not necessarily be a large number [103].

As was reported in chapter 2, homotopy algorithms can also perform as infinite horizon problems by replacing $\varepsilon$ with $1 - e^{-t}$, where $t \in [0, +\infty]$ is the new homotopy variable. The advantage of the homotopy method in the last case over the ode15s-type method is that the trajectory on the manifold defined in $\mathfrak{S}^{2n \times 1}$ Euclidean space almost always passes through the desired solution when this exists, regardless of the controllability of $(A,D)$.

More practical case studies of generalized AREs, considering robust control design problems, will be given in chapter 8.
Chapter 7

Numerical Solution & Analysis to Scientific Problems

7.1 Introduction

In this chapter, important scientific problems are formulated and solved as special cases of (2.1). These problems are from the area of finance, applied mathematics/mechanics, chemical engineering and chaotic dynamical systems.

Each problem is presented, via a numerical example from the literature, in a separate section. More specifically, section 7.2 presents the numerical solution and analysis to the computation of feedback Nash equilibria in scalar infinite horizon LQ-games [51], [15]. Section 7.3 deals with the numerical solution of the second order polynomial algebraic matrix equation [39], [43], [42], and the solution of the quadratic eigenvalue problem [79], [80], which can be used in measuring mechanical vibrations. In section 7.4, an example from chemical kinetics [121] is solved. Next, section 7.5 presents the computation of the equilibrium points of the famous Lorenz attractor reported in [161] and its relevant references. Every section contains relevant discussions and general conclusions are given in the last section 7.6.

The MATLAB driver m-files for all example are given in appendix D. The computations were implemented in a two parallel PIII 550 MHz personal computer using MATLAB v. 5.3.1.29215a (R11.1) (Unix version) with unit roundoff $u_r = 2^{-53}$ under the operating system RedHat Linux v. 6.2.
Chapter 7  Numerical Solution & Analysis to Scientific Problems  246

7.2 Feedback Nash equilibria in scalar infinite horizon LQ-games

7.2.1 Problem Formulation and Solution

Consider two players with each one of them controlling a different set of inputs to the first order scalar dynamical system

\[
\frac{dx(t)}{dt} = ax(t) + b_1 u_1(t) + b_2 u_2(t), \quad x(0), \quad t \in \mathbb{R}_+.
\]  

(7.1)

where \( x \) is the state of the system, \( u_i \) is the control variable of the \( i = 1, 2 \) player, and \( a, b_1, b_2 \) are constant parameters.

The problem for each player is to try to minimize his individual quadratic performance criterion

\[
J_i(u_1, u_2) = \frac{1}{2} \int_0^\infty \left( q_i x^2(t) + \eta_i u_i^2(t) \right) dt.
\]

where \( q_i, \eta_i \in \mathbb{R}_+, \quad b_i \in \mathbb{R}^+ \), \( i = 1, 2 \) [51].

The above problem is often met in finance and system theory and its solution is given via proposition (6.8) of [15] and it is associated with the solution with respect to \( x_1, x_2 \), of the following set of algebraic scalar equations (7.2) and (7.3).

\[
-2ax_1 - q_1 + s_1 x_1^2 + 2s_2 x_1 x_2 = 0
\]

\[
-2ax_2 - q_2 + s_2 x_2^2 + 2s_1 x_1 x_2 = 0, \quad (7.2)
\]

\[
2 - 1
\]

where \( s_i = b_i^2 \eta_i^{-1}, \quad i = 1, 2 \). Note that (7.2) and (7.3) above can be considered as scalar coupled AREs.

According to proposition (6.8) of [15] and as reported in [51] the minimizing control strategy of each player can be determined from theorem 7.1 below.

**Theorem 7.1** [51]:

Let \( \bar{x}_i \in \mathbb{R}_+, \quad i = 1, 2 \) solve (7.2) and (7.3). Then the pair of strategies \( u_i(t) = -b_i \eta_i^{-1} \bar{x}_i x(t), \quad t \in \mathbb{R}_+ \cup \{0\}, \quad i = 1, 2 \), provide a feedback Nash equilibrium, leading to the minimum cost \( J_i(u_1, u_2) = \bar{x}_i x^2(0) \) for the \( i \) player. Moreover, the resulting system dynamics described by \( \frac{dx(t)}{dt} = a_{cl} x(t), \quad x(0), \quad t \in \mathbb{R}_+ \cup \{0\}, \) where \( a_{cl} := a - s_1 \bar{x}_1 - s_2 \bar{x}_2 \) is asymptotically stable (i.e. \( a_{cl} < 0 \)).
Proof.
For the proof, see [15].

In addition to theorem 7.1, results about the number and the kind of solutions are derived for the above two-player game in [51].

Now, equations (7.2) and (7.3) can be written as a non-square algebraic quadratic matrix equation (2.1) with dimensions $n = 2$, $p = 1$, and with $\omega = 1$, $\xi = 2$, where

$$
A_1 := \begin{bmatrix} -2a & 0 \\ 0 & -2a \end{bmatrix}, \quad B_1 = E_1 = E_2 = 1, \quad C_1 := \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad C_2 := \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad D_1 := [s_1 \ s_2], \quad B_2 = a,
$$

$$
D_2 := [2s_1 \ s_2], \quad G := \begin{bmatrix} -q_1 \\ -q_2 \end{bmatrix}. \quad \text{In order to reduce pessimism as much as possible, when performing the numerical analysis of the computed solutions (see chapters 4 and 6), (7.2) and (7.3) can be equivalently considered as (2.1) with dimensions } n = 2, \quad p = 1, \quad \text{and with } \omega = 2, \quad \xi = 2, \quad \text{where}
$$

$$
A_1 := \begin{bmatrix} -a & 0 \\ 0 & -a \end{bmatrix}, \quad A_2 := \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}, \quad B_1 = 1, \quad B_2 = a, \quad C_1 := \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad C_2 := \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix},
$$

$$
C_2 := \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad D_1 := [s_1 \ 2s_2], \quad D_2 := [2s_1 \ s_2], \quad E_1 := 5, \quad E_2 := 2, \quad G := \begin{bmatrix} -q_1 \\ -q_2 \end{bmatrix}.
$$

Hence, the GQME-Toolbox can be used in order to obtain all or a subset of solutions of (7.2) and (7.3). Note that according to the theory in chapter 2, (7.2) and (7.3) can have at least $2^2 = 4$ solutions, since they are equivalent to the solution of (2.1) with dimension $2 \times 1$ (i.e. $n = 2, \quad p = 1$)

### 7.2.2 Numerical Example 7.1

In this subsection the numerical example in [51] with data $a = 2, \quad s_1 = 1, \quad s_2 = 2, \quad q_1 = 1, \quad q_2 = \frac{1}{4}$, is considered. Hence, the coefficients of (2.1) are:

$$
A_1 := \begin{bmatrix} -2 & 0 \\ 0 & -2 \end{bmatrix}, \quad A_2 := \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}, \quad B_1 = 1, \quad B_2 = 2, \quad C_1 := \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad C_2 := \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix},
$$

$$
C_2 := \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad D_1 := [1 \ 4], \quad D_2 := [2 \ 2], \quad E_1 := 5, \quad E_2 := 2, \quad G := \begin{bmatrix} -1 \\ -2 \end{bmatrix}, \quad H := \begin{bmatrix} -1 \\ -2 \end{bmatrix}.
$$
For this example, $2^2 = 4$ solutions $X_1, X_2, X_3, X_4 \in \mathbb{C}^2$ are computed as $X_{c_1}, X_{c_2}, X_{c_3}, X_{c_4} \in \mathbb{C}^2$ via the polynomial homotopy algorithm of GQME-Toolbox with data $\gamma = 1 + \text{i}.

The solutions have been computed considering homogeneous projective transformation with the random selected data (using MATLAB random generator $\text{rand}$) $A_2 := \begin{bmatrix} 9.5013 \times 10^{-1} & 2.3114 \times 10^{-1} \\ 5.013 \times 10^{-1} & 2.3114 \times 10^{-1} \end{bmatrix}$ and $a_{f_3} := 6.0684 \times 10^{-1} \text{i}$. Also, the solutions of the homotopy equation have been computed as $X_i(\epsilon)$ or $\tilde{X}_i(\epsilon)$, $i = 1, 2, 3, 4$, when considering respectively no scaling and either one of the scalings (3.31) or (3.63); and as $\hat{X}_i(\epsilon)$ or $\hat{\tilde{X}}_i(\epsilon)$, $i = 1, 2, 3, 4$, when considering respectively homogeneous projective transformation with no scaling or with either one of the scalings (3.31) or (3.63). Since it is known a priori that there are no solutions at infinity, computations without homogeneous projective transformation will not cause problems.

In all cases a posteriori numerical analysis of the computed solutions $X_i$, $i = 1, 2, 3, 4$, have been conducted.

Now, the exact solutions $X_i$, $i = 1, 2, 3, 4$, have been computed via symbolic computation in MATLAB. These converted to float number arithmetic are shown below up to four decimal digits.

$$
X_1 = \begin{bmatrix} 1.2564 \times 10^0 \\ 8.8490 \times 10^{-1} \end{bmatrix}, \quad X_2 = \begin{bmatrix} 2.9422 \times 10^{-1} \\ 1.7762 \times 10^0 \end{bmatrix}, \quad X_3 = \begin{bmatrix} 4.0078 \times 10^0 \\ 6.0439 \times 10^{-2} \end{bmatrix}, \quad X_4 = \begin{bmatrix} 2.2501 \times 10^{-1} \\ 5.4829 \times 10^{-2} \end{bmatrix}.
$$

There are four different initial conditions $X(0) \in \mathbb{C}^2$, which are determined from the solution of the easy problem $X(0) \circ X(0) - \begin{bmatrix} 1 \epsilon - 016 \\ 1 \epsilon - 016 \end{bmatrix} = 0_{2 \times 1}$, according to subsection 3.4.2 in chapter 3, when no homogeneous projective transformation is used. These are computed as:

$X_{c_1}(0) = \begin{bmatrix} 1 \epsilon - 008 \\ 1 \epsilon - 008 \end{bmatrix}, \quad X_{c_2}(0) = \begin{bmatrix} 1 \epsilon - 008 \\ 1 \epsilon - 008 \end{bmatrix}, \quad X_{c_3}(0) = \begin{bmatrix} 1 \epsilon - 008 \\ 1 \epsilon - 008 \end{bmatrix}, \quad X_{c_4}(0) = \begin{bmatrix} 1 \epsilon - 008 \\ 1 \epsilon - 008 \end{bmatrix}.$

When homogeneous projective transformation is used, the initial conditions are computed from the solution of the easy problem $X(0) \circ X(0) - (A_2 \text{vec}(X(0)) + a_{f_3})^2 \begin{bmatrix} 1 \epsilon - 016 \\ 1 \epsilon - 016 \end{bmatrix} = 0_{2 \times 1}$ as:

$X_{c_1}(0) = \begin{bmatrix} 6.0684 \times 10^{-9} \text{i} \\ 6.0684 \times 10^{-9} \text{i} \end{bmatrix}, \quad X_{c_2}(0) = \begin{bmatrix} -6.0684 \times 10^{-9} \text{i} \\ 6.0684 \times 10^{-9} \text{i} \end{bmatrix},$

$X_{c_3}(0) = \begin{bmatrix} 6.0684 \times 10^{-9} \text{i} \\ -6.0684 \times 10^{-9} \text{i} \end{bmatrix}, \quad X_{c_4}(0) = \begin{bmatrix} 6.0684 \times 10^{-9} \text{i} \\ 6.0684 \times 10^{-9} \text{i} \end{bmatrix}.$
Each one of $X_{c1}(0), X_{c2}(0), X_{c3}(0), X_{c4}(0) \in \mathbb{C}^2$ is expected to initiate a homotopy path that leads to $X_{c1}, X_{c2}, X_{c3}, X_{c4} \in \mathbb{C}^2$, respectively.

The driver to run the present example in GQME-Toolbox is the m-file Example71.m, given in appendix F. The user has only to type Example71 and to press Enter in the MATLAB workspace where the GQME-Toolbox is installed.

The computations of the scaling problems (3.31) and (3.63) gave the following:

(i) Scaling (3.31): $0.02 \text{ (sec) CPU time and 3712 number of FLOPS},$

$$S(u_X, u_C)^{(u_X, u_C)=(0,0)} = 1.0168e+001, \quad \min\{S(u_X, u_C)\} = 2.0146e+000.$$  

$$u_X = \begin{bmatrix} 3.9267e-001 \\ 5.5287e-001 \end{bmatrix}, \quad u_C = \begin{bmatrix} 2.7246e-001 \\ 2.5755e-001 \end{bmatrix}.$$  

(ii) Scaling (3.63): $1.5276e+002 \text{ (sec) CPU time and 6395490 number of FLOPS},$

$$S(u_X, u_C)^{(u_X, u_C)=(0,0)} = 8.9380e+001, \quad \min\{S(u_X, u_C)\} = 5.4394e+001.$$  

$$u_X = \begin{bmatrix} 3.9267e-001 \\ 5.5287e-001 \end{bmatrix}, \quad u_C = \begin{bmatrix} 2.7246e-001 \\ 2.5755e-001 \end{bmatrix}.$$  

Note that both scaling problems (3.31) and (3.63) possess almost identical solutions. This is natural since there are no big magnitude variations between the equation’s coefficients.

At this point, it should be said that the arithmetic used in the present example is the MATLAB fixed complex arithmetic and \texttt{flh=1}. No homotopy step corrector was posed (i.e. \texttt{flagcor=1}) and the ‘end game’ tolerance was set to \texttt{efinal=1e-20} with the jacobian formula not provided in the ‘end game’ solver (i.e. \texttt{jacobiangqme=’off’}) under a large scale optimization method \texttt{largescale=’on’}. The maximum allowed number of steps and the minimum allowed integration step of the IVP routine RK45GQMEHOM1.m were set to the MATLAB real maximum and minimum number values respectively (i.e. \texttt{maxnsteps=realmax, minstep=realmin}). Also, no discrete homotopy in the data was considered, thus \texttt{step=1} and obviously, no directionality rules used (i.e. \texttt{pdef=0}).

The computational results are presented in figures 7.1-7.6 and in tables 7.1.1-7.1.6. The notation used in the sequel is the same as in chapter 6 and therefore will not redefined here.
Figure 7.1. Homotopy paths for example 7.1 (n.s.n.p).
Figure 7.2. Homotopy paths for example 7.1 (n.s.p.).
Figure 7.3. Homotopy paths for example 7.1 (s.n.p.).
Figure 7.4. Homotopy paths for example 7.1 (s.p.)
Figure 7.5. Homotopy paths for example 7.1 (s/v, n, p).
Figure 7.6. Homotopy paths for example 7.1 (sv. p.).
Table 7.1.1. Computation results for example 7.1.

<table>
<thead>
<tr>
<th>$X$</th>
<th>$X_1$</th>
<th>$X_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\det(J_{X_c})$</td>
<td>n.s.n.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$3.3971e+000$</td>
<td>$2.4667e+001$</td>
</tr>
<tr>
<td></td>
<td>$-2.5235e-014$</td>
<td>$+3.6416e-022$</td>
</tr>
<tr>
<td></td>
<td>n.s.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$3.3971e+000$</td>
<td>$2.4667e+001$</td>
</tr>
<tr>
<td></td>
<td>$+6.7733e-016$</td>
<td>$-2.9980e-015$</td>
</tr>
<tr>
<td>$L(X_c)$</td>
<td>n.s.n.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$1$</td>
<td>$1$</td>
</tr>
<tr>
<td></td>
<td>n.s.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$9.4645e-016$</td>
<td>$-3.9353e-015$</td>
</tr>
<tr>
<td></td>
<td>$+1.5238e-000$</td>
<td>$-1.9581e-000$</td>
</tr>
<tr>
<td>$|\text{Residual}(X_c)|_F$</td>
<td>n.s.n.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$5.6583e-015$</td>
<td>$1.2333e-016$</td>
</tr>
<tr>
<td>$|X|_F$</td>
<td>n.s.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$3.3155e-016$</td>
<td>$6.2407e-016$</td>
</tr>
<tr>
<td>$|X-X_c|_F$</td>
<td>n.s.n.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$7.2597e-015$</td>
<td>$1.3789e-016$</td>
</tr>
<tr>
<td>$|X|_F$</td>
<td>n.s.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$2.6323e-016$</td>
<td>$2.3021e-016$</td>
</tr>
<tr>
<td>$C_\text{m}(X_c)$</td>
<td>n.s.n.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$1.3560e+001$</td>
<td>$1.5621e+001$</td>
</tr>
<tr>
<td>$E_{\text{prf max}}(X_c)$</td>
<td>n.s.n.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$6.8265e-016$</td>
<td>$1.2285e-017$</td>
</tr>
<tr>
<td>$E_{\text{prf max}}(X_c)$</td>
<td>n.s.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$3.4133e-017$</td>
<td>$6.4744e-017$</td>
</tr>
<tr>
<td>$E_{\text{prf max}}(X_c)$</td>
<td>n.s.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$5.2115e-015$</td>
<td>$4.4524e-015$</td>
</tr>
<tr>
<td>$E_{\text{prf max}}(X_c)$</td>
<td>n.s.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$4.5630e-015$</td>
<td>$4.5049e-015$</td>
</tr>
<tr>
<td>$E_{\text{rb max}}(X_c)$</td>
<td>n.s.n.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$2.0511e-015$</td>
<td>$1.1246e-017$</td>
</tr>
<tr>
<td>$E_{\text{rb max}}(X_c)$</td>
<td>n.s.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$1.0685e-016$</td>
<td>$5.9233e-017$</td>
</tr>
<tr>
<td>$E_{\text{rb max}}(X_c)$</td>
<td>n.s.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$3.0278e-015$</td>
<td>$1.7290e-017$</td>
</tr>
<tr>
<td>$E_{\text{rb max}}(X_c)$</td>
<td>n.s.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$1.5140e-016$</td>
<td>$9.1122e-017$</td>
</tr>
<tr>
<td>$E_{\text{rb max}}(X_c)$</td>
<td>n.s.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$2.3115e-014$</td>
<td>$6.2665e-015$</td>
</tr>
<tr>
<td>$E_{\text{rb max}}(X_c)$</td>
<td>n.s.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$2.0239e-014$</td>
<td>$6.3403e-015$</td>
</tr>
<tr>
<td>$\text{NS}$</td>
<td>n.s.n.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$1$</td>
<td>$1$</td>
</tr>
<tr>
<td>$\text{NS P}$</td>
<td>n.s.n.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$1$</td>
<td>$1$</td>
</tr>
<tr>
<td>$|\text{etol JVP}|_F$</td>
<td>n.s.n.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$1.0000e-014$</td>
<td>$1.0000e-014$</td>
</tr>
<tr>
<td>$n_{\text{steps JVP}}$</td>
<td>n.s.n.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$1564$</td>
<td>$1700$</td>
</tr>
<tr>
<td>$\text{min step JVP}$</td>
<td>n.s.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$1187$</td>
<td>$1683$</td>
</tr>
<tr>
<td>$t_{\text{CPU JVP}}(\text{sec})$</td>
<td>n.s.n.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$74$</td>
<td>$79.7400$</td>
</tr>
<tr>
<td>$n_{\text{FLOPS JVP}}$</td>
<td>n.s.n.p</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$17512681$</td>
<td>$19028945$</td>
</tr>
<tr>
<td></td>
<td>$15334361$</td>
<td>$21700521$</td>
</tr>
</tbody>
</table>
### Table 7.1.2

<table>
<thead>
<tr>
<th>$X$</th>
<th>$X_3$</th>
<th>$X_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\det(J_{X_c})$</td>
<td>n.s.n.p</td>
<td>1.7670e + 001 + 7.6132e - 015i</td>
</tr>
<tr>
<td></td>
<td>n.s.p</td>
<td>1.7670e + 001 + 1.4026e - 012i</td>
</tr>
<tr>
<td>$L(X_c)$</td>
<td>n.s.n.p</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>n.s.p</td>
<td>-8.2511e - 015 + 2.1505e - 001i</td>
</tr>
<tr>
<td>$|\text{Residual}(X_c)|_F$</td>
<td>n.s.n.p</td>
<td>9.2028e - 016</td>
</tr>
<tr>
<td></td>
<td>n.s.p</td>
<td>3.4844e - 014</td>
</tr>
<tr>
<td>$|X_c|_F$</td>
<td>n.s.n.p</td>
<td>4.8817e - 016</td>
</tr>
<tr>
<td></td>
<td>n.s.p</td>
<td>3.6671e - 014</td>
</tr>
<tr>
<td>$C_n(X_c)$</td>
<td>n.s.n.p</td>
<td>1.1443e + 001</td>
</tr>
<tr>
<td></td>
<td>n.s.p</td>
<td>1.1443e + 001</td>
</tr>
<tr>
<td>$E_{prf_{\text{max}}}(X_c)$</td>
<td>n.s.n.p</td>
<td>5.3827e - 017</td>
</tr>
<tr>
<td></td>
<td>n.s.p</td>
<td>1.9066e - 015</td>
</tr>
<tr>
<td>$E_{prf_{\text{limax}}}(X_c)$</td>
<td>n.s.n.p</td>
<td>2.6190e - 015</td>
</tr>
<tr>
<td></td>
<td>n.s.p</td>
<td>4.4718e - 015</td>
</tr>
<tr>
<td>$E_{rb}(X_c)$</td>
<td>n.s.n.p</td>
<td>1.6098e - 016</td>
</tr>
<tr>
<td></td>
<td>n.s.p</td>
<td>9.0688e - 015</td>
</tr>
<tr>
<td>$E_{prb_{\text{max}}}(X_c)$</td>
<td>n.s.n.p</td>
<td>3.0761e - 016</td>
</tr>
<tr>
<td></td>
<td>n.s.p</td>
<td>1.0896e - 014</td>
</tr>
<tr>
<td>$E_{prb_{\text{limax}}}(X_c)$</td>
<td>n.s.n.p</td>
<td>1.4967e - 014</td>
</tr>
<tr>
<td></td>
<td>n.s.p</td>
<td>2.5556e - 014</td>
</tr>
<tr>
<td>$NS$</td>
<td>n.s.n.p</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>n.s.p</td>
<td>1</td>
</tr>
<tr>
<td>$NS_{fl}$</td>
<td>n.s.n.p</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>n.s.p</td>
<td>1</td>
</tr>
<tr>
<td>$e_{tol_{\text{IVP}}}$</td>
<td>n.s.n.p</td>
<td>1.0000e - 014</td>
</tr>
<tr>
<td></td>
<td>n.s.p</td>
<td>1.0000e - 014</td>
</tr>
<tr>
<td>$n_{\text{steps}_{\text{IVP}}}$</td>
<td>n.s.n.p</td>
<td>1709</td>
</tr>
<tr>
<td></td>
<td>n.s.p</td>
<td>1126</td>
</tr>
<tr>
<td>$\min_{\text{step}_{\text{IVP}}}$</td>
<td>n.s.n.p</td>
<td>3.7086e - 017</td>
</tr>
<tr>
<td></td>
<td>n.s.p</td>
<td>4.1170e - 017</td>
</tr>
<tr>
<td>$t_{\text{CPU}_{\text{IVP}} (sec)}$</td>
<td>n.s.n.p</td>
<td>54.6100</td>
</tr>
<tr>
<td></td>
<td>n.s.p</td>
<td>54.6100</td>
</tr>
<tr>
<td>$n_{\text{FLOPS}_{\text{IVP}}}$</td>
<td>n.s.n.p</td>
<td>19129286</td>
</tr>
<tr>
<td></td>
<td>n.s.p</td>
<td>14551426</td>
</tr>
</tbody>
</table>

*Table 7.1.2. Computation results for example 7.1.*
Table 7.1.3. Computation results for example 7.1.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>\textit{X}</td>
<td>\textit{X}_1</td>
</tr>
<tr>
<td>$\det(J_{X_c})$</td>
<td>s.n.p</td>
<td>$3.3971e+000$</td>
</tr>
<tr>
<td></td>
<td>s.p</td>
<td>$-2.6588e-021$</td>
</tr>
<tr>
<td>$L(X_c)$</td>
<td>s.n.p</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>s.p</td>
<td>$1.4559e-015$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$+2.2653e-001$</td>
</tr>
<tr>
<td>$|\textit{Residual}(X_c)|_F$</td>
<td>s.n.p</td>
<td>$5.7797e-016$</td>
</tr>
<tr>
<td></td>
<td>s.p</td>
<td>$1.3298e-014$</td>
</tr>
<tr>
<td>$|\textit{X}_c|_F$</td>
<td>s.p</td>
<td>$1.6570e-014$</td>
</tr>
<tr>
<td>$|\textit{X} - \textit{X}_c|_F$</td>
<td>s.n.p</td>
<td>$2.7844e-022$</td>
</tr>
<tr>
<td></td>
<td>s.p</td>
<td>$1.6570e-014$</td>
</tr>
<tr>
<td>$C_n(X_c)$</td>
<td>s.n.p</td>
<td>$1.3560e+001$</td>
</tr>
<tr>
<td></td>
<td>s.p</td>
<td>$1.3560e+001$</td>
</tr>
<tr>
<td>$E_{\text{prf}_\text{max}}(X_c)$</td>
<td>s.n.p</td>
<td>$5.1567e-017$</td>
</tr>
<tr>
<td></td>
<td>s.p</td>
<td>$1.4165e-015$</td>
</tr>
<tr>
<td>$E_{\text{prf}_\text{max}}(X_c)$</td>
<td>s.n.p</td>
<td>$4.5804e-015$</td>
</tr>
<tr>
<td></td>
<td>s.p</td>
<td>$5.9454e-015$</td>
</tr>
<tr>
<td>$E_{\text{rb}}(X_c)$</td>
<td>s.n.p</td>
<td>$1.5541e-016$</td>
</tr>
<tr>
<td></td>
<td>s.p</td>
<td>$4.4231e-015$</td>
</tr>
<tr>
<td>$E_{\text{prb}_\text{max}}(X_c)$</td>
<td>s.n.p</td>
<td>$2.2872e-016$</td>
</tr>
<tr>
<td></td>
<td>s.p</td>
<td>$6.2892e-015$</td>
</tr>
<tr>
<td>$E_{\text{prb}_\text{limax}}(X_c)$</td>
<td>s.n.p</td>
<td>$2.0316e-014$</td>
</tr>
<tr>
<td></td>
<td>s.p</td>
<td>$2.6370e-014$</td>
</tr>
<tr>
<td>$\text{NS}$</td>
<td>s.n.p</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>s.p</td>
<td>1</td>
</tr>
<tr>
<td>$\text{NS}_f$</td>
<td>s.n.p</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>s.p</td>
<td>1</td>
</tr>
<tr>
<td>$e_{\text{tol}_{\text{IVP}}}$</td>
<td>s.n.p</td>
<td>$1.0000e-014$</td>
</tr>
<tr>
<td></td>
<td>s.p</td>
<td>$1.0000e-014$</td>
</tr>
<tr>
<td>$n_{\text{steps}_{\text{IVP}}}$</td>
<td>s.n.p</td>
<td>1784</td>
</tr>
<tr>
<td></td>
<td>s.p</td>
<td>1361</td>
</tr>
<tr>
<td>$\text{min}<em>{\text{step}</em>{\text{IVP}}}$</td>
<td>s.n.p</td>
<td>$1.9797e-017$</td>
</tr>
<tr>
<td></td>
<td>s.p</td>
<td>$2.1265e-017$</td>
</tr>
<tr>
<td>$t_{\text{CPU}_{\text{IVP}}}(\text{sec})$</td>
<td>s.n.p</td>
<td>84.0500</td>
</tr>
<tr>
<td></td>
<td>s.p</td>
<td>63.8900</td>
</tr>
<tr>
<td>$n_{\text{FLOPS}_{\text{IVP}}}$</td>
<td>s.n.p</td>
<td>19959676</td>
</tr>
<tr>
<td></td>
<td>s.p</td>
<td>17567651</td>
</tr>
<tr>
<td>$\text{Table 7.1.4.}$ Computation results for example 7.1.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$X$</td>
<td>$X_3$</td>
<td>$X_4$</td>
</tr>
<tr>
<td>$\text{det} (J_{Xc})$</td>
<td>$s.n.p$</td>
<td>$1.7670e + 001$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$+1.7270e - 015i$</td>
</tr>
<tr>
<td></td>
<td>$s.p$</td>
<td>$1.7670e + 001$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$+2.3807e - 012i$</td>
</tr>
<tr>
<td>$L(X_c)$</td>
<td>$s.n.p$</td>
<td>$1$</td>
</tr>
<tr>
<td></td>
<td>$s.p$</td>
<td>$-1.1474e - 014$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$+7.1775e - 002i$</td>
</tr>
<tr>
<td>$|\text{Residual}(X_c)|_F$</td>
<td>$s.n.p$</td>
<td>$6.6626e - 016$</td>
</tr>
<tr>
<td></td>
<td>$s.p$</td>
<td>$1.0293e - 013$</td>
</tr>
<tr>
<td>$|X_c|_F$</td>
<td>$s.n.p$</td>
<td>$6.8888e - 016$</td>
</tr>
<tr>
<td></td>
<td>$s.p$</td>
<td>$1.0157e - 013$</td>
</tr>
<tr>
<td>$C_n(X_c)$</td>
<td>$s.n.p$</td>
<td>$1.1443e + 001$</td>
</tr>
<tr>
<td></td>
<td>$s.p$</td>
<td>$1.1443e + 001$</td>
</tr>
<tr>
<td>$E_{\text{prf}}$</td>
<td>$s.n.p$</td>
<td>$3.1675e - 017$</td>
</tr>
<tr>
<td></td>
<td>$s.p$</td>
<td>$5.4271e - 015$</td>
</tr>
<tr>
<td>$E_{\text{prfmax}}(X_c)$</td>
<td>$s.n.p$</td>
<td>$2.5969e - 015$</td>
</tr>
<tr>
<td></td>
<td>$s.p$</td>
<td>$7.9923e - 015$</td>
</tr>
<tr>
<td>$E_{\text{rb}}(X_c)$</td>
<td>$s.n.p$</td>
<td>$1.6429e - 016$</td>
</tr>
<tr>
<td></td>
<td>$s.p$</td>
<td>$2.6531e - 014$</td>
</tr>
<tr>
<td>$E_{\text{prb}}(X_c)$</td>
<td>$s.n.p$</td>
<td>$1.8102e - 016$</td>
</tr>
<tr>
<td></td>
<td>$s.p$</td>
<td>$3.1015e - 014$</td>
</tr>
<tr>
<td>$E_{\text{prbmax}}(X_c)$</td>
<td>$s.n.p$</td>
<td>$1.4841e - 014$</td>
</tr>
<tr>
<td></td>
<td>$s.p$</td>
<td>$4.5675e - 014$</td>
</tr>
<tr>
<td>$NS$</td>
<td>$s.n.p$</td>
<td>$1$</td>
</tr>
<tr>
<td></td>
<td>$s.p$</td>
<td>$1$</td>
</tr>
<tr>
<td>$NS_{fl}$</td>
<td>$s.n.p$</td>
<td>$0$</td>
</tr>
<tr>
<td></td>
<td>$s.p$</td>
<td>$1$</td>
</tr>
<tr>
<td>$e_{\text{toljvp}}$</td>
<td>$s.n.p$</td>
<td>$1.0000e - 014$</td>
</tr>
<tr>
<td></td>
<td>$s.p$</td>
<td>$1.0000e - 014$</td>
</tr>
<tr>
<td>$n_{\text{steps_{jvp}}}$</td>
<td>$s.n.p$</td>
<td>$1903$</td>
</tr>
<tr>
<td></td>
<td>$s.p$</td>
<td>$1222$</td>
</tr>
<tr>
<td>$\text{min_{step_{jvp}}}$</td>
<td>$s.n.p$</td>
<td>$1.9797e - 017$</td>
</tr>
<tr>
<td></td>
<td>$s.p$</td>
<td>$2.1265e - 017$</td>
</tr>
<tr>
<td>$t_{\text{CPU_{jvp}} (sec)}$</td>
<td>$s.n.p$</td>
<td>$88.2800$</td>
</tr>
<tr>
<td></td>
<td>$s.p$</td>
<td>$57.3000$</td>
</tr>
<tr>
<td>$n_{\text{FLOPS_{jvp}}}$</td>
<td>$s.n.p$</td>
<td>$21286407$</td>
</tr>
<tr>
<td></td>
<td>$s.p$</td>
<td>$15783586$</td>
</tr>
<tr>
<td>$X$</td>
<td>$X_1$</td>
<td>$X_2$</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>$\text{det}(J_{X_c})$</td>
<td>$sv.n.p$</td>
<td>$3.3971e+000$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$+1.4296e-015i$</td>
</tr>
<tr>
<td></td>
<td>$sv.p$</td>
<td>$3.3971e+000$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$-9.7280e-015i$</td>
</tr>
<tr>
<td>$L(X_c)$</td>
<td>$sv.n.p$</td>
<td>$1$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$2.2452e-015$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$+2.2653e-001i$</td>
</tr>
<tr>
<td>$|\text{Residual}(X_c)|_F$</td>
<td>$sv.n.p$</td>
<td>$2.7369e-015$</td>
</tr>
<tr>
<td>$|X_c|_F$</td>
<td>$sv.p$</td>
<td>$7.8283e-015$</td>
</tr>
<tr>
<td>$|X - X_c|_F$</td>
<td>$sv.n.p$</td>
<td>$1.3560e+001$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$1.3560e+001$</td>
</tr>
<tr>
<td>$E_{pr f_{\text{max}}}(X_c)$</td>
<td>$sv.n.p$</td>
<td>$4.8656e-015$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$5.2825e-015$</td>
</tr>
<tr>
<td>$E_{pr f_{\text{max}}}(X_c)$</td>
<td>$sv.n.p$</td>
<td>$9.9059e-016$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$2.4108e-015$</td>
</tr>
<tr>
<td>$C_n(X_c)$</td>
<td>$sv.n.p$</td>
<td>$1.4935e-014$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$2.3431e-014$</td>
</tr>
<tr>
<td>$E_{pr f_{\text{max}}}(X_c)$</td>
<td>$sv.n.p$</td>
<td>$1.0000e-014$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$1.0000e-014$</td>
</tr>
<tr>
<td>$t_{\text{CPU}_{\text{IVP}}}$</td>
<td>$sv.n.p$</td>
<td>$1784$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$1361$</td>
</tr>
<tr>
<td>$n_{\text{steps}_{\text{IVP}}}$</td>
<td>$sv.n.p$</td>
<td>$1.9797e-017$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$2.1265e-017$</td>
</tr>
<tr>
<td>$n_{\text{FLOPS}_{\text{IVP}}}$</td>
<td>$sv.n.p$</td>
<td>$831.1100$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$64.1600$</td>
</tr>
</tbody>
</table>

Table 7.1.5. Computation results for example 7.1.
<table>
<thead>
<tr>
<th>( d e t(J_{X_C}) )</th>
<th>( X_3 )</th>
<th>( X_4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( sv. n. p )</td>
<td>1.7670e +001 +8.0157e -015i</td>
<td>2.2265e +001 +1.4559e -022i</td>
</tr>
<tr>
<td>( sv. p )</td>
<td>1.7670e +001 +2.5400e -012i</td>
<td>2.2265e +001 +3.3874e -022i</td>
</tr>
<tr>
<td>( L(X_C) )</td>
<td>( sv. n. p )</td>
<td>1</td>
</tr>
<tr>
<td>( sv. p )</td>
<td>-1.1399e -014 +7.1775e -002i</td>
<td>2.1270e -015 -3.8572e -001i</td>
</tr>
<tr>
<td>( | Residual(X_C) |_F )</td>
<td>( sv. n. p )</td>
<td>2.5852e -016</td>
</tr>
<tr>
<td>( | X |_F )</td>
<td>( sv. p )</td>
<td>1.0501e -013</td>
</tr>
<tr>
<td>( | X - X_C |_F )</td>
<td>( sv. n. p )</td>
<td>2.1507e -016</td>
</tr>
<tr>
<td>( | X |_F )</td>
<td>( sv. p )</td>
<td>1.0388e -013</td>
</tr>
<tr>
<td>( C_n(X_C) )</td>
<td>( sv. n. p )</td>
<td>1.1443e +001</td>
</tr>
<tr>
<td>( sv. p )</td>
<td>1.1443e +001</td>
<td>2.2233e +000</td>
</tr>
<tr>
<td>( E_{Prf_{max}}(X_C) )</td>
<td>( sv. n. p )</td>
<td>1.2670e -017</td>
</tr>
<tr>
<td>( sv. p )</td>
<td>5.5262e -015</td>
<td>2.3474e -023</td>
</tr>
<tr>
<td>( E_{Prf_{lm_{max}}}(X_C) )</td>
<td>( sv. n. p )</td>
<td>2.5779e -015</td>
</tr>
<tr>
<td>( sv. p )</td>
<td>8.0914e -015</td>
<td>1.7820e -015</td>
</tr>
<tr>
<td>( E_{rb}(X_C) )</td>
<td>( sv. n. p )</td>
<td>6.4568e -017</td>
</tr>
<tr>
<td>( sv. p )</td>
<td>2.7046e -014</td>
<td>5.7233e -023</td>
</tr>
<tr>
<td>( E_{Prb_{max}}(X_C) )</td>
<td>( sv. n. p )</td>
<td>7.2409e -017</td>
</tr>
<tr>
<td>( sv. p )</td>
<td>3.1581e -014</td>
<td>8.6296e -023</td>
</tr>
<tr>
<td>( E_{Prb_{lm_{max}}}(X_C) )</td>
<td>( sv. n. p )</td>
<td>1.4732e -014</td>
</tr>
<tr>
<td>( sv. p )</td>
<td>4.6240e -014</td>
<td>6.5510e -015</td>
</tr>
<tr>
<td>( NS )</td>
<td>( sv. n. p )</td>
<td>1</td>
</tr>
<tr>
<td>( sv. p )</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( NS_{fl} )</td>
<td>( sv. n. p )</td>
<td>0</td>
</tr>
<tr>
<td>( sv. p )</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( e_{tol_{IVP}} )</td>
<td>( sv. n. p )</td>
<td>1.0000e -014</td>
</tr>
<tr>
<td>( sv. p )</td>
<td>1.0000e -014</td>
<td>1.0000e -014</td>
</tr>
<tr>
<td>( n_{steps_{IVP}} )</td>
<td>( sv. n. p )</td>
<td>1903</td>
</tr>
<tr>
<td>( sv. p )</td>
<td>1222</td>
<td>1161</td>
</tr>
<tr>
<td>( min_{step_{IVP}} )</td>
<td>( sv. n. p )</td>
<td>1.9797e -017</td>
</tr>
<tr>
<td>( sv. p )</td>
<td>2.1265e -017</td>
<td>2.1265e -017</td>
</tr>
<tr>
<td>( t_{CPU_{IVP}}(sec) )</td>
<td>( sv. n. p )</td>
<td>89.1400</td>
</tr>
<tr>
<td>( sv. p )</td>
<td>57.4100</td>
<td>54.7700</td>
</tr>
<tr>
<td>( n_{FLOPS_{IVP}} )</td>
<td>( sv. n. p )</td>
<td>21286407</td>
</tr>
<tr>
<td>( sv. p )</td>
<td>15783586</td>
<td>15000651</td>
</tr>
</tbody>
</table>

Table 7.1.6. Computation results for example 7.1.
From figures 7.1-7.6, it can be observed that both scaling and homogeneous projective transformation change the length and the curvature of the homotopy paths. This was mentioned also in chapter 3.

Now, scaling ‘magnifies’ the homotopy paths to be close as possible to the unit ball. Hence, scaling is helpful not only in situations where the magnitudes of the data are large (see example in chapter 3) but also in the cases where the magnitudes of the data are small.

From tables 7.1.1-7.1.6, it is apparent that, all cases provided acceptable solutions and indicated numerical stability. Some pessimism appears in some cases where the results are computed considering floating point rounding errors. However, this is eliminated when using both scaling (3.63) and homogeneous projective transformation. In addition, the last case provides the best overall performance in terms of speed and accuracy. Hence, for the particular setup of the present example the best results can be obtained only with the combination of scaling (3.63) and the homogeneous projective transformation. Note also that all results are in accordance with the theory presented in chapters 2-4. However, it is suggested to the reader to carry out his/her own experiments with different setups in the driver Example71.m, in order to investigate more the behaviour of the algorithms in different situations.

Finally, it should be said that higher dimensional problems could also be tackled. In general, the case of $l \in \mathbb{Z}^+$ players in a scalar game will eventually lead to a quadratic algebraic matrix equation with dimension $l \times 1$. Note that, symbolic computation will be impractical to tackle cases with $l \geq 7$, hence the usefulness of the GQME-Toolbox. However the present example, has been chosen with small dimensions in order to compute the exact solutions via symbolic computation and then compare them with the computed solutions that GQME-Toolbox provides.

### 7.3 Second Order Polynomial Algebraic Matrix Equation and the Quadratic Eigenvalue Problem

Consider the second order quadratic algebraic matrix equation

$$AX^2 + BX + C = 0,$$  \hspace{1cm} (7.4)

where $A, B, C, X \in \mathbb{C}^{n \times n}$, $n \in \mathbb{Z}^+$. Equation (7.4) is closely related to the quadratic eigenvalue problem

$$\lambda^2 Ax + \lambda Bx + Cx = 0,$$  \hspace{1cm} (7.5)

where $A, B, C \in \mathbb{C}^{n \times n}$, $x \in \mathbb{C}^n$, $n \in \mathbb{Z}^+$, $\lambda \in \mathbb{C}$. The quadratic eigenvalue problem (7.5) arises in the analysis of structural systems and vibration problems [105]. One way to solve (7.5) is to
transform it to a generalized eigenproblem $Gx = \lambda Hx$ with double dimension $2n$. Another approach is to find the solution $X$ of (7.4) and then write

$$\lambda^2 A + \lambda B + C = -(AX + AA + B)(X - \lambda I_n).$$  \hspace{1cm} (7.6)

From the above, it follows that the eigenvalues of (7.5) are those of $X$ and those of the generalized eigenvalue problem $-(AX + B)x = \lambda Ax$. Note that now both problems are of dimension $n \times n$.

A solution $X$ of (7.4) is called a solvent, and the solvent with all eigenvalues greater in magnitude then the eigenvalues of any other solvent is called the dominant solvent [80]. For (7.4), there are theoretical results about the existence of solutions and some of these are reported in [80] and its references. One important result, which determines the number of finite solvents, is theorem 7.2.

**Theorem 7.2 [80];**

Suppose $Q(\lambda) := \lambda^2 A + \lambda B + C$ has $v \in N^*$ distinct eigenvalues $\{\lambda_i\}_{i=1}^v$, with $n \leq v \leq 2n$, and that the corresponding set of $v$ eigenvectors $\{u_i\}_{i=1}^v$, satisfies the Haar condition (that is, every subset of $n$ of them is linearly independent). Then there are at least $\frac{v!}{(v-n)!n!}$ different solvents of (7.4), and exactly this many if $v = 2n$, which are given by

$$X = W\text{diag}[\sigma_1 \cdots \sigma_n]W^{-1}, \quad W = [w_1 \cdots w_n],$$

where the eigenpairs $(\sigma_i, w_i)_{i=1}^n$ are chosen from among the eigenpairs $(\lambda_i, u_i)_{i=1}^v$ of $Q(\lambda)$.

**Proof.**

The proof is provided in [80].

Equation (7.4), is studied and numerically solved in the past using mainly Newton’s method and the generalization of Bernoulli’s method for scalar algebraic polynomials, for the general matrix polynomial algebraic equation

$$\sum_{i=0}^n A_i X^i = 0,$$  \hspace{1cm} (7.7)

where $A_i, X \in C^{n \times n}, i, m \in Z_+$, [39], [43], [42]. Note also, that the probability-1 homotopy algorithms in their general form as presented in chapter 2 can also be used for the numerical solution of (7.7). Recently in [79] and [80], (7.4) is studied via Newton’s method with exact line search. Also, [80] presents the numerical analysis of (7.4) subject to first order approximations.
All the methods mentioned above are able to provide in theory one solvent, and are not
global convergent methods. A tricky point with those methods is also the choice of different
initial conditions in order to compute more than one solvent. For example, it is not theoretically
justified how to choose different initial conditions in order not to compute the same solvent.
However, it should be said that these methods are fast and can handle problems of high
dimensions.

Now, the contribution of this section is that the GQME-Toolbox is able via polynomial
homotopies to compute all the solvents of (7.4), if there is a finite number of them. In the latter
case, there are at most $2n^2$ solvents. If there is an infinite number of solvents then, at least $2n^2$
solvents can be computed. The computations can encounter solutions at infinity and solutions
with multiplicities too. Moreover, there is the guarantee of convergence with probability one.
Also, the fixed point homotopy algorithms of the GQME-Toolbox can be used if only one
solvent is desirable to be computed. It is obvious that (2.1) can be easy specialized to (7.4) in
many ways. The aim in writing this specialization will be as in previous examples of this thesis
to avoid pessimism during the posteriori numerical analysis of (7.4).

Both polynomial and fixed point homotopies of the GQME-Toolbox will be illustrated
in the next subsections in terms of two numerical examples 7.2 and 7.3 respectively, taken from
the literature.

### 7.3.1 Numerical Example 7.2
In this subsection, a numerical example for (7.4), taken from [79], [39], is considered. The data
of (7.4) are: $A = B = I_2$, $C = \begin{bmatrix} -8 & -12 \\ -18 & -26 \end{bmatrix}$. In terms of (2.1) and hence in GQME-Toolbox and
in order to avoid pessimism when performing the numerical analysis of the computed solutions,
(7.4) can be written as (2.1) with data:

$$
A_1 = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 0.7 & 0 \\ 0 & 0.7 \end{bmatrix}, \quad B_1 = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},
$$

$$
C_1 = \begin{bmatrix} 0.4 & 0 \\ 0 & 0.4 \end{bmatrix}, \quad C_2 = \begin{bmatrix} 0.6 & 0 \\ 0 & 0.6 \end{bmatrix}, \quad D_1 = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}, \quad D_2 = \begin{bmatrix} 5 & 0 \\ 0 & 5 \end{bmatrix},
$$

$$
E_1 = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}, \quad E_2 = \begin{bmatrix} 0.2 & 0 \\ 0 & 0.2 \end{bmatrix}, \quad G = \begin{bmatrix} -8 & -12 \\ -18 & -26 \end{bmatrix}.
$$

Note that it was found out that the twice repeating matrix $B_1 = E_1 = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$, do not
cause pessimism during the numerical analysis of the computed solutions.
For this example, there are at least \( 2^{2(2)} = 16 \) solutions \( X_i \in \mathbb{C}^{2 \times 2}, \ i = 1, \ldots, 16 \) which are computed as \( X_{e_i} \in \mathbb{C}^{2 \times 2} \) via the polynomial homotopy algorithm of GQME-Toolbox with data \( \gamma = 1 + li \). Now, there are four finite solvents all real which are denoted without loss of generality \( X_1, X_2, X_3, X_4 \in \mathbb{R}^{2 \times 2} \) as the exact solvents and \( X_{e1}, X_{e2}, X_{e3}, X_{e4} \in \mathbb{C}^{2 \times 2} \) as the respective computed ones.

The solutions have been computed considering homogeneous projective transformation with the random selected data (using MATLAB random generator \textit{rand})

\[
\begin{align*}
A_a &= \begin{bmatrix}
3.5287 \cdot 10^{-1} & 8.1317 \cdot 10^{-1} \\
9.8613 \cdot 10^{-3} & 1.3889 \cdot 10^{-1}
\end{bmatrix},
\end{align*}
\]

Also, the solutions of the homotopy equation have been computed as \( X_{ic}(\epsilon) \) or \( \bar{X}_{ic}(\epsilon) \), \( i = 1, 2, 3, 4 \), when considering respectively no scaling or the scaling \((3.31)\), and as \( \hat{X}_{ic}(\epsilon) \) or \( \hat{\bar{X}}_{ic}(\epsilon) \), \( i = 1, 2, 3, 4 \), when considering respectively homogeneous projective transformation with no scaling or with the scaling \((3.31)\).

In all cases, a posteriori numerical analysis of the computed solutions \( X_{ic}, \ i = 1, 2, 3, 4 \), have been conducted. The results are presented in this section for the finite solutions \( X_{e1}, X_{e2}, X_{e3}, X_{e4} \in \mathbb{C}^{2 \times 2} \) only. The other solutions \( X_{e5}, \ldots, X_{e16} \in \mathbb{C}^{2 \times 2} \) are found to be at infinity and corresponding to these solutions only the homogeneous projective transformations and the Frobenious norm of the residual of \((7.4)\) are reported.

Now, the exact finite solutions \( X_i, \ i = 1, 2, 3, 4 \), have been computed via symbolic computation in MATLAB. These are converted to float number arithmetic and are shown below up to four decimal digits.

\[
\begin{align*}
X_1 &= \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, & X_2 &= \begin{bmatrix} 8.0558 \cdot 10^{-1} & 2.0889e+000 \\ 3.1334e+000 & 3.9390e+000 \end{bmatrix}, \\
X_3 &= \begin{bmatrix} 2 & 2 \\ 3 & 5 \end{bmatrix}, & X_4 &= \begin{bmatrix} 1.8056e+000 & 2.0889e+000 \\ 3.1334e+000 & 4.9390e+000 \end{bmatrix}.
\end{align*}
\]

There are sixteen different initial conditions \( X(0) \in \mathbb{C}^{2 \times 2} \), which are determined from the solution of the easy problem \( X(0) \circ X(0) - U_{2 \times 2} = 0_{2 \times 2} \), according to subsection 3.4.2 in chapter 3, when no homogeneous projective transformation is used. From these initial conditions the ones that lead to the finite solutions \( X_{e1}, X_{e2}, X_{e3}, X_{e4} \in \mathbb{C}^{2 \times 2} \) are respectively:
When homogeneous projective transformation is used, the initial conditions are computed from the solution of the easy problem $X(0) \cdot X(0) - \left( A_{a} \cdot \text{vec}(X(0)) + a_{f} \right)^{2} U_{2 \times 2} = 0_{2 \times 2}$, according to subsection 3.4.2 in chapter 3. From these initial conditions the ones that lead to the finite solutions $X_{c1}, X_{c2}, X_{c3}, X_{c4} \in C^{2 \times 2}$ are respectively:

$$X_{c1}(0) = \begin{pmatrix} 6.4414 e^{-001} \cos \theta & 1.523 e^{-001} \sin \theta \\ -1.523 e^{-001} \cos \theta & 6.4414 e^{-001} \sin \theta \end{pmatrix} U_{2 \times 2}, \quad X_{c2}(0) = \begin{pmatrix} 3.0322 e^{-001} \cos \theta & 8.7598 e^{-002} \sin \theta \\ -8.7598 e^{-002} \cos \theta & 3.0322 e^{-001} \sin \theta \end{pmatrix} U_{2 \times 2}.$$  

Each one of $X_{c1}(0), X_{c2}(0), X_{c3}(0), X_{c4}(0) \in C^{2 \times 2}$ is expected to initiate a homotopy path that leads to $X_{c1}, X_{c2}, X_{c3}, X_{c4} \in C^{2 \times 2}$, respectively.

The driver to run the present example in GQME-Toolbox is the m-file Example72.m, given in appendix D. The user has only to type Example72 and to press Enter in the MATLAB workspace where the GQME-Toolbox is installed.

The computation of the scaling problem (3.31) produced the following:

All cases took 0.03 (sec) CPU time and 16077 number of FLOPS and

$$S(u_{X}, u_{C}) = 6.0300 e + 001, \quad u_{X} = \begin{pmatrix} 2.3292 e^{-001} U_{2 \times 2} \end{pmatrix}, \quad u_{C} = \begin{pmatrix} 2.7707 e^{-001} & 2.8136 e^{-001} \\ 2.8566 e^{-001} & 2.8955 e^{-001} \end{pmatrix}.$$  

The arithmetic used in the present example is the MATLAB fixed complex arithmetic and $flh=1$. No homotopy step corrector was posed (i.e. $flagcor=1$) and the ‘end game’ tolerance was set to $efinal=1e-16$ with the Jacobian formula not provided in the ‘end game’ solver (i.e. $jacobianqme=’off’$) under a large scale optimization method $largescale=’on’$. The maximum allowed number of steps and the minimum allowed integration step of the IVP routine RK45GQMEHOM1.m were set to the MATLAB real maximum and minimum number values respectively (i.e. $maxnsteps=realmax$, $minstep=realmin$).

Also, no discrete homotopy in the data was considered, thus $step=1$ and obviously, no directionality rules were used (i.e. $pdef=0$).

The computational results are presented in figures 7.7-7.9 and in tables 7.2.1 through 7.2.5. The notation is the same as in example 7.1.
Figure 7.7. Homotopy paths of finite solutions for example 7.2 ($n.s.n.p.$).
Figure 7.8. Homotopy paths of finite solutions for example 7.2 (n.s.p).
Figure 7.9. Homotopy paths of finite solutions for example 7.2 (s.p.).
<table>
<thead>
<tr>
<th></th>
<th>X</th>
<th>X₁</th>
<th>X₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{det}(J_{X_c}) ) n.s.p</td>
<td>2.5574 \times 10^1</td>
<td>-4.5543 \times 10^2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-2.4125 \times 10^1 i</td>
<td>+9.2804 \times 10^1 i</td>
<td></td>
</tr>
<tr>
<td>s.p</td>
<td>2.5574 \times 10^1</td>
<td>-4.5543 \times 10^2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>+1.2713 \times 10^1 i</td>
<td>-1.4891 \times 10^1 i</td>
<td></td>
</tr>
<tr>
<td>( L(X_c) ) n.s.p</td>
<td>5.1116 \times 10^0</td>
<td>-1.2018 \times 10^0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>+8.5642 \times 10^2 i</td>
<td>+8.4490 \times 10^2 i</td>
<td></td>
</tr>
<tr>
<td>s.p</td>
<td>3.9246 \times 10^-07</td>
<td>2.1073 \times 10^-07</td>
<td></td>
</tr>
<tr>
<td></td>
<td>+2.0910 \times 10^-01 i</td>
<td>+2.051 \times 10^-01 i</td>
<td></td>
</tr>
<tr>
<td>( \left| \text{Residual}(X_c) \right|_F ) n.s.p</td>
<td>6.2792 \times 10^-16</td>
<td>1.3681 \times 10^-15</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6.6929 \times 10^-16</td>
<td>9.6738 \times 10^-16</td>
<td></td>
</tr>
<tr>
<td>( \left| X_c \right|_F ) s.p</td>
<td>2.1897 \times 10^-15</td>
<td>2.0652 \times 10^-16</td>
<td></td>
</tr>
<tr>
<td>( \left| X - X_c \right|_F ) n.s.p</td>
<td>8.4630 \times 10^-16</td>
<td>7.1312 \times 10^-16</td>
<td></td>
</tr>
<tr>
<td>( \left| X \right|_F ) s.p</td>
<td>2.3106 \times 10^-17</td>
<td>3.9677 \times 10^-17</td>
<td></td>
</tr>
<tr>
<td>( C_{n}(X_c) ) n.s.p</td>
<td>3.5895 \times 10^0</td>
<td>3.6112 \times 10^0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3.5895 \times 10^0</td>
<td>3.6112 \times 10^0</td>
<td></td>
</tr>
<tr>
<td>( E_{Prf_{\text{max}}}(X_c) ) s.p</td>
<td>2.7580 \times 10^-17</td>
<td>5.6102 \times 10^-17</td>
<td></td>
</tr>
<tr>
<td>( E_{Prf_{\text{max}}}(X_c) ) n.s.p</td>
<td>2.7376 \times 10^-16</td>
<td>5.2321 \times 10^-16</td>
<td></td>
</tr>
<tr>
<td>( E_{rb}(X_c) ) s.p</td>
<td>2.7374 \times 10^-16</td>
<td>4.4581 \times 10^-16</td>
<td></td>
</tr>
<tr>
<td>( E_{Prb_{\text{max}}}(X_c) ) n.s.p</td>
<td>9.1347 \times 10^-16</td>
<td>1.5924 \times 10^-15</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6.9494 \times 10^-16</td>
<td>1.3265 \times 10^-15</td>
<td></td>
</tr>
<tr>
<td>( E_{Prb_{\text{max}}}(X_c) ) s.p</td>
<td>7.0730 \times 10^-14</td>
<td>7.0582 \times 10^-14</td>
<td></td>
</tr>
<tr>
<td>( E_{Prb_{\text{max}}}(X_c) ) n.s.p</td>
<td>7.0512 \times 10^-14</td>
<td>7.0316 \times 10^-14</td>
<td></td>
</tr>
<tr>
<td>( NS ) s.p</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>( NS_{\beta} ) n.s.p</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>( e_{tol_{IVP}} ) s.p</td>
<td>1.0000 \times 10^-06</td>
<td>1.0000 \times 10^-06</td>
<td></td>
</tr>
<tr>
<td>( n_{steps_{IVP}} ) n.s.p</td>
<td>35</td>
<td>39</td>
<td></td>
</tr>
<tr>
<td>( min_{step_{IVP}} ) s.p</td>
<td>3.2</td>
<td>3.8</td>
<td></td>
</tr>
<tr>
<td>( n_{FLOPS_{IVP}} ) n.s.p</td>
<td>2.771290</td>
<td>2.761829</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2394686</td>
<td>2681306</td>
<td></td>
</tr>
</tbody>
</table>

*Table 7.2.1. Computation results for example 7.2.*
<table>
<thead>
<tr>
<th></th>
<th>( n.s.p )</th>
<th>( s.p )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \det (J_{X_c}) )</td>
<td>( 6.1638e+002 )</td>
<td>( -2.8507e+001 )</td>
</tr>
<tr>
<td></td>
<td>( -1.6057e-015i )</td>
<td>( -2.2478e-011i )</td>
</tr>
<tr>
<td>( L(X_c) )</td>
<td>( 9.2691e-007 )</td>
<td>( 4.2952e-008 )</td>
</tr>
<tr>
<td></td>
<td>( -4.1727e-002i )</td>
<td>( -4.1452e-002i )</td>
</tr>
<tr>
<td></td>
<td>( 4.7486e-007 )</td>
<td>( -7.6390e-008 )</td>
</tr>
<tr>
<td></td>
<td>( -6.2250e-002i )</td>
<td>( -6.1891e-002i )</td>
</tr>
<tr>
<td>( |\text{Residual}(X_c)|_F )</td>
<td>( n.s.p )</td>
<td>( s.p )</td>
</tr>
<tr>
<td>( |X_c|_F )</td>
<td>( 2.7410e-016 )</td>
<td>( 3.9290e-015 )</td>
</tr>
<tr>
<td></td>
<td>( 6.7013e-014 )</td>
<td>( 1.1972e-006 )</td>
</tr>
<tr>
<td>( |X - X_c|_F )</td>
<td>( 1.1869e-016 )</td>
<td>( 2.3420e-015 )</td>
</tr>
<tr>
<td>( |X|_F )</td>
<td>( 2.6058e-013 )</td>
<td>( 4.8351e-006 )</td>
</tr>
<tr>
<td>( C_n(X_c) )</td>
<td>( n.s.p )</td>
<td>( s.p )</td>
</tr>
<tr>
<td></td>
<td>( 3.0186e+001 )</td>
<td>( 3.0273e+001 )</td>
</tr>
<tr>
<td></td>
<td>( 3.0186e+001 )</td>
<td>( 3.0275e+001 )</td>
</tr>
<tr>
<td>( E_{Pr_{\text{max}}}(X_c) )</td>
<td>( n.s.p )</td>
<td>( s.p )</td>
</tr>
<tr>
<td></td>
<td>( 9.1984e-018 )</td>
<td>( 1.7890e-016 )</td>
</tr>
<tr>
<td></td>
<td>( 3.1266e-015 )</td>
<td>( 5.3037e-008 )</td>
</tr>
<tr>
<td>( E_{Pr_{\text{max}}}(X_c) )</td>
<td>( n.s.p )</td>
<td>( s.p )</td>
</tr>
<tr>
<td></td>
<td>( 3.3589e-015 )</td>
<td>( 3.5301e-015 )</td>
</tr>
<tr>
<td></td>
<td>( 6.4763e-015 )</td>
<td>( 5.3037e-008 )</td>
</tr>
<tr>
<td>( E_{rb}(X_c) )</td>
<td>( n.s.p )</td>
<td>( s.p )</td>
</tr>
<tr>
<td></td>
<td>( 1.2359e-016 )</td>
<td>( 1.7632e-015 )</td>
</tr>
<tr>
<td></td>
<td>( 3.1359e-014 )</td>
<td>( 4.5246e-007 )</td>
</tr>
<tr>
<td>( E_{Pr_{b_{\text{max}}}}(X_c) )</td>
<td>( n.s.p )</td>
<td>( s.p )</td>
</tr>
<tr>
<td></td>
<td>( 2.5781e-016 )</td>
<td>( 7.0000e-015 )</td>
</tr>
<tr>
<td></td>
<td>( 1.1607e-013 )</td>
<td>( 2.0595e-006 )</td>
</tr>
<tr>
<td>( E_{Pr_{b_{\text{max}}}}(X_c) )</td>
<td>( n.s.p )</td>
<td>( s.p )</td>
</tr>
<tr>
<td></td>
<td>( 9.4125e-014 )</td>
<td>( 1.0073e-013 )</td>
</tr>
<tr>
<td></td>
<td>( 2.0994e-013 )</td>
<td>( 2.0595e-006 )</td>
</tr>
<tr>
<td>( NS )</td>
<td>( n.s.p )</td>
<td>( s.p )</td>
</tr>
<tr>
<td></td>
<td>( 1 )</td>
<td>( 1 )</td>
</tr>
<tr>
<td>( NS_{j_b} )</td>
<td>( n.s.p )</td>
<td>( s.p )</td>
</tr>
<tr>
<td></td>
<td>( 1 )</td>
<td>( 1 )</td>
</tr>
<tr>
<td>( e_{10}^{I_{\text{IVP}}} )</td>
<td>( n.s.p )</td>
<td>( s.p )</td>
</tr>
<tr>
<td></td>
<td>( 1.0000e-006 )</td>
<td>( 1.0000e-006 )</td>
</tr>
<tr>
<td></td>
<td>( 1.0000e-006 )</td>
<td>( 1.0000e-006 )</td>
</tr>
<tr>
<td>( n_{\text{steps}<em>{I</em>{\text{IVP}}} )</td>
<td>( n.s.p )</td>
<td>( s.p )</td>
</tr>
<tr>
<td></td>
<td>( 25 )</td>
<td>( 24 )</td>
</tr>
<tr>
<td></td>
<td>( 21 )</td>
<td>( 21 )</td>
</tr>
<tr>
<td>( \min_{\text{step}<em>{I</em>{\text{IVP}}} )</td>
<td>( n.s.p )</td>
<td>( s.p )</td>
</tr>
<tr>
<td></td>
<td>( 9.3084e-003 )</td>
<td>( 9.3294e-003 )</td>
</tr>
<tr>
<td></td>
<td>( 1.5867e-002 )</td>
<td>( 1.5363e-002 )</td>
</tr>
<tr>
<td>( t_{CPU_{I_{\text{IVP}}} )</td>
<td>( n.s.p )</td>
<td>( s.p )</td>
</tr>
<tr>
<td></td>
<td>( 1.3900 )</td>
<td>( 1.4400 )</td>
</tr>
<tr>
<td></td>
<td>( 1.1500 )</td>
<td>( 1.3100 )</td>
</tr>
<tr>
<td>( n_{FLOPS_{I_{\text{IVP}}} )</td>
<td>( n.s.p )</td>
<td>( s.p )</td>
</tr>
<tr>
<td></td>
<td>( 1817697 )</td>
<td>( 1858623 )</td>
</tr>
<tr>
<td></td>
<td>( 1490008 )</td>
<td>( 1670828 )</td>
</tr>
</tbody>
</table>

Table 7.2.2. Computation results for example 7.2.
<table>
<thead>
<tr>
<th>$X$</th>
<th>$X_1$</th>
<th>$X_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>det($J X_c$)</td>
<td>n.s.n.p</td>
<td>2.5574e + 001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+1.4776e - 013i</td>
</tr>
<tr>
<td>$L(X_c)$</td>
<td>n.s.n.p</td>
<td>1</td>
</tr>
<tr>
<td>$|\text{Residual}(X_c)|_F$</td>
<td>n.s.n.p</td>
<td>1.2973e - 015</td>
</tr>
<tr>
<td>$|X|_F$</td>
<td>n.s.n.p</td>
<td>1.6767e - 016</td>
</tr>
<tr>
<td>$C_n(X_c)$</td>
<td>n.s.n.p</td>
<td>3.5895e + 001</td>
</tr>
<tr>
<td>$E_{\text{prf}}_{\text{max}}(X_c)$</td>
<td>n.s.n.p</td>
<td>4.3994e - 017</td>
</tr>
<tr>
<td>$E_{\text{prb}}_{\text{max}}(X_c)$</td>
<td>n.s.n.p</td>
<td>2.9041e - 015</td>
</tr>
<tr>
<td>$E_{\text{rb}}(X_c)$</td>
<td>n.s.n.p</td>
<td>5.3047e - 016</td>
</tr>
<tr>
<td>$E_{\text{prb}}_{\text{max}}(X_c)$</td>
<td>n.s.n.p</td>
<td>1.0791e - 015</td>
</tr>
<tr>
<td>$E_{\text{prf}}_{\text{max}}(X_c)$</td>
<td>n.s.n.p</td>
<td>7.0896e - 014</td>
</tr>
<tr>
<td>NS</td>
<td>n.s.n.p</td>
<td>1</td>
</tr>
<tr>
<td>$\text{NSfl}$</td>
<td>n.s.n.p</td>
<td>1</td>
</tr>
<tr>
<td>$\epsilon_{\text{tol}}_{\text{IVP}}$</td>
<td>n.s.n.p</td>
<td>1.0000e - 006</td>
</tr>
<tr>
<td>$n_{\text{steps}_{\text{IVP}}}$</td>
<td>n.s.n.p</td>
<td>33</td>
</tr>
<tr>
<td>$\min_{\text{step}}_{\text{IVP}}$</td>
<td>n.s.n.p</td>
<td>5.7183e - 003</td>
</tr>
<tr>
<td>$t_{\text{CPU}}_{\text{IVP}}$</td>
<td>n.s.n.p</td>
<td>1.9400</td>
</tr>
<tr>
<td>$n_{\text{FLOPS}}_{\text{IVP}}$</td>
<td>n.s.n.p</td>
<td>1.9400</td>
</tr>
</tbody>
</table>

Table 7.2.3. Computation results for example 7.2.
<table>
<thead>
<tr>
<th></th>
<th>$X_1$</th>
<th>$X_3$</th>
<th>$X_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{det}(J_{X_c})$</td>
<td>n.s.n.p</td>
<td>6.1638e+002</td>
<td>$-2.8507e+001$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$-2.7732e-01i$</td>
<td>$+3.4286e-014i$</td>
</tr>
<tr>
<td>$L(X_c)$</td>
<td>n.s.n.p</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$|\text{Residual}(X_c)|_F$</td>
<td>n.s.n.p</td>
<td>1.1718e-015</td>
<td>1.0985e-015</td>
</tr>
<tr>
<td>$|X_c|_F$</td>
<td>n.s.n.p</td>
<td>1.7995e-015</td>
<td>7.6799e-017</td>
</tr>
<tr>
<td>$|X-X_c|_F$</td>
<td>n.s.n.p</td>
<td>1.7995e-015</td>
<td>7.6799e-017</td>
</tr>
<tr>
<td>$|X|_F$</td>
<td>n.s.n.p</td>
<td>1.7995e-015</td>
<td>7.6799e-017</td>
</tr>
<tr>
<td>$C_n(X_c)$</td>
<td>n.s.n.p</td>
<td>3.0186e+001</td>
<td>3.0273e+001</td>
</tr>
<tr>
<td>$E_{\text{prf}_{\text{max}}}(X_c)$</td>
<td>n.s.n.p</td>
<td>4.7784e-017</td>
<td>3.6910e-017</td>
</tr>
<tr>
<td>$E_{\text{prf}_{\text{max}}}(X_c)$</td>
<td>n.s.n.p</td>
<td>3.3975e-015</td>
<td>3.3881e-015</td>
</tr>
<tr>
<td>$E_{\text{rb}}(X_c)$</td>
<td>n.s.n.p</td>
<td>5.3366e-016</td>
<td>4.5717e-016</td>
</tr>
<tr>
<td>$E_{\text{prb}_{\text{max}}}(X_c)$</td>
<td>n.s.n.p</td>
<td>1.6560e-015</td>
<td>1.0333e-015</td>
</tr>
<tr>
<td>$E_{\text{prb}_{\text{max}}}(X_c)$</td>
<td>n.s.n.p</td>
<td>9.5523e-014</td>
<td>9.4761e-014</td>
</tr>
<tr>
<td>$NS$</td>
<td>n.s.n.p</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$NS_{fl}$</td>
<td>n.s.n.p</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$e_{\text{tol}_{\text{IVP}}}$</td>
<td>n.s.n.p</td>
<td>1.0000e-006</td>
<td>1.0000e-006</td>
</tr>
<tr>
<td>$n_{\text{steps}_{\text{IVP}}}$</td>
<td>n.s.n.p</td>
<td>30</td>
<td>32</td>
</tr>
<tr>
<td>$m_{\text{step}_{\text{IVP}}}$</td>
<td>n.s.n.p</td>
<td>5.3185e-003</td>
<td>5.0244e-003</td>
</tr>
<tr>
<td>$t_{\text{CPU}_{\text{IVP}}}$</td>
<td>n.s.n.p</td>
<td>1.7200</td>
<td>1.7700</td>
</tr>
<tr>
<td>$n_{\text{FLOPS}_{\text{IVP}}}$</td>
<td>n.s.n.p</td>
<td>1943675</td>
<td>2018297</td>
</tr>
</tbody>
</table>

Table 7.2.4. Computation results for example 7.2.
<table>
<thead>
<tr>
<th>$X$</th>
<th>$L(X_c)$</th>
<th>Residual($X_c$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_5$</td>
<td>$1.3652e-004$</td>
<td>$1.6635e+001$</td>
</tr>
<tr>
<td>$X_6$</td>
<td>$4.2039e-005$</td>
<td>$2.2011e+001$</td>
</tr>
<tr>
<td>$X_7$</td>
<td>$5.0689e-005$</td>
<td>$1.9588e+001$</td>
</tr>
<tr>
<td>$X_8$</td>
<td>$1.8770e-004$</td>
<td>$1.5697e+001$</td>
</tr>
<tr>
<td>$X_9$</td>
<td>$1.5052e-004$</td>
<td>$2.4697e+001$</td>
</tr>
<tr>
<td>$X_{10}$</td>
<td>$1.2287e-004$</td>
<td>$1.7501e+001$</td>
</tr>
<tr>
<td>$X_{11}$</td>
<td>$4.3456e-004$</td>
<td>$2.4697e+001$</td>
</tr>
<tr>
<td>$X_{12}$</td>
<td>$1.4637e-004$</td>
<td>$2.4751e+001$</td>
</tr>
<tr>
<td>$X_{13}$</td>
<td>$1.2752e-004$</td>
<td>$2.4700e+001$</td>
</tr>
<tr>
<td>$X_{14}$</td>
<td>$7.649e-005$</td>
<td>$2.4760e+001$</td>
</tr>
<tr>
<td>$X_{15}$</td>
<td>$6.7464e-004$</td>
<td>$2.4308e+001$</td>
</tr>
<tr>
<td>$X_{16}$</td>
<td>$6.0138e-005$</td>
<td>$1.7783e+001$</td>
</tr>
</tbody>
</table>

Table 7.2.5. Computation results for example 7.2.
From figures 7.7-7.8, it is apparent that the homogeneous projective transformation reduces the length and changes the curvature of the homotopy paths. Now when in addition scaling is used the homotopy paths remain almost the same, as can be seen in figure 7.9. Hence, the homogeneous projective transformation in his case dominates over scaling the behaviour of the homotopy paths and hence the behaviour of the computational process. This is something that might be expected, since the scaling does not change the original equation much. In this respect note that, \( \min\{S(u_X, u_C)\} \) is close to \( S(u_X, u_C) = (0,0) \).

From tables 7.2.1-7.2.4, it is obvious that in all cases very good performance (speed), robustness and accuracy were obtained. All cases are characterized as numerically stable and the overall results are impressive. One may perhaps argue that the computation of \( X_{e_4} \) when scaling is used is not accurate, because the errors are not so small as they are for the other solutions, as shown in table 7.2.2. This could be a valid statement if there was a big difference between the actual and the estimated backward error. On the contrary, this is not the case in this example, since the estimation of errors is very good and by no means pessimistic. However, the reader will discover, by performing the computations with smaller \( \epsilon_{tol}^{IVP} \), that all errors are reduced further and higher accuracy can be obtained.

Now in table 7.2.5, the values of the magnitude of homogeneous projective transformations and the Frobenious norms of the original equation's residual, are reported for the solutions \( X_{e_i} \in \mathbb{C}^{2 \times 2}, i = 5,...,16 \). Note that since the finite solutions are only the \( X_{e_i} \in \mathbb{C}^{2 \times 2}, i = 1,...,4 \), and have no multiplicities, the solutions \( X_{e_i} \in \mathbb{C}^{2 \times 2}, i = 5,...,16 \), should be at infinity. Hence, table 7.2.5 reveals the following ‘rule of thumb’ for judging solutions at infinity:

"A computed solution can be judged as being at infinity if with respect to this solution, the value of the magnitude of homogeneous projective transformation is reasonably small, and simultaneously the Frobenious norm of the equation's residual is reasonably large".

The above rule is implemented in GQMEHOM1.m in the GQME-Toolbox, via messages during the end of the computation of a solution (see appendix C).

### 7.3.2 Numerical Example 7.3

In this subsection a conjugate pair of solutions for an example taken from [80], is computed, using the fixed point homotopy in the GQME-Toolbox. The data of (7.4) for this example are:
Chapter 7 Numerical Solution & Analysis to Scientific Problems 276

Note that, the above example is related to the mechanical engineering problem of modeling the oscillations in an airplane wing (see [105]).

In terms of (2.1) and hence in GQME-Toolbox and in order to avoid pessimism when performing the numerical analysis of the computed solutions, (7.4) can be written as (2.1) with data:

\[ A_i := B, \quad A_2 := 0.6B, \quad B_1 := 0.4I_3, \quad B_2 := I_3, \quad C_1 := 0.4A, \quad C_2 := 0.6A \]

\[ D_1 := 10I_3, \quad D_2 := 5I_3, \quad E_1 := 0.1I_3, \quad E_2 := 0.2I_3, \quad G := C. \]

For this example, the complex conjugate pair of solutions in [105], is computed using the fixed point homotopy algorithm of GQME-Toolbox with \( \gamma = 1 + li \) and \( \gamma = -1 - li \). The solution of the homotopy equation is computed as \( X_c(\epsilon) \) or \( \overline{X_c}(\epsilon) \), when considering respectively no scaling or the scaling (3.31). Each case has been computed subject to two different values (one relatively large and one small) of the tolerance \( \varepsilon_{tol,IVP} \) of the IVP.

In all cases, a posteriori numerical analysis of the computed solution \( X_c \) has been conducted. Note that since exact solutions are not known, the actual (exact) backward error is absent from the numerical analysis.

The initial conditions for both setups with respect to \( \gamma = 1 + li \) and \( \gamma = -1 - li \) are chosen as the solution of the easy problem \( X(0) - 0_{3 \times 3} = 0_{3 \times 3} \); i.e. \( X(0) = 0_{3 \times 3} \) (zero initial conditions). From these initial conditions and for \( \gamma = 1 + li \) and \( \gamma = -1 - li \) the solutions obtained are:

(i) \( \gamma = 1 + li \): \( X_c = \text{Re} \left[ \begin{bmatrix} -3.6568e + 001 & 5.7955e - 003 & 5.0661e - 002 \\ 2.3728e - 001 & -5.8156e - 001 & 2.5516e - 001 \\ 1.0061e + 000 & -4.7679e - 002 & -7.6087e - 001 \end{bmatrix} \right] \)

\[ + \text{Im} \left[ \begin{bmatrix} -3.2084e + 000 & -1.9770e - 001 & 7.2923e - 001 \\ 2.0937e + 000 & -1.4042e + 000 & 2.2633e + 000 \\ 2.3554e + 000 & -1.0218e - 001 & -8.1123e + 000 \end{bmatrix} \right] \]
(ii) $y = 1 + li : \quad X_C = \text{Re} \begin{pmatrix}
-3.6568e-001 & 5.7955e-003 & 5.0661e-002 \\
2.3728e-001 & -5.8156e-001 & 2.5516e-001 \\
1.0061e+000 & -4.7679e-002 & -7.6087e-001
\end{pmatrix}
+ \text{Im} \begin{pmatrix}
3.2084e+000 & 1.9770e-001 & -7.2923e-001 \\
-2.0937e+000 & 1.4042e+000 & -2.2633e+000 \\
-2.3554e+000 & 1.0218e-001 & 8.1123e+000
\end{pmatrix}.

The eigenvalues of these solutions are:

Case (i) - $y = 1 + li :
-8.84836e-001 - 8.44156e+000i,
9.47226e-002 + 2.52296e+000i,
-9.18006e-001 + 1.76066e+000i.

Case (ii) - $y = 1 + li :
-8.84836e-001 + 8.44156e+000i,
9.47226e-002 + 2.52296e+000i,
-9.18006e-001 + 1.76066e+000i.

All the above eigenvalues are the eigenvalues of the quadratic eigenvalue problem (7.5).

The driver to run the present example in GQME-Toolbox is the m-file Example73.m, given in appendix D. The user has only to type Example73 and to press Enter in the MATLAB workspace where the GQME-Toolbox is installed.

The computation of the scaling problem (3.31) provides the following:

All cases took 0.05 $\text{sec}$ CPU time and 140437 number of FLOPS and

$$S(u_X, u_C) \big|_{(u_X, u_C)=(0,0)} = 9.2616e+002, \quad \min \left\{ S(u_X, u_C) \right\} = 6.3725e+002,$$

$$u_X = \begin{bmatrix}
-1.6174e-001 & -1.6174e-001 & -1.6174e-001 \\
1.7703e-001 & 1.7703e-001 & 1.7703e-001 \\
1.4393e-001 & 1.4393e-001 & 1.4393e-001
\end{bmatrix},$$

$$u_C = \begin{bmatrix}
-4.1077e-001 & -4.0632e-001 & -4.0590e-001 \\
3.4458e-001 & 3.4220e-001 & 3.4922e-001 \\
2.2491e-001 & 2.2776e-001 & 2.2428e-001
\end{bmatrix}.$$

The arithmetic used in the present example is the MATLAB fixed complex arithmetic and $\text{flh} = 0$. No homogeneous projective transformation used. No homotopy step corrector was posed (i.e. $\text{flagcor} = 1$) and the ‘end game’ tolerance was set to $\text{efinal} = 1e-20$ with the jacobian formula not provided in the ‘end game’ solver (i.e. $\text{jacobianqme}'=\text{'off'}$) under a large scale optimization method $\text{largescale}'=\text{'on'}$. The maximum allowed number of steps and the minimum allowed integration step of the IVP routine RK45GQMEHOM1.m.
Figure 7.10. Homotopy paths for example 7.3.
<table>
<thead>
<tr>
<th></th>
<th>$X$ ($\gamma = 1 + \text{i}$)</th>
<th>$X$ ($\gamma = -1 - \text{i}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\det(JX_c)$</td>
<td>$n.s.n.p$</td>
<td>$n.s.n.p$</td>
</tr>
<tr>
<td></td>
<td>$-3.8740 \times 10^9 + 1.1083 \times 10^2 \text{i}$</td>
<td>$-3.8740 \times 10^9 - 1.1083 \times 10^2 \text{i}$</td>
</tr>
<tr>
<td>$L(X_c)$</td>
<td>$n.s.n.p$</td>
<td>$n.s.n.p$</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$|\text{Residual}(X_c)|_F$</td>
<td>$n.s.n.p$</td>
<td>$n.s.n.p$</td>
</tr>
<tr>
<td></td>
<td>1.7205 $\times 10^{-7}$</td>
<td>1.8735 $\times 10^{-7}$</td>
</tr>
<tr>
<td>$|X_c|_F$</td>
<td>$s.n.p$</td>
<td>$s.n.p$</td>
</tr>
<tr>
<td></td>
<td>2.7716 $\times 10^{-7}$</td>
<td>2.8056 $\times 10^{-7}$</td>
</tr>
<tr>
<td>$C_n(X_c)$</td>
<td>$n.s.n.p$</td>
<td>$n.s.n.p$</td>
</tr>
<tr>
<td></td>
<td>5.3532 $\times 10^{-1}$</td>
<td>5.3532 $\times 10^{-1}$</td>
</tr>
<tr>
<td>$E_{\text{prf}_{\text{max}}}(X_c)$</td>
<td>$n.s.n.p$</td>
<td>$n.s.n.p$</td>
</tr>
<tr>
<td></td>
<td>4.6615 $\times 10^{-10}$</td>
<td>5.1403 $\times 10^{-10}$</td>
</tr>
<tr>
<td>$E_{\text{prf}_{\text{rmax}}}(X_c)$</td>
<td>$n.s.n.p$</td>
<td>$n.s.n.p$</td>
</tr>
<tr>
<td></td>
<td>6.8663 $\times 10^{-10}$</td>
<td>7.2659 $\times 10^{-10}$</td>
</tr>
<tr>
<td>$E_{\text{rb}}(X_c)$</td>
<td>$n.s.n.p$</td>
<td>$n.s.n.p$</td>
</tr>
<tr>
<td></td>
<td>1.6152 $\times 10^{-8}$</td>
<td>2.0214 $\times 10^{-8}$</td>
</tr>
<tr>
<td>$E_{\text{prb}_{\text{max}}}(X_c)$</td>
<td>$n.s.n.p$</td>
<td>$n.s.n.p$</td>
</tr>
<tr>
<td></td>
<td>2.4308 $\times 10^{-8}$</td>
<td>2.6754 $\times 10^{-8}$</td>
</tr>
<tr>
<td>$E_{\text{prb}_{\text{rmax}}}(X_c)$</td>
<td>$n.s.n.p$</td>
<td>$n.s.n.p$</td>
</tr>
<tr>
<td></td>
<td>2.0968 $\times 10^{-7}$</td>
<td>2.1880 $\times 10^{-7}$</td>
</tr>
<tr>
<td>$\text{NS}$</td>
<td>$n.s.n.p$</td>
<td>$n.s.n.p$</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$\text{NS}_{\text{fl}}$</td>
<td>$n.s.n.p$</td>
<td>$n.s.n.p$</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$\varepsilon_{\text{tol}_{\text{IVP}}}$</td>
<td>$n.s.n.p$</td>
<td>$n.s.n.p$</td>
</tr>
<tr>
<td></td>
<td>1.0000 $\times 10^{-6}$</td>
<td>1.0000 $\times 10^{-6}$</td>
</tr>
<tr>
<td>$n_{\text{steps}_{\text{IVP}}}$</td>
<td>$n.s.n.p$</td>
<td>$n.s.n.p$</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>29</td>
</tr>
<tr>
<td>$\min_{\text{step}_{\text{IVP}}}$</td>
<td>$n.s.n.p$</td>
<td>$n.s.n.p$</td>
</tr>
<tr>
<td></td>
<td>2.7449 $\times 10^{-3}$</td>
<td>3.2332 $\times 10^{-3}$</td>
</tr>
<tr>
<td>$t_{\text{CPU}_{\text{IVP}}}$</td>
<td>$n.s.n.p$</td>
<td>$n.s.n.p$</td>
</tr>
<tr>
<td></td>
<td>1.7800</td>
<td>1.7300</td>
</tr>
<tr>
<td>$n_{\text{FLOPS}_{\text{IVP}}}$</td>
<td>$n.s.n.p$</td>
<td>$n.s.n.p$</td>
</tr>
<tr>
<td></td>
<td>11288221</td>
<td>11314412</td>
</tr>
</tbody>
</table>

Table 7.3.1. Computation results for example 7.3.
<table>
<thead>
<tr>
<th></th>
<th>$X$</th>
<th>$X$ ($\gamma = 1 + li$)</th>
<th>$X$ ($\gamma = -1 - li$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\det(J_{X_c})$</td>
<td></td>
<td>$-3.8740e+009 +1.1083e+010i$</td>
<td>$-3.8740e+009 -1.1083e+010i$</td>
</tr>
<tr>
<td>$L(X_c)$</td>
<td></td>
<td>$1$</td>
<td>$1$</td>
</tr>
<tr>
<td>$|\text{Residual}(X_c)|_F$</td>
<td>n.s.n.p</td>
<td>$9.5691e-013$</td>
<td>$1.0448e-012$</td>
</tr>
<tr>
<td>$|X_c|_F$</td>
<td>s.n.p</td>
<td>$2.3767e-012$</td>
<td>$2.2400e-012$</td>
</tr>
<tr>
<td>$C_n(X_c)$</td>
<td></td>
<td>$5.3532e+001$</td>
<td>$5.3532e+001$</td>
</tr>
<tr>
<td>$E_{prf_{\max}}(X_c)$</td>
<td>n.s.n.p</td>
<td>$2.5883e-015$</td>
<td>$2.8713e-015$</td>
</tr>
<tr>
<td>$E_{prf_{\max}}(X_c)$</td>
<td>s.n.p</td>
<td>$5.8195e-015$</td>
<td>$5.5093e-015$</td>
</tr>
<tr>
<td>$E_{rb}(X_c)$</td>
<td></td>
<td>$7.4285e-014$</td>
<td>$1.0400e-013$</td>
</tr>
<tr>
<td>$E_{prb_{\max}}(X_c)$</td>
<td>n.s.n.p</td>
<td>$7.3848e-013$</td>
<td>$8.4834e-013$</td>
</tr>
<tr>
<td>$E_{prb_{\max}}(X_c)$</td>
<td>s.n.p</td>
<td>$1.7942e-012$</td>
<td>$1.7030e-012$</td>
</tr>
<tr>
<td>$NS$</td>
<td></td>
<td>$1$</td>
<td>$1$</td>
</tr>
<tr>
<td>$NS_{fl}$</td>
<td></td>
<td>$1$</td>
<td>$1$</td>
</tr>
<tr>
<td>$e_{tol_{IVP}}$</td>
<td></td>
<td>$1.0000e-012$</td>
<td>$1.0000e-012$</td>
</tr>
<tr>
<td>$n_{steps_{IVP}}$</td>
<td>n.s.n.p</td>
<td>$381$</td>
<td>$373$</td>
</tr>
<tr>
<td>min$<em>{step</em>{IVP}}$</td>
<td>s.n.p</td>
<td>$301$</td>
<td>$292$</td>
</tr>
<tr>
<td>$t_{CPU_{IVP}}$</td>
<td></td>
<td>$16.903e-004$</td>
<td>$1.9813e-004$</td>
</tr>
<tr>
<td>$t_{CPU_{IVP}}$</td>
<td>s.n.p</td>
<td>$4.7556e-004$</td>
<td>$5.6856e-004$</td>
</tr>
<tr>
<td>$n_{FLOPS_{IVP}}$</td>
<td></td>
<td>$22.1300$</td>
<td>$21.6600$</td>
</tr>
<tr>
<td>$n_{FLOPS_{IVP}}$</td>
<td>s.n.p</td>
<td>$17.6500$</td>
<td>$17.1400$</td>
</tr>
</tbody>
</table>

Table 7.3.2. Computation results for example 7.3.
were set to the MATLAB real maximum and minimum number values respectively (i.e. `maxnsteps=realmax, minstep=realmin`). Also, no discrete homotopy in the data was considered, thus `step=1` and obviously, no directionality rules used (i.e. `pdef=0`). The computational results are presented in figure 7.10 and in tables 7.3.1, 7.3.2. The rest notation is the same as in the previous examples.

From figures 7.10, it is clear that scaling reduces the length and changes the curvature of the homotopy paths by making them smoother. This improves the speed of the computations and it is more noticeable from table 7.3.2, when the demand in accuracy is high ($\text{etol}_{\text{IVP}} = 1.0000e-012$). In addition scaling also, preserves the numerical stability and accuracy of computations for all cases and overall is the best setup.

From tables 7.3.1 and 7.3.2, it is observed that increasing the demand for accuracy (i.e. larger $\text{etol}_{\text{IVP}}$) resulting an increase to the number of floating point operations and consequently an increase in computation times.

All, the above are in accordance with the theory developed in chapters 2-4 and generally validating the good performance and robustness of GQME-Toolbox.

### 7.4 Example from a study of a Chemical Kinetics Model

This section solves the numerical example from [121], which arises in the study of of a chemical kinetics model. This involves the numerical solution of a system of two algebraic scalar quadratic equations

$$a_{11}x_1^2 + a_{12}x_2^2 + a_{13}x_1x_2 + a_{14}x_1 + a_{15}x_2 + a_{16} = 0$$

and

$$a_{21}x_1^2 + a_{22}x_2^2 + a_{23}x_1x_2 + a_{24}x_1 + a_{25}x_2 + a_{26} = 0.$$  (7.9)

The system of (7.8) and (7.9) can be written as equation (2.1) with dimensions $n=2$, $p=1$, and with $\omega=2$, $\xi=2$, $X = \begin{bmatrix} x_{11} \\ x_{21} \end{bmatrix}$.

$$A_1 := \begin{bmatrix} a_{14} & 0 \\ a_{24} & 0 \end{bmatrix}, \quad A_2 := \begin{bmatrix} 0 & a_{15} \\ 0 & a_{25} \end{bmatrix}, \quad B_1 = E_1 = E_2 := 1, \quad C_1 := \begin{bmatrix} a_{11} & -a_{12} \\ a_{21} & -a_{22} \end{bmatrix},$$

$$C_2 := \begin{bmatrix} 0 & a_{11} + a_{12} + a_{13} \\ 0 & a_{21} + a_{22} + a_{23} \end{bmatrix}, \quad D_1 := [1 \ 0], \quad D_2 := [1 \ -1], \quad G := \begin{bmatrix} a_{16} \\ a_{26} \end{bmatrix}.$$  

Now, since in most of the cases the above matrices are different from each other apart from $B_1, E_1, E_2$ pessimism is less likely to occur in most of the cases when performing the numerical analysis to the computed solutions.
7.4.1 Numerical Example 7.4

For (7.8) and (7.9), consider the numerical values:

\[ a_{11} = -0.229242874000584 \times 10^{-3}, \quad a_{12} = 0.239288083622779 \times 10^{-4}, \]
\[ a_{13} = -0.273493390049709 \times 10^{-2}, \quad a_{14} = -0.553727730639579 \times 10^{-4}, \]
\[ a_{15} = 0.276995297192016 \times 10^{2}, \quad a_{16} = 0.1425211110826190 \times 10^{2}, \]
\[ a_{21} = -0.719432423708819 \times 10^{-4}, \quad a_{22} = 0.239288083622779 \times 10^{-4}, \]
\[ a_{23} = -0.273494828914557 \times 10^{-2}, \quad a_{24} = 0.55372773056810 \times 10^{4}, \]
\[ a_{25} = -0.276995297192090 \times 10^{7}, \quad a_{26} = 0.141753439263839 \times 10^{2}. \]

For this example, \( 2^2 = 4 \) solutions \( x_1, x_2, x_3, x_4 \in \mathbb{C} \) are computed via the polynomial homotopy algorithm of GQME-Toolbox with data \( y = 1 + li \).

The solutions have been computed considering homogeneous projective transformation with the random selected data (using MATLAB random generator \( \text{rand} \)) \( A_{da} = [3.6775e - 001 \ 6.2080e - 001] \) and \( a_{13} = 7.2155e - 001 \). Also, the solutions of the homotopy equation have been computed as \( \mathbf{X}_{i,c}(\epsilon) \), \( i = 1,2,3,4 \), when considering either scaling (3.31) or (3.63); and as \( \hat{\mathbf{X}}_{i,c}(\epsilon) \) or \( \tilde{\mathbf{X}}_{i,c}(\epsilon) \), \( i = 1,2,3,4 \), when considering respectively homogeneous projective transformation with no scalings or with either one of the scaling (3.31) or (3.63).

In all cases a posteriori numerical analysis of the computed solutions \( \mathbf{X}_{i,c} \), \( i = 1,2,3,4 \), have been conducted. The exact solutions are not known and hence the actual backward errors were not considered in the numerical analysis.

There are four different initial conditions \( \mathbf{X}(0) \in \mathbb{C}^2 \), which are determined from the solution of the easy problem \( \mathbf{X}(0) \circ \mathbf{X}(0) - (A_a \text{vec}(\mathbf{X}(0)) \circ a_{13} \mathbf{1} \times 008) = 0_{2 \times 1} \), according to subsection 3.4.2 in chapter 3, when no homogeneous projective transformation is used. These are computed as:

\[
\mathbf{X}_{c1}(0) = \begin{bmatrix} 7.2162e - 005 \\ 7.2162e - 005 \end{bmatrix}, \quad \mathbf{X}_{c2}(0) = \begin{bmatrix} -7.2157e - 005 \\ 7.2157e - 005 \end{bmatrix},
\]
\[
\mathbf{X}_{c3}(0) = \begin{bmatrix} 7.2153e - 005 \\ -7.2153e - 005 \end{bmatrix}, \quad \mathbf{X}_{c4}(0) = \begin{bmatrix} 7.2148e - 005 \\ 7.2148e - 005 \end{bmatrix}.
\]
Each one of \( X_{c1}(0), X_{c2}(0), X_{c3}(0), X_{c4}(0) \in \mathbb{C}^2 \) is expected to initiate a homotopy path that leads to \( X_{c1}, X_{c2}, X_{c3}, X_{c4} \in \mathbb{C}^2 \), respectively.

The driver to run the present example in GQME-Toolbox is the m-file `Example74.m`, given in appendix D. The user has only to type `Example74` and to press `Enter` in the MATLAB workspace where the GQME-Toolbox is installed.

The computations of the scaling problems (3.31) and (3.63) provide the following:

(i) Scaling (3.31): 0.02 (sec) CPU time and 3715 number of FLOPS, 

\[
S(u_X,u_C)\big|_{(u_X,u_C)=(0,0)} = 1.0379e+003, \quad \min\{S(u_X,u_C)\} = 7.2992e+002, \\
\]

\[
u_X = \begin{bmatrix} 3.6066e+000 \\ 5.1952e+000 \end{bmatrix}, \quad u_C = \begin{bmatrix} 5.0546e+000 \\ 4.9770e+000 \end{bmatrix}, 
\]

(ii) Scaling (3.63): 1.4618e+002 (sec) CPU time and 6577141 number of FLOPS, 

\[
S(u_X,u_C)\big|_{(u_X,u_C)=(0,0)} = 1.7638e+004, \quad \min\{S(u_X,u_C)\} = 1.9708e+004, \\
\]

\[
u_X = \begin{bmatrix} 3.6066e+000 \\ 5.1952e+000 \end{bmatrix}, \quad u_C = \begin{bmatrix} 5.0546e+000 \\ 4.9770e+000 \end{bmatrix}, 
\]

hence, no minimum obtained.

Note that both scaling problems (3.31) and (3.63) possess almost identical solutions and that in case (ii) there is no minimum solution.

At this point, it should be said that the arithmetic used in the present example is the MATLAB fixed complex arithmetic and \( \text{flh}=1 \). No homotopy step corrector was posed (i.e. \( \text{flagcor}=1 \)) and the ‘end game’ tolerance was set to \( \text{efinal}=le-50 \) with the Jacobian formula not provided in the ‘end game’ solver (i.e. \( \text{jacobiangqme='off'} \)) under a large scale optimization method \( \text{largescale='on'} \). The maximum allowed number of steps and the minimum allowed integration step of the IVP routine RK45GQMEHOM1.m were set to the MATLAB real maximum (i.e. \( \text{maxnsteps}=realmax \)) and \( \text{minstep}=le-60 \) respectively. Also, no discrete homotopy in the data was considered, thus \( \text{step}=1 \) and obviously, no directionality rules were used (i.e. \( \text{pdef}=0 \)). All computations were performed twice with \( \text{flpseudos}=1 \) with \( \text{pseudofin}=2 \) and without pseudo-arc length parameterization.
The computational results are presented in figures 7.11-7.13 and in tables 7.4.1-7.4.6. The notation used is the same as in the previous examples.

From figures 7.11-7.13, it can be observed that both scaling and homogeneous projective transformation change the length and the curvature of the homotopy paths, as in the previous examples.

From tables 7.4.1-7.4.6, and from the rule of thumb in subsection 7.3.1, solution $X_{e3}$ can be considered as being at infinity, while the rest of the solutions are finite.

Now, both scalings improve computation times with little degradation in accuracy. All cases reveal very good performance, numerical stability and robustness. An exemption can be considered the case of pseudo-arc parameterisation with no scaling for the computation of $X_{e3}$. The results of the last case are reported in table 7.4.2 with bold-faced numerals. In this case, there is a run time error since $X_{e3}$ is not computed at infinity but is computed identical to $X_{e1}$. In other words two different homotopy paths starting from different initial conditions converge to the same solution. According to the theory in chapters 2 and 3 this should not have happened. Thus as was mentioned in chapter 2 tracking the homotopy paths subject to the homotopy variable and not to the pseudo-arc length can be a more robust method. This contradicts the habit of many researchers who use pseudo-arc length parameterisation as a natural way of tracing homotopy paths (see [182], [183], [32], [33]).

One way of resolving the above problem is to reduce more $\epsilon_{tol/IVP}$ with the expense of increasing the computational burden. When scaling is used, the above problem is eliminated. Hence scaling is a useful way of avoiding run time errors and convergence to the same solution. Also, both scalings reduce the computational burden and overall are the best setup for the GQM EHOM1.m.

All the above are in accordance with the theoretical analysis in chapters 2-4 and vindicate the discussions made in those chapters.

### 7.5 Computation of Equilibrium Points of the Lorenz Attractor

The famous Lorenz attractor is a chaotic dynamical system of three quadratic ordinary differential equations [161]; namely

\[
\frac{d(x_1(t))}{dt} = -10x_1(t) + 10x_2(t) \tag{7.10}
\]

\[
\frac{d(x_2(t))}{dt} = 28x_1(t) - x_2(t) - x_1(t)x_3(t) \tag{7.11}
\]
Figure 7.11. Homotopy paths for example 7.4 (n.s. p.).
Figure 7.12. Homotopy paths for example 7.4 (s.p.).
Figure 7.13. Homotopy paths for example 7.4 (sv. p.).
<table>
<thead>
<tr>
<th>(X(n.s.p))</th>
<th>(X_1)</th>
<th>(X_2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\det(J_{X_c}))</td>
<td>(\varepsilon)</td>
<td>(9.7709e+006)</td>
</tr>
<tr>
<td></td>
<td>(s(\varepsilon))</td>
<td>(-8.9254e-002i)</td>
</tr>
<tr>
<td>(L(X_c))</td>
<td>(\varepsilon)</td>
<td>(-1.4602e-001)</td>
</tr>
<tr>
<td></td>
<td>(s(\varepsilon))</td>
<td>(+4.3767e-007i)</td>
</tr>
<tr>
<td>(|\text{Residual}(X_c)|_F)</td>
<td>(\varepsilon)</td>
<td>(2.9683e-008)</td>
</tr>
<tr>
<td></td>
<td>(s(\varepsilon))</td>
<td>(1.5663e-007)</td>
</tr>
<tr>
<td>(|X_c|_F)</td>
<td>(\varepsilon)</td>
<td>(6.2728e+003)</td>
</tr>
<tr>
<td></td>
<td>(s(\varepsilon))</td>
<td>(6.2728e+003)</td>
</tr>
</tbody>
</table>

Table 7.4.1. Computation results for example 7.4.
<table>
<thead>
<tr>
<th>$X(\text{n.s.p})$</th>
<th>$X_3$</th>
<th>$X_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\det(J_{X_c})$</td>
<td>$\varepsilon$</td>
<td>$4.4763e+017$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$-2.5131e+017i$</td>
</tr>
<tr>
<td>$L(X_c)$</td>
<td>$\varepsilon$</td>
<td>$-3.2911e-010$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$+1.6876e-010i$</td>
</tr>
<tr>
<td>$|Residual(X_c)|_F$</td>
<td>$\varepsilon$</td>
<td>$3.9173e+006$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$3.4998e-009$</td>
</tr>
<tr>
<td>$C_n(X_c)$</td>
<td>$\varepsilon$</td>
<td>$3.1029e+004$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$6.2728e+003$</td>
</tr>
<tr>
<td>$|X_c|_F$</td>
<td>$s(\varepsilon)$</td>
<td>$8.9163e-016$</td>
</tr>
<tr>
<td>$E_{\text{prf}}(X_c)$</td>
<td>$\varepsilon$</td>
<td>$3.2229e-005$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$8.9163e-016$</td>
</tr>
<tr>
<td>$E_{\text{prf}_{\text{max}}}(X_c)$</td>
<td>$\varepsilon$</td>
<td>$3.2229e-005$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$2.2285e-015$</td>
</tr>
<tr>
<td>$E_{\text{rb}}(X_c)$</td>
<td>$\varepsilon$</td>
<td>$4.5582e-005$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$6.3236e-013$</td>
</tr>
<tr>
<td>$E_{\text{prb}_{\text{max}}}(X_c)$</td>
<td>$\varepsilon$</td>
<td>$8.7252e-005$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$1.1047e-012$</td>
</tr>
<tr>
<td>$E_{\text{prb}<em>{\text{II}}</em>{\text{max}}}(X_c)$</td>
<td>$\varepsilon$</td>
<td>$8.7252e-005$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$2.7610e-012$</td>
</tr>
<tr>
<td>$\text{NS}$</td>
<td>$\varepsilon$</td>
<td>$1$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$1$</td>
</tr>
<tr>
<td>$\text{NS}_{\text{fl}}$</td>
<td>$\varepsilon$</td>
<td>$1$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$1$</td>
</tr>
<tr>
<td>$\epsilon_{\text{tolIVP}}$</td>
<td>$\varepsilon$</td>
<td>$1.0000e-006$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$1.0000e-006$</td>
</tr>
<tr>
<td>$n_{\text{stepsIVP}}$</td>
<td>$\varepsilon$</td>
<td>$60$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$57$</td>
</tr>
<tr>
<td>$\min_{\text{stepIVP}}$</td>
<td>$\varepsilon$</td>
<td>$5.6844e-011$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$1.0537e-004$</td>
</tr>
<tr>
<td>$t_{\text{CPUIVP}}$</td>
<td>$\varepsilon$</td>
<td>$3.5400$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$3.2200$</td>
</tr>
<tr>
<td>$n_{\text{FLOPSIVP}}$</td>
<td>$\varepsilon$</td>
<td>$932674$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$908312$</td>
</tr>
</tbody>
</table>

Table 7.4.2. Computation results for example 7.4.
Table 7.4.3. Computation results for example 7.4.
| TABLE 7.4.4 |
|---|---|---|
| $X(s,p)$ | $X_3$ | $X_4$ |
| $\det(J_{X_c})$ | $\epsilon$ | $\pm 5.1335e+028$ |
|  |  | $+4.6813e+027i$ |
|  |  | $-6.8245e-003i$ |
|  | $s(\epsilon)$ | $3.2442e+021$ |
|  |  | $-8.8964e+023i$ |
|  |  | $9.7709e+006$ |
|  |  | $-1.1335e-005i$ |
| $L(X_c)$ | $\epsilon$ | $2.3926e-012$ |
|  |  | $+5.2578e-011i$ |
|  |  | $+2.1426e-007i$ |
|  | $s(\epsilon)$ | $-8.9674e-009$ |
|  |  | $+8.9576e-009i$ |
|  |  | $-6.1064e-005$ |
| $\|\text{Residual}(X_c)\|_F$ | $\epsilon$ | $3.9173e+006$ |
|  |  | $4.4517e-012$ |
|  | $\|X_c\|_F$ | $s(\epsilon)$ | $3.9173e+006$ |
|  |  | $2.9458e-012$ |
| $C_n(X_c)$ | $\epsilon$ | $3.4178e+010$ |
|  |  | $4.7573e+000$ |
|  | $s(\epsilon)$ | $1.4186e+008$ |
|  |  | $6.2728e+003$ |
| $E_{pr_lmax}(X_c)$ | $\epsilon$ | $2.9259e-011$ |
|  |  | $2.3200e-021$ |
|  | $s(\epsilon)$ | $7.0490e-009$ |
|  |  | $7.5047e-019$ |
| $E_{pr_lmax}(X_c)$ | $\epsilon$ | $2.9261e-011$ |
|  |  | $1.3342e-015$ |
|  | $s(\epsilon)$ | $7.0490e-009$ |
|  |  | $1.3377e-015$ |
| $E_{rb}(X_c)$ | $\epsilon$ | $4.1378e-011$ |
|  |  | $6.5056e-016$ |
|  | $s(\epsilon)$ | $9.9688e-009$ |
|  |  | $5.3774e-016$ |
| $E_{pr_lmax}(X_c)$ | $\epsilon$ | $7.9207e-011$ |
|  |  | $1.0591e-015$ |
|  | $s(\epsilon)$ | $1.9082e-008$ |
|  |  | $9.2978e-016$ |
| $E_{pr_lmax}(X_c)$ | $\epsilon$ | $7.9213e-011$ |
|  |  | $6.0907e-010$ |
|  | $s(\epsilon)$ | $1.9082e-008$ |
|  |  | $1.6573e-012$ |
| $NS$ | $\epsilon$ | 1 |
|  |  | 1 |
|  | $s(\epsilon)$ | 1 |
|  |  | 1 |
| $NS_{fl}$ | $\epsilon$ | 1 |
|  |  | 1 |
|  | $s(\epsilon)$ | 1 |
|  |  | 1 |
| $e_{tol_{IVP}}$ | $\epsilon$ | $1.0000e-006$ |
|  |  | $1.0000e-006$ |
|  | $s(\epsilon)$ | $1.0000e-006$ |
|  |  | $1.0000e-006$ |
| $n_{steps_{IVP}}$ | $\epsilon$ | 61 |
|  |  | 80 |
|  | $s(\epsilon)$ | 53 |
|  |  | 67 |
| $min_{step_{IVP}}$ | $\epsilon$ | $3.8966e-011$ |
|  |  | $3.7723e-011$ |
|  | $s(\epsilon)$ | $1.0528e-004$ |
|  |  | $1.0532e-004$ |
| $CPU_{IVP}$ | $\epsilon$ | 3.4400 |
|  |  | 4.4300 |
|  | $s(\epsilon)$ | 3.0400 |
|  |  | 3.7400 |
| $n_{FLOPS_{IVP}}$ | $\epsilon$ | 944494 |
|  |  | 1176891 |
|  | $s(\epsilon)$ | 880412 |
|  |  | 1072161 |

Table 7.4.4. Computation results for example 7.4.
<table>
<thead>
<tr>
<th>$X_{sv,p}$</th>
<th>$X_1$</th>
<th>$X_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\det(J_{Xc})$</td>
<td>$\varepsilon$</td>
<td>9.7709e + 006</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+1.1782e + 000i</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>9.7709e + 006</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+1.2425e - 005i</td>
</tr>
<tr>
<td>$L_{Xc}$</td>
<td>$\varepsilon$</td>
<td>7.2261e - 001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-2.8987e - 009i</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>7.2261e - 001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-4.5270e - 006i</td>
</tr>
<tr>
<td>$|Residual(Xc)|_F$</td>
<td>$\varepsilon$</td>
<td>1.3110e - 006</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>9.0277e - 011</td>
</tr>
<tr>
<td>$|Xc|_F$</td>
<td>$\varepsilon$</td>
<td>6.2728e + 003</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>6.2728e + 003</td>
</tr>
<tr>
<td>$C_n(Xc)$</td>
<td>$\varepsilon$</td>
<td>3.2714e - 013</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>2.2998e - 017</td>
</tr>
<tr>
<td>$\varepsilon_{prf_{\text{max}}}(Xc)$</td>
<td>$\varepsilon$</td>
<td>3.2848e - 013</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>2.3031e - 017</td>
</tr>
<tr>
<td>$\varepsilon_{prf_{\text{max}}}(Xc)$</td>
<td>$\varepsilon$</td>
<td>1.3599e - 015</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>1.3401e - 015</td>
</tr>
<tr>
<td>$\varepsilon_{rb}(Xc)$</td>
<td>$\varepsilon$</td>
<td>2.4418e - 010</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>1.6384e - 014</td>
</tr>
<tr>
<td>$\varepsilon_{prb_{\text{max}}}(Xc)$</td>
<td>$\varepsilon$</td>
<td>4.0531e - 010</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>2.8493e - 014</td>
</tr>
<tr>
<td>$\varepsilon_{prb_{\text{max}}}(Xc)$</td>
<td>$\varepsilon$</td>
<td>4.0697e - 010</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>2.8521e - 014</td>
</tr>
<tr>
<td>$\alpha_{\text{NS}}$</td>
<td>$\varepsilon$</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>1</td>
</tr>
<tr>
<td>$\alpha_{\text{NS}}$</td>
<td>$\varepsilon$</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>1</td>
</tr>
<tr>
<td>$\alpha_{\text{etol_{IVP}}}$</td>
<td>$\varepsilon$</td>
<td>1.0000e - 006</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>1.0000e - 006</td>
</tr>
<tr>
<td>$\alpha_{n_{\text{steps_{IVP}}}}$</td>
<td>$\varepsilon$</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>23</td>
</tr>
<tr>
<td>$\alpha_{\text{min_{step_{IVP}}}}$</td>
<td>$\varepsilon$</td>
<td>8.5693e - 011</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>2.1464e - 005</td>
</tr>
<tr>
<td>$\alpha_{\text{CPU_{IVP}}}$</td>
<td>$\varepsilon$</td>
<td>1.9200</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>1.4100</td>
</tr>
<tr>
<td>$\alpha_{n_{\text{FLOPS_{IVP}}}}$</td>
<td>$\varepsilon$</td>
<td>524684</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>401632</td>
</tr>
</tbody>
</table>

Table 7.4.5. Computation results for example 7.4.
### Table 7.4.6

Computation results for example 7.4.

<table>
<thead>
<tr>
<th></th>
<th>$X_{3}$</th>
<th>$X_{4}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$det(J_{X_C})$</td>
<td>$\varepsilon$</td>
<td>$-5.1335e+028$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$3.2442e+021$</td>
</tr>
<tr>
<td></td>
<td>$L(X_C)$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$-8.9749e-009$</td>
</tr>
<tr>
<td></td>
<td>$|\text{Residual}(X_C)|_F$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td></td>
<td>$|X_C|_F$</td>
<td>$s(\varepsilon)$</td>
</tr>
<tr>
<td></td>
<td>$C_{n}(X_C)$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$1.4186e+008$</td>
</tr>
<tr>
<td></td>
<td>$E_{prf_{\text{max}}}(X_C)$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$7.0490e-009$</td>
</tr>
<tr>
<td></td>
<td>$E_{prf_{\text{max}}}(X_C)$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$7.0490e-009$</td>
</tr>
<tr>
<td></td>
<td>$E_{rb}(X_C)$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$9.9688e-009$</td>
</tr>
<tr>
<td></td>
<td>$E_{prb_{\text{max}}}(X_C)$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$1.9082e-008$</td>
</tr>
<tr>
<td></td>
<td>$E_{prb_{\text{max}}}(X_C)$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$1.9082e-008$</td>
</tr>
<tr>
<td></td>
<td>$NS$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$1$</td>
</tr>
<tr>
<td></td>
<td>$NS_{fl}$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$1$</td>
</tr>
<tr>
<td></td>
<td>$\epsilon_{\text{tol IVP}}$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$1.0000e-006$</td>
</tr>
<tr>
<td></td>
<td>$n_{\text{steps IVP}}$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$53$</td>
</tr>
<tr>
<td></td>
<td>$\text{min}_{\text{step IVP}}$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$1.0528e-004$</td>
</tr>
<tr>
<td></td>
<td>$t_{\text{CPU IVP}}$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$3.0600$</td>
</tr>
<tr>
<td></td>
<td>$n_{\text{FLOPS IVP}}$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td></td>
<td>$s(\varepsilon)$</td>
<td>$880412$</td>
</tr>
</tbody>
</table>
\[
\frac{d(x_2(t))}{dt} = -\frac{8}{3}x_3(t) + x_1(t)x_2(t)
\] (7.12)

The above system is a classical system in the study of chaos and many physical and mechanical systems can be modelled as the above structure with perhaps different coefficients (see [161] and its references).

### 7.5.1 Numerical Example 7.5

Consider equation (2.1) with \( \omega = \xi = 2 \), \( n = 3 \), \( p = 1 \) and

\[
A_1 := \begin{bmatrix} -4 & 4 & 0 \\ 112 & -0.4 & 0 \\ 0 & 0 & 3.2 \end{bmatrix}, \quad A_2 := \begin{bmatrix} -10 & 10 & 0 \\ 28 & -10 & 0 \\ 0 & 0 & -8 \end{bmatrix}, \quad B_1 := 1, \quad B_2 := 0.6, 
\]

\[
C_1 := \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad C_2 := \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \quad D_1 := \begin{bmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix}, \quad D_2 := \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}
\]

\[
E_1 = E_2 := 1, \quad G := 0_{3 \times 1}, \quad \text{where } X \in \mathbb{C}^{3 \times 1}.
\]

Setting \( X = \begin{bmatrix} x_{11} \\ x_{21} \\ x_{31} \end{bmatrix} := \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} \), it is easy to verify that the system of (7.10)-(7.12), can be written equivalently as

\[
\frac{dX}{dt} = \sum_{i=1}^{2} A_i X B_i + \sum_{i=1}^{2} C_i X D_i X E_i + G.
\] (7.13)

An important computational task in nonlinear dynamical systems as above is the computation of equilibrium points, since these determine the steady states of the system. Now, this is equivalent to finding the solutions to \( \sum_{i=1}^{2} A_i X B_i + \sum_{i=1}^{2} C_i X D_i X E_i + G = 0 \) and therefore the GQME-Toolbox can be used. Note that in general the GQME-Toolbox can be used for the calculation of the equilibrium points of any quadratic time invariant dynamical system.

Note that, because of the chaotic nature of many systems as the above, techniques of computing the equilibrium points based on flow methods similar to the Davidon method in chapter 6 can not applied with confidence.

For the present example, it was found that the repeating matrices \( B_1 = E_1 = E_2 \), do not cause pessimism during the numerical analysis of the computed solutions. There are at least \( 2^3 = 8 \) solutions \( X_i \in \mathbb{C}^3 \), \( i = 1, \ldots, 8 \) which are computed as \( X_{\epsilon_i} \in \mathbb{C}^3 \) via the polynomial
homotopy algorithm of GQME-Toolbox with data \( \gamma = 1 + li \). Now, there are three finite solutions all real which are denoted without loss of generality \( X_1, X_2, X_3 \in \mathbb{R}^3 \) as the exact solutions and \( X_{c1}, X_{c2}, X_{c3} \in \mathbb{C}^3 \) as the respective computed ones.

The solutions have been computed considering homogeneous projective transformation with the random selected data (using MATLAB random generator \( \text{rand} \))

\[
A_2 := \begin{bmatrix} 2.6775 \times 10^{-1} & 3.2080 \times 10^{-1} & 9.2670 \times 10^{-1} \\
1 & 0 & 0 \\
0 & 1 & 0 
\end{bmatrix}
\]

and \( a_{13} := 5.4000 \times 10^{-1} \). The solutions of the homotopy equation have been computed as \( \tilde{X}_{i\epsilon}(\epsilon) \) or \( \tilde{X}_{i\epsilon}(\epsilon) \), \( i = 1, \ldots, 8 \), when considering respectively homogeneous projective transformation with no scaling or with the scaling (3.31).

In all cases, a posteriori numerical analysis of the finite computed solutions \( X_{i\epsilon}, \)

\( i = 1, \ldots, 3 \), have been conducted. The other solutions \( X_{c4}, \ldots, X_{c8} \in \mathbb{C}^3 \) are found to be at infinity and corresponding to these solutions only the homogeneous projective transformations and the Frobenious norm of the residual of the right hand side of (7.13) are reported.

Now, the exact finite solutions \( X_i, \ i = 1, 2, 3, 4 \), have been computed via symbolic computation in MATLAB. These are converted to float number arithmetic and shown below up to four decimal digits.

\[
X_1 = \begin{bmatrix} 8.4853 \times 10^0 \\
8.4853 \times 10^0 \\
2.7000 \times 10^1 
\end{bmatrix},
X_2 = \begin{bmatrix} -8.4853 \times 10^0 \\
-8.4853 \times 10^0 \\
2.7000 \times 10^1 
\end{bmatrix},
X_3 = 0_{3 \times 1}
\]

There are eight different initial conditions \( X(0) \in \mathbb{C}^3 \), which are determined from the solution of the easy problem \( X(0) \cdot X(0) - (A_2 \cdot X(0)) + a_{14}^2 U_{3 \times 1} = 0_{3 \times 1} \), according to subsection 3.4.2 in chapter 3. From these initial conditions the ones that lead to the finite solutions \( X_{c1}, X_{c2}, X_{c3} \in \mathbb{C}^3 \) are respectively:

\[
X_{c1}(0) = \begin{bmatrix} 1.0480 \times 10^0 \\
1.0480 \times 10^0 \\
1.0480 \times 10^0 
\end{bmatrix},
X_{c2}(0) = \begin{bmatrix} -2.6667 \times 10^1 \\
2.6667 \times 10^1 \\
2.6667 \times 10^1 
\end{bmatrix},
X_{c3}(0) = \begin{bmatrix} -8.1589 \times 10^{-1} \\
-8.1589 \times 10^{-1} \\
8.1589 \times 10^{-1} 
\end{bmatrix}
\]

Each one of \( X_{c1}(0), X_{c2}(0), X_{c3}(0) \in \mathbb{C}^3 \) is expected to initiate a homotopy path that leads to \( X_{c1}, X_{c2}, X_{c3} \in \mathbb{C}^3 \), respectively.

The driver to run the present example in GQME-Toolbox is the m-file \textit{Example75.m}, given in appendix D. The user has only to type \textit{Example75} and to press \textit{Enter} in the MATLAB workspace where the GQME-Toolbox is installed.
The computation of the scaling problem (3.31) provides the following:

All cases took 0.03 (sec) CPU time and 16077 number of FLOPS and

\[ S(u_X, u_C)(u_X, u_C) = 4.9851e + 000, \quad \min_{(u_X, u_C)} S(u_X, u_C) = 3.8638e + 000, \]

\[ u_X = [1.5699e - 001, 2.2590e - 001, 2.4376e - 001]^T, \]

\[ u_C = [-4.7132e - 001, 4.2525e - 001, 3.5673e - 001]^T. \]

The arithmetic used in the present example is the MATLAB fixed complex arithmetic and \( flh = 1 \). No homotopy step corrector was posed (i.e. flagcor = 1) and the ‘end game’ tolerance was set to \( efinal = 1e - 20 \) with the jacobian formula not provided in the ‘end game’ solver (i.e. jacobiang = ‘off’) under a large scale optimization method largescale = ‘on’. The maximum allowed number of steps and the minimum allowed integration step of the IVP routine RK45GQMEHOM1.m were set to the MATLAB real maximum and minimum number values respectively (i.e. maxnsteps = realmax, minstep = realmin). Also, no discrete homotopy in the data was considered, thus step = 1 and obviously, no directionality rules used (i.e. pdef = 0).

The computational results are presented in figures 7.14, 7.15 and in tables 7.5.1-7.5.2. The notation is the same as in the previous examples.

From figures 7.14 and 7.15, scaling does not by affect much the curvature and the length of the homotopy paths. This is something that might be expected, since the scaling does not change by much the original equation. Also note that, \( \min_{(u_X, u_C)} S(u_X, u_C) \) is close to \( S(u_X, u_C)(u_X, u_C) = (0, 0) \):

Now, from table 7.5.1 it is apparent that the results characterize a good computation in terms of performance, accuracy and numerical stability. Accuracy can be further improved using smaller \( etol_{IVP} \) at the expense of an increased computational burden and hence computation times. In these results, \( X_3 \) in table 7.5.1 may cause confusion, because of the big error values. Note however that this is due to the fact that \( X_{c3} \) tends to zero (since it is the computed zero solution \( X_3 \)) and the analysis in chapter 4 assumes the solution to be different from zero.

Finally, note that, from table 7.5.2 and the rule of thumb in subsection 7.3.1, the solution \( X_{ci} \), \( i = 4,..,8 \) can be considered as being at infinity.
Figure 7.14. Homotopy paths for example 7.5 (n.s.p).
Figure 7.15. Homotopy paths for example 7.5 (s.p.).
Table 7.5.1. Computation results for example 7.5.
### Table 7.5.2

<table>
<thead>
<tr>
<th>$X$</th>
<th>$L(X_c)$</th>
<th>$|\text{Residual}(X_c)|_F$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_4$ n.s.p</td>
<td>$1.2962e^{-009}$</td>
<td>$1.5559e+010$</td>
</tr>
<tr>
<td>s.p</td>
<td>$2.3118e^{-009}$</td>
<td>$1.2523e+010$</td>
</tr>
<tr>
<td>$X_5$ n.s.p</td>
<td>$4.2525e^{-012}$</td>
<td>$8.3762e+014$</td>
</tr>
<tr>
<td>s.p</td>
<td>$3.7393e^{-009}$</td>
<td>$1.6149e+009$</td>
</tr>
<tr>
<td>$X_6$ n.s.p</td>
<td>$2.8623e^{-009}$</td>
<td>$5.5500e+008$</td>
</tr>
<tr>
<td>s.p</td>
<td>$2.9832e^{-009}$</td>
<td>$1.8800e+009$</td>
</tr>
<tr>
<td>$X_7$ n.s.p</td>
<td>$1.5842e^{-009}$</td>
<td>$1.8638e+009$</td>
</tr>
<tr>
<td>s.p</td>
<td>$5.7544e^{-012}$</td>
<td>$4.0228e+014$</td>
</tr>
<tr>
<td>$X_8$ n.s.p</td>
<td>$1.7152e^{-009}$</td>
<td>$9.1543e+008$</td>
</tr>
<tr>
<td>s.p</td>
<td>$1.7248e^{-009}$</td>
<td>$1.6624e+009$</td>
</tr>
</tbody>
</table>

*Table 7.5.2. Computation results for example 7.5.*

### 7.6 Synopsis

In this chapter, various scientific problems have been numerically solved and analyzed under a common framework using the GQME-Toolbox. In general, the results indicated very good performance, accuracy and numerical stability of GQME-Toolbox for this class of problems. In all cases, the theory and discussions of chapters 2-4 have been vindicated by many points.

The reader can perform more experiments using the drivers of the examples given in appendix D, by modifying the setup of each problem.
8.1 Introduction

In this chapter, it will be shown how (2.1) can be used in the design of robust control systems. More specifically, the design of fixed order $H_2$ and anti-windup control systems subject to saturated actuators will be addressed.

The proposed control system design methodology constitutes the combination of different control system design methods under a practical optimization based framework. The different methods that are combined are the:

- Anti-windup control systems subject to independent saturated control inputs [174].
- Observer based anti-windup control systems subject to radial saturated control inputs [172].
- Optimal pole placement for state feedback systems possessing integrity [153].
- Optimal pole placement method of [13].
- Loop-transfer recovery method of [45], [46].

The proposed methodology will be applied to two industrial systems, both of which require fast response and robust control systems.
The first case study is in the area of modern electro-technological processing of materials. More specifically the automatic control of the power of a laser, which is used in welding, cutting and surface treatment. Since the research for this kind of systems is relatively recent, only a few control techniques have been applied so far (most of them based on classical PID controller design [108]). Note that the techniques so far proposed can not easily cope with saturated actuators and meet specific design specifications.

The second case study is taken from the area of electrical power systems. More specifically it considers the design of a fast and robust control system for a single synchronous electric generator connected to an infinite bus.

This chapter is organized as follows. In section 8.2, an introduction to the actuator saturation problem will be presented together with a proposed technique for preventing saturation of stable linear actuators. Next, section 8.3 presents the proposed design of the control system. Section 8.4, illustrates how the proposed control system design can be numerically solved using the GQME-Toolbox (see appendix C). In section 8.5, an introduction is given to the fast modern electro-technological manufacturing process of the present laser system. In the same section, the system’s design specifications are posed and the controller is computed. The numerical analysis of the computations and the simulation of the overall system are reported at the end of section 8.5 together with general conclusions. The methodology is similarly applied to the electrical power system in section 8.5. Finally, general conclusions are given in section 8.7.

8.2 The Actuator Saturation Problem and Possible Solutions

All control actuation devices are subject to amplitude saturation, since they are technologically and therefore practically energy limited. Therefore, in any practical system, unexpectedly large amplitudes of disturbances can cause saturation of the actuators and push the system to operate in a mode for which it was not originally designed. In some cases, the system is able to recover from the saturation event [167], but when it can not then the performance may be bad or even catastrophic. Some of the saturation symptoms are large steady state errors, limit cycles, a catastrophe of system robustness and performance leading to instability [12]. All these constrain the ability of a system to respond appropriately to commands and disturbances. The saturation problem is more difficult when the open loop system is unstable or when the closed-loop system contains unstable controllers which can cause windup [12].

A general policy for a good linear design is the use of an economic control action, which does not push the actuator into saturation. This is one of the main considerations when selecting penalty weights in modern [7] and post-modern [188], [155] control techniques such
as $LQG$, $LQR$, $H_2$, $H_\infty$, etc. Very often, in order to improve system performance, large penalties on the states result in large control signal inputs, and hence actuator saturation. This is why the compromise between the weighting of control inputs and states and between system performance and robustness is usual a difficult task during the design.

Another design policy is to modify the reference signal (when dealing with tracking problems) so that saturation does not occur. However, even when a designer manages to design a system that normally will not saturate (according to the above considerations for example), unexpected large amplitude or frequent disturbances, that were impossible to take into account in the design, may occur and push the actuators of the system into saturation. Hence the need for controllers that can somehow eliminate or reduce the effects of actuator saturation.

There are two major philosophies of designing controllers for systems with saturating actuators. The first philosophy [101] is to design a linear controller, assuming no presence of saturation, and then to modify the closed-loop system in order to counteract possible saturation events. This modification usually consists of augmenting additional parts to the linear controller that they will only be activated in the event of saturation with the mission to compensate the saturation. In this respect, classical anti-windup as proposed many years ago and has been unified and reformulated in a modern manner over the last three years.

The other philosophy [172] is to design a controller that takes the actuator saturation event as a priori part of the design. Recent progress on the saturation problem together with progress in the area of nonlinear systems design, has lead to the synthesis of nonlinear multivariable design techniques that account a priori for the presence of saturation nonlinearities in contrast with the classical anti-windup designs. For example, the modern anti-windup design of [172] could be characterised as an extension of the classical anti-windup approach [12], [101] to the modern nonlinear designs of [69], [70], [95].

It is evident that actuator saturation has become a major problem of interest to the control community over the last decade, and is still under investigation. A large number of relevant papers have been published and a good list of references that contains most of them is found in [22], [101].

So far the majority of design methods do not take into account constraints on an actuator's states and rate of states and do not view the actuator as a dynamical system. But for the problem of control signal amplitude saturation, state of the art designs have been developed in a series of papers [22], [101].

When unrealistic actuator models are used in the design, big problems may occur. Of particular concern are non modelled limitations due to actuator constraints on its states and rate of states which may be vital to a system’s response.

The design framework presented below considers the problem of actuator saturation, with constraints on the actuators’ dynamical system; these are constraints on the actuators’
states, rate of states, outputs and also constraints on the plant (not actuator) states which are present in the actuator’s dynamical model.

The crux of the design is to a priori estimate the limits of the actuators’ inputs such that saturation is not taking place on the actuators’ states, their rates and outputs. Hence, the problem can be transformed as an amplitude signal saturation problem. The technique in limiting the actuators’ inputs (i.e. controller outputs) can be applied for both continuous and discrete time systems. This is based on a practical optimization procedure and is presented in the next subsection.

8.2.1 Preventing Saturation in Asymptotically Stable Plants via Limited Inputs

Consider the linear time invariant dynamical system

\[
\frac{dx_a(t)}{dt} = A_a x_a(t) + B_a u(t) + \Gamma \alpha_1 w_{a1}(t), \quad x_a(0) = x_{a0}, \quad u(0) = u_0, \quad t \in \mathbb{R}_+, \quad (8.1)
\]

\[
y_a(t) = C_a x_a(t) + D_a u(t) + \Gamma \alpha_2 w_{a2}(t), \quad (8.2)
\]

where \(x_a(t) \in \mathbb{R}^{n_a}, \quad u(t) \in \mathbb{R}^m, \quad y_a(t) \in \mathbb{R}^{n_a}, \quad w_{a1}(t) \in \mathbb{R}^{l_a}, \quad w_{a2}(t) \in \mathbb{R}^{q_a}\), are defined as \(n_a, m, l_a, q_a \in \mathbb{Z}_+\). Now, \(A_a \in \mathbb{R}^{n_a \times n_a}, \quad B_a \in \mathbb{R}^{n_a \times m}, \quad \alpha_1 \in \mathbb{R}^{n_a \times l_a}, \quad C_a \in \mathbb{R}^{P_a \times n_a}\), and

\(D_a \in \mathbb{R}^{P_a \times m}, \quad \alpha_2 \in \mathbb{R}^{P_a \times q_a}\) are the corresponding constant matrix coefficients of the above vectors and \(t\) is the time variable.

For the above system assume that all eigenvalues of \(A_a \in \mathbb{R}^{n_a \times n_a}\) have strictly negative real parts (i.e. \(\text{Re}(\lambda_i(A_a)) < 0, \quad \forall i = 1, \ldots, n\)), and that \(\forall t \in \mathbb{R}_+ \cup \{0\}, \quad |\Gamma \alpha_i w_{ai}(t)| \leq \bar{w}_{ai} \geq 0, \quad \forall i = 1, 2\) and \(|x_{a0}| \leq \bar{x}_a\). Also, let \(\bar{x}_a, \bar{y}_a \in \mathbb{R}_+^{n_a}\) and \(\bar{y}_a \in \mathbb{R}_+^{P_a}\) define the limits (i.e. saturation) of the system’s state, rate of state and output. Then the problem of limiting \(u(t) \in \mathbb{R}^m\) such that \(|x_a(t)| \leq \bar{x}_a, \quad |y_a(t)| \leq \bar{y}_a, \quad \left| \frac{dx_a(t)}{dt} \right| \leq \bar{x}_a\), can formulated as

**Problem 8.1:** Let system \((8.1), (8.2)\) be controllable (i.e. \(\text{rank} \left[ A_a - v I_{n_a} \quad B_a \right] = n_a\), \(\forall v \in \mathbb{C}\) with \(\text{Re}(\lambda_i(A_a)) < 0, \quad \forall i = 1, \ldots, n\) and \(|\Gamma \alpha_i w_{ai}(t)| \leq \bar{w}_{ai} \geq 0, \quad \forall i = 1, 2, \quad \forall t \in \mathbb{R}_+ \cup \{0\}\). Given \(\bar{x}_a, \bar{y}_a \in \mathbb{R}_+^{n_a}, \quad |x_{a0}| \leq \bar{x}_a, \quad \bar{y}_a \in \mathbb{R}_+^{P_a}, \quad n_a, m, l_a, q_a \in \mathbb{Z}_+\), find
\[
\bar{u} := \max_{t \in \mathbb{R}_+ \cup \{0\}} \left| \left( u(t) \right) \right| \text{ such that } \left| x_a(t) \right| \leq \bar{x}_a \quad \text{and} \quad \left| y_a(t) \right| \leq \bar{y}_a \quad \text{and} \quad \left| \frac{dx_a(t)}{dt} \right| \leq \bar{x}_a, \quad \forall t \in \mathbb{R}_+ \cup \{0\}.
\]

**Approximate solution.**

In order to produce an ‘approximate’ solution for problem 8.1, the analysis/synthesis procedure (well known in designs in Euclidean geometry, [52]) is applied.

a) **Analysis:**

System (8.1), (8.2) can be approximated by a linear discrete time system with sampling time \( t_s \in \mathbb{R}_+ \) reasonable small according to Nyquist theorem [104]; namely

\[
x_a(k + 1) = A_{ad} x_a(k) + B_{ad} u(k) + \Gamma \alpha_{d1} w_{a1}(k), \quad x_a(0) = x_{a0}, \quad u(0) = u_0, \quad k \in \mathbb{Z}_+ \cup \{0\}
\]

\[
y_a(k) = C_{ad} x_a(k) + D_{ad} u(k) + \Gamma \alpha_{d2} w_{a2}(k),
\]

where \( x_a(k) \in \mathbb{R}^{n_a}, \; u(k) \in \mathbb{R}^m, \; y_a(k) \in \mathbb{R}^{p_a}, \; w_{a1}(k) \in \mathbb{R}^{l_a}, \; w_{a2}(k) \in \mathbb{R}^{q_a} \), are defined as the systems’ state, input, output, state disturbance and output disturbance respectively and \( n_a, m, p_a, l_a, q_a \in \mathbb{Z}_+ \) as before. Now, \( A_{ad} \in \mathbb{R}^{n_a \times n_a}, \; B_{ad} \in \mathbb{R}^{n_a \times m}, \; \Gamma \alpha_{d1} \in \mathbb{R}^{n_a \times l_a}, \; C_{ad} \in \mathbb{R}^{p_a \times n_a}, \; D_{ad} \in \mathbb{R}^{p_a \times m}, \; \Gamma \alpha_{d2} \in \mathbb{R}^{p_a \times q_a} \) are the corresponding constant matrix coefficients of the above vectors in the discrete time domain \( k \in \mathbb{Z}_+ \cup \{0\} \). These are:

\[
A_{ad} := e^{A_{ad} t_s}, \quad \Gamma \alpha_{d1} := \left( \int_0^{t_s} \left( e^{A_{ad} s} \right) ds \right) \Gamma \alpha_1, \quad C_{ad} := C_a, \quad D_{ad} := D_a \quad \text{and} \quad \Gamma \alpha_{d2} := \Gamma \alpha_2.
\]

For the system (8.3), (8.4) the corresponding assumptions stated previously for (8.1), (8.2) obviously turn out to be

\[
\left| x_i (A_{ad}) \right| \leq 1, \; \forall i = 1, \ldots, n, \; \left| \Gamma \alpha_{d1} w_{a1}(k) \right| \leq \bar{w}_{a1} \geq 0, \; \forall i = 1, 2, \; \forall k \in \mathbb{Z}_+ \cup \{0\} \quad \text{and} \quad \left| x_{a0} \right| \leq \bar{x}_a. \quad \text{In terms of the limits on the state, rate of state and output of (8.3), (8.4) they are:} \quad \left| x_a(k) \right| \leq \bar{x}_a, \quad \left| y_a(k) \right| \leq \bar{y}_a, \quad \left| \Gamma \alpha_{d2} w_{a2}(k) \right| \leq \bar{w}_{a2}, \quad \forall k \in \mathbb{Z}_+ \cup \{0\}, \quad \text{where} \quad \bar{x}_a(k) := \frac{1}{t_s}(x_a(k+1) - x_a(k)), \quad \left| x_{a0} \right| \leq \bar{x}_a.
\]

(8.3) and (8.4) can equivalently be written by induction \( \forall k \in \mathbb{Z}_+ \) as
\[ x_a(k) = A_{ad}^k x_{a0} + B_{ad} u(k-1) + \Gamma \alpha_{d1} w_{a1}(k-1) + \sum_{i=1}^{k-1} \left( A_{ad}^i B_{ad} u(k-i-1) \right) \]

\[ + \sum_{i=1}^{k-1} \left( A_{ad}^i \Gamma \alpha_{d1} w_{a1}(k-i-1) \right) \]

(8.5)

\[ y_a(k) = C_{ad} A_{ad}^k x_o + (C_{ad} B_{ad} + D_{ad}) u(k-1) + C_{ad} \Gamma \alpha_{d1} w_{a1}(k-1) + \Gamma \alpha_{d2} w_{a2}(k-1) \]

\[ + C_{ad} \sum_{i=1}^{k-1} \left( A_{ad}^i B_{ad} u(k-i-1) \right) + C_{ad} \sum_{i=1}^{k-1} \left( A_{ad}^i \Gamma \alpha_{d1} w_{a1}(k-i-1) \right) . \]

(8.6)

where \( u(i):= 0 \), \( w_{a1}(i):= 0 \), \( w_{a2}(i):= 0 \), \( \forall i < 0 \).

Moreover, from (8.3), (8.5) and \( \| \Gamma \alpha_{d1} w_{a1}(k) \| \leq \overline{w}_{a1} \), \( \forall i = 1, 2 \) we have

\[ \hat{x}_a(k) = \frac{1}{t_s} \left( x_a(k+1) - x_a(k) \right) = \frac{1}{t_s} \left( A_{ad} x(k) + B_{ad} u(k) + \Gamma \alpha_{d1} w_{a1}(k) - x_a(k) \right) \]

\[ \Rightarrow \hat{x}_a(k) = \frac{1}{t_s} \left( \left( A_{ad} - I_{na} \right) A_{ad}^k x_{a0} + B_{ad} u(k) + \Gamma \alpha_{d1} w_{a1}(k) \right) \]

\[ + \left( A_{ad} - I_{na} \right) B_{ad} u(k-1) + \left( A_{ad} - I_{na} \right) \Gamma \alpha_{d1} w_{a1}(k-1) \]

\[ + \left( A_{ad} - I_{na} \right) \sum_{i=1}^{k-1} \left( A_{ad}^i B_{ad} u(k-i-1) \right) \]

\[ + \left( A_{ad} - I_{na} \right) \sum_{i=1}^{k-1} \left( A_{ad}^i \Gamma \alpha_{d1} w_{a1}(k-i-1) \right) \]

\[ \Rightarrow \hat{x}_a(k) = \frac{1}{t_s} \left( A_{ad}^{k+1} - A_{ad}^k \right) x_{a0} + B_{ad} u(k) + \Gamma \alpha_{d1} w_{a1}(k) \]

\[ + \frac{1}{t_s} \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^i \right) B_{ad} u(k-i-1) \]

\[ + \frac{1}{t_s} \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^i \right) \Gamma \alpha_{d1} w_{a1}(k-i-1) \]

(8.7)

From (8.5) we have,

\[ |x_a(k)| = \left| A_{ad}^k x_{a0} + B_{ad} u(k-1) + \Gamma \alpha_{d1} w_{a1}(k-1) + \sum_{i=1}^{k-1} \left( A_{ad}^i B_{ad} u(k-i-1) \right) \right| \]

\[ + \sum_{i=1}^{k-1} \left( A_{ad}^i \Gamma \alpha_{d1} w_{a1}(k-i-1) \right) \]
\[
\leq \left| A_{ad}^k x_0 + B_{ad} u(k-1) \right| + \left| \Gamma \alpha_d w a_1(k-1) \right|
\]
\[
+ \left| \sum_{i=1}^{k-1} A_{ad}^i B_{ad} u(k-i-1) \right| + \left| \sum_{i=1}^{k-1} \left( A_{ad}^i \Gamma \alpha_d \right) w a_1(k-i-1) \right|
\]
\[
\leq \left| A_{ad}^k x_0 + B_{ad} u(k-1) + \bar{w} a_1 \right| + \left| \sum_{i=1}^{k-1} (A_{ad}^i B_{ad} u(k-i-1)) + \left( I_{na} + \sum_{i=1}^{k-1} A_{ad}^i \right) \bar{w} a_1 \right|
\]
\[
\Rightarrow |x_a(k)| \leq \left| A_{ad}^k x_0 + B_{ad} u(k-1) + \left| \sum_{i=1}^{k-1} (A_{ad}^i B_{ad} u(k-i-1)) + \left( I_{na} + \sum_{i=1}^{k-1} A_{ad}^i \right) \bar{w} a_1 \right| \leq \bar{x}_a \quad (8.8)
\]
\[
\Rightarrow |x_a(k)| \leq \left| A_{ad}^i x_0 + B_{ad} u(k-1) + \left| \sum_{i=1}^{k-1} (A_{ad}^i B_{ad} u(k-i-1)) + \left( I_{na} + \sum_{i=1}^{k-1} A_{ad}^i \right) \bar{w} a_1 \right| \leq \bar{x}_a \quad (8.9)
\]

Hence, from (8.8) and (8.9) we have \( \forall k \in \mathbb{Z}_+ \), the induction (8.10) and (8.11) respectively.

\[
|i_x(k)| = \left| C_{ad} A_{ad}^k x_0 + (C_{ad} B_{ad} + D_{ad}) u(k-1) + C_{ad} \Gamma \alpha_d w a_1(k-1) + \Gamma \alpha_d w a_2(k-1) \right|
\]
\[
+ C_{ad} \sum_{i=1}^{k-1} (A_{ad}^i B_{ad} u(k-i-1)) + C_{ad} \sum_{i=1}^{k-1} \left( A_{ad}^i \Gamma \alpha_d \right) w a_1(k-i-1) \right|
\]
\[
\Rightarrow |y_a(k)| \leq \left| C_{ad} A_{ad}^k x_0 + (C_{ad} B_{ad} + D_{ad}) u(k-1) + C_{ad} \Gamma \alpha_d w a_1(k-1) + \left| \Gamma \alpha_d w a_2(k-1) \right| \right|
\]
\[
+ \left| C_{ad} \sum_{i=1}^{k-1} (A_{ad}^i B_{ad} u(k-i-1)) + C_{ad} \sum_{i=1}^{k-1} \left( A_{ad}^i \Gamma \alpha_d \right) w a_1(k-i-1) \right|
\]
\[
\leq \left| C_{ad} A_{ad}^k x_0 + (C_{ad} B_{ad} + D_{ad}) u(k-1) + C_{ad} \left| \Gamma \alpha_d w a_1(k-1) \right| + \left| \Gamma \alpha_d w a_2(k-1) \right| \right|
\]
\[
+ \left| C_{ad} \sum_{i=1}^{k-1} (A_{ad}^i B_{ad} u(k-i-1)) + C_{ad} \sum_{i=1}^{k-1} \left( A_{ad}^i \Gamma \alpha_d \right) w a_1(k-i-1) \right|
\]
\[
\leq \left| C_{ad} A_{ad}^k x_0 + (C_{ad} B_{ad} + D_{ad}) u(k-1) \right| + C_{ad} \bar{w} a_1 + \bar{w} a_2
\]
\[ \left| C_{ad} \sum_{i=1}^{k-1} A_{ad}^{i} B_{ad} u(k-i-1) \right| + \left| C_{ad} \sum_{i=1}^{k-1} A_{ad}^{i} \right| \bar{w}_{a1} \]

\[ = \left| C_{ad} A_{ad}^{k} x_{a0} + (C_{ad} B_{ad} + D_{ad}) u(k-1) \right| + \left| C_{ad} \sum_{i=1}^{k-1} A_{ad}^{i} B_{ad} u(k-i-1) \right| \]

\[ + C_{ad} \left( I_{na} + \sum_{i=1}^{k-1} A_{ad}^{i} \right) \bar{w}_{a1} + \bar{w}_{a2} \]

\[ \Rightarrow \left| y_{a}(k) \right| \leq \left| C_{ad} A_{ad}^{k} x_{a0} + (C_{ad} B_{ad} + D_{ad}) u(k-1) \right| + \left| C_{ad} \sum_{i=1}^{k-1} A_{ad}^{i} B_{ad} u(k-i-1) \right| \]

\[ + C_{ad} \left( I_{na} + \sum_{i=1}^{k-1} A_{ad}^{i} \right) \bar{w}_{a1} + \bar{w}_{a2} \]

\[ \Rightarrow \left| y_{a}(k) \right| \leq \left| C_{ad} A_{ad}^{k} x_{a0} \right| + \left| C_{ad} B_{ad} + D_{ad} \right| u(k-1) + \left| C_{ad} \sum_{i=1}^{k-1} A_{ad}^{i} B_{ad} u(k-i-1) \right| \]

\[ + C_{ad} \left( I_{na} + \sum_{i=1}^{k-1} A_{ad}^{i} \right) \bar{w}_{a1} + \bar{w}_{a2} \]

Hence, from the last two inequalities above we have \( \forall k \in \mathbb{Z}_{+} \) the induction (8.12) and (8.13) respectively.

\[ \left| C_{ad} A_{ad}^{k} x_{a0} + (C_{ad} B_{ad} + D_{ad}) u(k-1) \right| + \left| C_{ad} \sum_{i=1}^{k-1} A_{ad}^{i} B_{ad} u(k-i-1) \right| \]

\[ + C_{ad} \left( I_{na} + \sum_{i=1}^{k-1} A_{ad}^{i} \right) \bar{w}_{a1} + \bar{w}_{a2} - \bar{y}_{a} \leq 0 \Rightarrow \left| y_{a}(k) \right| \leq \bar{y}_{a} \quad (8.12) \]

\[ \left| C_{ad} A_{ad}^{k} x_{a0} + C_{ad} B_{ad} D_{ad} \right| u(k-1) + \left| C_{ad} \sum_{i=1}^{k-1} A_{ad}^{i} B_{ad} u(k-i-1) \right| \]

\[ + C_{ad} \left( I_{na} + \sum_{i=1}^{k-1} A_{ad}^{i} \right) \bar{w}_{a1} + \bar{w}_{a2} - \bar{y}_{a} \leq 0 \Rightarrow \left| y_{a}(k) \right| \leq \bar{y}_{a} \quad (8.13) \]

Furthermore, from (8.7) we have

\[ \left| \tilde{e}_{a}(k) \right| = \left| \frac{1}{t_s} \left( A_{ad}^{k+1} - A_{ad}^{k} \right) x_{a0} + B_{ad} u(k) + \Gamma_{ad} \bar{w}_{a1}(k) \right| \]

\[ + \frac{1}{t_s} \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^{i} \right) B_{ad} u(k-i-1) + \frac{1}{t_s} \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^{i} \right) \Gamma_{ad} \bar{w}_{a1}(k-i-1) \]
\[\Rightarrow |\delta x_a(k)| \leq \frac{1}{t_s} \left( \left( A_{ad}^{k+1} - A_{ad}^k \right)x_{ao} + B_{ad}u(k) \right) + \Gamma_{ad} w_a(k) + \left| \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^i \right) B_{ad}u(k-i-1) \right| + \left| \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^i \right) w_a(k) \right| \]

\[= \frac{1}{t_s} \left( \left( A_{ad}^{k+1} - A_{ad}^k \right)x_{ao} + B_{ad}u(k) \right) + \left| \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^i \right) B_{ad}u(k-i-1) \right| + \left| \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^i \right) w_a(k) \right| \]

\[= \frac{1}{t_s} \left( \left( A_{ad}^{k+1} - A_{ad}^k \right)x_{ao} + B_{ad}u(k) \right) + \left| \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^i \right) B_{ad}u(k-i-1) \right| + \left| \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^i \right) w_a(k) \right| \]

\[\Rightarrow |\delta x_a(k)| \leq \frac{1}{t_s} \left( \left( A_{ad}^{k+1} - A_{ad}^k \right)x_{ao} + B_{ad}u(k) \right) + \left| \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^i \right) B_{ad}u(k-i-1) \right| + \left| \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^i \right) w_a(k) \right| \]

\[\Rightarrow |\delta x_a(k)| \leq \frac{1}{t_s} \left( \left( A_{ad}^{k+1} - A_{ad}^k \right)x_{ao} + B_{ad}u(k) \right) + \left| \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^i \right) B_{ad}u(k-i-1) \right| + \left| \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^i \right) w_a(k) \right| \]

Hence, from (8.14) and (8.15) we have \( \forall k \in \mathbb{Z}_+ \) the induction (8.16) and (8.17) respectively.

\[
\left( A_{ad}^{k+1} - A_{ad}^k \right)x_{ao} + B_{ad}u(k) + \left| \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^i \right) B_{ad}u(k-i-1) \right| + \left| \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^i \right) w_a(k) \right| \]

\[+ \left( I_{na} + \left| \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^i \right) w_a(k) \right| \right) \leq 0 \Rightarrow |\delta x_a(k)| \leq \bar{x}_a \quad (8.16)\]

\[
\left( A_{ad}^{k+1} - A_{ad}^k \right)x_{ao} + B_{ad}u(k) + \left| \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^i \right) B_{ad}u(k-i-1) \right| + \left| \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^i \right) w_a(k) \right| \]

\[+ \left( I_{na} + \left| \sum_{i=1}^{k-1} \left( A_{ad}^{i+1} - A_{ad}^i \right) w_a(k) \right| \right) \leq 0 \Rightarrow |\delta x_a(k)| \leq \bar{x}_a \quad (8.17)\]
In applications to fast modern electro-technological processes and power systems, consider the above analysis and the following optimization problems:

**Problem 8.2:** ∀ ∈ \( N \), find \( \bar{u}_{k-1} := \max_{k \in \mathbb{N}} \left| u(k-1) \right| \) subject to the inequality constraints of (8.10) and (8.12) and subject to \( \left| u(k-1) \right| > 0 \).

**Problem 8.3:** Find \( \bar{u}_k := \max_{k \in \mathbb{N}} \left| u(k) \right| \) subject to the inequality constraints of (8.14).

Now, let

\[
S_{O_1} := \left\{ \bar{u}_{k-1} := \max_{k \in \mathbb{N}} \left| u(k-1) \right| \mid \forall k \in \mathbb{N}, \text{inequality constraints (8.10) and (8.12) hold and} \left| u(k-1) \right| \geq 0 \right\}
\]

and

\[
S_{O_2} := \left\{ \bar{u}_k := \max_{k \in \mathbb{N}} \left| u(k) \right| \mid \forall k \in \mathbb{N}, \text{inequality constraint (8.14) hold and} \left| u(k) \right| \geq 0 \right\}
\]

be the solution sets of the optimization problems (8.2) and (8.3) respectively. Furthermore, define their union

\[
S_{O} := S_{O_1} \cup S_{O_2} := \left\{ \bar{u}_i \mid \forall i \in \mathbb{N}, \bar{u}_i \in S_{O_1} \text{ or } \bar{u}_i \in S_{O_2} \right\}
\]

From the above it follows that an approximate solution to problem (8.1) is given by

\[
\bar{u} := \min_{\bar{u}_i \in S_{O}} \left( u_i \right).
\]

Note that problems (8.2) and (8.3) constitute nonlinear programming problems. In addition, if instead of the inequality constraints (8.10), (8.12) and (8.14), (8.11), (8.13) and (8.15) respectively will used in problems 8.2 and 8.3, then these will be reduced to linear programming problems. In the last case, (8.18) and (8.19) are replaced respectively with

\[
S_{O_1} := \left\{ \bar{u}_{k-1} := \max_{k \in \mathbb{N}} \left| u(k-1) \right| \mid \forall k \in \mathbb{N}, \text{inequality constraints (8.11) and (8.13) hold and} \left| u(k-1) \right| \geq 0 \right\}
\]

and

\[
S_{O_2} := \left\{ \bar{u}_k := \max_{k \in \mathbb{N}} \left| u(k) \right| \mid \forall k \in \mathbb{N}, \text{inequality constraint (8.15) hold and} \left| u(k) \right| \geq 0 \right\}
\]
Note that conservatism can be introduced when using (8.22), (8.23) instead of (8.18), (8.19). Also, when solving practically the above optimization problems (8.2) and (8.3) it is assumed that \( k \in [0,k_{\text{max}}] \), where \( k_{\text{max}} \in \mathbb{Z}_+ \) is a sufficiently large integer subject to which the system is approximately at steady state. In other words, since the system is asymptotically stable, the infinite horizon discrete time domain \( \mathbb{Z}_+ \cup \{0\} = \mathbb{N} \) is replaced with its closed subset \( [0,k_{\text{max}}] \), where \( k_{\text{max}} \in \mathbb{Z}_+ \) can be considered as the discrete settling time. This in practice does not cause any problem since all the system’s state, rate of state and output maximum values take place within \( [0,k_{\text{max}}] \).

Finally, it should be said that the above solution is an approximation to the real one, since the whole process considers a discretization of (8.1), (8.2). The above results can obviously be applied directly to linear time invariant discrete time systems.

Now, problem 8.1 can also be solved as a linear programming problem in the continuous time domain. In that case, from (8.1), (8.2) we have ([164]),

\[
x_a(t) = e^{A_a t} x_a(0) + \int_0^t \left( e^{A_a (t-\sigma)} B_a u(\sigma) \right) d\sigma + \int_0^t \left( e^{A_a (t-\sigma)} F \alpha \Gamma_{a1} \tilde{w}_{a1}(\sigma) \right) d\sigma.
\]

Having in mind that any signal can be approximated as a sequence of pulse inputs, (8.24) subject to

\[
u(t) := \begin{cases} \bar{u}_t, & 0 \leq t \leq t_{u_{\text{max}}} \\ 0, & t > t_{u_{\text{max}}} \end{cases}
\]

and

\[
\tilde{w}_{a1}(t) := \begin{cases} \bar{w}_{a1}, & 0 \leq t \leq t_{w_{a1_{\text{max}}}} \\ 0, & t > t_{w_{a1_{\text{max}}}} \end{cases}
\]

where \( \bar{u}_t \in \mathbb{R}^m, \bar{w}_{a1} \in \mathbb{R}^{n_a}. t_{u_{\text{max}}}, t_{w_{a1_{\text{max}}}} \in \mathbb{R}_+ \), gives

\[
x_a(t) = e^{A_a t} x_a(0) + \frac{1 - \text{sign}(t-t_{u_{\text{max}}})}{2} A_a^{-1} \left( e^{A_a (t-I_n_a)} B_a \bar{u}_t \right)
\]

\[
+ \frac{1 - \text{sign}(t-t_{w_{a1_{\text{max}}}})}{2} A_a^{-1} \left( e^{A_a (t-I_n_a)} \Gamma \alpha \Gamma_{a1} \tilde{w}_{a1} \right).
\]

Now, setting \( \bar{u} := \max_{t \in \mathbb{R}_+ \cup \{0\}} \left( \| u(t) \| \right) = \max_{t \in \mathbb{R}_+ \cup \{0\}} \| u_t \| \), it follows from (8.27) that

\[
\| x_a(t) \| \leq \| e^{A_a t} x_a(0) \| + \| A_a^{-1} \left( e^{A_a (t-I_n_a)} B_a \right) \| \bar{u} \| + \| A_a^{-1} \left( e^{A_a (t-I_n_a)} \Gamma \alpha \Gamma_{a1} \tilde{w}_{a1} \right) \| \].
\[ |x_a(t)| \leq e^{A_a t} x_{a0} + |A_a^{-1}(e^{A_a t} - I_{na})B_a| \bar{u} + |A_a^{-1}(e^{A_a t} - I_{na})| \bar{w}_{a1}. \] (8.28)

From (8.28) it is obvious that the following holds
\[ e^{A_a t} x_{a0} + |A_a^{-1}(e^{A_a t} - I_{na})B_a| \bar{u} + |A_a^{-1}(e^{A_a t} - I_{na})| \bar{w}_{a1} \leq 0 \Rightarrow |x_a(t)| \leq \bar{x}_a. \] (8.29)

Working similarly for the output \( y(t) \), and assuming a disturbance \( \bar{w}_{a2} \), \( 0 < t < t_{wa2_{max}} \), it follows from (8.2) and (8.27) that
\[ y_a(t) = C_a e^{A_a t} x_{a0} + \frac{1}{2} \left( 1 - \text{sign}(t - t_{u_{max}}) \right) \left( C_a A_a^{-1}(e^{A_a t} - I_{na})B_a + D_a \right) \bar{u}_t + \frac{1}{2} \left( 1 - \text{sign}(t - t_{wa2_{max}}) \right) C_a A_a^{-1}(e^{A_a t} - I_{na}) \Gamma a_1 \bar{w}_{a1} + \frac{1}{2} \left( 1 - \text{sign}(t - t_{wa2_{max}}) \right) \Gamma a_2 \bar{w}_{a2}. \] (8.31)

As before, (8.31) gives
\[ |y_a(t)| \leq |C_a e^{A_a t} x_{a0}| + |C_a A_a^{-1}(e^{A_a t} - I_{na})B_a + D_a| \bar{u} + |C_a A_a^{-1}(e^{A_a t} - I_{na})| \bar{w}_{a1} + \bar{w}_{a2} \leq 0 \Rightarrow |y_a(t)| \leq \bar{y}_a. \] (8.32)

From (8.32), it is obvious that the following holds
\[ |C_a e^{A_a t} x_{a0}| + |C_a A_a^{-1}(e^{A_a t} - I_{na})B_a + D_a| \bar{u} + |C_a A_a^{-1}(e^{A_a t} - I_{na})| \bar{w}_{a1} + \bar{w}_{a2} - \bar{y}_a \leq 0 \Rightarrow |y_a(t)| \leq \bar{y}_a \] (8.33)

Moreover, from (8.1) via (8.24) we have,
\[ \frac{d(x_a(t))}{dt} = A_a e^{A_a t} x_{a0} + \frac{1}{2} \left( 1 - \text{sign}(t - t_{u_{max}}) \right) \left( (e^{A_a t} - I_{na})B_a + B_a \right) \bar{u}_t + \frac{1}{2} \left( 1 - \text{sign}(t - t_{wa1_{max}}) \right) \left( (e^{A_a t} - I_{na}) \Gamma a_1 + \Gamma a_1 \right) \bar{w}_{a1}_t + \frac{1}{2} \left( 1 - \text{sign}(t - t_{wa2_{max}}) \right) e^{A_a t} \Gamma a_2 \bar{w}_{a2}_t. \]

\[ \Rightarrow \frac{d(x_a(t))}{dt} = A_a e^{A_a t} x_{a0} + \frac{1}{2} \left( 1 - \text{sign}(t - t_{u_{max}}) \right) e^{A_a t} B_a \bar{u}_t + \frac{1}{2} \left( 1 - \text{sign}(t - t_{wa1_{max}}) \right) e^{A_a t} \Gamma a_1 \bar{w}_{a1}_t + \frac{1}{2} \left( 1 - \text{sign}(t - t_{wa2_{max}}) \right) e^{A_a t} \Gamma a_2 \bar{w}_{a2}_t. \] (8.34)
From (8.34), it is obvious that the following holds
\[ A_a e^{A_a t} x_{a0} + e^{A_a t} B_a \bar{u} + e^{A_a t} \bar{w}_a 1 - \bar{x}_a \leq 0 \Rightarrow \left| \frac{d(x_a(t))}{dt} \right| \leq \bar{x}_a. \] (8.35)

Hence,
\[ \bar{u} := \left\{ \max(\bar{u}) \right\}, \quad \text{(8.29) and (8.33) and (8.35) hold}. \] (8.36)

In the special case of \( A_a \) being diagonal, since \( e^{A_a t} \leq I_{n_a} \quad \forall t \in \mathbb{R}_+ \cup \{0\}, \) (8.29), (8.33) and (8.35) can be replaced by (8.37), (8.38) and (8.39) respectively.

\[ x_{a0} + A_a^{-1} B_a \bar{u} + A_a^{-1} \bar{w}_a 1 - \bar{x}_a \leq 0 \Rightarrow \left| x_a(t) \right| \leq \bar{x}_a \] (8.37)
\[ C_a x_{a0} + C_a A_a^{-1} B_a + D_a \bar{u} + C_a A_a^{-1} \bar{w}_a 1 + \bar{w}_a 2 - \bar{y} \leq 0 \Rightarrow \left| y_a(t) \right| \leq \bar{y}_a \] (8.38)
\[ A_a x_{a0} + B_a \bar{u} + \bar{w}_a 1 - \bar{x}_a \leq 0 \Rightarrow \left| \frac{d(x_a(t))}{dt} \right| \leq \bar{x}_a \] (8.39)

In conclusion, whenever the previous optimization problems are feasible subject to the initial conditions of (8.1), (8.2), and under a sufficient system discretization in section 8.2.1, a limit to the control input signal can be obtained. With such a limit, saturation will never occur to the system’s state, rate of state and output.

Therefore, assuming (8.1), (8.2) are the actuators’ model in a general system design, the whole problem of designing an overall control system by preventing actuator saturation is effectively transformed to a control system design subject to amplitude saturated control signals. Hence, any postmodern design technique (e.g. anti-windup control designs [22]) can be applied.

### 8.3 Proposed Control System Design

After transforming the original control problem to a control system design problem subject to amplitude saturated control signals, the particular design that is used for the last problem is described in this section.

#### 8.3.1 Fixed Order Observer Based Dynamic Output Feedback \( H_2 \)/Anti-windup Controllers with Integrators for Systems with Amplitude Saturated Control Inputs

Consider a plant described by the dynamical system
\[ \frac{dx_p(t)}{dt} := A_p x_p(t) + B_p \sigma(u(t)) + \Gamma_p w_p(t), \quad x_p(0) := x_{p0}, \; u(0) := u_0, \; t \in \mathbb{R}_+ \cup \{0\}. \] (8.40)
\[ y_p(t) = C_p x_p(t) + \Gamma_p w_p(t), \quad (8.41) \]

where \( x_p(t) \in \mathbb{R}^{n_p} \), \( u(t) \in \mathbb{R}^m \), \( y_p(t) \in \mathbb{R}^{p_p} \), \( w_p(t) \in \mathbb{R}^{l_p} \), \( w_2(t) \in \mathbb{R}^{q_p} \), are defined as the plant’s state, control input, output, state disturbance and output disturbance respectively and \( n_p, m, p_p, l_p, q_p \in \mathbb{Z}_+ \). \( A_p \in \mathbb{R}^{n_p \times n_p} \), \( B_p \in \mathbb{R}^{n_p \times m} \), \( \Gamma_p \in \mathbb{R}^{n_p \times l_p} \), \( C_p \in \mathbb{R}^{p_p \times n_p} \), \( \Gamma_p \in \mathbb{R}^{p_p \times q_p} \) are the corresponding constant matrix coefficients of the above vectors and \( t \) is the time.

Now, \( \sigma: \mathbb{R}^m \to \mathbb{R}^m \) is a multivariable saturation nonlinearity (see [174]) having the independent symmetric saturation function

\[ \sigma(u_i(t)) = \left[ \begin{array}{c} \sigma(u_{i1}(t)) \\ \vdots \\ \sigma(u_{im}(t)) \end{array} \right]^T, \quad (8.42) \]

\[ \sigma_{i1}(u_{i1}(t)) = \begin{cases} u_{i1}(t), & |u_{i1}(t)| \leq \bar{u}_{i1} \\ \text{sign}(u_{i1}(t))\bar{u}_{i1}, & |u_{i1}(t)| > \bar{u}_{i1} \end{cases}, \quad (8.43) \]

where \( \bar{u}_{i1} \in \mathbb{R}_+ \). Note that for \( m \geq 2 \), the saturation function \( \sigma(u(t)) \) is not necessarily in the same direction as \( u(t) \) is [174].

System (8.40), (8.41) is assumed further to be controllable (i.e. \( \text{rank} \begin{bmatrix} A_p - v I n_p \\ B_p \end{bmatrix} = n_p, \quad \forall v \in C \)), and observable (i.e. \( \text{rank} \begin{bmatrix} A_p - v I n_p \\ C_p \end{bmatrix} = n_p, \quad \forall v \in C \)).

For tracking control of the output determined by (8.41), the integrator dynamical system below is used

\[ \frac{d x_I(t)}{dt} = A_I y_I(t), \quad x_I(0) = x_I0, \quad t \in \mathbb{R}_+ \cup \{0\}, \quad (8.44) \]

\[ y_I(t) = x_I(t) + \Gamma_I w_I(t), \quad (8.45) \]

where \( x_I(t) \in \mathbb{R}^{p_p} \), \( y_I(t) \in \mathbb{R}^{p_p} \), \( w_I(t) \in \mathbb{R}^{q_I} \), are defined as the integrator’s state, output and output disturbance respectively and \( l_I, q_I, p_p \in \mathbb{Z}_+ \). \( A_I \in \mathbb{R}^{p_p \times p_p} \), \( \Gamma_I \in \mathbb{R}^{p_p \times q_I} \) are the corresponding constant matrix coefficients of the above vectors and \( t \) is time.

The augmented system of the plant (8.40), (8.41) and the integrator (8.44), (8.45) is described from the dynamical system

\[ \frac{dx(t)}{dt} := Ax(t) + B \sigma(u(t)) + \Gamma_I w_I(t), \quad x(0) = x_0, \quad u(0) = u_0, \quad t \in \mathbb{R}_+ \cup \{0\}, \quad (8.46) \]

\[ y(t) := Cx(t) + \Gamma_2 w_2(t), \quad (8.47) \]
where \( x(t) = \begin{bmatrix} x_p(t) \\ x_1(t) \end{bmatrix}, \ u(t) \in \mathbb{R}^m, \ y(t) = \begin{bmatrix} w_{p1}(t) \\ w_{p2}(t) \end{bmatrix}, \ w_1(t) = w_1(t), \ w_2(t) = w_2(t), \) are defined as the plant’s state, control input, output, state disturbance and output disturbance respectively and \( x(t) \in \mathbb{R}^n, \ y(t) \in \mathbb{R}^p, \ w_1(t) \in \mathbb{R}^l, \ w_2(t) \in \mathbb{R}^q, \ n = n_p + p_p, \ p = p_p, \ l = l_p + q_p, \ q = q_1, \ n,m,l,q \in \mathbb{Z}_+. \) Now, \( A := \begin{bmatrix} A_p & 0_{p_p \times p_p} \\ A_p C_p & 0_{p_p \times p_p} \end{bmatrix}, \ B := \begin{bmatrix} B_p \\ A_I D_p \end{bmatrix}, \)

\[
A_1 := \begin{bmatrix} \Gamma_p_1 & 0_{n_p \times q_p} \\ 0_{p_p \times l_p} & \Lambda_I \Gamma_p_2 \end{bmatrix}, \ C := \begin{bmatrix} 0_{p_p \times n_p} & 1_{p_p} \end{bmatrix}, \ C_1 \in \mathbb{R}^{p \times n}, \Gamma_2 := \Gamma_1 \] are the corresponding constant matrix coefficients of the above vectors, where \( A \in \mathbb{R}^{n \times n}, \ B \in \mathbb{R}^{n \times m}, \Gamma_1 \in \mathbb{R}^{n \times l}, \ C \in \mathbb{R}^{p \times n}, \Gamma_2 \in \mathbb{R}^{p \times q}. \) In the sequel, system (8.46), (8.47) will be referred to as the open loop system and it is assumed that it is controllable (i.e. \( \text{rank}([A - v I_n B]) = n, \ \forall v \in C \)), and observable (i.e. \( \text{rank} \left[ \begin{bmatrix} A - v I_n \\ C \end{bmatrix} \right] = n, \ \forall v \in C \)).

For the open loop system (8.46), (8.47), consider now the observer based ‘pseudo linear’ controller

\[
\frac{dx_c(t)}{dt} := A_c x_c(t) + B_c y(t) + E_c (\sigma(u(t)) - u(t)), \quad x_c(0) = x_{c_0}, \quad t \in \mathbb{R}_+ \cup \{0\},
\]

\[
u(t) := C_c x_c(t),
\]

where \( x_c(t) \in \mathbb{R}^{n_c}, \) is the controller’s state and \( n_c \in \mathbb{Z}_+, \) \( n_c \leq n \) and \( A_c \in \mathbb{R}^{n_c \times n_c}, \ B_c \in \mathbb{R}^{n_c \times p}, \ E_c \in \mathbb{R}^{n_c \times m}, \ C_c \in \mathbb{R}^{m \times n_c}. \)

A transfer function block diagram of (8.48), (8.49) in the Laplace transform domain is shown in figure 8.1.

![Figure 8.1. The observer based 'pseudo linear' controller (8.48), (8.49) in Laplace domain.](image)
The resulting closed loop system of (8.46)-(8.49) can be written as the dynamical system

$$\frac{d\tilde{x}(t)}{dt} = \tilde{A}\tilde{x}(t) + \tilde{B}\left(\sigma(u(t)) - u(t)\right) + \tilde{w}(t), \quad \tilde{x}(0) = \tilde{x}_0, \quad t \in \mathbb{R}_+ \cup \{0\}.$$  \hspace{1cm} (8.50)

$$u(t) = \tilde{C}\tilde{x}(t),$$  \hspace{1cm} (8.51)

where $\tilde{x}(t) = \begin{bmatrix} x_p(t) \\ x_e(t) \end{bmatrix}$ is the closed loop system’s state, $\tilde{w}(t) = \begin{bmatrix} w_1(t) \\ w_2(t) \end{bmatrix}$, $\tilde{w}(t) \in \mathbb{R}^{l+q}$ is the closed loop system’s disturbance, while its output is considered as $u(t) \in \mathbb{R}^{m}$. Furthermore, $\tilde{A} := \begin{bmatrix} A & BC_c \\ B_c C & A_c \end{bmatrix}, \quad \tilde{B} := \begin{bmatrix} B \\ E_c \end{bmatrix}, \quad \tilde{B} \in \mathbb{R}^{(n+n_c) \times m}$, $\tilde{\Gamma} := \begin{bmatrix} \Gamma_1 & 0_{n \times q} \\ 0_{p \times 1} & \Gamma_2 \end{bmatrix}, \quad \tilde{\Gamma} \in \mathbb{R}^{(n+p) \times (l+q)}$ and $\tilde{C} := \begin{bmatrix} 0_{m \times n} & C_c \end{bmatrix}, \quad \tilde{C} \in \mathbb{R}^{m \times (n+n_c)}$.

A transfer function block diagram of the closed loop system (8.50), (8.51) in the Laplace transform domain, indicating the plant, integrators and controller is shown in figure 8.2.

The design of the controller (8.48), (8.49) for the (8.50), (8.51) is provided by the next theorem.

![Figure 8.2. Closed loop system (8.50), (8.51) in Laplace domain.](image-url)
Theorem 8.1; Consider all the assumptions for systems (8.46)-(8.49) stated so far and let $R_1, V_1 \in \mathbb{R}^{n \times n}$, $R_1 \geq 0$, $V_1 := \Gamma_1 \Gamma_1^T$, $R_2 \in \mathbb{R}^{m \times m}$, $R_2 > 0$, $V_2 \in \mathbb{R}^{p \times p}$, $V_2 := \Gamma_2 \Gamma_2^T$ and $V_2 > 0$, $\theta \in \mathbb{R}_+$. Suppose that there are $P, Q, \hat{P}, \hat{Q} \in \mathbb{R}^{n \times n}$, $P \geq 0$, $Q \geq 0$, $\hat{P} \geq 0$, $\hat{Q} \geq 0$ satisfying

$$A^TP + PA - (1 - \theta^{-2})PBR_2^{-1}B^TP + \tau_1^TPBR_2^{-1}B^TP\tau_1 + R_1 = 0,$$  

(8.52)

$$AQ + QA^T - QC^TV_2^{-1}CQ + \tau_1QC^TV_2^{-1}CQ\tau_1 + V_1 = 0.$$  

(8.53)

$$(A - QC^TV_2^{-1}C)^T \hat{P} + \hat{P}(A - QC^TV_2^{-1}C) + PBR_2^{-1}B^TP - \tau_1^TPBR_2^{-1}B^TP\tau_1 = 0.$$  

(8.54)

$$(A - BR_2^{-1}B^TP)\hat{Q} + \hat{Q}(A - BR_2^{-1}B^TP)^T + QC^TV_2^{-1}CQ - \tau_2QC^TV_2^{-1}CQ\tau_1 = 0.$$  

(8.55)

$$\tau_1 := I_n - \hat{Q}\hat{P}(\hat{Q}\hat{P})^\dagger = I_n - \tau_1^T \tau_2.$$  

(8.56)

where $(.)^\dagger$ denotes the Moore-Penrose generalized inverse of a matrix [126], [140], [21].

Furthermore, define

$$\tilde{P} := \begin{bmatrix} P + \hat{P} & -\hat{P}\tau_1^T \\ -\tau_1\hat{P} & \tau_1\hat{P}\tau_1 \\ \end{bmatrix},$$

$$E_c := B,$$  

(8.57)

$$C_c := -R_2^{-1}B^TP\tau_1^T.$$  

(8.58)

$$B_c := \tau_2QC^TV_2^{-1}.$$  

(8.59)

$$A_c := \tau_2A\tau_1^T + \tau_2BC_c - B_cC\tau_1^T.$$  

(8.60)

$$\tilde{R} := \begin{bmatrix} R_1 & 0_{n \times n_c} \\ 0_{n_c \times n} & C_c^TR_2C_c \\ \end{bmatrix}.$$  

$$\beta_0(\theta) := \begin{cases} 1 - \theta^{-1}, & \theta > 1 \\ 0, & 0 < \theta \leq 1 \end{cases},$$

and suppose that $(\tilde{A}, \tilde{R})$ is observable (i.e. $\text{rank} \left( \begin{bmatrix} \tilde{A} - \lambda I_{n+n_c} \\ \tilde{R} \end{bmatrix} \right) = n + n_c$, $\forall \lambda \in \mathbb{C}$). Then the closed loop system (8.50), (8.51) is asymptotically stable if its state initial conditions satisfy
\[ \tilde{x}^T(0)\tilde{P}\tilde{x}(0) < \min_{i=1, \ldots, m} \left\{ \frac{\tilde{u}_i^2}{\beta_i^2(\vartheta)\text{row}_i(\tilde{C})\tilde{P}^{-1}\text{row}_i(\tilde{C})^T} \right\}. \]

Furthermore, the $H_2$-type cost functional

\[ J(\tilde{x}(t)) = \int_0^{+\infty} \left( x^T(t)R_1x(t) + u^T(t)R_2u(t) + \vartheta^{-2}\tilde{x}^T(t)\tilde{P}\tilde{B}R_1^{-1}\tilde{B}^T\tilde{x}(t) + 2\tilde{x}^T(t)\tilde{P}\tilde{B}(u(t) - \sigma(u(t))) \right) dt \]

is minimized to $J(\tilde{x}(0)) = \tilde{x}^T(0)\tilde{P}\tilde{x}(0)$ and $\left\| \tilde{C}(sI_{n+c} - \tilde{A})\tilde{B} \right\|_{\infty} \leq \vartheta$, where $s$ denotes the Laplace transform variable \cite{149} and $\| \cdot \|_{\infty}$ the $H_\infty$ norm of a matrix that is a function of $s$ \cite{81}, \cite{47}.

**Proof.**

The proof of this theorem is immediate from proposition 4.2 of \cite{172}, theorem 2.1 of \cite{174} and theorem 2.1 of \cite{172}, after the observation that equation (2.6) of \cite{174} with $\gamma = \vartheta$, $\beta_o = 1$ and $R_o = \frac{1}{2}/\vartheta^2 R_2$ is identical to equation (8) 2.1 of \cite{172} \cite{172}, \cite{174}).

The matrices $R_1, V_1 \in \mathbb{R}^{n \times n}$, $R_2 \in \mathbb{R}^{m \times m}$, $V_2 \in \mathbb{R}^{p \times p}$, play the role of penalty matrices. $R_1 \in \mathbb{R}^{n \times n}$ and $R_2 \in \mathbb{R}^{m \times m}$ can be selected arbitrarily, or in a systematic optimal way, as it will be shown in next section.

The set

\[ \Psi_s := \left\{ \tilde{x}(0) \in \mathbb{R}^{n+c} \mid \tilde{x}^T(0)\tilde{P}\tilde{x}(0) < \min_{i=1, \ldots, m} \left\{ \frac{\tilde{u}_i^2}{\beta_i^2(\vartheta)\text{row}_i(\tilde{C})\tilde{P}^{-1}\text{row}_i(\tilde{C})^T} \right\} \right\} \]

defines a subset of the domain of attraction of the closed loop system. Since theorem 8.1 provides sufficient conditions for asymptotic stability, it is possible for the closed loop system to be asymptotically stable, for initial conditions outside $\Psi_s$.

The result $\left\| \tilde{C}(sI_{n+c} - \tilde{A})\tilde{B} \right\|_{\infty} \leq \vartheta$ can be viewed as an application of the small gain theorem \cite{69} to a system with input $u(t) - \sigma(u(t))$ and output $u(t)$. Hence, one may argue that theorem 8.1 enforces an $H_\infty$-type minimization on saturation events. For $\vartheta \to +\infty$ theorem 8.1 specializes to the fixed order $H_2$ problem and equations (8.52)-(8.56) are identical with those
associated with the fixed order $H_2$ control problem [7], [188]. In this case, for initial conditions that belong to $\Psi_s$, according to theorem 8.1, the closed loop system never saturates [172].

Controller (8.48), (8.49) has an observer structure, and it is obvious that for the cases of $C \neq I_n$ the estimated state is provided by the controller itself. Also, it is easy to show that the so called separation principle [130] holds with (8.52) the control equation and (8.53) the estimator equation [7], [130]. Hence, a loop transfer recovery (LTR) process [45], [46] can be posed in order to obtain certain performance specifications, with the aid of state feedback design laws, such as [153], [13] etc. In this situation, the closed loop system in general will not be optimal subject to (8.61), and a compromise between stability margins and disturbance attenuation takes place. Finally, it is noted that various block diagrammatic illustrations of the closed loop system with the controller (8.40), (8.41) and more details about the above structures can be found in [172], [174], [188] and [7].

From a computational viewpoint, the main task in the control system design is to solve (8.52)-(8.56). Note that (8.52), (8.53) are optimal projection equations (OPE) (see chapter 6), and (8.54), (8.55) can be considered as algebraic Lyapunov equations. These computations are presented next.

### 8.4 Control System Design Computation

The penalty matrices $R_1, V_1 \in \mathbb{R}^{n \times n}$, $R_2 \in \mathbb{R}^{n \times m}$, $R_1$ and $R_2$ are chosen in such a way that if we consider an LQR state feedback law for (8.46), (8.47) (subject to $R_1$ and $R_2$), then the resulting closed loop system will have all its eigenvalues optimally placed in the complex plane region shown in grey in figure 8.3.

$h$ (in sec$^{-1}$ units) in figure 8.3 is the degree of relative stability for the designed closed loop system [153] and hence it is a design parameter. When the closed loop system is approximated via its dominant pair of eigenvalues, then its settling time to the steady state within an error of ±5% is approximately $t_s = \frac{3}{h}$ [40]. Such eigenvalue placement will hopefully provide a closed loop system with an appropriate settling time and good damping of oscillations.

Hence, with initial random $R_1 \geq 0$ and $R_2 = I_m$ the optimal pole placement method of [13] is used first to stabilise the open loop system (if it is unstable) with a state feedback law. Then, the optimal pole placement method of [153] is used keeping $R_2 = I_m$ and modifying $R_1$ of the first obtained stable system in order to place the eigenvalues in the grey region of figure 8.3.
Note that in the last case, the resulting closed loop system under state feedback is given by

\[ A_{cl} := A - BB^TP, \]

where \( P \geq 0 \) satisfies \( A^TP + PA - PBB^TP + R_I = 0 \) which is (8.52) with \( \vartheta \to +\infty \) and \( \tau_\perp = 0_{n\times n} \). Hence, \( R_I \) is determined by \( R_I = -A^TP - PA + PBB^TP \). The combination of these methods and the details of the respective algorithms can be found in [166].

After the above, \( R_I \) is further modified such that under the minimum possible \( \vartheta > 0 \), \( A_{cl} := A - \left(1 - \vartheta^{-2}\right)BB^TP \) has all its eigenvalues in the gray region in figure 8.3. For the last case, \( A^TP + PA - \left(1 - \vartheta^{-2}\right)PBB^TP + R_I = 0 \), hence \( R_I = -A^TP - PA + \left(1 - \vartheta^{-2}\right)PBB^TP \).

Since the methods of [13], [153] are based in LQR optimization, infinite gain margin and at least 60° phase margin are guaranteed in each input channel for \( A_{cl} := A - \left(1 - \vartheta^{-2}\right)BB^TP \).
Note that, with the above determination of $R_1$, $R_2$ and $\theta > 0$, (8.52) holds with $\tau_\perp = 0_{n \times n}$ (i.e. $n_c = n$ full order controller case). Hence, it is obvious that for the full order controller case $n_c = n$, the loop-transfer recovery (LTR) method of [46] can be used in order that the closed loop system (8.50), (8.51) recovers as much as possible the dynamics and the relative stability properties of the closed loop system $A_{cl} := A - \left(1 - \theta^{-2}\right)BB^TP$ with state feedback.

In order to use the LTR method of [46], $V_I \in \mathbb{R}^{n \times n}$ is modified as

$$V_I := \Gamma_I \Gamma_I^T + \ell^2 BB^T,$$

where $\ell \in \mathbb{R}_+$. $\ell = 0$ corresponds to optimal estimation of the missing states but in general results in poor recovery of state feedback stability margins (i.e. robustness). When $\ell \to \infty$, there is very good recovery of the state feedback stability margins but there is poor estimation of the missing states (i.e. poor performance). Hence, a compromise between robustness and performance should take place when selecting $\ell \in \mathbb{R}_+$ (see [46] for relevant discussions).

After determining $R_1, V_I \in \mathbb{R}^{n \times n}, R_2 \in \mathbb{R}^{m \times m}, V_2 \in \mathbb{R}^{p \times p}$ and $\theta \in \mathbb{R}_+$, the major stage of the controller design according to theorem 8.1 is the computation of (8.52)-(8.56). Note that in general (8.52) and (8.53) can not be solved as algebraic Riccati equations. This is exactly where the GQME-Toolbox can be used; to solve (8.52) and (8.53) as special symmetric cases of (2.1). Also, (8.54) and (8.55) can be solved as algebraic Lyapunov equations using state-of-the-art software. Now, $\tau_\perp$ in (8.56) is determined in an iterative manner as proposed in [84], [168] or in [86]. Overall, we have two possible iteration algorithms shown below.

**Algorithm 8.1 (Eigenvalue Ranking [84]):**

1. Specify the order of the compensator $n_c \leq n$ and a convergence criterion $\epsilon_{tol} > 0$.
   
   Set $k = 0$, $\tau_\perp := 0_n$. and go to step 2.

2. Set $k = k + 1$. With $\tau_{\perp k} = \tau_{\perp k-1}$

   2a. Select $X(0) = P_k^0$, $\gamma$ and solve the optimal projection equation (see chapter 6)
   
   (8.52) for $X = P_k = P$ as special case to (2.1).

   2b. Select $X(0) = Q_k^0$, $\gamma$ and solve the optimal projection equation (see chapter 6)
   
   (8.53) for $X = Q_k = Q$ as special case to (2.1).

   2c. Solve (8.54) for $\tilde{P}_k = \tilde{P}$ and (8.55) for $\tilde{Q}_k = \tilde{Q}$ as algebraic Lyapunov equations using standard software.
2d. Go to step 3.

3. Form the spectral decomposition of $\hat{Q}_k \hat{P}_k$, as $S. \tilde{D} [\tilde{Q}_k \tilde{P}_k] = V_{l_k} \tilde{A}_k V_{r_k}^T$.

$$\tilde{A}_k = \text{diag} \left( [\tilde{\lambda}_1, \ldots, \tilde{\lambda}_{n_k}, \tilde{\lambda}_{n_k+1}, \ldots, \tilde{\lambda}_n] \right), \tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \ldots \geq \tilde{\lambda}_n > 0,$$

$$V_{r_k}^T V_{l_k} = I_n.$$ 

Set $Y_{k_i} = \frac{\tilde{\lambda}_{k_i}}{\chi_{k_i}}, i = n_{c} + 1, \ldots, n$ and $Y_k = \text{diag} \left( [Y_{k_{n_{c}+1}}, \ldots, Y_{k_n}] \right)$.

$$\tau_{\perp_k} = I_n - V_{l_k} \begin{bmatrix} I_{n_c} & 0_{n_c \times (n-n_c)} \\ 0_{(n-n_c) \times n_c} & Y_k \end{bmatrix} V_{l_k}^T.$$ 

4. If $Y_{k_{n_{c}+1}} > \varepsilon_{\text{tol}}$ return to step 2. Else, go to step 5.

5. Stop: the process is complete with $\tau_1 = \begin{bmatrix} \text{row}_1 \left( V_{l_k}^T \right) \\ \vdots \\ \text{row}_{n_c} \left( V_{l_k}^T \right) \end{bmatrix}$, $\tau_2 = \begin{bmatrix} \text{row}_1 \left( V_{r_k}^T \right) \\ \vdots \\ \text{row}_{n_c} \left( V_{r_k}^T \right) \end{bmatrix}$, $\tau_{\perp} = \tau_{\perp_k}$.

$$P = P_k \cdot Q = Q_k \cdot \tilde{P} = \tilde{P}_k \cdot \tilde{Q} = \tilde{Q}_k.$$

*Algorithm 8.2 (Component-Cost Ranking [86]):*

1. Specify the order of the compensator $n_c \leq n$ and a convergence criterion $\varepsilon_t > 0$.

Set $k = 0$, $\tau_{\perp} = 0_n$, and go to step 2.

2. Set $k = k + 1$. With $\tau_{\perp_k} = \tau_{\perp_{k-1}}$

2a. Select $X(0) = P_k^{\text{opt}}$, $\gamma$ and solve the optimal projection equation (see chapter 6)

(8.52) for $X = P_k = P$ as special case to (2.1).

2b. Select $X(0) = Q_k^{\text{opt}}$, $\gamma$ and solve the optimal projection equation (see chapter 6)

(8.53) for $X = Q_k = Q$ as special case to (2.1).

2c. Solve (8.54) for $\tilde{P}_k = \tilde{P}$ and (8.55) for $\tilde{Q}_k = \tilde{Q}$ as algebraic Laypunov equations using standard software.

2d. Go to step 3.

3. Form the spectral decomposition of $\tilde{Q}_k \tilde{P}_k$, as $S. \tilde{D} [\tilde{Q}_k \tilde{P}_k] = \tilde{V}_{l_k} \tilde{\tilde{A}}_k \tilde{V}_{r_k}^T$.

$$\tilde{\tilde{A}}_k = \text{diag} \left( [\tilde{\tilde{\lambda}}_1, \ldots, \tilde{\tilde{\lambda}}_{n_k}, \tilde{\tilde{\lambda}}_{n_k+1}, \ldots, \tilde{\tilde{\lambda}}_n] \right), \tilde{\tilde{\lambda}}_1 \geq \tilde{\tilde{\lambda}}_2 \geq \ldots \geq \tilde{\tilde{\lambda}}_n > 0,$$

$$\tilde{V}_{r_k}^T \tilde{V}_{l_k} = I_n.$$
Set $\tilde{A}_k := -\tilde{A}_k \tilde{V}_k [(A - BR_2^{-1}B^TP_k - Q_k C^TV_2^{-1}C)^T \tilde{V}_k^T]$ and let $S \tilde{A}_k := \{ \tilde{A}_{ki} \}, \ i = 1, \ldots, n$.

Set $\tilde{A}_k = \text{diag} \left( \left[ \tilde{\lambda}_{k1} \cdots \tilde{\lambda}_{kn_c} \tilde{\lambda}_{kn_c+1} \cdots \tilde{\lambda}_{kn} \right] \right)$, where

$\tilde{\lambda}_{ki} := \left\{ \begin{array}{ll}
1, & -\sqrt{\tilde{\lambda}_{ki}} \text{ is among the } n_c \text{ largest elements of } S \tilde{A}_k \\
\max_{i=1, \ldots, n_c} (\tilde{\lambda}_{ki}), & -\sqrt{\tilde{\lambda}_{ki}} \text{ is not among the } n_c \text{ largest elements of } S \tilde{A}_k
\end{array} \right.$

Form the spectral decomposition of $\tilde{Q}_k \tilde{P}_k$, as $SD \left[ \tilde{V}_k \tilde{A}_k \tilde{V}_k^T \right] = \tilde{V}_k \tilde{A}_k \tilde{V}_k^T$.

$A_k = \text{diag} \left( \left[ \lambda_{k1} \cdots \lambda_{kn_c} \lambda_{kn_c+1} \cdots \lambda_{kn} \right] \right)$, $\lambda_{k1} \geq \lambda_{k2} \geq \ldots \geq \lambda_{kn} > 0$,

$V_k^T \tilde{V}_k = I_n$.

Set $Y_k = \text{diag} \left( \left[ \lambda_{kn_c+1} \cdots \lambda_{kn} \right] \right)$, $\tau_k = I_n - V_k \left[ \begin{array}{cc}
I_{n_c} & 0_{n_c \times (n-n_c)} \\
0_{(n-n_c) \times n_c} & Y_k
\end{array} \right] V_k^T$.

4. If $\lambda_{kn_c+1} > \epsilon_{tol}$ return to step 2. Else, go to step 5.

5. Stop: the process is complete with $\tau = \begin{bmatrix}
\text{row}_1 \left( V_k^T \right) \\
\vdots \\
\text{row}_{n_c} \left( V_k^T \right)
\end{bmatrix}$, $\tau = \begin{bmatrix}
\text{row}_1 \left( V_k^T \right) \\
\vdots \\
\text{row}_{n_c} \left( V_k^T \right)
\end{bmatrix}$, $\tau_{\perp} = \tau_{\perp_k}$.

$P = P_k$, $Q = Q_k$, $\tilde{P} = \tilde{P}_k$, $\tilde{Q} = \tilde{Q}_k$.

With the above algorithms, (8.52)-(8.55) are solved for a specific iterate $\tau_{\perp_k}$. As each algorithm progresses and the tolerance in step 4 gets closer to $\epsilon_{tol}$, $\tau_{\perp_k}$ tends to the idempotent matrix in (8.56) with rank equal to $n - n_c$.

The setup and the specialization of (2.1) in (8.52) and (8.53) are in accordance with the setup for OPEs described in chapter 6. Next the proposed computational method is illustrated with two examples.

### 8.5 Robust Control of Fast Modern Electro-technological Manufacturing Processes: High Powered Lasers

Industry currently uses high-powered lasers in welding, cutting and surface treatment [114]. Although laser systems are normally stable by construction, the responses of some of them are
slow. This affects the overall cost of production and therefore the need to speed up response times. This is true for all fast modern electro-technological manufacturing processes. Hence, the needs for robust control systems, which will effectively control the speed of the response of the particular components of such systems in accordance with the production demand.

One of the major parameters of a high-powered laser is the laser power itself. It is known that the laser power and power distribution have profound effects on the production quality [108]. Laser power is the outcome of synthetic interactions within the laser machine such as the excitation current, plasma density and structure, gas compositions, feedback radiation etc. The power of a high-powered laser machine is normally described by a first order linear differential equation. This is by nature an asymptotically stable system but with reasonably large startup times [158]. For example, for a 2 KW laser machine the startup time is about 15 minutes (very large for fast automation processes [66]). Therefore, the control of the laser power in order to reduce settling times and also to increase the accuracy between changes in power levels is very important.

For control of laser power, in-process laser power monitoring and feedback control methods have been applied in the past [114] and sensor technology has contributed to the feasibility implementing PID controllers in real time. The laser material processing technologies are still relatively new (in terms of applications), and therefore modern robust control methods have not yet been used. So far the most popular controller that has been used is the PID controller ([108]).

The main problem with using a PID controller for the laser power control, is that the controlled power can overflow the maximum allowed power of the machine so that the control can call on higher excitation currents causing an overload situation.

### 8.5.1 System Description and Design Specifications

In the present study, the above problem is solved using the control design proposed in the earlier sections. This is illustrated on a model of a real industrial high powered system taken from the literature [108]. In [108], the power of the laser is described by a dynamical system in the form of (8.40), (8.41) with $A_p = -0.5$, $B_p = 0.08$, $r_{p1} = 1$, $C_p = 1$, $D_p = 0$, $r_{p2} = 1$, referred to as the plant. In order to track power reference signals, an integrator like system (8.44.), (8.45) is used with $A_I = -0.1$, $C_I = 1$, $r_I = 0$. Hence, the open loop system (8.46), (8.47) is determined as

$$A = \begin{bmatrix} -0.5 & 0 \\ -0.1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0.08 \\ 0 \end{bmatrix}, \quad \Gamma_I = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad C = \begin{bmatrix} 0 & 1 \end{bmatrix}, \quad D = 0, \quad \Gamma_2 = 1.$$

The eigenvalues of $A$ are 0, -0.5, so the open loop system is unstable. Now, the controller (8.48), (8.49) is to be designed. The generating control signal (i.e. the controller's output is to be added to the reference signal of power levels $r$ and the resulting signal is then input to the plant.
The design specifications for the above system are the tracking of power levels with steady state error less than 0.2% and settling time less than or equal to 10 seconds, under small disturbances to the open loop system. In addition, the maximum allowed power is 20 KW and in any case, the power should not overflow the reference power levels during changes in these levels. The last constraint is important for not damaging the processed material subject to an undesirable power magnitude that might take place through a change in the power level during the process. Due to the above specifications, a first order controller ($n_c = 1$) (8.48), (8.49) should be designed.

### 8.5.2 Control System Computation

In order to keep the power of the system below or equal to the reference level $r$, (8.36) and (8.37) are used in order to limit the control signal according to subsection 8.2.1. Note that, in the present case it happens that the actuator model (8.1), (8.2) is actually the plant (8.40), (8.41). Hence, assuming a maximum disturbance of magnitude 0.001, the maximum limit of the control signal for which (8.37) hold can be analytically determined as

$$
\overline{u}_c := \frac{A_p}{B_p} \left( |r| - |x_p(0)| - \frac{w_{p,r}}{B_p} \right) = 6.25 |r| - |x_p(0)| - 0.002.
$$

Note that, the limit in (8.43) is now a function of the reference level and the plant's initial state. In other words, (8.43) is a varying limiter.

The rest of the design now follows according to section 8.4. In order to meet the requirement of a settling time of 10 seconds, a degree of relative stability (see figure 8.3) $\delta = \frac{3}{10} = 0.3$ is selected. Hence, using the methods [153], [13], [168] it is found that with a value of $\delta \to +\infty$, $R_f = \begin{bmatrix} 1.4111e+001 & -7.0551e+001 \\ -7.0551e+001 & 3.5270e+002 \end{bmatrix}$, $R_2 = 1$ a state feedback law will place all the eigenvalues of the closed loop system in the grey region of figure 8.4. Although, $R_f$ can be further modified such that under a minimum $\theta > 0$ the eigenvalues of the closed loop system remain in the above region, simulations indicated very good results with $R_f$, $R_2$ as above and $\theta = 1.1$. Hence, these last values were finally chosen.

Subject to the LTR in section 8.4 we select $\ell^2 = 1000$. Hence, (8.62) gives $V_1 = \begin{bmatrix} 7.4 & 0 \\ 0 & 0 \end{bmatrix}$, $V_2 = 1$.

For the computation of (8.52)-(8.56), algorithm 8.2 is used. In this algorithm the setup in terms of the GQME-Toolbox for (8.52) and (8.56) is subject to a fixed homotopy without
scaling and is the same as in example 6.4 (see chapter 6). Furthermore, \( P_k^0 = P_{k-1} \), \( Q_k^0 = Q_{k-1} \) for \( k > 1 \), \( P_k^0 = Q_k^0 = 1e-008I_8 \), \( n_c = 1 \), \( e_{tol} = 1e-005 \).

Algorithm 8.2 converged in \( k = 57 \) steps in a total of 84.79 CPU sec. Computations took place in a two parallel PIII 550 MHz personal computer using MATLAB v. 5.3.1.29215a (R11.1) (Unix version) with unit roundoff \( u_r = 2^{-53} \) under the operating system RedHat Linux v. 6.2. Next only the solutions \( P \), \( Q \) and \( \tau_1 \), \( \tau_2 \) are shown and a posteriori numerical analysis for \( P \) and \( Q \) is presented similar to example 6.4 in chapter 6. Note that, having this information the reader can easily evaluate \( \hat{P} \) and \( \hat{Q} \) and therefore the controller gains in theorem 8.1.

It was found that

\[
P = \begin{bmatrix} 1.6080 e + 002 & -6.0153 e + 002 \\ -6.0153 e + 002 & 2.8561 e + 003 \end{bmatrix}, \quad Q = \begin{bmatrix} 8.8253 e + 000 & -3.1540 e - 001 \\ -3.1540 e - 001 & 5.1918 e - 001 \end{bmatrix},
\]

\[
\tau_1 = \begin{bmatrix} -6.8661 e - 001 \\ -7.2703 e - 001 \end{bmatrix}, \quad \tau_2 = \begin{bmatrix} 7.7571 e - 001 \\ -2.1080 e + 000 \end{bmatrix}.
\]

The numerical analysis results are shown in table 8.1 below. The notation used here is the one adopted from the previous chapters.

<table>
<thead>
<tr>
<th>( n.s.n.p )</th>
<th>( X = P )</th>
<th>( X = Q )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \det(J_{X_c}) )</td>
<td>1.8338e - 001</td>
<td>1.5469e + 000</td>
</tr>
<tr>
<td>( |\text{Residual}(X_c)|_F )</td>
<td>3.9785e - 015</td>
<td>6.2075e - 015</td>
</tr>
<tr>
<td>( |X_c|_F )</td>
<td>1.0703e + 002</td>
<td>1.8078e + 001</td>
</tr>
<tr>
<td>( C_n(X_c) )</td>
<td>4.5482e - 016</td>
<td>8.6787e - 016</td>
</tr>
<tr>
<td>( E_{prf_{\text{max}}}(X_c) )</td>
<td>6.1475e - 015</td>
<td>5.1923e - 015</td>
</tr>
<tr>
<td>( E_{prf_{\text{limax}}}(X_c) )</td>
<td>2.3079e - 015</td>
<td>4.6085e - 015</td>
</tr>
<tr>
<td>( E_{prb_{\text{max}}}(X_c) )</td>
<td>3.5116e - 014</td>
<td>4.5059e - 014</td>
</tr>
<tr>
<td>( E_{prb_{\text{limax}}}(X_c) )</td>
<td>3.6299e - 013</td>
<td>2.0646e - 013</td>
</tr>
<tr>
<td>( NS )</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( NS_{fl} )</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( e_{tol_{IVP}} )</td>
<td>1.0000e - 012</td>
<td>1.0000e - 012</td>
</tr>
</tbody>
</table>

Table 8.1. Computation results.
Table 8.1 above indicates that the GQME-Toolbox provided computations which are numerically stable and accurate.

A simulation of the closed loop system subject to a reference power from 0 KW to 1.8 KW and then to 400 KW; that is

\[ r(t) = \begin{cases} 
1.8, & 0 \leq t \leq 30 \\
0.4, & t > 30 
\end{cases} \text{ KW}, \]

subject to

\[ x(0) = \begin{cases} 
0, & 0 \leq t \leq 30 \\
r(30), & t > 30 
\end{cases} \]

\[ w_p(t) = \begin{cases} 
0.001, & 0 \leq t \leq 5 \\
0, & t > 5 
\end{cases} \]

is shown in figure 8.4.

![Figure 8.4. Closed loop simulation.](image)

8.6 Robust Control of a Synchronous Electric Generator/Infinite Bus Power System

The dynamic stability of a synchronous electric generator connected to an infinite bus through a transmission line (see figure 8.5) has been investigated in great detail in [41], [50], [9], [20], [129], [148] and their references.

In such a system, load disturbances, self-excited oscillations and other phenomena, which perturb the system from its normal operation, can drive the system actuators into undesirable saturation. This is true, even when the system operates under relatively small variations of its dynamics.

So far the majority of controller designs aim to limit the feedback control action in such a way that saturation is unlikely to occur. For example, power system stabilizers put limits on voltage feedback signals in order to avoid exciter saturation [129]. In linear quadratic regulator (LQR) and \( H_2 \) designs, the weight selections are made in such a way that the control signal does not
cause saturation. The above strategy usually gives reasonable designs in terms of saturation avoidance but in some cases, it may limit the system performance.

![Figure 8.5. System under study.](image)

### 8.6.1 System Description and Modeling

In this section, the description of the system under study is given together with numerical data and design specifications. (see [41], [9], [50], [129], [88], [89], [169] and [170]).

Essentially, the control of the steady state normal operation of a synchronous electric generator connected to an infinite bus through a transmission line is implemented with two major control loops: an automatic voltage regulator (AVR) and an automatic load-frequency controller (ALFC) [50], [9].

| **TABLE 8.2**  
| **System Nomenclature** |
| --- | --- | --- | --- |
| $e_t$ | Terminal Voltage | $R$ | Regulation Droop |
| $E$ | Infinite Bus voltage | $G$ | Exciter Gain |
| $X_e$ | Series Transmission Line Reactance | $P$ | Real Power |
| $X_d$ | d-axis Synchronous Reactance | $Q$ | Reactive Power |
| $X'_d$ | d-axis Transient Reactance | $T_e$ | Electromagnetic Torque |
| $X_q$ | q-axis Synchronous Reactance | $\delta$ | Rotor Angle |
| $T_G$ | Speed Governor Time Constant | $\omega$ | Rotor Speed |
| $T_T$ | Turbine Time Constant | $E_q$ | Generated Field Voltage |
| $T_{do}$ | Generator Field Time Constant | $T_m$ | Prime Mover Torque |
| $M$ | Inertia Coefficient | $X_E$ | Steam Control Valve |
| $D$ | Damping Coefficient | $E_{fd}$ | Excitation Voltage |
| $T_L$ | Load Disturbance | $w_2$ | Measurement Disturbance |

*Table 8.2. System Nomenclature.*
A power system transfer function block diagram is shown with these two loops in figure 8.6 and from now on it will be referred to as the plant. Useful nomenclature is given in table 8.2.

![Block diagram of the plant under study in the Laplace domain.](image)

**Figure 8.6.** Transfer function block diagram of the plant under study in the Laplace domain.

It is supposed that the system operates under small dynamic variation. A linear mathematical model in the form of (8.40), (8.41) can be obtained (see [41], [88], [89], [9], [20]) with:

\[ x_p(t) = \begin{bmatrix} \Delta \delta(t) & \Delta \omega(t) & \Delta E_q(t) & \Delta T_m(t) & \Delta X_E(t) & \Delta E_{fd}(t) \end{bmatrix}^T, \quad u(t) = \begin{bmatrix} \Delta u_1(t) & \Delta u_2(t) \end{bmatrix}^T, \]
Chapter 8  Case Studies in Robust Control System Design: Applications 330 to Fast Modern Electro-technological Processes and Power Systems

\[ y_p(t) = [\Delta \xi(t) \Delta \delta(t)]^T, \quad w_{p2}(t) = [\Delta w_{p21}(t) \Delta w_{p22}(t)]^T, \]

\[ A_p = \begin{bmatrix} 0 & 314 & 0 & 0 & 0 & 0 \\ -K_1 & -D & -K_2 & 0 & 0 & 0 \\ M & M & M & M & 0 & 0 \\ -K_4 & 0 & 0 & 0 & 1 & -1 \\ T_{do} & T_{do}K_3 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad B_p = \begin{bmatrix} 0 & 0 & 0 & 0 & G/T^T \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \]

\[ \Gamma_{p1} = \begin{bmatrix} 0 & -1/M & 0 & 0 & 0 & 0 \end{bmatrix}^T, \quad C_p = \begin{bmatrix} K_5 & 0 & K_6 & 0 & 0 & 0 \end{bmatrix}, \quad \Gamma_{p2} = \begin{bmatrix} \Gamma_{p21} & 0 \\ 0 & \Gamma_{p21} \end{bmatrix}. \]

In the above notation \( \Delta \alpha := \alpha - \alpha^0 \) denotes the deviation of \( \alpha \) from its normal value of operation \( \alpha^0 \). The constants \( K_1 - K_6 \) can be calculated from formulas in terms of the system numerical data and normal point of operation [41].

Now, the present system above contains the actuator dynamics (i.e. exciter and turbine valve) as part of the model (8.40), (8.41). More precisely, the actuator model is associated with the last two rows of the present model. Hence, the actuator model can be written in the form of (8.1), (8.2) as

\[ x_a(t) = [\Delta X_E(t) \Delta E_{fd}(t)]^T, \quad w_a(t) = [\Delta \delta(t) \Delta \omega(t) \Delta E_q(t)]^T, \]

\[ A_a := \begin{bmatrix} -1 & 0 \\ T_G & 0 \end{bmatrix}, \quad B_a := \begin{bmatrix} 0 & 1/T_G \\ G/T & 0 \end{bmatrix}, \quad \Gamma_{\alpha1} := \begin{bmatrix} 0 & -1/R_T \\ -G K_5/T & 0 \\ -G K_6/T & 0 \end{bmatrix}, \quad \Gamma_{\alpha2} := 0_{2 \times 2}. \]

\[ C_a := I_2, \quad D_a := 0_{2 \times 2}, \quad \Gamma_{\alpha1} := 0_{2 \times 2}. \]

In order to track the output variations from the normal operation of the plant to zero, integrators are added to the plant’s output. These integrators are mathematically expressed in terms of the system (8.44), (8.45) with

\[ A_I := \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}, \quad x_I(0) := 0_{2 \times 2}, \quad \Gamma_I := \begin{bmatrix} \Gamma_{I1} & 0 \\ 0 & \Gamma_{I2} \end{bmatrix}, \quad w_I(t) := \begin{bmatrix} w_{I1}(t) \\ w_{I2}(t) \end{bmatrix}. \]

The system numerical data is taken from [146] as: \( X_d = 1.03 \) pu, \( \dot{X}_d = 0.247 \) pu, \( X_q = 0.612 \) pu, \( T_{do} = 6.05 \) sec, \( X_e = 0.3 \) pu, \( G = 100 \), \( T = 0.05 \) sec, \( T_T = 1 \) sec, \( T_G = 0.1 \) sec,
\( R = 0.04 \text{ pu}, \ D = 3, \ M = 10 \text{ sec} \), where pu denotes per unit format [50]. The normal steady state operating point is subject to a lagging power factor \( P + Qi = 1.2 + 0.4i \text{ pu} \) and a terminal voltage 1 pu. With respect to the previous values, \( K_1 - K_6 \) are computed [41] as: \( K_1 = 1.0515, \ K_2 = 1.2960, \ K_3 = 0.5997, \ K_4 = 1.0147, \ K_5 = -0.1408, \ K_6 = 0.4824 \).

For the present study let the saturation of the exciter and the turbine steam control valve have the following limits:

**Exciter:**
\[ \Delta E_{fd max} = 0.5 \text{ pu}, \ \Delta E_{fd min} = -0.5 \text{ pu}. \]

**Turbine steam control valve:**
\[ \Delta X_E max = 0.05 \text{ pu}, \ \Delta X_E min = -0.05 \text{ pu}, \ \Delta X_E max = 0.5 \text{ pu/sec}, \ \Delta X_E min = -0.5 \text{ pu/sec}. \]

Hence, according to subsection 8.2.1, \( \bar{x}_a = [0.05 \ 0.5]^T, \ \bar{x}_a = [+\infty \ 0.5]^T. \)

Finally, the measurement disturbance matrix is equal to \( \Gamma_{p_2} = I_2. \)

Considering all the previous data the overall open loop system model in the form of (8.46), (8.47) is determined as

\[
A = \begin{bmatrix}
0 & 3.1400e+002 & 0 & 0 & 0 & 0 & 0 \\
-1.0515e-001 & -3.0000e-001 & -1.2960e-001 & 1.0000e-001 & 0 & 0 & 0 \\
-1.6772e-001 & 0 & -2.7558e-001 & 0 & 0 & 1.6529e-001 & 0 & 0 \\
0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \\
0 & -2.5000e+002 & 0 & 0 & -1 & 0 & 0 & 0 \\
2.8177e+002 & 0 & -9.6483e+002 & 0 & 0 & -20 & 0 & 0 \\
1.4088e-001 & 0 & -4.8242e-001 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 
\end{bmatrix}
\]

\[
B = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
10 & 0 & 0 & 0 & 0 & 0 \\
2000 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 
\end{bmatrix}
\]

\[ \Gamma_1 = \begin{bmatrix}
I_6 & 0_{6 \times 2} \\
0_{2 \times 6} & -I_2 
\end{bmatrix}, \ C = \begin{bmatrix}
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
1 & 0 \\
0 & 1 
\end{bmatrix}, \ \Gamma_2 = I_2. \]

The eigenvalues of the open loop system (i.e. the eigenvalues of \( A \) in (8.46) are:

\[
4.3696e-001 \pm 6.4385e+000i, \ 0, \ 0, \ -9.4298e-001, \ -1.0170e+001, \ -1.0668e+001 \pm 8.1620e+000i.
\]

Obviously, the open loop system at the present operating condition is unstable.

In the present study, the control problem is the design of a control system to stabilize and optimize the dynamic response of the open loop system, under relatively small deviations of
its normal operation and under the action of small load disturbances. The control system should be able to avoid saturation of the system exciter and turbine, and to provide settling times of 1 to 2 sec approximately under reasonable disturbance situations [50] and less than 15 sec under large disturbances.

8.6.2 Control System Computation

In order to avoid saturation, since $C_a = I_2$, (8.36), (8.37), (8.39) are used in order to limit the control signal according to subsection 8.2.1. Note that in all expected cases the penalization of states during the design computation will be such that

$$|w_a(t)| = \left[ |\Delta \delta(t)| \quad |\Delta \phi(t)| \quad |\Delta E_q(t)| \right]^T \leq [0.01 \quad 0.001 \quad 0.001]^T, \quad \forall t \in \mathbb{R}_+.$$  

With respect to the above data and assuming $x_a(0) = 0_{2 \times 1}$, the maximum limit of the control signal for which (8.37) and (8.39) are both satisfied is found to be $\mathbf{u}_* = \begin{bmatrix} 3.109e-003 \\ 2.500e-002 \end{bmatrix}$.

The rest of the design now follows according to section 8.4. In order to meet the requirement of settling time between of 1 and 2 sec a degree of relative stability (see figure 8.3) $h = \frac{3}{1.5} = 2$ is selected. Hence, using the methods [153], [13], [166] it is found that with minimum value of $\vartheta = 5.3622$ and $R_2 = I_2$, $R_1 = \begin{bmatrix} R_{11} & R_{12} \\ R_{21}^T & R_{13} \end{bmatrix}$, where

$$R_{11} = \begin{bmatrix} 1.8440e+000 & -2.2592e+001 & 2.3025e+000 & -1.9992e+000 \\ -2.2592e+001 & 8.0524e+003 & -1.3063e+002 & 1.1870e+002 \\ 2.3025e+000 & -1.3063e+002 & 4.4689e+000 & -2.2043e+000 \\ -1.9992e+000 & 1.1870e+002 & -2.2043e+000 & 1.4783e+001 \end{bmatrix},$$

$$R_{12} = \begin{bmatrix} -1.5436e-001 & 4.8018e-003 & -2.6190e+000 & -1.1428e+000 \\ 4.8886e+000 & -2.2504e-001 & -4.3759e+002 & -3.8270e+002 \\ -6.8331e-002 & 8.6332e-003 & -1.4966e+000 & 2.1768e+000 \\ 1.4286e+000 & -4.6279e-003 & 3.5262e+001 & -1.6286e+001 \end{bmatrix},$$

$$R_{13} = \begin{bmatrix} 1.4587e-001 & -1.8757e-004 & -3.4185e+000 & -1.4117e+000 \\ -1.8757e-004 & 9.9707e-004 & -4.4216e-003 & 2.7165e-003 \\ -1.8757e-004 & -4.4216e-003 & 1.2700e+002 & 6.3582e+001 \\ -1.4117e+000 & 2.7165e-003 & 6.3582e+001 & 3.6782e+001 \end{bmatrix},$$

a state feedback law will place all the eigenvalues of the closed loop system in the grey region of figure 8.4 subject to $h = 2$.
Subject to the LTR in section 8.4 we select $\ell^2 = 10$. Hence, (8.62) gives $V_1 = \text{diag}([1 \ 0.01 \ 1 \ 1001 \ 4e+007 \ 1 \ 1])$, and $V_2 = \Gamma_2 \Gamma_2^T = I_2$.

For the computation of (8.52)-(8.56), algorithm 8.2 is used. In this algorithm the setup in terms of the GQME-Toolbox for (8.52) and (8.56) is subject to a fixed homotopy and is the same as in example 6.4 (see chapter 6). Furthermore, scaling (3.31) (see chapter 3) was implemented on each iteration of the algorithm 8.2, and $P_k = P_{k-1}, \ Q_k = Q_{k-1}$ for $k > 1$.

Algorithm 8.2 converged in $k = 3$ steps in a total of 289.55 CPU sec. Computations took place in a two parallel PIII 550 MHz personal computer using MATLAB v. 5.3.1.29215a (R11.1) (Unix version) with unit roundoff $u_r = 2^{-53}$ under the operating system RedHat Linux v. 6.2. Next only the solutions $P, Q$ and $\tau_1, \tau_2$ are shown and a posteriori numerical analysis for $P$ and $Q$ is presented similar to example 6.4 in chapter 6. Note that, having this information the reader can easily evaluate $\hat{P}$ and $\hat{Q}$ and therefore the controller gains in theorem 8.1.

It was found that

\[
P = \begin{bmatrix} P_1 & P_2 \\ P_2^T & P_3 \end{bmatrix}, \quad Q = \begin{bmatrix} Q_1 & Q_2 \\ Q_2^T & Q_3 \end{bmatrix}, \quad \tau_1 = \begin{bmatrix} \tau_{11} & \tau_{12} \end{bmatrix}, \quad \tau_2 = \begin{bmatrix} \tau_{21} & \tau_{22} \end{bmatrix},
\]

where

\[
P_1 = \begin{bmatrix} 3.1203e-001 & 8.4145e+000 & 2.0442e-001 & -2.0563e-002 \\ 8.4145e+000 & 2.7925e+003 & -2.3230e+001 & 4.7706e+001 \\ 2.0442e-001 & -2.3230e+001 & 6.4245e-001 & -2.6624e-001 \\ -2.0563e+002 & 4.7706e+001 & -2.6624e-001 & 4.0561e+000 \end{bmatrix},
\]

\[
\]

\[
P_3 = \begin{bmatrix} 3.8913e-002 & -1.6240e-005 & -1.1514e+000 & -5.6006e-001 \\ -1.6240e-005 & 1.4929e-005 & 4.7106e-004 & 1.3689e-003 \\ -1.1514e+000 & 4.7106e-004 & 4.6517e+001 & 2.4666e+001 \\ -5.6006e-001 & -1.3689e-003 & 2.4666e+001 & 1.4344e+001 \end{bmatrix},
\]

\[
Q_1 = \begin{bmatrix} 3.7813e+002 & 2.1446e+000 & 9.2157e+000 & -1.3525e+001 \\ 2.1446e+000 & 1.1899e-001 & -2.4783e+000 & 7.2180e-002 \\ 9.2157e+000 & -2.4783e+000 & 1.7558e+002 & -2.1302e+000 \\ -1.3525e+001 & 7.2180e-002 & -2.1302e+000 & 5.4636e+000 \end{bmatrix}.
\]
The numerical analysis results are shown in table 8.3. The notation used here is the one adopted from the previous chapters. From table 8.3 below it is clear that the GQME-Toolbox provides computations which can be fairly characterized as numerically stable and robust. But note the very large condition numbers that indicate an ill conditioned numerical problem. To this end, it should be said that experiments without using scaling in the computations have increased the computation times dramatically. This is possibly attributed to the ill conditioning of the problem, which may be removed with scaling (3.31).

\[
Q_2 = \begin{bmatrix}
-1.0212e+002 & 5.8027e+003 & 1.9833e+001 & -3.0954e+001 \\
-1.9389e+000 & 1.1149e+002 & 2.2019e+001 & 3.2328e-001 \\
2.1970e+001 & 2.0905e+002 & -5.6386e+000 & -6.9849e+000 \\
5.7587e+000 & -1.4605e+002 & -1.1161e+000 & 1.3401e+000 \\
\end{bmatrix}
\]

\[
Q_3 = \begin{bmatrix}
1.0354e+002 & -2.6160e+003 & -6.2233e+000 & 7.3997e-001 \\
-2.6160e+000 & 1.0739e+006 & 4.8683e+002 & -2.4878e+002 \\
-6.2233e+000 & 4.8683e+002 & 2.8671e+000 & -1.9795e+000 \\
7.3997e-001 & -2.4878e+002 & -1.9795e+000 & 7.6888e+000 \\
\end{bmatrix}
\]

\[
\tau_{I_1} = \begin{bmatrix}
-1.3327e-001 & -1.2438e-003 & -1.0462e-001 & 1.2012e-002 \\
3.6757e-001 & -1.2341e-002 & 5.7796e-001 & 8.1301e-002 \\
4.4072e-002 & 1.2565e-004 & 4.5225e-002 & 1.1556e-002 \\
1.0312e-001 & 1.8480e-003 & -7.7165e-002 & -1.5694e-002 \\
1.2513e-001 & 1.7761e-003 & 9.4110e-002 & -1.0592e-002 \\
\end{bmatrix}
\]

\[
\tau_{I_2} = \begin{bmatrix}
2.6031e-002 & -4.3783e-001 & 2.2373e-001 & -5.3072e-001 \\
\end{bmatrix}
\]

\[
\tau_{2_1} = \begin{bmatrix}
1.3082e+000 & 3.2668e+002 & -1.4480e+000 & 5.3110e+001 \\
1.0464e+000 & -5.1774e+000 & -5.3338e-001 & 2.1977e+000 \\
-2.1982e-014 & 1.6470e+001 & 2.9172e+000 & 4.8541e+001 \\
6.0194e+000 & -2.6502e+001 & -6.9216e+000 & 4.8046e+000 \\
-1.9629e-013 & -3.1864e+001 & 7.5551e+000 & 1.2470e+001 \\
\end{bmatrix}
\]

\[
\tau_{2_2} = \begin{bmatrix}
5.5011e-000 & -1.2297e-002 & -6.9271e+000 & -2.9053e+001 \\
2.2744e-001 & -1.5190e-002 & -8.1011e+000 & -4.6747e+000 \\
5.4640e+000 & 7.2765e-002 & 4.9691e+001 & 3.1386e+001 \\
4.9113e-001 & -1.3073e-001 & 3.6172e+000 & -3.5968e-001 \\
1.5011e+000 & 1.1836e-001 & -1.7925e+001 & 3.2983e+000 \\
\end{bmatrix}
\]
<table>
<thead>
<tr>
<th>TABLE 8.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>( X ) (s.n.p)</td>
</tr>
<tr>
<td>( \det(J_{X_c}) )</td>
</tr>
<tr>
<td>( | Residual(X_c) |_F )</td>
</tr>
<tr>
<td>( | X_c |_F )</td>
</tr>
<tr>
<td>( C_n(X_c) )</td>
</tr>
<tr>
<td>( E_{\text{prf} \max}(X_c) )</td>
</tr>
<tr>
<td>( E_{\text{prf} \min}(X_c) )</td>
</tr>
<tr>
<td>( E_{\text{rb}}(X_c) )</td>
</tr>
<tr>
<td>( E_{\text{prb} \max}(X_c) )</td>
</tr>
<tr>
<td>( N_S )</td>
</tr>
<tr>
<td>( N_S \text{fl} )</td>
</tr>
<tr>
<td>( e_{\text{tol IVP}} )</td>
</tr>
</tbody>
</table>

Table 8.3. Computation results.

A simulation of the closed loop system subject to \( x(0) = [0.001 0.001 0 0 0 0 0 0]^T \),

\[
w_1(t) = \begin{cases} 
[0 0.02 0 0 0 0 0.0001 0.0001]^T, & 0 \leq t \leq 5, \\
0_{8 \times 1}, & t > 5 
\end{cases}
\]

\[
w_2(t) = \begin{cases} 
[0.0001 0.0001]^T, & 0 \leq t \leq 5, \\
0_{2 \times 1}, & t > 5 
\end{cases}
\]

is shown in figure 8.7.

8.7 Synopsis

In this chapter it was illustrated how the GQME-Toolbox and the whole theoretical developments of this thesis can be applied to the computation of robust control systems. Also, a robust control design method has been proposed and implemented via two examples from industry. Results indicated both numerical accuracy and stability for the computed designs.
Figure 8.7. Closed loop simulation.
The solution of (8.52)-(8.56) in theorem 8.1 constitutes a non-convex problem as was pointed out in [173]. That means that there are more than one non-negative definite solutions for each of (8.52)-(8.56). Early computations in [168], have showed that if the initial conditions in GQMEHOM1.m are very small (close to zero) nonnegative matrices, in each step of algorithms 8.1 or 8.2, then the solutions obtained correspond to the global minimum cost in the control system design. This is an empirical observation and more research has to be done in order to establish rigorously which initial conditions lead to the global minimum cost.

Finally, it should be stated that there are many other control design problems that are associated with equations similar to (8.52)-(8.56). Some of these are [71]-[74] and [87], which similarly can be solved using the GQME-Toolbox as shown in this chapter. To this end, note that the algorithms proposed for the computation of such designs in the previous references are sensitive due to the initialization. This is in contrast to the approach presented here. In addition, in these other methods, numerical analysis studies of the computed solutions are not provided at all. Hence, it is difficult to judge the numerical stability and accuracy of these algorithms. Fortunately, this can now be overcome with the general results of chapter 4 of the present thesis. Therefore, a large variety of control design computations can be numerically checked and conclusions about the numerical behaviour of the computation methods can take place. It should also be said that the numerical examples of [21], [37] have been solved successful using both algorithms 8.1 and 8.2 of section 8.4 of this chapter. These were not presented here, but the reader can verify this using the GQME-Toolbox.
Chapter 9

Numerical Solution & Analysis of
Coupled Generalized Algebraic Quadratic Matrix Equations

9.1 Introduction

In this short chapter, it will be shown how a system of coupled generalized algebraic quadratic matrix equations can be equivalently written as equation (2.1). Therefore, the numerical solution and numerical analysis of such systems follows from chapters 2-4.

Systems of coupled algebraic quadratic matrix equations arise in various multiobjective control system design problems and in multivariable game theory. A list of relevant references and comments will be provided in the sequel to this chapter. The analysis in the present chapter will concentrate on two coupled generalized algebraic quadratic matrix equations. Note that the extension to systems of more than two equations follows in a similar manner.

At this point it should be said that for a system of two or more coupled generalized algebraic quadratic matrix equations, the GQME-Toolboox (see appendix C) must be modified in such a way that GQMEHOM.m (see chapter 5, appendix C) can accommodate more than two linear and quadratic terms. Currently the software is limited to a maximum of $\omega = \xi = 2$ linear and quadratic terms. For this reason numerical examples will not presented. The modifications of the GQME-Toolboox and all the relevant investigations for such systems will be the subject of future research.
Chapter 9 Numerical Solution & Analysis of Coupled Generalized Algebraic Quadratic Matrix Equations

Section 9.2 will present the problem of two-coupled matrix equations. Next, in section 9.3 it will be discussed how various control and game theory design problems can be formulated and effectively solved. Conclusions and discussions will be given in the last section 9.4.

9.2 Coupled Generalized Quadratic Algebraic Matrix Equations

Consider the following system of two coupled generalized quadratic algebraic matrix equations

\[
\begin{align*}
\sum_{i=1}^{\omega_1} A_{i1} X_1 B_{i1} + \sum_{i=1}^{\xi_1} C_{i1} X_1 D_{i1} X_1 E_{i1} + \sum_{i=1}^{\omega_2} K_{i1} X_1 L_{i1} + \sum_{i=1}^{\xi_2} M_{i1} X_1 N_{i1} X_2 P_{i1} \\
+ \sum_{i=1}^{\omega_3} Q_{i1} X_1 R_{i1} X_2 S_{i1} + \sum_{i=1}^{\xi_3} T_{i1} X_2 V_{i1} X_1 W_{i1} + G_j = 0, \quad j = 1, 2,
\end{align*}
\]

where \(\omega_1, \omega_2, \xi_1, \xi_2, \xi_3, \xi_4 \in \mathbb{N}, \omega_j + \xi_j + \xi_j + \xi_j + \xi_j + \xi_j + \xi_j \not= 0, \ n, p \in \mathbb{Z}_+,\)

and \(A_{i1}, C_{i1}, K_{i1}, M_{i1}, Q_{i1}, T_{i1} \in \mathbb{C}^{n \times n}, B_{i1}, E_{i1}, L_{i1}, P_{i1}, S_{i1}, W_{i1} \in \mathbb{C}^{p \times p},\)

\(D_{j1} \cdot N_{j1} \cdot R_{j1} \cdot V_{j1} \in \mathbb{C}^{p \times n}, G_j \in \mathbb{C}^{n \times p},\) are the constant matrix coefficients of the equation

and \(X_1, X_2 \in \mathbb{C}^{n \times p}\) are the unknowns.

It is obvious that the equalities below hold.

\[
\begin{align*}
\sum_{i=1}^{\omega_1} A_{i1} X_1 B_{i1} + \sum_{i=1}^{\xi_1} C_{i1} X_1 D_{i1} X_1 E_{i1} + \sum_{i=1}^{\omega_2} K_{i1} X_1 L_{i1} + \sum_{i=1}^{\xi_2} M_{i1} X_1 N_{i1} X_2 P_{i1} = 0 \\
+ \sum_{i=1}^{\omega_3} Q_{i1} X_1 R_{i1} X_2 S_{i1} + \sum_{i=1}^{\xi_3} T_{i1} X_2 V_{i1} X_1 W_{i1} + G_j = 0, \quad j = 1, 2,
\end{align*}
\]

where \(\omega_1, \omega_2, \xi_1, \xi_2, \xi_3, \xi_4 \in \mathbb{N}, \omega_j + \xi_j + \xi_j + \xi_j + \xi_j + \xi_j + \xi_j \not= 0, \ n, p \in \mathbb{Z}_+,\)

and \(A_{i1}, C_{i1}, K_{i1}, M_{i1}, Q_{i1}, T_{i1} \in \mathbb{C}^{n \times n}, B_{i1}, E_{i1}, L_{i1}, P_{i1}, S_{i1}, W_{i1} \in \mathbb{C}^{p \times p},\)

\(D_{j1} \cdot N_{j1} \cdot R_{j1} \cdot V_{j1} \in \mathbb{C}^{p \times n}, G_j \in \mathbb{C}^{n \times p},\) are the constant matrix coefficients of the equation

and \(X_1, X_2 \in \mathbb{C}^{n \times p}\) are the unknowns.
Chapter 9 Numerical Solution & Analysis of Coupled Generalized Algebraic Quadratic Matrix Equations

\[
\begin{align*}
\sum_{i=1}^{\xi_{12}} M_{i1} X_1 N_{i1} X_2 P_{i1} & + \sum_{i=1}^{\xi_{12}} M_{i2} X_1 N_{i2} X_2 P_{i2} \\
\sum_{i=1}^{\xi_{13}} Q_{i1} X_1 R_{i1} X_2 S_{i1} & + \sum_{i=1}^{\xi_{13}} Q_{i2} X_1 R_{i2} X_2 S_{i2} \\
\sum_{i=1}^{\xi_{14}} T_{i1} X_1 V_{i1} X_2 W_{i1} & + \sum_{i=1}^{\xi_{14}} T_{i2} X_1 V_{i2} X_2 W_{i2}
\end{align*}
\]

(9.5) (9.6) (9.7)

From the above it is apparent that, in view of (9.2)-(9.7) the system (9.1) can be written as (2.1) with \( X := \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}, \quad \omega = \sum_{i=1}^{2} \left( \sum_{j=1}^{2} \left( \alpha_{ij} \right) \right), \quad \xi = \sum_{i=1}^{4} \left( \sum_{j=1}^{4} \left( \xi_{ij} \right) \right). \) Hence, system (9.1) can be solved with the methods of chapters 2 and 3 and numerically analyzed as proposed in chapter 4.

Note that now, (2.1) in general consists of two linear and four quadratic quantities. For this reason the GQME-Toolbox cannot be used as is given in appendix C simply because it considers only two quadratic quantities in (2.1). In order to implement (9.1) as (2.1) in GQME-Toolbox, GQMEHOM1.m has to be modified slightly to incorporate another two quadratic quantities (i.e. another six matrices). The same is true for SGQME.m, NAGQME.m and all the relevant functions (subroutines) in GQMEHOM1.m. In the case of more than two coupled generalized algebraic quadratic matrix equations system, the process is similar. Again, these equations can be written as (2.1) with \( X := \begin{bmatrix} X_1 & \cdots & X_k \end{bmatrix}, \) where \( k \in N_0 \) is the number of equations.

The advantage of the above formulation is that equations are written in compact form and hence can be more easily manipulated in terms of analysis and algebra. The disadvantage is that the dimension of (2.1) is proportionally increased with the number of equations. However, whatever the formulation this is unavoidable by nature.
9.3 Applications to Control and Game Theory Design Problems

9.3.1 Multi-objective Control Systems Design Computations

With the formulation in section 9.2, various multi-objective control design problems can be computed. For example, [21], [49], [54], [68], [71]-[74], [87], [111], [172], [173], [174].

The main computational task in all the above problems subject to the design of full order controllers is usually to solve the following set of coupled algebraic quadratic matrix equations.

\[ f_2(X_1, X_2) = A_1 X_1 + X_1 A_1^T + B_1 X_2 X_1 + X_1 X_2 B_2 - X_1 B_2 X_1 + G_1 = 0 \] (9.8)

and

\[ f_2(X_1, X_2) = \tilde{A}_2 X_2 + X_2 \tilde{A}_2 - \tilde{B}_2 X_1 X_2 - X_2 X_1 \tilde{B}_2 + X_2 \tilde{B}_1 X_2 + \tilde{G}_2 = 0 \] (9.9)

where \( \forall i = 1, 2 \), \( X_i, A_i, B_i, G_i \in \mathbb{R}^{n \times n}, n \in \mathbb{N}, X_i \geq 0, B_i > 0, G_i \geq 0 \).

When the fixed order compensator design is addressed in the above problems than (9.8) and (9.9) are modified as

\[ f_2(X_1, X_2) = \tilde{A}_1 X_1 + X_1 \tilde{A}_1^T + \tilde{B}_1 X_2 X_1 + X_1 X_2 \tilde{B}_2 - X_1 \tilde{B}_2 X_1 + \tau_1 X_1 \tilde{B}_2 X_1 \tau_1^T + \tilde{G}_1 = 0 \] (9.10)

and

\[ f_2(X_1, X_2) = \tilde{A}_2 X_2 + X_2 \tilde{A}_2 - \tilde{B}_2 X_1 X_2 - X_2 X_1 \tilde{B}_2 + X_2 \tilde{B}_1 X_2 - \tau_2^T X_2 \tilde{B}_1 X_2 \tau_2 + \tilde{G}_2 = 0 \] (9.11)

where \( \tau_\perp \) is an idempotent matrix as in chapter 8.

According to section 9.2, (9.10) and (9.11) can be formulated as a special case of (2.1) as shown below.

\[
\begin{bmatrix}
\tilde{A}_1 & 0 \\
0 & \tilde{A}_2 \\
\end{bmatrix} X_1 I_n + 
\begin{bmatrix}
0 & X_1 \\
I_n & 0 \\
\end{bmatrix} A_1^T + 
\begin{bmatrix}
0 & A_2^T \\
0 & 0 \\
\end{bmatrix} X_1 I_n + 
\begin{bmatrix}
0 & 0 \\
I_n & 0 \\
\end{bmatrix} X_1 \tilde{A}_2
\]

\[
+ \begin{bmatrix}
I_n & 0 \\
0 & 0 \\
\end{bmatrix} X_1 \tilde{B}_2 + \begin{bmatrix}
0 & X_1 \\
I_n & 0 \\
\end{bmatrix} A_1^T + \begin{bmatrix}
0 & 0 \\
0 & X_1 \\
\end{bmatrix} \tilde{A}_2 + \begin{bmatrix}
0 & X_1 \\
I_n & 0 \\
\end{bmatrix} \tau_1^T
\]

\[
+ \begin{bmatrix}
0 & I_n \\
0 & 0 \\
\end{bmatrix} X_2 \tilde{B}_1 + \begin{bmatrix}
1 & I_n \\
0 & 0 \\
\end{bmatrix} X_2 \tilde{B}_2 + \begin{bmatrix}
0 & 0 \\
0 & I_n \\
\end{bmatrix} X_1 \tilde{B}_1 + \begin{bmatrix}
0 & I_n \\
0 & 0 \\
\end{bmatrix} X_2 \tilde{B}_2
\]

\[
\begin{bmatrix}
\tilde{G}_1 \\
\tilde{G}_2 \\
\end{bmatrix} = 0
\]
\[ f(X) = \sum_{i=1}^{4} A_i X B_i + \sum_{i=1}^{8} C_i X D_i X E_i + G = 0, \quad (9.12) \]

where

\[ \omega = 4, \quad \xi = 8, \quad f(X) = \begin{bmatrix} f_1(X_1, X_2) \\ f_2(X_1, X_2) \end{bmatrix}, \quad X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}. \]

Now, (9.12) is solved by the steps of algorithms 8.1 or 8.2 of chapter 8. The numerical analysis of the computed solutions follows from chapter 4.

### 9.3.2 Computation of Two Player Feedback Nash Equilibria in Infinite Horizon Multivariable LQ-Games

This section is the generalization of subsection 7.21 from the scalar to the multivariable (matrix) case.

**Problem 9.1:**

Consider the two-player quantitative infinite horizon dynamic game [15] defined in continuous time, and described by the state equation

\[ \frac{d(x(t))}{dt} := \bar{A}x(t) + \sum_{i=1}^{2} \bar{B}_i u_i(t), \quad x(0) = x_0. \]
where \( t \in \mathbb{R}_+ \cup \{0\} \), \( \hat{A} \in \mathbb{R}^{n \times n} \), \( \hat{B}_1, \hat{B}_2 \in \mathbb{R}^{n \times m} \), \( x(t) \in \mathbb{R}^n \), \( n, m \in \mathbb{N}_+ \), and for \( i = 1, 2 \) the cost functionals \( J_i(t) = \frac{1}{2} \int_0^{\infty} \left( x^T(t)G_ix(t) + \sum_{j=1}^{2} u_j^T(t)R_{ij}u_j(t) \right) dt \), where \( G_i \in \mathbb{R}^{n \times n} \), \( G_i \geq 0 \), \( R_{ij} \in \mathbb{R}^{m \times m} \), \( R_{ij} > 0 \). \( \forall i = 1, 2 \) find \( u_i(t) \in \mathbb{R}^m \), that provide a Nash equilibrium which minimizes \( J_i(t) \).

The solution to problem 9.1 is provided by theorem 9.1, which is the generalization of theorem 7.1 in chapter 7.

**Theorem 9.1 [15]:**

Let \( X_i \in \mathbb{R}^{n \times n} \), \( X_i \geq 0 \) the set of positive semidefinite solutions of the coupled equations

\[
\begin{aligned}
f_i(X_1, X_2) &= X_i \left( \hat{A} - \sum_{i=1}^{N} \left( \hat{B}_i R_{ii}^{-1} \hat{B}_i^T X_i \right) \right) + \left( \hat{A} - \sum_{i=1}^{N} \left( \hat{B}_i R_{ii}^{-1} \hat{B}_i^T X_i \right) \right)^T X_i \\
&\quad + \sum_{j=1}^{2} \left( X_j \hat{B}_j R_{jj}^{-1} \hat{B}_j^T X_j \right) + G_i = 0 \ , \ i = 1, 2 .
\end{aligned}
\]  
(9.13)

Furthermore, assume that \( \forall i = 1, 2 \),

\[
\begin{aligned}
\text{rank} \left[ \begin{bmatrix} \hat{A} - \sum_{j=1}^{2} \left( \hat{B}_j R_{jj}^{-1} \hat{B}_j^T X_j \right) \end{bmatrix} - \lambda I_n \ 
\hat{B}_i \end{bmatrix} \right] &= n , \ \text{rank} \left[ \begin{bmatrix} \hat{A} - \sum_{j=1}^{2} \left( \hat{B}_j R_{jj}^{-1} \hat{B}_j^T X_j \right) \end{bmatrix} - \lambda I_n \ 
G_i \end{bmatrix} \right] = n ,
\end{aligned}
\]

\( \forall \text{Re}(\lambda) \in \mathbb{R}_+ \cup \{0\} \).

Then, \( u(t) := R_{ii}^{-1} \hat{B}_i^T X_i x(t) \), \( i = 1, 2 \), provide a feedback Nash equilibrium, leading to the minimum costs \( \min_{t \in \mathbb{R}_+ \cup \{0\}} \{ J_i(t) \} = x_0^T X_i x_0 \), \( i = 1, 2 \). Moreover, the resulting system

\[
\frac{dx(t)}{dt} := \left( \hat{A} - \sum_{i=1}^{2} \left( \hat{B}_i R_{ii}^{-1} \hat{B}_i^T X_i \right) \right) x \ , \ x(0) = x_0 \text{ is asymptotically stable.}
\]

**Proof:**

The proof is given in [15].
Chapter 9  Numerical Solution & Analysis of Coupled Generalized Algebraic Quadratic Matrix Equations

The main computational task in the above problem is to solve the coupled algebraic quadratic matrix equations (9.13). Doing all the algebra in (9.13) we have equivalently the system of (9.14) and (9.15) below.

\[
\begin{align*}
\hat{A}^T X_1 + X_1 \hat{A} - X_1 \tilde{B}_1 R_{11} \tilde{B}_1^T X_1 - X_1 \tilde{B}_2 R_{22} \tilde{B}_2^T X_2 - X_2 \tilde{B}_2 R_{22} \tilde{B}_2^T X_1
+ & X_2 \tilde{B}_2 R_{22} R_{12} \tilde{B}_2^T X_2 + G_1 = 0 \\
\text{and} \\
\hat{A}^T X_2 + X_2 \hat{A} - X_2 \tilde{B}_2 R_{22} \tilde{B}_2^T X_2 - X_2 \tilde{B}_1 R_{11} \tilde{B}_1^T X_1 - X_2 \tilde{B}_1 R_{11} \tilde{B}_1^T X_1
+ & X_1 \tilde{B}_1 R_{11} \tilde{B}_1^T X_1 + G_2 = 0
\end{align*}
\]  (9.14)

According to section 9.2, (9.14) and (9.15) in view of (9.2)-(9.7) can be equivalently written as

\[
f(X) := \sum_{i=1}^{4} A_i X B_i + \sum_{i=1}^{8} C_i X D_i E_i + G = 0,
\]  (9.16)

where

\[
\omega = 4, \quad \xi = 8, \quad f(X) := \begin{bmatrix} f_1(X_1, X_2) \\
                        f_2(X_1, X_2) \end{bmatrix}, \quad X := \begin{bmatrix} X_1 \\
                                                  X_2 \end{bmatrix},
\]

\[
A_1 := \begin{bmatrix} \hat{A}^T & 0 \\
                      0 & 0 \end{bmatrix}, \quad A_2 := \begin{bmatrix} I_n & 0 \\
                                                  0 & 0 \end{bmatrix}, \quad A_3 := \begin{bmatrix} 0 & 0 \\
                                                  0 & \hat{A}^T \end{bmatrix}, \quad A_4 := \begin{bmatrix} 0 & 0 \\
                                                  0 & I_n \end{bmatrix},
\]

\[
B_1 := I_n, \quad B_2 := A, \quad B_3 := B_1, \quad B_4 := B_2,
\]

\[
C_1 := \begin{bmatrix} I_n & 0 \\
                      0 & 0 \end{bmatrix}, \quad C_2 := \begin{bmatrix} 0 & 0 \\
                                                  0 & I_n \end{bmatrix}, \quad C_3 := C_1, \quad C_4 := C_2,
\]

\[
C_5 := C_1, \quad C_6 := C_2, \quad C_7 := C_1, \quad C_8 := C_2,
\]

\[
D_1 := \begin{bmatrix} -\tilde{B}_1 R_{11} \tilde{B}_1^T & 0 \\
                      0 & \tilde{B}_1 R_{11} \tilde{B}_2 R_{22} \tilde{B}_2^T \end{bmatrix}, \quad D_2 := \begin{bmatrix} -\tilde{B}_1 R_{11} \tilde{B}_2 R_{22} \tilde{B}_2^T & 0 \\
                                                  0 & \tilde{B}_2 R_{22} \tilde{B}_2^T \end{bmatrix},
\]

\[
D_3 := \begin{bmatrix} \tilde{B}_2 R_{22} \tilde{B}_2 R_{22} \tilde{B}_2^T & 0 \\
                          0 & \tilde{B}_1 R_{11} \tilde{B}_1^T \end{bmatrix}, \quad D_4 := \begin{bmatrix} \tilde{B}_2 R_{22} \tilde{B}_2 R_{22} \tilde{B}_2^T & 0 \\
                                                  0 & \tilde{B}_1 R_{11} \tilde{B}_1^T \end{bmatrix},
\]

\[
D_5 := \begin{bmatrix} \tilde{B}_2 R_{22} \tilde{B}_2 R_{22} \tilde{B}_2^T & 0 \\
                          0 & \tilde{B}_1 R_{11} \tilde{B}_1^T \end{bmatrix}, \quad D_6 := \begin{bmatrix} -\tilde{B}_1 R_{11} \tilde{B}_1^T & 0 \\
                                                  0 & \tilde{B}_2 R_{22} \tilde{B}_2^T \end{bmatrix}, \quad D_7 := D_5, \quad D_8 := D_6,
\]

\[
E_1 := B_1, \quad E_2 := B_1, \quad E_3 := B_1, \quad E_4 := B_1, \quad E_5 := B_1, \quad E_6 := B_1, \quad E_7 := B_1, \quad E_8 := B_1,
\]

\[
G := \begin{bmatrix} G_1 \\
                  G_1 \end{bmatrix}.
\]

Hence, the numerical solution and analysis of (9.16) follow from chapters 2-4.
9.4 Synopsis

From the previous sections it is apparent that (2.1) can accommodate systems of coupled algebraic quadratic matrix equations. Moreover, it is clear that (2.1) can be viewed as a unified framework for the numerical analysis and solution of any system of quadratic algebraic matrix equations. Specialization within this framework provides solutions to many control system design and game theory problems as shown.

Future research can now proceed in implementing these problems in the GQME-Toolbox and in conducting experiments on some real world design problems. Since many of the above-mentioned problems result in symmetric equations, the computational burden can be reduced removing the symmetric parts from the computations. This will require further modifications in GQMEHOM1.m in the GQME-Toolbox (see chapter 5, appendix C).
Chapter 10

Epilogue

10.1 General Comments

This thesis has presented the numerical solution and analysis of a generalized quadratic matrix equation, namely (1.1), together with the development of relevant computer software. The contribution of the thesis can be roughly divided into three parts:

- The theoretical developments and design of algorithms (chapters 2, 3, 4, 9).
- Software design and code implementation (chapter 5).
- Applications to engineering and scientific problems (chapters 6, 7, 8, 9).

It has been shown that virtually all algebraic quadratic equations can be unified under (1.1). Hence, the thesis constitutes a unified framework for the numerical solution and analysis of generalized algebraic systems of quadratic equations.

The major numerical methods that are applied are the probability-1 homotopy methods. Based on these methods, algorithms for the numerical solution of (1.1) have been developed and numerical analysis studies have been conducted for the first time.

Numerical examples from various fields in engineering (robust control system design, modern electro-technological manufacturing, power systems) and science (systems theory, matrix algebraic Riccati equations, noncooperative game theory, quadratic eigenvalue problem, chemical kinetics, chaos theory) have indicated that the proposed methods are characterized by numerical robustness and efficiency and that they are proven to be good competitors with already existing state-of-the-art methods for special problems. It should be noted that, the methods presented in this thesis cover the general case of (1.1) rather than specific cases. This
allows the flexibility of including and adapting possible future problems within the proposed framework.

During the exposition of this thesis, detailed discussions and conclusions have been given along with summaries in the synopsis section of each chapter. They are therefore not repeated here. Instead, the thesis ends with thoughts on possible future research directions.

10.2 Future research

Future research should include further experimentation with the new results, using the GQME-Toolbox to solve and analyse real engineering and scientific problems, which can then be compared with other methods. To this end, note that because of the global convergence features of the algorithms the new software can be helpful when other software fails, or more importantly when no software is available.

Problems with complex data (not reported in this thesis) might be worth investigating. Relevant examples can be found in bifurcation computations of complex dynamical systems, (e.g. determining organizing centers for passive optical systems [145]). Although such cases have not been reported, initial experiments have been successfully conducted but more experiments will give a further insight into these problems.

The GQME-Toolbox is written for the case of (1.1) with two linear and two quadratic parts and therefore modifications are required in order to accommodate more parts, which would be useful for mixed objective control system design and game theory problems (see chapter 9). For example, the general algebraic Riccati equation for the computation of the static output feedback stabilization problem in [165], [163] can be considered as a special case of (1.1) with four linear terms and one quadratic. In the latter case, the GQME-Toolbox must modified to include two more additional linear terms.

It is obvious that when specializing to a specific problem computation time can be saved. For example, matrix equations in control system designs (e.g. algebraic Riccati equations) are symmetric. Hence, the symmetric parts can be removed in order to save computation time. To be more specific an $n^{th}$ order symmetric algebraic Riccati equation corresponds to $n^2$ scalar equations. On removing from this equation the symmetric part, the number of corresponding scalar equations is reduced to $\frac{1}{2}n(n + 1)$. In addition, investigations into specialized numerical algebra routines for symmetric problems (e.g. symmetric matrix inversion and eigenvalue problems) should be made in order to choose the best routine in terms of numerical efficiency, speed and robustness. Therefore, special routines for each special problem can be written and libraries of these can be incorporated into the GQME-Toolbox. This
will help the specialized engineer and scientist to use an option of the software especially designed for him or her, instead of having to adapt the features of a multipurpose software.

Another research task would be to symbolically generate tables of the data the matrices need for the evaluation of scaling coefficients of different scaling problems. Therefore, a library could be incorporated into the GQME-Toolbox and computation time saved when computing scaling problems.

Another possible direction is the implementation of the GQME-Toolbox in a lower level language than MATLAB. C++ and FORTAN programming languages may be good options, for improving computation speed. Note that MATLAB already provides translators which translate its code to C++. Unfortunately, the outcome of this translation is not an optimized C++ code and the improvements compared with the original high level MATLAB code are very small if not null in many cases, as numerical experiments (not reported in this thesis) on all the examples in this thesis have shown. As for FORTAN (the most used language in scientific computations) this would seem to be a good choice because state-of-the-art ordinary differential equation and linear algebra routines are already available e.g. the collection of subroutines in LAPACK [8], NAG library and ODEPACK library. Note that the NAG ordinary differential equation routines, which are provided with MATLAB have been tested within the GQME-Toolbox with no improvements at all. Hence, writing the code in C++ or FORTRAN from scratch is necessary in order to establish if improvements in computation speed can be made.

Another subject of research is to write the software code for parallel computation. Parallel computation is a relatively new subject, and is very promising in the light of computation times and efficiency. It would be very useful for fast and efficient Jacobean matrix computations and ordinary differential equation solvers [30] especially for large dimensional problems.

Further improvements can be made in order to make the GQME-Toolbox more user friendly. Window based environments for the problem setup would help inexperienced user.

On the theoretical side it would make sense to investigate other possible homotopy mappings than the ones are used in this thesis, as was suggested in chapter 2. These investigations could be problem dependent in order to analyse which homotopy mapping is the best choice for a particular problem. Hence, the numerical analysis results of chapter 4 can be specialized according to each specific problem.

Another research area is the use of second order perturbation analysis in the homotopy equations of the homotopy mappings, in order to obtain second order error bounds similar to [100]. Then comparisons with the existing numerical analysis framework in chapter 4 can be made.
For local homotopy step correctors (see chapter 5) and discrete tracking methods of the homotopy paths (see chapters 2, 3), the use of interval arithmetic [120] may be useful to guarantee convergence of the local solver and hence to relax by a factor the dependence of the initial conditions of the local solver on the previous computed homotopy path point. This will effectively allow larger discrete steps to be taken during homotopy path tracking.

The selection of the local solver method for step correction and the homotopy end game needs further investigation. It is known from [138] that Newton-based local solvers, can cycle with their iterates forming periodic and non periodic (e.g. fractals, strange attractors) loops, and never settle down to a solution. Hence, Newton-based methods might not always be the best choice.

For symmetric problems, Lie group algebra and the results therein can be applied in order to define homotopy mappings as symmetric flows [35] on specific manifolds and to take advantage of some special features of Lie algebra as pointed out in [92]. In addition, a study of the characteristics of Kronecker and Hadamard operators (see appendix B) that appear in the jacobian of the homotopy mappings, especially for symmetric problems, could lead to ways of calculating the inverses of these jacobians more economically and faster.

Potentially large field of research is the existence and geometry of solutions of (1.1) in its general form. This will probably require the development of results based on geometric arguments from fields of abstract mathematics, such as algebraic and differential topology, algebraic geometry, set theory, manifold calculus etc. Some examples of such studies for special types of matrix equations are given in [151], [152].

Finally, the reader can use this thesis as a reference or guide to perform similar research on other than quadratic forms of matrix equations (e.g. affine matrix equations etc), since many of the theoretical results hold for general nonlinear algebraic systems of equations as was shown in chapter 2.
Bibliography


[38] **B. D. Craven.** ‘Functions of several variables’. Chapman and Hall, 1981.


[51] **J. Engwerda.** ‘Feedback Nash equilibria in the scalar infinite horizon LQ-game’.
Automatica, Vol. 36, 2000, pp. 135-139.


M. M. Konstantinov, P. Hr. Petkov and D. W. Gu. ‘Improved perturbation bounds for general quadratic matrix equations’. Numerical Functional Analysis and
[100] **M. Konstantinov, V. Angelova, P. Petkov, D. W. Gu, and V. Tsachouridis.**


[137] M. J. D. Powell. ‘A fast algorithm for nonlinearly constrained optimization
calculations’. Numerical Analysis, G. A. Watson Ed., Lecture Notes in Mathematics,


1998.

[143] S. L. Richter. ‘A homotopy algorithm for solving the optimal projection
equations for fixed-order dynamic compensation. Existence, convergence and
global optimality’. American Control Conference, June 1987 Mineapolis, FA1-12:00.

compensator design using the optimal projection equations’. Proceedings of the 28th
506-511.

[145] D. Roose, B. D. Dier and A. Spence. ‘Continuation and Bifurcations: Numerical

controller of saturated synchronous machine under different loading’. IEEE
Bibliography


Index of Theorems Propositions and Lemmas
# Index of Theorems Propositions and Lemmas

## Chapter 2
- **Theorem 2.1**, 19
- **Theorem 2.2**, 34
- **Theorem 2.3**, 48
- **Theorem 2.4**, 49
- **Lemma 2.1**, 40

## Chapter 3
- **Theorem 3.1**, 67
- **Theorem 3.2**, 73
- **Theorem 3.3**, 82
- **Theorem 3.4**, 90
- **Theorem 3.5**, 102
- **Theorem 3.6**, 103
- **Proposition 3.1**, 69
- **Proposition 3.2**, 97
- **Proposition 3.3**, 98
- **Lemma 3.1**, 80

## Chapter 4
- **Theorem 4.1**, 121
- **Theorem 4.2**, 128
- **Theorem 4.3**, 133
- **Proposition 4.1**, 168
- **Lemma 4.1**, 117
- **Lemma 4.2**, 117
- **Lemma 4.3**, 117
- **Lemma 4.4**, 118
- **Lemma 4.5**, 119
- **Lemma 4.6**, 119
- **Lemma 4.7**, 127
- **Lemma 4.8**, 138

## Chapter 6
- **Theorem 6.1**, 205
- **Theorem 6.2**, 206

## Appendix A
- **Theorem A.3.1**, 376
- **Theorem A.4.1**, 379
- **Theorem A.4.2**, 380
- **Theorem A.4.3**, 380
- **Theorem A.4.4**, 381
- **Theorem A.4.5**, 383

## Appendix B
- **Theorem B.2.1**, 392
- **Theorem B.2.2**, 393
- **Proposition B.2.1**, 391
- **Proposition B.2.2**, 392
Appendix

by

Vassilios A. Tsachouridis M.Sc (Salford)
Department of Engineering
University of Leicester

February 2002
Appendix A

Algebraic & Differential Topology Results

A.1 Introduction

In this appendix, some relevant results from algebraic and differential topology will be presented. They are used mainly in chapters 2 and 3.

The appendix begins with elementary issues from set theory in section A.2. Section A.3 considers algebraic polynomial systems and the characterization of their solutions. This section ends with the Bezout theorem and the Cauchy-Riemann equations. Section A.4 introduces the topological (Brouwer) degree, the homotopy invariance theorem and presents Sard's theorem and its variations. This section ends with the definition of a norm-coercive mapping. The appendix concludes with section A.5, where references are provided for further details.

A.2 Set Theory Elements

The style of presentation is adopted from functional analysis, as for example in [102]. The results are for general abstract metric spaces, for which the real and complex Euclidean spaces are just special cases. Therefore, all the following definitions and results apply for any real and/or complex Euclidean space.

Definition A.2.1 (Metric Space & Metric [102]):

A metric space is a pair \((X,d)\), where \(X\) is an abstract set (i.e. set of elements whose nature is left unspecified) and \(d\) is a metric on \(X\) (or distance function on \(X\)),
that is, a function defined on $X \times X$ such that for all $x, y, z \in X$ we have:

(M1) $d$ is real-valued, finite and nonnegative.

(M2) $d(x, y) = 0$ if and only if $x = y$.

(M3) $d(x, y) = d(y, x)$ (Symmetry).

(M4) $d(x, y) \leq d(x, z) + d(z, y)$ (Triangle Inequality).

Note that, a subspace $(Y, \tilde{d})$ of $(X, d)$ is obtained by taking a subset $Y \subseteq X$ and restrict $d$ to $Y \times Y$.

**Definition A.2.2 (Euclidean Real & Complex Spaces [102]):**

The real Euclidean space $\mathbb{R}^n$, $n \in \mathbb{N}$, is obtained if we take the set of all ordered $n$-tuples of real numbers, written $x = (\xi_1, \ldots, \xi_n)$, $y = (\eta_1, \ldots, \eta_n)$, etc., and the Euclidean metric defined by $d(x, y) = \sqrt{\sum_{i=1}^{n} (\xi_i - \eta_i)^2} \geq 0$.

The complex Euclidean space $\mathbb{C}^n$, $n \in \mathbb{N}$, is obtained if we take the set of all ordered $n$-tuples of complex numbers, written $x = (\xi_1, \ldots, \xi_n)$, $y = (\eta_1, \ldots, \eta_n)$, etc., and the metric defined by $d(x, y) = \sqrt{\sum_{i=1}^{n} |\xi_i - \eta_i|^2} \geq 0$.

In definition A.2.2, $n = 1$ generates the real line $\mathbb{R}$ and the complex plane $\mathbb{C}$.

**Definition A.2.3 (Diameter & Bounded Set [102]):**

The diameter $\delta(A)$ of a nonempty set $A$ ($A \neq \emptyset$) in a metric space $(X, d)$ is defined to be $\delta(A) = \sup_{x, y \in A} (d(x, y))$. A is said to be bounded if $\delta(A) < \infty$.

**Definition A.2.4 (Ball & Sphere [102]):**

Given a point $x_0 \in X$ and $r \in \mathbb{R}_+$, the sets below are defined as:

(a) $B(x_0; r) = \{x \in X | d(x, x_0) < r\}$ (Open Ball)

(b) $\overline{B}(x_0; r) = \{x \in X | d(x, x_0) \leq r\}$ (Closed Ball)

(c) $S(x_0; r) = \{x \in X | d(x, x_0) = r\}$ (Sphere)
In all three cases above, $x_o$ is called the center and $r$ the radius.

**Definition A.2.5 (Open & Closed Set [102]);**

A subset $M$ of a metric space $X$ is said to be open if it contains an open ball about each of its points. A subset $K$ of $X$ is said to be closed if its complement $K^C$ (in $X$) is open; that is, $K^C = X - K$ is open.

**Definition A.2.6 ($\varepsilon$-neighborhood & neighborhood [102]);**

An open ball $B(x_o; \varepsilon)$ is called an $\varepsilon$-neighborhood of $x_o$. A neighborhood of $x_o$ is any subset of $X$, which contains an $\varepsilon$-neighborhood of $x_o$.

**Definition A.2.7 (Interior Point, Interior & Exterior of a Set [102], [109]);**

The point $x_o$ is called an interior point of a set $M \subseteq X$ if $M$ is a neighborhood of $x_o$. The interior of $M$ is the set of all interior points of $M$ and it is denoted by $\text{int}\{M\}$. The exterior of $M$ (in $X$), denoted $\text{ext}\{M\}$, is the interior of the complement of $M$, that is, $\text{ext}\{M\} = \text{int}\{M^C\} = \text{int}\{X - M\}$.

**Definition A.2.8 (Accumulation Point & Closure of a Set [102]);**

Let $M$ be a subset of a metric space $X$. Then a point $x_o$ of $X$ (which may or not be a point of $M$) is called an accumulation point of $M$ (or limit point of $M$) if every neighborhood of $x_o$ contains at least one point $y \in M$ distinct from $x_o$. The set consisting of all points of $M$ and the accumulation points of $M$ is called the closure of $M$ and is denoted by $\overline{M}$.

Note that $\overline{M}$ is the smallest closed set containing $M$ and $\text{int}\{M\}$ is the largest open set contained in $M$.

**Definition A.2.9 (Countable Set [102]);**

A set $M$ is said to be countable if $M$ is finite (i.e. has finitely many elements) or if positive integer numbers can be associated with the elements of $M$ so that each element of $M$ corresponds to a unique positive integer number and vice versa.
**Definition A.2.10 (Dense & Separable Set [102]);**

A subset $\mathcal{M}$ of a metric space $\mathcal{X}$ is said to be dense in $\mathcal{X}$ if $\overline{\mathcal{M}} = \mathcal{X}$. $\mathcal{X}$ is said to be separable if it has a countable subset which is dense in $\mathcal{X}$.

**Definition A.2.11 (Boundary Point & Boundary of a Set [102]);**

A boundary point $x$ of a set $\mathcal{A} \subseteq (\mathcal{X}, d)$ is a point of $\mathcal{X}$ (which may or not belong to $\mathcal{A}$) such that every neighborhood of $x$ contains points of $\mathcal{A}$ as well as points not belonging to $\mathcal{A}$. The boundary (or frontier) of $\mathcal{A}$ is the set of all boundary points of $\mathcal{A}$ and it is denoted by $\partial \mathcal{A}$.

Note that, it is apparent from definitions A.2.7 and A.2.10, that $\partial \mathcal{A} = \{ x \mid x \in \text{int} \{ \mathcal{A} \} \cup \text{ext} \{ \mathcal{A} \} \}$.

**Definition A.2.12 (Convergent Sequence & Limit [102]);**

A sequence $\{x_n\}$ in a metric space $\mathcal{X} = (\mathcal{X}, d)$ is said to be convergent if there is a $x \in \mathcal{X}$ such that $\lim_{n \to \infty} (d(x_n, x)) = 0$. $x$ is called the limit of $\{x_n\}$ and we write

$$\lim_{n \to \infty} (x_n) = x \text{ or, simply } x_n \to x.$$

**Definition A.2.13 (Compactness [102]);**

A metric space $\mathcal{X}$ is said to be compact if every sequence in $\mathcal{X}$ has a convergent subsequence. A subset $\mathcal{M}$ of $\mathcal{X}$ is said to be compact if $\mathcal{M}$ is compact considered as a subspace of $\mathcal{X}$.

**Definition A.2.14 (Set of Measure Zero [102]);**

A subset $\mathcal{A}$ of an n-dimensional space $\mathcal{V}^n$ is said to have (Lebesque) measure zero of $\forall \varepsilon \in \mathbb{R}_+$ there exists a collection of countable rectangular solids\(^\dagger\) in $\mathcal{V}^n$,

$$\{ S_i \mid i \in \mathbb{N}_+ \}, \text{ such that } \mathcal{A} \subseteq \bigcup_{i=1}^{\infty} S_i \text{ and } \sum_{i=1}^{\infty} \text{vol}(S_i) < \varepsilon.$$

\(^\dagger\) A rectangular solid $S_i$ in a space $\mathcal{V}^n$, is a cartesian product of $n$ intervals in $\mathcal{V}^1$, and its volume $\text{vol}(S_i)$ is the product of the lengths of the $n$ intervals.
Because of definition A.2.14, saying that $A$ has measure zero is like saying that $A$ has zero volume.

### A.3 Nonlinear Algebraic Polynomial Systems and Equations

We begin with a series of definitions:

**Definition A.3.1 (Nonlinear Algebraic Polynomial System [124]):**

Consider the system of scalar algebraic equations

$$f_i(x) = \sum_{k=1}^{v_i} \left( a_{ik} \prod_{j=1}^{m} \xi_{ij}^{k} \right), \quad i = 1, \ldots, m,$$

(A.3.1)

where $x \in \mathcal{C}^n$, $x := [x_{11} \ldots x_{m1}]^T$, $a_{ik} \in \mathcal{C}$, $v_i, k, m, j, \xi_{ikj} \in \mathbb{Z}_+$. (A.3.1) is called a nonlinear algebraic polynomial system or simply an algebraic polynomial system and is denoted in vector form as $f(x)$, where $f(x) := [f_{11}(x) \ldots f_{m1}(x)]^T$. Furthermore, $x_j$ and $a_{ik}$ are called respectively the variables and coefficients (also called as data) of (A.3.1).

**Definition A.3.2 (Total & Individual Degree of an Nonlinear Algebraic Polynomial System [123]):**

Consider the nonlinear algebraic polynomial system (A.1). Then the numbers

$$\ell \in \mathbb{N}$$

and $d_i \in \mathbb{N}$ defined as

$$\ell := \prod_{j=1}^{m} d_i,$$

(A.3.2)

$$d_i := \max_{k \in [1,v_i]} \left\{ \sigma_k \sigma_k := \sum_{j=1}^{m} \xi_{ikj} \right\},$$

(A.3.3)

are called the total degree and the individual degree of (A.3.1) respectively.

**Definition A.3.3 (Homogeneous Part of a Nonlinear Algebraic Polynomial System):**

Consider the nonlinear algebraic polynomial system (A.3.1). The homogeneous part of A.3.1 is defined as the nonlinear algebraic polynomial system

$$\bar{f}_{ii}(x) := f_{ii}(x), \quad \xi_{ikj} < d_i \Rightarrow a_{ik} = 0,$$

(A.3.4)
where \( d_i \) is the individual degree of (A.3.1) given by (A.33). Furthermore, (A.3.4) is
denoted in vector form as \( \tilde{f}(x) \), where \( \tilde{f}(x):=[\tilde{f}_{11}(x) \ldots \tilde{f}_{m1}(x)]^T \) and \( x_j \) and \( a_{ik} \) are called respectively the variables and coefficients (also called data) of (A.3.4).

In other words, the homogeneous part \( \tilde{f}(x) \) of \( f(x) \), is the nonlinear algebraic polynomial system derived from \( f(x) \) by setting the lower degree terms in each element \( f_{ii}(x) \) of \( f(x) \) equal to zero (see [124], [123]).

**Definition A.3.4 (Nonlinear Algebraic Polynomial Vector Equation);**
A nonlinear algebraic polynomial system as (A.3.1) set equal to zero (i.e. \( f(x) = 0 \)) is called a nonlinear algebraic polynomial vector equation.

Instead of the term nonlinear algebraic vector equation, often the term polynomial system is used, by already assuming that (A.3.1) is being set to zero.

**Definition A.3.5 (Homogeneous Nonlinear Algebraic Polynomial Vector Equation);**
The homogeneous part (A.3.4) of a nonlinear algebraic polynomial system (A.3.1), being set equal to zero (i.e. \( \tilde{f}(x) = 0 \)) is called a homogeneous nonlinear algebraic polynomial vector equation.

**Definition A.3.6 (Geometrically Isolated Solution of a Nonlinear Algebraic Polynomial Vector Equation [123]);**
A solution to a nonlinear algebraic polynomial vector equation as in definition A.3.4 is called geometrically isolated if there is a ball around the solution that contains no other solution of the same equation.

Definition A.3.6 states that a solution \( x = x^* \) of \( f(x) = 0 \) above is nonsingular if \( f(x^* + \delta x^*) \neq 0 \); that is \( f(x^*) = 0 \) with respect to sufficiently small perturbations of \( x^* \) by \( \delta x^* \in \mathbb{C}^n \), produce no other solution than \( x^* \).
Definition A.3.7 (Singular Solution of a Nonlinear Algebraic Polynomial Vector Equation [123]):

A solution to a nonlinear algebraic polynomial vector equation \( f(x) = 0 \), as in definition A.3.4, is called singular if \( \det(J_f(x)) = 0 \) where \( J_f(x) \) is the Jacobian of \( f(x) \) with respect to \( f(x) = 0 \) (see definition B.3.1 in appendix B).

Note that, a geometrically isolated solution can be singular and a not geometrically isolated solution is always singular (see [123]).

Definition A.3.8 (Multiplicity of a Solution of a Nonlinear Algebraic Polynomial Vector Equation [123]):

Let \( x = x^* \) be a geometrically isolated solution of the nonlinear algebraic vector equation \( f(x) = 0 \). Now, perturb all coefficients (including the zero ones) of \( f(x) \) by adding arbitrary small numbers to each coefficient (i.e. \( a_{ik} + \delta a_{ik} \), with \( \delta a_{ik} \in \mathbb{C} \) sufficiently small) in such a way that the resulting perturbed equation has only nonsingular solutions. Then, the number \( m_{x^*} \in \mathbb{N} \) of these solutions arbitrary close to \( x^* \) is called the multiplicity of \( x^* \).

It is worth mentioning, that definition A.3.6 deals with the perturbations of the variables of nonlinear algebraic vector equation \( f(x) = 0 \), while definition A.3.8 deals with the perturbations of coefficients (data) of \( f(x) = 0 \).

Definition A.3.9 (Solution of a Nonlinear Algebraic Polynomial Vector Equation at Infinity [121], [123], [124]):

Consider definitions A.3.4 and A.3.5. The non zero solution \( x = x^* \), \( x^* \in \mathbb{C}^n \) of \( \tilde{f}(x) = 0 \) is defined as a solution of \( f(x) = 0 \) at infinity.

It is apparent that any nonzero multiple of a solution at infinity of \( f(x) = 0 \) will satisfy \( \tilde{f}(x) = 0 \).

Theorem A.3.1 (Theorem of Bezout [121], [123]):
Appendix A  Algebraic & Differential Topology Results 377

Let \( f(x) = 0 \) be a nonlinear algebraic polynomial vector equation according to definition A.3.4. Assume that \( f(x) = 0 \) can have in general finite solutions and solutions at infinity (see definition A.3.9). Then the total number of geometrically isolated solutions (counting multiplicities) of \( f(x) = 0 \) is no more than its total degree \( \ell \) (see A.3.2). Furthermore, if \( f(x) = 0 \) has a finite number of solutions, then the total number of geometrically isolated solutions (counting multiplicities) is exactly equal to its total degree \( \ell \) (see A.3.2).

**Proof.**

The proof can be found in [33] and in [123].

**Definition A.3.10 (Cauchy-Riemann Equations [2], [160]):**

Consider the complex analytic mapping \( f: \mathbb{C}^n \to \mathbb{C}^m, n,m \in \mathbb{N}_* \), for a function of \( x \in \mathbb{C}^n \), \( f(x) \). Furthermore consider the equivalent expression of \( f \) as the real mapping \( g: \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^m, g(x_1,x_2) = u(x_1,x_2) + i v(x_1,x_2) = f(x) \), where \( u: \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^m, v: \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^m \), \( x_1 = \text{Re}(x), x_2 = \text{Im}(x) \) (note that, \( x = x_1 + x_2i \)). Then the equations

\[
\frac{\partial (u(x_1, x_2))}{\partial x_1} = \frac{\partial (v(x_1, x_2))}{\partial x_2}, \tag{A.3.5}
\]

\[
\frac{\partial (u(x_1, x_2))}{\partial x_2} = -\frac{\partial (v(x_1, x_2))}{\partial x_1}, \tag{A.3.6}
\]

are known as the Cauchy-Riemann differential equations or simply the Cauchy-Riemann equations.

### A.4 Algebraic & Differential Topology Theorems

Before presenting the relevant theorems, the concept of topological (Brower) degree of real mappings is presented first.

**Definition A.4.1 (Topological (Brouwer) Degree [110], [119], [131]):**

Consider the \( C^1 \) (continuous) mapping \( M: S \to T, M(x) = y \), where \( x \in S, y \in T, S, T \subseteq \mathbb{R}^n, n \in \mathbb{N}_* \), and \( S \) is open and bounded \((\hat{S} \neq \emptyset)\). Furthermore, let
crease(M):= \left\{ M(x) | \forall x \in \overline{S}, \frac{d(M(x))}{d(x^T)} = 0_{n \times n} \right\} \text{ and suppose that } y \notin \text{crease}(M) \\

and that \( \forall x \in \overline{S}, M(x) \neq y \). Then the topological or Brouwer degree of the mapping \( M \), subject to the domain \( S \) and the regular value \( y \), is defined as

\[
\deg(M(x), S, y) := \sum_{i=1}^m \text{sign} \left( \det \left( \left\{ \frac{d(M(x))}{d(x^T)} \right\}_{x=x_i|M(x_i)=y} \right) \right),
\]

(A.4.1)

where \( m \in \mathbb{N} \) and \( \forall a \in \mathbb{R}, \text{sign}(a) := \begin{cases} +1, & a > 0 \\ 0, & a = 0 \\ -1, & a < 0 \end{cases} \).

As proven in theorem 1.1.2 of [110], the condition \( y \notin \text{crease}(M) \) together with the fact that \( \overline{S} \) is compact (true since \( \overline{S} \subseteq \mathbb{R}^n \) and \( \overline{S} \) is bounded, see [186]) in definition A.4.1 is vital since, it implies a finite number \( m \in \mathbb{N} \) of solutions of \( M(x) = y \) in (A.4.1) in \( S \). Also, it is apparent from (A.4.1) that \( \deg(M(x), S, y) \in \mathbb{Z} \).

Moreover, the topological degree of a mapping can also be defined for unbounded domains \( S \). In the last case, the definition is different from definition A.4.1 above. For more details see [110].

Now, in view of more generalized mappings, definition A.4.1 is restated as

**Definition A.4.2 (Topological (Brouwer) Degree of a Homotopy Mapping):**

Consider the \( C^1 \) (continuous) mapping \( M: \mathbb{S} \times \mathbb{W} \rightarrow \mathbb{T}, M(x(\varepsilon), \varepsilon) = y \), where \( \varepsilon \in \mathbb{W}, x(\varepsilon) \in \mathbb{S}, y \in \mathbb{T}, \mathbb{W} \subseteq \mathbb{R}, \mathbb{S}, \mathbb{T} \subseteq \mathbb{R}^n, n \in \mathbb{N}, \) and \( \mathbb{S} \) is open and bounded (i.e. \( \mathbb{S} \neq \emptyset \)). Furthermore, let

\[ \text{crease}_\varepsilon(M):= \left\{ M(x(\varepsilon), \varepsilon) | \forall x(\varepsilon) \in \overline{S}, \frac{d(M(x(\varepsilon), \varepsilon))}{d(x(\varepsilon)^T)} = 0_{n \times n} \right\} \text{ and suppose that } \]

\( y \notin \text{crease}_\varepsilon(M) \) and that \( \forall x(\varepsilon) \in \overline{S}, M(x(\varepsilon), \varepsilon) \neq y \). Then the topological or Brouwer degree of the mapping \( M \) at \( \varepsilon \), subject to the domain \( S \) and the regular value \( y \), is defined as
\[ \deg(M(x(\varepsilon), \varepsilon), S, y) := \sum_{i=1}^{m} \text{sign} \left( \det \left( \frac{\partial(M(x(\varepsilon), \varepsilon))}{\partial(x(\varepsilon)^T)} \right) \bigg|_{x(\varepsilon) = x_i(\varepsilon)} = y \right) \] 

(A.4.2)

where \( m \in \mathbb{N}^* \) and \( \forall a \in \mathbb{R}, \text{sign}(a) := \begin{cases} +1, & a > 0 \\ 0, & a = 0 \\ -1, & a < 0 \end{cases} \)

It is obvious that definition A.4.2, can be applied to appropriate homotopy mappings with bounded domains (e.g. (2.36) in chapter 2) for every value of the homotopy variable. Again, note that from (A.4.2) it follows that 
\[ \deg(M(x(\varepsilon), \varepsilon), S, y) \in \mathbb{Z} \]. A very important result considering the topological degree, is the homotopy invariance theorem stated below.

**Theorem A.4.1 (Homotopy Invariance Theorem [131]):**

Let \( S \subseteq \mathbb{R}^n, n \in \mathbb{N}^* \), be open and bounded \((\hat{S} \neq \emptyset)\) and let the \( C^1 \) (continuous) mapping \( M : \hat{S} \times \bar{U} \rightarrow \mathbb{R}^n, M(x(\varepsilon), \varepsilon) = y \), where \( \varepsilon \in \bar{U}, \bar{U} := [0,1] \), \( x(\varepsilon) \in S, y \in \mathbb{R}^n \). Suppose, further, that \( \forall (x(\varepsilon), \varepsilon) \in \hat{S} \times \bar{U}, M(x(\varepsilon), \varepsilon) \neq y \) and that \( \forall \varepsilon \in \bar{U}, y \in \text{crease}_e(M) \). Then \( \deg(M(x(\varepsilon), \varepsilon), S, y) = \text{const}, \forall \varepsilon \in \bar{U} \).

**Proof.**

For the proof, see theorem 6.2.2 of [131].

Generally, the topological degree and the homotopy invariance theorem can be defined for mappings in complex Euclidean spaces. In the last case, the complex mappings are equivalently expressed as real mappings with double dimensions resulting from their real and complex parts (e.g. see (2.29) and (2.36) in chapter 2). Hence, definitions A.4.1, A.4.2, and theorem A.4.1 above can be applied.

Next the Sard's density theorem, also known as Sard's theorem, and its variations are presented.
Theorem A.4.2 (Sard's Density Theorem [1]):

Consider the (continuous) mapping $M: S \to \mathbb{C}^p$, $M(x) = y$, where $x \in S$, $y \in \mathbb{C}^p$, $S \subseteq \mathbb{C}^n$ is open and $n, p \in \mathbb{N}_*$. Furthermore, let $M$ be $C^r$, where $r > \max(0, n - p)$. Then the set $\left\{ M(x) \big| \forall x \in S, \ \text{Re} |\text{rank} \left( \frac{d(M(x))}{d(x^T)} \right) | < 2p \right\}^\dagger$ has measure zero in $\mathbb{C}^p$.

Proof.

For the proof, see [1] and its references. Note that in [1], theorem A.4.2 and its proof are presented for real Euclidean spaces. However, as was mentioned previously, complex mappings can be equivalently written as real mappings with double dimensions. Hence, the proof can follow.

In theorem A.4.2, $\left\{ M(x) \big| \forall x \in S, \ \text{Re} |\text{rank} \left( \frac{d(M(x))}{d(x^T)} \right) | < 2p \right\}$ is known as the set of critical values of $M$. Note that, detailed discussions and variations of theorem A.4.2 can be found in [1], [67], [119] and [123]. A very useful variation of theorem A.4.2 is Sard's transversality theorem shown next.

Theorem A.4.3 (Sard's Transversality or Parameterized Sard's Theorem [32]):

Consider the $C^r$ (continuous) mapping $M: S \times T \to \mathbb{C}^p$, $M(x, b) = y$, where $x \in S$, $b \in T$, $y \in \mathbb{C}^p$, $S \subseteq \mathbb{C}^n$ and $T \subseteq \mathbb{C}^q$ are open and $r > \max(0, q - p)$, $n, p, q \in \mathbb{N}_*$. Suppose, further, that $0_{p \times 1} \in M(x, b) \big| \forall (x, b) \in S \times T, \ \text{Re} |\text{rank} \left( \frac{d(M(x, b))}{d(x^T)} \right) | < 2p$, and define $M_b: S \to \mathbb{C}^p$, $M_b(x) = M(x, b)$. Then

\[ 1 \text{ For the operator } \text{Re} |\text{rank}(\cdot) \text{ see appendix B.} \]
\[
\begin{cases}
b \in T | 0 \times 1 \in \left\{ M_b(x) | \forall x \in S, \text{ Re}|\text{rank} \left( \frac{d(M_b(x))}{d(x^T)} \right) < 2p \right\}\end{cases}
\]

has measure zero in \( T \).

\textbf{Proof.}

For a proof see [32], [67]. Again, note that in [32], theorem A.4.3 and its proof are presented for real Euclidean spaces. However, as was mentioned previously, complex mappings can be equivalently written as real mappings with double dimensions. Hence, the proof can follow. Also, in [67] the proof is more abstract and considers more generalized spaces.

An immediate consequence of theorem A.4.3 is theorem A.4.4 below.

\textbf{Theorem A.4.4 (First Variation of Parameterized Sard’s Theorem);}

Consider the \( C^2 \) homotopy mappings \( H: C^m \times [0,1] \times C^m \rightarrow C^m \), \( H(x, \varepsilon, \alpha) = 0 \) and \( H_{\alpha}: C^m \times [0,1] \rightarrow C^m \), \( H_{\alpha}(x, \varepsilon) = H(x, \varepsilon, \alpha) \), \( m \in \mathbb{N} \). If \( H \) is transversal \( ^\dagger \) to zero then \( \forall \alpha \in C^m \), \( H_{\alpha} \) is also transversal to zero with probability 1.

\textbf{Proof.}

Let \( T = \left\{ b = [\varepsilon, \alpha]^T | (\varepsilon, \alpha) \in [0,1] \times C^m \right\} \), \( S \subseteq C^n \), \( y = 0 \times 1 \), \( p = n = m \), \( q = m + 1 \). Obviously, \( S \subseteq C^n \) and \( T \subseteq C^q \) are open and \( n, p, q \in \mathbb{N} \). \( H \) and \( H_{\alpha} \) can be represented equivalently as \( M: S \times T \rightarrow C^p \), \( M(x, b) = H(x, \varepsilon, \alpha) = y = 0 \times 1 \), where \( x \in S \), \( b \in T \) and \( M_b: S \rightarrow C^p \), \( M_b(x) = M(x, b) \) or equivalently, since \( H_{\alpha}(x, \varepsilon) = H(x, \varepsilon, \alpha) \), \( M_b(x) = H_{\alpha}(x, \varepsilon) = y = 0 \times 1 \). Hence, \( M \) and \( M_b \) are \( C^2 \).

\( M_b \) is \( C^2 \) and in view of theorem A.4.2 we have, \( r = 2 \) and \( \max(0, n - p) = \max(0, m - m) = 0 \). Therefore, \( r > \max(0, n - p) \) and theorem A.4.2

\( ^\dagger \) For the definition of a mapping being transversal to zero see chapter 2.
implies that \( \{ M_b(x) \mid \forall x \in S, \text{ Re}\, \text{rank} \left( \frac{d(M_b(x))}{d(x^T)} \right) < 2p \} \) has measure zero in \( C^p \),

which means that \( \forall x \in S, \text{ Re}\, \text{rank} \left( \frac{d(M_b(x))}{d(x^T)} \right) = 2p \) with probability 1. Now, \( H \) being transversal to zero results by definition

\[
\{ H(x, \varepsilon, \alpha) \mid \forall (x, \varepsilon, \alpha) \in S \times [0,1) \times S, \text{ Re}\, \text{rank} \left( \begin{bmatrix} \frac{d(H(x, \varepsilon, \alpha))}{d(x^T)} & \frac{d(H(x, \varepsilon, \alpha))}{d(\varepsilon)} & \frac{d(H(x, \varepsilon, \alpha))}{d(\alpha^T)} \end{bmatrix} \right) < 2p \}.
\]

In view of \( M(x,b) := H(x,\varepsilon,\alpha) = y = 0_{p \times 1} \), the last result is equivalent to

\[
\{ M(x,b) \mid \forall (x,b) \in S \times T, \text{ Re}\, \text{rank} \left( \begin{bmatrix} \frac{d(M(x,b))}{d(x^T)} & \frac{d(M(x,b))}{d(b^T)} \end{bmatrix} \right) < 2p \},
\]

which means that \( \forall (x,b) \in S \times T, \text{ Re}\, \text{rank} \left( \begin{bmatrix} \frac{d(M(x,b))}{d(x^T)} & \frac{d(M(x,b))}{d(b^T)} \end{bmatrix} \right) = 2p \) with probability 1. From the last result and from \( \text{ Re}\, \text{rank} \left( \frac{d(M_b(x))}{d(x^T)} \right) = 2p, \forall x \in S, \) we have

\[
\text{ Re}\, \text{rank} \left( \begin{bmatrix} \frac{d(M(x,b))}{d(x^T)} & \frac{d(M(x,b))}{d(b^T)} \end{bmatrix} \right) = \text{ Re}\, \text{rank} \left( \frac{d(M(x,b))}{d(x^T)} \right) = 2p, \forall (x,b) \in S \times T
\]

and thus \( \{ M(x,b) \mid \forall (x,b) \in S \times T, \text{ Re}\, \text{rank} \left( \frac{d(M(x,b))}{d(x^T)} \right) < 2p \} \). In addition, \( M \) is \( C^2 \) and in view of theorem A.4.3 we have, \( r = 2 \) and \( \max(0,q-p) = \max(0,m+1-m) = 1 \). Therefore, \( r > \max(0,q-p) \). Hence, theorem A.4.3 implies that \( \{ b \in T \mid \exists 0_{p \times 1} \in \{ M_b(x) \mid \forall x \in S, \text{ Re}\, \text{rank} \left( \frac{d(M_b(x))}{d(x^T)} \right) < 2p \} \} \) has
measure zero in $T$. The last result, in view of \( M_b(x) = H_\alpha(x, \varepsilon) = y = 0_{p \times 1} \), means that \( \forall \alpha \in \mathbb{C}^m, \: H_\alpha \) is transversal to zero with probability 1.

In the proofs of theorems 2.1 and 2.2 in chapter 2, a generalization of theorem A.4.4 is used. That is

**Theorem A.4.5 (Second Variation of Parameterized Sard's Theorem):**

Consider the \( C^2 \) homotopy mappings \( H: \mathbb{C}^m \times [0,1] \times \mathbb{C}^m \to \mathbb{C}^m \), \( H(x, \varepsilon, \alpha, \beta) = 0 \) and \( H_{\alpha}\beta: \mathbb{C}^m \times [0,1] \to \mathbb{C}^m \), \( H_{\alpha}\beta(x, \varepsilon) := H(x, \varepsilon, \alpha, \beta) \), as in definitions 2.1 and 2.2 in chapter 2. If \( H \) is transversal to zero then \( \forall (\alpha, \beta) \in \mathbb{C}^m \times \mathbb{C}^m, \: H_{\alpha}\beta \) is transversal to zero with probability 1.

**Proof.**

First assume \( \beta \in \mathbb{C}^m \) as being fixed, and consider the homotopy \( H_\beta: \mathbb{C}^m \times [0,1] \times \mathbb{C}^m \to \mathbb{C}^m \), \( H_\beta(x, \varepsilon, \alpha) = H(x, \varepsilon, \alpha, \beta) = 0 \). Note that, \( H_{\alpha}\beta \) can viewed equivalently as \( H_{\beta,\alpha}: \mathbb{C}^m \times [0,1] \to \mathbb{C}^m \), \( H_{\beta,\alpha}(x, \varepsilon) := H_\beta(x, \varepsilon, \alpha) \). By construction, \( H_\beta \) and \( H_{\beta,\alpha} \) are also \( C^2 \).

Let \( T := S \cup b := [\varepsilon, \alpha]^T \in [0,1] \times \mathbb{C}^m, \: S \subseteq \mathbb{C}^n, \: y = 0_{p \times 1}, \: p = n = m, \: q = m + 1 \). Hence, \( S \subseteq \mathbb{C}^n \) and \( T \subseteq \mathbb{C}^q \) are open and \( n, p, q \in N_* \). \( H_\beta \) and \( H_{\beta,\alpha} \) can be represented equivalently as \( M: S \times T \to \mathbb{C}^p \), \( M(x, b) := H_\beta(x, \varepsilon, \alpha) = y = 0_{p \times 1} \), where \( x \in S, \: b \in T \) and \( M_b: S \to \mathbb{C}^p \), \( M_b(x) := M(x, b) \) or equivalently, since \( H_{\beta,\alpha}(x, \varepsilon) := H_\beta(x, \varepsilon, \alpha) \), \( M_b(x) = H_{\beta,\alpha}(x, \varepsilon) = y = 0_{p \times 1} \). Hence, \( M \) and \( M_b \) are \( C^2 \).

\( M_b \) is \( C^2 \) and in view of theorem A.4.2 we have, \( r = 2 \) and \( \max(0, n - p) = \max(0, m - m) = 0 \). Therefore, \( r > \max(0, n - p) \) and theorem A.4.2 implies that \( M_b(x) \forall x \in S, \: \text{Re} \text{rank} \left( \frac{d(M_b(x))}{d(x^T)} \right) < 2p \) has measure zero in \( \mathbb{C}^p \).
which means that \( \forall x \in S, \text{Re} |\text{rank} \left( \frac{d(M_b(x))}{d(x^T)} \right) = 2p \) with probability 1. Now, \( H \)

being transversal to zero results by definition

\[
0_{p \times 1} \notin \{ H(x, \varepsilon, \alpha, \beta) \mid \forall (x, \varepsilon, \alpha, \beta) \in S \times [0,1) \times S \times S, \\
\text{Re} |\text{rank} \left( \begin{bmatrix} d(H(x, \varepsilon, \alpha, \beta)) \\ d(x^T) \\ d(\varepsilon) \\ d(\alpha^T) \\ d(\beta^T) \end{bmatrix} \right) < 2p \}
\]

For fixed \( \beta \in \mathbb{C}^m \), in view of \( M(x, b) := H_{\beta}(x, \varepsilon, \alpha) = y = 0_{p \times 1} \), the last result is equivalent to

\[
0_{p \times 1} \notin \{ M(x, b) \mid \forall (x, b) \in S \times [0,1) \times T, \\
\text{Re} |\text{rank} \left( \begin{bmatrix} d(M(x, b)) \\ d(x^T) \\ d(\varepsilon) \\ d(\alpha^T) \\ d(\beta^T) \end{bmatrix} \right) < 2p \}
\]

which means that for fixed \( \beta \in \mathbb{C}^m \), \( \forall (x, b) \in S \times T \).

\[
\text{Re} |\text{rank} \left( \begin{bmatrix} d(M(x, b)) \\ d(x^T) \\ d(\varepsilon) \\ d(\alpha^T) \\ d(\beta^T) \end{bmatrix} \right) = 2p
\]

with probability 1. From the last result and from \( \text{Re} |\text{rank} \left( \frac{d(M_b(x))}{d(x^T)} \right) = 2p, \forall x \in S, \)

we have for fixed \( \beta \in \mathbb{C}^m \) that \( \forall (x, b) \in S \times T, \)

\[
\text{Re} |\text{rank} \left( \begin{bmatrix} d(M(x, b)) \\ d(x^T) \\ d(\varepsilon) \\ d(\alpha^T) \\ d(\beta^T) \end{bmatrix} \right) = \text{Re} |\text{rank} \left( \frac{d(M_b(x))}{d(x^T)} \right) = 2p
\]

and thus \( 0_{p \times 1} \notin \{ M(x, b) \mid \forall (x, b) \in S \times T, \text{Re} |\text{rank} \left( \frac{d(M(x, b))}{d(x^T)} \right) < 2p \}. \)

In addition, \( M \) is \( C^2 \) and in view of theorem A.4.3 we have, \( r = 2 \) and \( \max(0, q - p) = \max(0, m + 1 - m) = 1 \). Therefore, \( r > \max(0, q - p) \). Hence, theorem A.4.3 implies that \( b \in T \)

\[
0_{p \times 1} \notin \{ M_b(x) \mid \forall x \in S, \text{Re} |\text{rank} \left( \frac{d(M_b(x))}{d(x^T)} \right) < 2p \}
\]

has
measure zero in $T$. The last result, in view of $M_b(x) = H_{\beta_\alpha}(x,e) = y = 0_p \times 1$, means that $\forall \alpha \in S$ and for a fixed $\beta \in C^m$, $H_{\beta_\alpha}$ is transversal to zero with probability 1.

Since the above results hold for a general fixed $\beta \in C^m$, it follows that $\forall \alpha \in S$ and $\forall \beta \in C^m$, $H_{\beta_\alpha}$ is transversal to zero with probability 1. The last result implies by construction of $H_{\beta_\alpha}$ that, $\forall (\alpha, \beta) \in C^m \times C^m$, $H_{\alpha \beta}$ is transversal to zero with probability 1.

**Definition A.4.3 (Norm-Coercive Mapping):**

Consider the mapping $M : S \times W \rightarrow T$, $M(x(e),e) = y$, where $e \in W$, $x(e) \in S$, $y \in T$, $W \subseteq C$, $S \subseteq T \subseteq C^n$, $n \in \mathbb{N}$, and $S$ is open and bounded (i.e. $S \neq \emptyset$). $M$ is said to be a norm-coercive mapping if and only if

$$\lim_{\|x(e)\|_2 \rightarrow \infty} \left( \|M(x(e),e)\|_2 \right) = \infty.$$

A similar definition of a norm-coercive mapping, subject to real Euclidean spaces, can be found in [131].

**A.5 Useful Bibliography**

The material presented in this appendix and detailed studies of relevant algebraic, differential and general topology subjects can be found in [102], [109], [157], [2], [160], [119], [110], [67], [1], [131] and [123].
Appendix B

Results in Matrix Analysis

B.1 Introduction

This appendix presents useful results in matrix analysis, which are applied in the main chapters. The results are subject to complex Euclidean spaces and hold for real Euclidean spaces too. Many of these results are novel and (the author believes) derived for the first time here. The appendix also serves as a tutorial in matrix differentiation involving Kronecker and Hadamard operators.

Section B.2 presents several matrix operators and their algebraic features. Section B.3 focuses on matrix calculus: mainly the differentiation of matrix functions and the derivation of analytic formulas. The appendix ends with section B.4, in which a useful bibliography is provided for more detailed study.

B.2 Matrix Operators & Algebra

First of all, let \( I_n \in \mathbb{Z}_+^{n \times n} \) be the identity matrix, let \( U_{n \times m} \in \mathbb{Z}_+^{n \times m} \) be the unity matrix (i.e. the matrix in which all elements are equal to 1), and define the multiplication of a scalar \( a \in \mathbb{C} \) and a matrix \( X \in \mathbb{C}^{n \times m} \), \( aX \), as \( [aX]_{ij} := aX_{ij} \),

\[
\forall i \in [1,n_X], \forall j \in [1,m_X], i, j \in \mathbb{Z}_+.
\]

B.2.1 Matrix Operators

Let \( x \in \mathbb{C}^{n_x} \), \( A,C \in \mathbb{C}^{n_A \times m_A} \), \( B \in \mathbb{C}^{n_B \times m_B} \), \( A_f \in \mathbb{R}^{n_{A_f} \times m_{A_f}} \), where
Appendix B Results in Matrix Analysis 387

\[ n_x, n_{A_1}, n_{A_2}, n_{B_1}, m_{A_1}, m_{B_1}, n_{A_2}, m_{A_2} \in N, \] and define \( [X]_{ij} := X_{ij} \). Then the following operators are defined:

**Kronecker Sum Operator** \( \oplus \):

\[
A \oplus B := \begin{bmatrix}
A_{11}U_{n_B \times m_B} & \cdots & A_{1m_{A_1}}U_{n_B \times m_B} & B \\
\vdots & \ddots & \vdots & \vdots \\
A_{n_{A_1}}U_{n_B \times m_B} & \cdots & A_{n_{A_1}m_{A_1}}U_{n_B \times m_B} & B
\end{bmatrix}._{n_{A_1} \times m_{A_2} \times n_{A_2} \times m_{A_2}}
\]

\( A \oplus B \in C^{(n_{A_1} + n_{A_2}) \times (m_{A_1} + m_{A_2})} \).

**Kronecker Product Operator** \( \otimes \):

\[
A \otimes B := \begin{bmatrix}
A_{11}B & \cdots & A_{1m_{A_1}}B \\
\vdots & \ddots & \vdots \\
A_{n_{A_1}}B & \cdots & A_{n_{A_1}m_{A_1}}B
\end{bmatrix}._{n_{A_1} \times m_{A_1} \times n_{A_2} \times m_{A_2}}
\]

\( A \otimes B \in C^{(n_{A_1} + n_{A_2}) \times (m_{A_1} + m_{A_2})} \).

**Hadamard Product Operator** \( \circ \):

\[
[A \circ C]_{ij} := A_{ij}C_{ij}, \quad \forall i \in [1, n_{A_1}] \quad \text{and} \quad \forall j \in [1, m_{A_1}], \quad i, j \in Z_+.
\]

\( (A \circ C) \in C^{n_{A_1} \times m_{A_1}} \).

**Hadamard Division Operator** \( + \):

\[
[A + C]_{ij} := A_{ij} / C_{ij}, \quad \text{only if} \quad C_{ij} \neq 0, \quad \forall i \in [1, n_{A_1}] \quad \text{and} \quad \forall j \in [1, m_{A_1}], \quad i, j \in Z_+.
\]

\( (A + C) \in C^{n_{A_1} \times m_{A_1}} \).

**Vector Matrix Operator** \( \text{vec}(\cdot): C^{n_{A_1} \times m_{A_1}} \to C^{n_{A_2} \times m_{A_2}} \):

\[
\text{vec}(A) := \begin{bmatrix}
col_1(A) \\
\vdots \\
col_{m_{A_1}}(A)
\end{bmatrix}, \text{ where } \ col_j(A) := \begin{bmatrix}
A_{ij} \\
\vdots \\
A_{n_{A_1}j}
\end{bmatrix}.
\]

\( \text{col}_j(A) \in C^{n_{A_1}} \) denote the \( j \) column of \( A \) \( \forall j \in [1, m_{A_1}], \quad j \in Z_+ \).

**Inverse Vector Matrix Operator** \( \text{vec}^{-1}(\cdot): C^{n_{A_2} \times m_{A_2}} \to C^{n_{A_1} \times m_{A_1}} \):

\[
\text{vec}^{-1}(\text{vec}(A)) := A.
\]
Appendix B

Vector Diagonalization Operator $\text{diag}(\cdot): \mathbb{C}^{n \times n} \to \mathbb{C}^{n \times n \times n}$ or

$\text{diag}(\cdot): \mathbb{C}^{1 \times n} \to \mathbb{C}^{n \times n}$:

$[\text{diag}(x)]_{ij} := \begin{cases} x_{ij}, & i = j \\ 0, & i \neq j \end{cases}$ or $[\text{diag}(x^T)]_{ij} := \begin{cases} x_{ij}, & i = j \\ 0, & i \neq j \end{cases}$

$\text{diag}(x^T) \in \mathbb{C}^{n \times n}$.

Absolute Value Operator $|\cdot|: \mathbb{C}^{n \times m} \to \mathbb{R}^{n \times m}$; $|A|_{ij} := |A_{ij}|$, $\forall i \in [1,n_A]$ and $\forall j \in [1,m_A]$, $i, j \in \mathbb{Z}^+$. $|A| \in \mathbb{R}^{n \times m_A}$.

Exponent Operator Base-10:

$\text{E}_{10}(\cdot): \mathbb{C}^{n \times m} \to \mathbb{C}^{n \times m}$; $[\text{E}_{10}(A)]_{ij} := 10^{A_{ij}}$, $\forall i \in [1,n_A]$ and $\forall j \in [1,m_A]$, $i, j \in \mathbb{Z}^+$. $\text{E}_{10}(A) \in \mathbb{C}^{n \times m_A}$.

Logarithmic Operator Base-10 $\text{L}_{10}(\cdot): \mathbb{C}^{n \times m} \to \mathbb{R}^{n \times m}$:

$[\text{L}_{10}(A)]_{ij} := \begin{cases} \log_{10}(|A_{ij}|), & A_{ij} \neq 0 \\ 0, & A_{ij} = 0 \end{cases}$, $\forall i \in [1,n_A]$ and $\forall j \in [1,m_A]$, $i, j \in \mathbb{Z}^+$. $\text{L}_{10}(A) \in \mathbb{R}^{n \times m_A}$. 

At this point it is useful to illustrate how the function $a^z$, where $a \in \mathbb{C}$ and $z \in \mathbb{C}$, can be calculated.

$a^z = a^{\text{Re}(z)+\text{Im}(z)i} = a^{\text{Re}(z)} a^{\text{Im}(z)i} = a^{\text{Re}(z)} \left( e^{\text{ln}(a)} \right)^{\text{Im}(z)i}$

$\Leftrightarrow a^z = a^{\text{Re}(z) \left( e^{\text{Im}(z)\text{ln}(a)} \right)^i} = a^{\text{Re}(z)} e^{(\text{Im}(z)\text{ln}(a))i}$

$\Leftrightarrow a^z = a^{\text{Re}(z) \left( \cos(\text{Im}(z)\text{ln}(a)) + i \sin(\text{Im}(z)\text{ln}(a)) \right)}$

$\Leftrightarrow a^z = a^{\text{Re}(z) \cos(\text{Im}(z)\text{ln}(a)) + ia\text{Re}(z) \sin(\text{Im}(z)\text{ln}(a))}$

When $a = 0$, then by definition $0^z = 0$. Note that the above result shows how each element of the exponent operator base-10 above can be computed.
Appendix B

Results in Matrix Analysis 389

Sign Operator \( S(\cdot) : \mathbb{C}^{n_A \times m_A} \rightarrow \mathbb{C}^{n_A \times m_A} ; \)

\[
[S(A)]_{ij} := \begin{cases} 
A_{ij} & , \quad A_{ij} \in \mathbb{C}^* , \quad \forall i \in [1, n_A] \text{ and } \forall j \in [1, m_A] , \quad i, j \in \mathbb{Z}^+ . \\
0, & , \quad A_{ij} = 0 
\end{cases}
\]

\( S(A) \in \mathbb{C}^{n_A \times m_A} . \)

B.2.2 Algebraic Results

Let \( a \in \mathbb{C}^{n_A} , \quad r \in \mathbb{C}^{n_R} , \quad A, C \in \mathbb{C}^{n_A \times m_A} , \quad B, D, E \in \mathbb{C}^{n_B \times m_B} , \quad R \in \mathbb{C}^{n_R \times m_R} , \quad S \in \mathbb{C}^{n_S \times m_S} , \quad T \in \mathbb{C}^{n_T \times m_T} \), where \( n_A, n_B, n_S, n_R, n_T, n_r, n_r, m_A, m_B, m_S, m_R, m_T \in \mathbb{N}^* . \)

For the matrix operators that have previously defined in subsection B.2.1, the following algebraic formulas can be proven to hold by construction.

\[
(A \circ C) T = A T \circ C T = (C \circ A) T \quad (B.2.2)
\]

\[
(A \circ C) T = A T \circ C T = (C \circ A) T \quad (B.2.3)
\]

\[
 vec(A \circ C) = vec(A) \circ vec(C) = vec(C) \circ vec(A) = vec(C \circ A) \quad (B.2.4)
\]

Assuming that \( n_S = n_A \), we have

\[
(A C) S \neq A (C S) . \quad (B.2.6)
\]

\[
(A \otimes C) T = A T \otimes C T \quad (B.2.7)
\]

\[
(A \otimes C) T_c = A T_c \otimes C T_c \quad (B.2.8)
\]

where \((\cdot)^T_c\) is the complex conjugate transpose of a general complex matrix and can be viewed as an operator, which is defined via the function

\[
(\cdot)^T_c : \mathbb{C}^{n_A \times m_A} \rightarrow \mathbb{C}^{n_A \times m_A} , \quad (A)^T_c = \text{Re}(A^T) - i\text{Im}(A^T) , \quad \forall A \in \mathbb{C}^{n_A \times m_A} .
\]

Now, assuming that \( m_A = n_S \) and \( m_B = n_T \), we have

\[
(A \otimes B)(S \otimes T) = (AS) \otimes (BT) \quad (B.2.9)
\]

Assuming that \( n_A = m_A , \quad n_B = m_B \) and that \( A^{-1} , \quad B^{-1} \) exist, we have

\[
(A \otimes B)^{-1} = A^{-1} \otimes B^{-1} \quad (B.2.10)
\]

Assuming that \( m_A = n_B \) and \( m_B = n_S \), we have
\[ \text{vec}(ABS) = (S^T \otimes A) \text{vec}(B). \]  
(B.2.11)

With respect to (B.2.11) above, and if in addition \( m_S = n_T \) and \( m_T = n_R \), it can easily be established that

\[ \text{vec}(ABSTR) = \left( R^T \otimes A \right) \left( T^T \otimes B \right) \text{vec}(S), \]  
(B.2.12)

\[ \text{vec}(ABSTR) = \left( R^T \otimes A \right) \left( T^T S^T \otimes I_{n_R} \right) \text{vec}(B), \]  
(B.2.13)

\[ \text{vec}(ABSTR) = \left( R^T \otimes A \right) \left( I_{n_T} \otimes (BS) \right) \text{vec}(T), \]  
(B.2.14)

Assuming that \( \text{Im}(A) = 0 \) and \( \text{Im}(B) = 0 \), we have (B.2.15)-(B.2.17).

\[ (E_{10}(A))^T = E_{10}(A^T) \]  
(B.2.15)

\[ E_{10}(A) \otimes E_{10}(B) = E_{10}(A \oplus B) \]  
(B.2.16)

\[ (S(A) \circ E_{10}(A)) \otimes (S(B) \circ E_{10}(B)) = (S(A) \circ S(B)) E_{10}(A \oplus B) \]  
(B.2.17)

Assuming that \( n_a = n_B, n_r = m_B, \text{Im}(a) = 0 \) and \( \text{Im}(D) = 0 \), we have

\[ E_{10}(a) \circ \left( (B \circ E_{10}(D)) r \right) = \left( B \circ E_{10}(D + aU_{1 \times m_B}) \right) r \]  
(B.2.18)

Assuming \( n_A = m_A = n_a = n_B m_B \), we have

\[ A \text{vec}(B \circ D) = A \left( \text{vec}(B) \circ \text{vec}(D) \right) = \left( \left( U_{n_A \times 1} \circ \text{vec}(B) \right)^T \circ \text{vec}(A) \right) \text{vec}(D) \]  
(B.2.19)

and

\[ (A \circ C) a = \left[ \text{diag} \left( \left[ \text{col}_1 \left( A \circ \left( U_{n_A \times 1} a^T \right) \right) \right] \right) \ldots \right. \]
\[ \left. \ldots \text{diag} \left( \left[ \text{col}_m A \circ \left( U_{n_A \times 1} a^T \right) \right] \right) \left] \text{vec}(C) \right. \]  
(B.2.20)

Furthermore, assuming that \( \text{Im}(a) = 0, \text{Im}(A) = 0 \) and \( n_A = m_A = n_a = n_B m_B \) as before, we have

\[ E_{10}(A) \circ \left( U_{n_A \times 1} a^T \right) = E_{10}(A + U_{n_A \times 1} a^T) \]  
(B.2.21)

and

\[ (E_{10}(A) \circ C) a = \left[ \text{diag} \left( E_{10} \left( \text{col}_1 \left( A \circ \left( U_{n_A \times 1} a^T \right) \right) \right] \right) \ldots \right. \]
\[ \left. \ldots \text{diag} \left( E_{10} \left( \text{col}_m A \circ \left( U_{n_A \times 1} a^T \right) \right) \right) \left] \text{vec}(C) \right. \]  
(B.2.22)
Now, combining (B.2.19)-(B.2.22) and assuming that $\text{Im}(A) = 0$, $\text{Im}(B) = 0$, $\text{Im}(r) = 0$ and $m_A = n_B = m_B = n_r$ we have

$$\left(E_{10}(A) \circ C\right)\left(E_{10}(B) \circ D \circ E\right)E_{10}(r) = \left(\left(U_{nA} \times I_{\text{vec}(E)}\right)^T \circ \left(U_{1 \times m_A} \otimes C\right)\right) \circ E_{10}\left(U_{1 \times m_A} \otimes A + U_{nA} \times I_{\text{vec}(B + U_{m_A} \times 1 r^T)^T}\right)\text{vec}(D).$$

(B.2.23)

Note that the proofs for (B.2.1)-(B.2.23) are evident by performing all algebraic calculations in both sides of the equations. Before moving to section B.3, the definition of the real rank of a complex matrix, $\text{Re} \text{rank}()$, is given below.

**Definition B.2.1 (Real Rank of a Complex Matrix):**

Let $A \in \mathbb{C}^{nA \times m_A}$. Then the real rank of a complex matrix, $\text{Re} \text{rank}() \in \mathbb{N}$, is defined as

$$\text{Re} \text{rank}(A) := \text{rank}\left(\begin{bmatrix} \text{Re}(A) & -\text{Im}(A) \\ \text{Im}(A) & \text{Re}(A) \end{bmatrix}\right).$$

(B.2.24)

It is apparent from definition B.2.1 that $\text{Re} \text{rank}()$ can be considered as an operator, which is defined by the function $\text{Re} \text{rank}() : \mathbb{C}^{nA \times m_A} \to \mathbb{N}$ or $\text{Re} \text{rank}() : \mathbb{R}^{2nA \times 2m_A} \to \mathbb{N}$ with respect to (B.2.24). Also, from (B.2.24) it follows that $\text{Re} \text{rank}(A) = \text{Re} \text{rank}(A^T)$ and $\text{Re} \text{rank}(A) = \text{rank}\left(\begin{bmatrix} \text{Re}(A) & \text{Im}(A) \\ -\text{Im}(A) & \text{Re}(A) \end{bmatrix}\right)$.

At this point it should be noted that the determinant of a matrix is considered as a matrix operator (function) too [113], [83]. Two useful results associated with the determinant of a matrix are presented next.

**Proposition B.2.1 (Determinant of a Partitioned Matrix [83]):**

Let $i, j, n_{A_{ij}}, m_{A_{ij}} \in \mathbb{N}$ and suppose that $A \in \mathbb{C}^{nA \times m_A}$ can be partitioned as

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix},$$

where $\forall i, j \in [1, 2]$, $A_{ij} \in \mathbb{C}^{n_{A_{ij}} \times m_{A_{ij}}}$ are matrices of appropriate dimensions with $n_{A_{11}} = m_{A_{11}}$ and $n_{A_{22}} = m_{A_{22}}$. If $A_{11}^{-1}$ exists then

$$\det(A) = \det(A_{11}) \det\left(A_{22} - A_{21}A_{11}^{-1}A_{12}\right).$$

(B.2.25)
Proof. 

The proof is given in section 0.8.5 of [83].

Proposition B.2.2:

Let \( n \in \mathbb{N}_* \), \( \tilde{A} \in \mathbb{R}^{n \times n} \) and \( A := \begin{bmatrix} \tilde{A} & -\tilde{A} \\ A & \tilde{A} \end{bmatrix} \), then \( \det(A) \geq 0 \).

Proof. 

Using the well known result that

\[
\det \left( \prod_{i=1}^{m} X_i^{n_i} \right) = \prod_{i=1}^{m} (\det(X_i))^{n_i},
\]

\( \forall n_i \in \mathbb{Z} \), where \( X_i \in \mathbb{C}^{n_i \times n_i} \), \( i, n \in \mathbb{N}_* \) (see [83], [113]), and notice that

\[
\det \left( \begin{bmatrix} I_n & 0_{n \times n} \\ -I_n & I_n \end{bmatrix} \right) = 1 \]

we have

\[
\det \left( \begin{bmatrix} \tilde{A} & -\tilde{A} \\ A & \tilde{A} \end{bmatrix} \right) = \det \left( \begin{bmatrix} \tilde{A} & -\tilde{A} \\ A & \tilde{A} \end{bmatrix} \right) \left( \begin{bmatrix} I_n & 0_{n \times n} \\ -I_n & I_n \end{bmatrix} \right) = \det \left( \begin{bmatrix} 2\tilde{A} & -\tilde{A} \\ 0_{n \times n} & \tilde{A} \end{bmatrix} \right) \right) \]

(B.2.26)

From (B.2.25) in proposition B.2.1, (B.2.26) becomes equivalently

\[
\det \left( \begin{bmatrix} \tilde{A} & -\tilde{A} \\ \tilde{A} & \tilde{A} \end{bmatrix} \right) = \det \left( \begin{bmatrix} 2\tilde{A} & -\tilde{A} \\ 0_{n \times n} & \tilde{A} \end{bmatrix} \right) \right) \]

(B.2.27)

Hence, (B.2.27) results in \( \det(A) \geq 0 \).

The Rayleigh-Ritz theorem and the Weyl inequality theorem are presented next.

Theorem B.2.1 (Rayleigh-Ritz Theorem [83]):

Let \( A \in \mathbb{C}^{n \times n} \), where \( n \in \mathbb{N}_* \), be Hermitian\(^\dagger\) and let the eigenvalues \( \lambda_i(A) \) be ordered as \( \lambda_{\min} = \lambda_1(\cdot) \leq \cdots \leq \lambda_n(\cdot) = \lambda_{\max}(\cdot) \). Then

\[
\lambda_1(A)x^Tc x \leq x^Tc Ax \leq \lambda_n(A)x^Tc x, \quad \forall x \in \mathbb{C}^n, \quad (B.2.27)
\]

\[
\lambda_{\max}(A) = \lambda_n(A) = \max_{x \in \mathbb{C}^n} \left( \frac{x^Tc Ax}{x^Tc x} \right) = \max_{x^Tc x = 1} (x^Tc Ax),
\]

\(^\dagger\) A matrix \( X \in \mathbb{C}^{n \times n} \) is said to be Hermitian when \( X = X^Tc \) [83].
\[ \lambda_{\min}(A) = \lambda_1(A) = \min_{x \in \mathbb{C}^n} \left( x^T A x / x^T x \right) = \min_{x^T x = 1} \left( x^T A x \right). \]

**Proof.**

The proof is given in section 4.2.2 of [83]. \[\Box\]

**Theorem B.2.2 (Weyl Inequality [83]):**

Let \( A, C \in \mathbb{C}^{n \times n} \), where \( n \in \mathbb{N}_* \), be Hermitian and let the eigenvalues \( \lambda_i(A) \), \( \lambda_i(B) \) and \( \lambda_i(A+B) \) be ordered as \( \lambda_{\min} = \lambda_1(\cdot) \leq \cdots \leq \lambda_n(\cdot) = \lambda_{\max}(\cdot) \). Then \( \forall i \in [1,n], i \in \mathbb{N}_* \) we have

\[ \lambda_i(A) + \lambda_1(B) \leq \lambda_i(A+B) \leq \lambda_i(A) + \lambda_n(B). \quad (B.2.28) \]

**Proof.**

The proof is given in section 4.3.1 of [83]. \[\Box\]

### B.3 Matrix Calculus

Initially, consider definitions B.3.1 and B.3.2 below.

**Definition B.3.1 (Jacobian of a Matrix):**

Let \( f(x_1, x_2, \ldots, x_n) \in \mathcal{C}^{n_f \times m_f} \) be a matrix function of \( x_i \in \mathcal{C}^{n_{x_i}} \), where \( i \in [1,n] \), \( n_f, m_f, n_{x_i}, i \in \mathbb{N}_* \). Then the Jacobian of \( f(x_1, x_2, \ldots, x_n) \in \mathcal{C}^{n_f \times m_f} \) with respect to \( x_i \in \mathcal{C}^{n_{x_i}} \), denoted as \( J_{f(x_1, x_2, \ldots, x_n)}|_{x_i} \in \mathcal{C}^{n_f \times (m_f + n_{x_i})} \), is defined by

\[
J_{f(x_1, x_2, \ldots, x_n)}|_{x_i} := \begin{bmatrix}
\frac{\partial f(x_1, x_2, \ldots, x_n)}{\partial x_{i11}} & \cdots & \frac{\partial f(x_1, x_2, \ldots, x_n)}{\partial x_{in_{x_i}1}}
\end{bmatrix},
\]

where

\[
\left[ \frac{\partial f(x_1, x_2, \ldots, x_n)}{\partial x_{i11}} \right]_{KL} := \frac{\partial \left[ f(x_1, x_2, \ldots, x_n) \right]_{KL}}{\partial x_{i11}}, \quad \forall k \in [1,n_f], \forall l \in [1,m_f],
\]

\( k, l \in \mathbb{N}_* \). Consequently, the Jacobian of \( f(x_1, x_2, \ldots, x_n) \in \mathcal{C}^{n_f \times m_f} \) with respect to \( x_1, x_2, \ldots, x_n \), denoted as \( J_{f(x_1, x_2, \ldots, x_n)}|_{(x_1, x_2, \ldots, x_n)} \in \mathcal{C}^{n_f \times (m_f + n_{x_i})} \), is defined by
\[
J_f(x_1, x_2, \ldots, x_n) := \begin{bmatrix}
J_f(x_1, x_2, \ldots, x_n) & \cdots & J_f(x_1, x_2, \ldots, x_n)
\end{bmatrix}.
\]

Note that, very often for the scalar case where \(f(x_1, x_2, \ldots, x_n) \in \mathbb{C}\), the notation \(\nabla f(x_1, x_2, \ldots, x_n) = J_f(x_1, x_2, \ldots, x_n)(x_1, x_2, \ldots, x_n)\) is used. Also, for the vector case \(f(x) \in \mathbb{C}^n\), \(x \in \mathbb{C}^n\), we often use the notation \(\frac{d(f(x))}{d(x^T)} := J_f(x)\).

**Definition B.3.2 (Frechet Derivative of a Matrix [131]):**

Consider the mapping \(f : \mathbb{C}^{n \times m} \rightarrow \mathbb{C}^{n \times m}, n_X, m_X, n, m \in \mathbb{Z}_+\), with image the matrix \(f(X) \in \mathbb{C}^{n \times m}\). A linear mapping (operator) \(f_X : \mathbb{C}^{n \times m} \rightarrow \mathbb{C}^{n \times m}\) with image \(f_X(X) \in \mathbb{C}^{n \times m}\), is said to be the Frechet derivative of matrix \(f(X)\) at \(X \in \mathbb{C}^{n \times m}\) if, \(\forall X \in \mathbb{C}^{n \times m}\), \(\lim_{\delta X \to 0} \left( \frac{1}{\|\delta X\|} \| f(X + \delta X) - f(X) - f_X(X) \| \right) = 0\), where \(\|\|\) is any reasonable norm.

Note that, although \(f(X)\) can be nonlinear in general, its Frechet derivative \(f_X(X)\) is always linear. The Frechet derivative, just as the derivative of a scalar function indicates how rapidly the function will change with respect to perturbations \(\delta X \in \mathbb{C}^{n \times m}\) in its argument \(X \in \mathbb{C}^{n \times m}\).

Because of definitions B.3.1 and B.3.2, there is a relation between the Jacobian and the Frechet derivative of a matrix. More specific, because \(f_X(X)\) is linear it can be written that \(\text{vec}(f_X(X)) = h(X)\text{vec}(\delta X)\). Now, it turns out by construction that

\[
\lim_{\delta X \to 0} \left( \frac{1}{\|\delta X\|} \| f(X + \delta X) - f(X) - \text{vec}^{-1} \left( J_{\text{vec}}(f(X))\text{vec}(\delta X) \right) \| \right) = 0.
\]

Hence,

---

\(\dagger\) A mapping \(f : S \rightarrow T\) from set \(S\) to \(T\), with image \(f(x) \in T\) is called linear mapping or linear operator, if \(\forall x, y \in S\) and for all scalars \(a\), \(f(x + y) = f(x) + f(y)\) and \(f(ax) = af(x)\).


\[
\begin{align*}
h(X) &= J_{\text{vec}(f(X))\text{vec}(X)} = \frac{d[\text{vec}(f(X))]}{d\left(\text{vec}(X)^T\right)}, \\
\text{and} \\
f_X(X) &= \text{vec}^{-1}\left(\frac{J_{\text{vec}(f(X))\text{vec}(X)}\text{vec}(\delta X)}{\text{vec}(X)}\right).
\end{align*}
\]

Next, some calculus differentiation results of vectored matrix functions are presented. The proof of these results is evident by construction of the relevant functions. In other words, if one performs all the algebra on both sides of the relevant equations, these sides turn out to be equal. Hence, for \( X, T \in \mathbb{C}^{n \times p} \), \( C \in \mathbb{C}^{m \times n} \), \( D \in \mathbb{C}^{p \times n} \), \( E \in \mathbb{C}^{p \times k} \), \( S \in \mathbb{C}^{m \times k} \), \( x, a, b \in \mathbb{C}^{np} \), \( c \in \mathbb{C}^{l} \), \( f_i(x), h(x), g(x) \in \mathbb{C}^{l} \), \( w(x) \in \mathbb{C} \), \( A \in \mathbb{C}^{q \times np} \), where \( m, n, p, k, l, i \in \mathbb{N}_* \), we have the formulas below.

\[
\frac{d}{dx} \left[ \sum_{i=1}^{v} f_i(x) \right] = \sum_{i=1}^{v} \frac{d(f_i(x))}{dx}, \quad \forall v \in \mathbb{N}_*, \ i \in \mathbb{N}_*
\]

(B.3.3)

\[
\frac{d}{dx} \left( h(x) \circ g(x) \right) = \frac{d(h(x))}{dx} \circ \left( g(x) U_{I \times (np)} \right) + \left( h(x) U_{I \times (np)} \right) \frac{d(g(x))}{dx}
\]

(B.3.4)

\[
\frac{d}{dx} (e \circ g(x)) = (e U_{I \times (np)}) \frac{d(g(x))}{dx}
\]

(B.3.5)

\[
\frac{d}{dx} (h(x) \circ w(x)) = \frac{d(h(x))}{dx} w(x) + \left( h(x) U_{I \times (np)} \right) \frac{d(w(x))}{dx}
\]

(B.3.6)

\[
\frac{d}{dx} (e \circ w(x)) = (e U_{I \times (np)}) \frac{d(w(x))}{dx}
\]

(B.3.7)

\[
\frac{d}{dx} (Ax) = A
\]

(B.3.8)

\[
\frac{d}{dx} \left[ \left( E^T \otimes C \right) \left( I_p \otimes (XD) \right) \text{vec}(X) \right] = \\
\frac{d}{dx} \left[ \text{vec}(X)^T \right] \\
\left( E^T \otimes C \right) \left[ \frac{d}{dx} \left( I_p \otimes (XD) \right) \text{vec}(X) + \left( I_p \otimes (XD) \right) \frac{d}{dx} \left[ \text{vec}(X) \right] \right] = ...
\]
Subject to the concept of the Fréchet derivative, (B.3.9) can be alternatively calculated considering (B.3.1). More specifically, the Fréchet derivative of \( f(X) := CXDXE \) can calculated from the perturbation \( f(X + \delta X) \) as shown next.

\[
f(X + \delta X) = C(X + \delta X)D(X + \delta X)E = CXDXE + C\delta XDXE + CXD\delta XE + C\delta X\delta XE
\]

Setting \( f_X(X) = C\delta XDXE + CXD\delta XE \) obviously satisfies

\[
\lim_{\delta X \to 0} \frac{1}{\|\delta X\|} \left\| f(X + \delta X) - f(X) - f_X(X) \right\| = 0, \text{ with respect to definition B.3.2.}
\]

Using (B.2.13) and (B.2.14), we have

\[
\begin{align*}
vec(f_X(X)) &= \left( E^T \otimes C \right) \left[ [I_p \otimes (XD)] + \left( (XD)^T \otimes I_n \right) \right] vec(\delta X) := h(X)vec(\delta X).
\end{align*}
\]

Hence, (B.3.1) results in \( h(X) = \frac{d(\vec{f}(X))}{d(\vec{vec}(X))^T} \), which leads to

the compact formula

\[
\frac{d\left[ \left( E^T \otimes C \right) \left[ [I_p \otimes (XD)] + \left( (XD)^T \otimes I_n \right) \right] \vec{vec}(X) \right]}{d\left[ \vec{vec}(X) \right]^T} = \left( E^T \otimes C \right) \left[ [I_p \otimes (XD)] + \left( (XD)^T \otimes I_n \right) \right]
\]

(B.3.10)

Including Hadamard products in \( CXDXE \), we have the formulas below for \( f(X) := C(T \circ X)D(T \circ X)E \).

\[
\begin{align*}
&\frac{d\left[ \left( E^T \otimes C \right) \left[ [I_p \otimes (T \circ X)D] \right] \vec{vec}(T \circ X) \right]}{d\left[ \vec{vec}(X) \right]^T} = \left( E^T \otimes C \right) \\
&\times \left[ \left[ \frac{d[I_p \otimes ((T \circ X)D)]}{d[\vec{vec}(X)]_{II}} \vec{vec}(T \circ X) \right] + \left[ I_p \otimes ((T \circ X)D) \right] \left( \vec{vec}(T) \circ \frac{d[\vec{vec}(X)]}{d[\vec{vec}(X)]_{II}} \right) \right] \ldots \\
&\ldots \left[ \frac{d[I_p \otimes ((T \circ X)D)]}{d[\vec{vec}(X)]_{(np)I}} \vec{vec}(T \circ X) \right] + \left[ I_p \otimes ((T \circ X)D) \right] \left( \vec{vec}(T) \circ \frac{d[\vec{vec}(X)]}{d[\vec{vec}(X)]_{(np)I}} \right) \right]
\end{align*}
\]

(B.3.11)
Similarly to (B.3.10), the Frechet derivative of \( f(X) := C(T \circ X)D(T \circ X)E \) can calculated from the perturbation \( f(X + \delta X) \) as shown next.

\[
f(X + \delta X) = C(T \circ (X + \delta X))D(T \circ (X + \delta X))E
\]

\[
= C(T \circ X)D(T \circ X)E + C(T \circ \delta X)D(T \circ X)E
\]

\[
+ C(T \circ X)D(T \circ \delta X)E + C(T \circ \delta X)D(T \circ \delta X)E
\]

Setting \( f_X(X) = C(T \circ X)D(T \circ \delta X)E + C(T \circ \delta X)D(T \circ X)E \), obviously satisfies

\[
\lim_{\delta X \to 0} \frac{1}{\|\delta X\|} \left\| f(X + \delta X) - f(X) - f_X(X) \right\| = 0, 
\]

with respect to definition B.3.2.

Using (B.2.13) and (B.2.14), we have

\[
\text{vec}(f_X(X)) = \left( E^T \otimes C \right) \left[ (I_p \otimes (T \circ X)D) + \left( (T \circ X)D \otimes I_n \right) \right] \text{vec}(T \circ \delta X).
\]

Now, using (B.2.19) the last equation becomes equivalently

\[
\text{vec}(f_X(X)) = \left( \left( U_{(np)} \times I \otimes \text{vec}(T) \right) \circ \left( E^T \otimes C \right) \times \left[ (I_p \otimes ((T \circ X)D)) + \left( (T \circ X)D \otimes I_n \right) \right] \right) \text{vec}(\delta X)
\]

Hence, (B.3.1) results in \( h(X) = J_{\text{vec}(f(X))}\text{vec}(X) = \frac{d(\text{vec}(f(X)))}{d(\text{vec}(X)^T)} \), which leads to the compact formula

\[
\frac{d}{\text{vec}(X)^T} \left[ \left( E^T \otimes C \right) \left[ (I_p \otimes ((T \circ X)D)) + \left( (T \circ X)D \otimes I_n \right) \right] \right] =
\]

\[
\left( U_{(np)} \times I \otimes \text{vec}(T) \right) \circ \left( E^T \otimes C \right) \left[ (I_p \otimes ((T \circ X)D)) + \left( (T \circ X)D \otimes I_n \right) \right] \right)
\]

(B.3.12)

### B.4 Useful Bibliography

For further study in relevant subjects with this appendix, very good references are [82], [83], [113], [65], [25]-[29], [128], [175]-[177], [14], [38], [140], [126],..
Appendix C

The GQME-Toolbox

C.1 Introduction

This appendix includes the complete set of software subroutines and the code which constitutes the GQME-Toolbox. In order to install the toolbox, all files should be installed into a common directory and all commands and runs in the MATLAB v. 5.3.1.29215a (R11.1) work space must take place in the same directory.

C.2 Software Code

The software code is given in the subsequent pages.
Appendix C

The GQME-Toolbox

% Finds using a Homotopic Continuation the solution of the General Quadratic Matrix Equation (GQME):
%  A1X+B1+C1X*D1+E1+G=0.
% Where A1,A2,C1,C2: n x n, B1,B2,E1,E2: p x p, D1,D2: p x n, G: n x p, complex matrices.
% Give the command:
% [error,X,JacobianX,LX,ux,eo,polyXa,plX,plepsilon,plpseudos,
nsteps,stepmin,tCPU,nFLOPS,tCPUscale,nFLOPScale]=GQMEHOM1
% (A1,B1,A2,B2,C1,D1,E1,C2,D2,E2,G,A1o,B1o,A2o,B2o,C1o,D1o,
% E1o,E2o,Eo,Go,flagv,flagproj,Aa,applus1,eps,flambda,flag,sux,
% seo,flagproj,Aa,applus1,flagx,flagXo,Yo,flagproj,Aa,applus1,
% efinal,flchom,flcarath,flpseudos,pseudosfin,maxnsteps,
% minnstep,flh,flchom,flagcor,largescale,jacobianx,linesearch,
% displayon,displayoff,plotdisplay);
% place.
% LX=0 when the solution obtained is of GQME at infinity.
% - ux,eo: the variable and equation scaling factors respectively.
% When no scaling used ux and eo are returned as zero arrays.
% See m-file SGQME.m (type help SGQME.m) for details.
% - polyXa=[repolyXa;impolyXa]: Matrix containing on each row
% the coefficients of the polynomial approximation to
% Homotopy paths of vecc(LX*X(epsilon)./(10.^ux)).
% The first np rows are with respect to the real part of
% vecc(LX*X(epsilon)./(10.^ux)), while the last np rows are
% with respect to the imagine part of
% vecc(LX*X(epsilon)./(10.^ux)).
% - plX, plepsilon, plpseudo: The ODE output (points of the
% Homotopy Paths).
% - nsteps: Number of integration steps used in the ODE
% routine RK45GQMEHOM1.m.
% - stepmin: Minimum integration step used in the integration
% ODE routine RK45GQMEHOM1.m.
% - tCPU: The computer CPU time required for tracing the
% homotopy paths when step=1.
% - nFLOPS: The number of FLOPS required for tracing the
% homotopy paths when step=1.
% - tCPUscale: The computer CPU time required for scaling.
% - nFLOPSscale: The number of FLOPS required for
% scaling.

%====================================================================
%INPUTS:
%====================================================================
% - A1o,B1o,A2o,B2o,C1o,D1o,E1o,C2o,D2o,E2o,Qo:
% Initial matrices such that
% A1oXB1o+A2oXB2o+C1oXD1oXE1o+C2oXD2oXE2o+Qo=0.
% - 1>step>0: Fixed step of the discrete Homotopy, from which
% A1,B1,A2,B2,C1,D1,E1,C2,D2,E2,Q, are obtained gradually
% from A1o,B1o,A2o,B2o,C1o,D1o,E1o,C2o,D2o,E2o,Qo, as
% D1=0,step,...,1.
% - Xo,Yo:nxp:Initial conditions.
% When n=p=m=1 and pdef=+-1: Xo>=0 if gam>0, Xo<=0 if
% gam<0.
% >=(<,=) means, Not-negative (Not-positive) definite if
% pdef=-1.
% >=(<,=) means, negative (positive) definite if pdef=1.
% If pdef=0: no specific kind of solution required.
% Also Xo, must have nonzero elements when flh<0.
% Yo>0, always.
% Xo, Yo, in the general case are chosen as random.
Appendix C  The GQME-Toolbox  401

- SignXoYo: nxp. Determines when flh<>0, if
  Xij(0) = (Xoij/Yoij)^0.5. or Xij(0) = -(Xoij/Yoij)^0.5.
  Hence SignXoYoj=1 for Xij(0) = (Xoij/Yoij)^0.5 and
  SignXoYoj=-1 for Xij(0) = -(Xoij/Yoij)^0.5.
- sux, seo, flagscale, flagv, eps, flagopt, flagsym:
  See m-file SGQME.m (Type, help SGQME.m).
- flagproj:
  0, if the projective transformation (Fhat, LX, np+1) is
  used for the solution.
  <>0, when the projective transformation (Fhat, LX, np+1) is
  not used for the solution.
  Where LX=Aa vec(X)+anpplusl.
  Aa=[a1 a2 ... anp] and anpplusl, are random complex
  numbers in general.
  When flagproj=0 then the solution to the easy problem
  (ODE initial condition) is
  automatically fixed by the program as:
  vec(X(0))=anpplusl(pinv(Inp-vec(Xo./Yo)Aa)
  *vec(Xo./Yo), when flh=0.
  vec(X(0))=anpplusl(pinv(Inp-(vec(SignXoYo)
  *vec(Xo./Yo).^0.5)*Aa)
  *(vec(SignXoYo)*vec(Xo./Yo).^0.5),
  when flh<>0.
  (Where ./ denotes element by element division)
  Note that in this case symmetry and special structures of
  Xo, Yo given by the user may be lost
  so the user must give a Xo such that vec(X(0)) is of the
  desired kind.
  Thesis and m-file INGQMEHOM1.m (Type, help INGQMEHOM1.m).
- flh (See m-file INGQMEHOM1.m (Type, help INGQMEHOM1.m)):
  The user has the option to select the
dX(epsilon)/epsilon computation for two kinds of
homotopies:
  1. H(X(epsilon),epsilon)=epsilonf(vecX(epsilon))
     -gam(1-epsilon)(Yo*X(epsilon)*X(epsilon)-Xo),
     when flh<>0.
  2. H(X(epsilon),epsilon)=epsilonf(vecX(epsilon))
     -gam(1-epsilon)(Yo*X(epsilon)-Xo), when flh=0.
  Or when pseudo-arc length parameterization s is used:
  1. H(X(epsilon(s)),epsilon(s))=
     epsilon(s)f(vecX(epsilon(s)))
     -gam(1-epsilon(s))(Yo*X(epsilon(s))*X(epsilon(s))-Xo),
     when flh<>0.
  2. H(X(epsilon(s)),epsilon(s))=
     epsilon(s)f(vecX(epsilon(s)))
     -gam(1-epsilon(s))(Yo*X(epsilon(s))-Xo), when flh=0.
- flchom=0: If H is a complex homotopy. flchom<>0 if H is a
  real homotopy.
- fclarith=0: If complex arithmetic is used. flcarith<>0 if
real arithmetic is used.

- flpsudos<>0: If pseudoarc-length parameterization is used, and flpsudos=0 otherwise.
  For the case flpsudos<>0, it is assumed that
  $H(x(\epsilon(s),\epsilon(s))$ is transversal to zero,
  and more important that $H^{-1}(0)$ is bounded, with no
  solutions at infinity.
  Otherwise the algorithm will no work in general.

- pseudosfin: Estimated pseudo-arc length when such parameterization is used.

- maxnsteps: Maximum number of integration steps allowed in
  the ODE routine RK45GQMEHOM1.m.

- minstep: Minimum integration step allowed in the
  integration ODE routine RK45GQMEHOM1.m.
  Note: maxnsteps & minstep are used together in
  combination for truncate very long paths assumed that
  they divergent to infinity. This is helpful when there
  are infinite number of solutions at infinity. Hence
  Projective Transformation may not be able of computing
  such solutions.

- flagcor: Flag for corrector in each step of the ODE
  RK45GQMEHOM1.m, (On when 0, Off when <>0).

- e: is the zero tolerance.

- efinal: is the zero tolerance for the solution obtained.

- largescale: It is given as string 'on' or 'off' when the
  local solver at epsilon uses a large or a medium scale
  optimization algorithm.
  For more details see MATLAB m-file FSOLVE.m.

- jacobianqme: It is given as string 'on' when the local
  solver at epsilon uses the jacobian explicit formulas, or
  'off' otherwise.

- linesearch: Line search type for the local solver at
  epsilon.
  It is given as a string 'quadcubic' or 'cubicpoly'.
  When jacobianqme='on' then Linesearch='quadcubic' and
  when jacobianqme='off' then Linesearch='cubicpoly', are
  preferable.

- displaysolver: It is given as string 'iter' if display of
  results, of the local solver at epsilon, on each
  iteration required, or 'off' otherwise.

The local solver at epsilon, is a subspace trust region
method, based on an interior-reflective Newton method
of the FSOLVE.m when largescale='on'.
When largescale='off', the local solver at epsilon is the
Gauss-Newton solver of the FSOLVE.m.
For details type: help fsolve and help optimset and view
m-file FGQMEHOM1.m.
Appendix C

- displayode: It is given as string 'on' if display of results of the ODE solver on each step required, or 'off' otherwise.

- plotdisplay: It is given as string 'on' for plotting the 3-D graph (epsilon, realX(epsilon), imagX(epsilon)), or 'off' otherwise.

% For tracking the homotopy paths the RK45GQMEHOM1.m ODE solver is used.
% Computations can take place in complex or real arithmetic in MALAB (version: 5.3.1.29215a (R11.1)).
% For more details see PhD Thesis:

Vassilios A. Tsachouridis.


Department of Engineering, University of Leicester, 2002

function [error,X,JacobianX,ux,polyXa,plX,plepsilon,plpseudo,nsteps,stepmin,tCPU,nFLOPS,tCPUscale,nFLOPSscale]=GQMEHOM1(A1,B1,A2,B2,C1,D1,E1,C2,D2,E2,G,A1o,B1o,A2o,B2o,C1o,D1o,E1o,C2o,D2o,E2o,Go,flagscale,flagv,eps,flagopt,flagsym,sux,seo,flagproj,Aa,anp,plus1,gam,pdef,Xo,Yo,SignXoYo,step,e,efinal,f1h,f1omp,f1cath,flpseudo,pseudoef,manxsteps,manstep,flagfrechet,flagcor,largezscale,jacobian,qme,linesearch,displaysolver,displayode,plotdisplay);

-------------Check the dimension of the input matrices.------------------

[n1, m1]=size(A1);
[n1, m1]=size(B1);
[n2, m2]=size(A2);
[n2, m2]=size(B2);
[n1, m1]=size(C1);
[n1, m1]=size(D1);
[n1, m1]=size(E1);
[n2, m2]=size(C2);
[n2, m2]=size(D2);
[n2, m2]=size(E2);
[n1, m1]=size(G);
[n1, m1]=size(A1o);
[n1, m1]=size(B1o);
[n2, m2]=size(A2o);
[n2, m2]=size(B2o);
[n2, m2]=size(C1o);
[n2, m2]=size(D1o);
[n1, m1]=size(E1o);
[n2, m2]=size(C2o);
[n2, m2]=size(D2o);
[n2, m2]=size(E2o);
[n1, m1]=size(Go);
[n1, m1]=size(Xo);
[nY0, mY0] = size(Y0);
[nSignX0Yo, mSignX0Yo] = size(SignX0Yo);
if (nA1 ~= nA2) | (nA2 ~= nC1) | (nC1 ~= nC2) | (mA1 ~= mA2) | (mA2 ~= mC1) | (mC1 ~= mC2)
disp('A1, A1o, A2, A2o, C1, C1o, and C2, C2o must have the same dimensions nxn.');?>
break
end
if (nB1 ~= nB2) | (nB2 ~= nE1) | (nE1 ~= nE2) | (mB1 ~= mB2) | (mB2 ~= mE1) | (mE1 ~= mE2)
disp('B1, B1o, B2, B2o, E1, E1o, and E2, E2o must have the same dimensions pxp.');?>
break
end
if (nD1 ~= nD2) | (mD1 ~= mD2) | (nD1o ~= nD1o) | (mD2o ~= mD2o)
disp('D1, D1o and D2, D2o must have the same dimensions pxn.');?>
break
end
if (nA1 ~= nG) | (mB1 ~= mG) | (nG ~= nGo) | (mG ~= mGo)
disp('G, Go must have the same number of rows with A1, A2, C1, C2 and the same number of columns with B1, B2, E1, E2; dimension npxp.');?>
break
end
if (nX0 ~= nG) | (mX0 ~= mG) | (nY0 ~= nG) | (mY0 ~= mG)
disp('X0 and Yo must have the same dimensions with G; dimension npxp.');?>
break
end
if (nSignX0Yo ~= nX0) | (mSignX0Yo ~= mX0) | (nSignX0Yo ~= nY0) | (mSignX0Yo ~= mY0)
disp('SignX0Yo must have the same dimensions with X0, Yo; dimension npxp.');?>
break
end
for i = 1:nSignX0Yo
  for j = 1:mSignX0Yo
    if (SignX0Yo(i, j) == 1) & (SignX0Yo(i, j) == -1)
      stopsign = 1;
    end
  end
end
stopsign = 0;
if (stopsign == 1)
disp('SignX0Yo must have elements +1 or -1.');
break
end
if (nA1 ~= mA1)
disp('A1, A1o, A2, A2o, C1, C1o, C2, C2o and Ao must have the same dimensions nxn.');?>
break
end
if (nB1 ~= mB1)
disp('B1, B1o, B2, B2o, E1, E1o, E2, E2o and Bo must have the same dimensions pxp.');?>
break
end
if (step > 1) | (step == 0)
Appendix C

```matlab
disp('step must be 0<step<=1.');
break
end
if (maxnsteps<=0)
disp('maxnsteps must be maxnsteps>0.');
break
end
if (minstep<0)
disp('minstep must be minstep>0.');
break
end
imeqdat1 = norm(imag(A1), 'fro') + norm(imag(A2), 'fro') + norm(imag(B1), 'fro') + norm(imag(B2), 'fro') + norm(imag(C1), 'fro') + norm(imag(C2), 'fro');
imeqdat2 = norm(imag(D1), 'fro') + norm(imag(D2), 'fro') + norm(imag(E1), 'fro') + norm(imag(E2), 'fro') + norm(imag(G), 'fro');
imeqdat3 = norm(imag(Alo), 'fro') + norm(imag(A2o), 'fro') + norm(imag(Blo), 'fro') + norm(imag(B2o), 'fro') + norm(imag(Clo), 'fro') + norm(imag(C2o), 'fro');
imeqdat4 = norm(imag(Dlo), 'fro') + norm(imag(D2o), 'fro') + norm(imag(Elo), 'fro') + norm(imag(E2o), 'fro') + norm(imag(Go), 'fro');
imeqdat5 = norm(imag(Xo), 'fro') + norm(imag(Yo), 'fro') + norm(imag(Anaplus1), 'fro') + norm(imag(SignXoYo), 'fro') + norm(imag(SignXoY), 'fro') + norm(imag(SignXoY2), 'fro');
if (imeqdat1+imeqdat2+imeqdat3+imeqdat4+imeqdat5) ~ 0 & (f1chom=0)
disp('H is a complex homotopy. Give f1chom=0.');
break
end
if (sum(largyscale) ~ = sum('off')) & (sum(largyscale) ~ = sum('on'))
disp('largyscale must be the string 'on' or 'off');
break
end
if (sum(jacobianqme) ~ = sum('off')) & (sum(jacobianqme) ~ = sum('on'))
disp('jacobianqme must be the string 'on' or 'off');
break
end
if (sum(linesearch) ~ = sum('quadcubic')) & (sum(linesearch) ~ = sum('cubicpoly'))
disp('linesearch must be the string 'quadcubic' or 'cubicpoly');
break
end
if (sum(displaysolver) ~ = sum('iter')) & (sum(displaysolver) ~ = sum('off'))
disp('displaysolver must be the string 'iter' or 'off');
break
end
if (sum(displayode) ~ = sum('on')) & (sum(displayode) ~ = sum('off'))
disp('displayode must be the string 'on' or 'off');
break
end
if (sum(plotdisplay) ~ = sum('on')) & (sum(plotdisplay) ~ = sum('off'))
disp('plotdisplay must be the string 'on' or 'off');
break
end
```
Appendix C

The GQME-Toolbox 406

%---------------------------------------------------
% Save dimension numbers, m,n,p,l.
%---------------------------------------------------

m = nA1;
n = mA1;
p = nB1;
l = mB1;
np = n*p;
lm = l*m;
Im = eye(m,m);
In = eye(n,n);
Inn = In;
Ip = eye(p,p);
Inp = eye(np,np);

% Set initial value.

if (flagproj==0)
  if (anplus1==0)
    disp('anplus1 must be nonzero.');
    break
  end
  if (fh==0)
    X = invvec(anplus1*pinv(Inp-
    vecc(Xo./Yo)*Aa)*vecn(Xo./Yo),n,p);
  else
    if (nnz(Xo)<n*p)
      disp('For homotopy map 1. Xo must have nonzero
      elements.');
      break
    end
    if (nnz(Yo)<n*p)
      disp('For homotopy map 1. Yo must have nonzero
      elements.');
      break
    end
    X = invvec(anplus1*pinv(Inp-
    (vecn(SignXoYo).*vecn(Xo./Yo).^0.5)*Aa)*vecn(Xo./Yo).^0.5,n,p);
    Yo.*X.*X-Xo*(Aa*vecn(X)+anplus1)^2
  end
else
  if (fh==0)
    X = Xo./Yo;
  else
    if (nnz(Xo)<n*p)
      disp('For homotopy map 1. Xo must have nonzero
      elements.');
      break
    end
    if (nnz(Yo)<n*p)
      disp('For homotopy map 1. Yo must have nonzero
      elements.');
      break
    end
    X = SignXoYo.*((Xo./Yo).^0.5);
end
end
if (n==p) & (m==l)
eX = eig(X);
for h=1:n
  if (real(eX(h))<0) & (gam>0) & (pdef==1)
    disp('Xo and gam must be consisted.');
    break
  end
end
end
if (real(eX(h))>0) & (gam<0) & (pdef==1)
disp('Xo and gam must be consisted.');
break
end

A1f=A1;
B1f=B1;
A2f=A2;
B2f=B2;
Clf=C1;
D1f=D1;
E1f=E1;
C2f=C2;
D2f=D2;
E2f=E2;
GF=G;

%------------------------Check if initial value is a solution.--------
error=abs(A1f*X*B1f+A2f*X*B2f+C1f*X*D1f*X*E1f+C2f*X*D2f*X*E2f+Gf);
if (norm(error)<=e)
ployXa=0;
else

%------------------------Outer-Loop:Discrete Homotopy-----------------
if (step==1)
D1=1;
flagend=1;
else
D1=0;
flagend=0;
end
while (D1<=1) & (flagend<=1)
fprintf(2, '[Discrete homotopy for data step]=%e\n',D1);
A1=A1o+D1*(A1f-A1o);
B1=B1o+D1*(B1f-B1o);
A2=A2o+D1*(A2f-A2o);
B2=B2o+D1*(B2f-B2o);
Cl=C1o+D1*(C1f-C1o);
D1=D1o+D1*(D1f-D1o);
E1=E1o+D1*(E1f-E1o);
C2=C2o+D1*(C2f-C2o);
D2=D2o+D1*(D2f-D2o);
E2=E2o+D1*(E2f-E2o);
G=Go+D1*(Gf-Go);

%------------------------Make store file storeg empty of any old data.--------
clear storeg
%------------------------Save equations constant data to file storeg.--------
save storeg A1 B1 A2 B2 C1 D1 E1 C2 D2 E2 G p
%------------------------Scale equation.--------------------------
[Abar,Bbar,Cbar,ux,eo,S,Sm,Alhat,B1hat,A2hat,B2hat,Clhat,D1hat,
E1hat,C2hat,D2hat,Ghat,tCPUscale,nFLOPScale]=SGQME(A1,B1,
A2,B2,C1,D1,E1,C2,D2,E2,G,flagscale,flagv,eps,flagopt,flagsym,
sux,seo);
ux10=10.^ux;
veccux=vecc(ux);
vecc10ux=10.^'(veccux);
U1mx1=ones(lm,1);
U1mx1vecc10ux=U1mx1*vecc10ux';
diagvecc10ux=diag(vecc10ux);
eo10=10.^eo;
Appendix C

The GQME-Toolbox 408

Ul\times np=\text{ones}(1,\text{np});
eol\times np1=\text{eo10}\ast Ul\times np;
eol\times np1\times Ul\times mLvecc10uxt=eo10Unp1\ast Ul\times mLvecc10uxt;
%-------------------------Make store file storepg empty of any old data.-----
clear storepg
%-------------------------Save equations constant data to file storepg.-------
save storepg A1 B1 A2 B2 C1 D1 E1 C2 D2 E2 G n p flagproj Aa
anpluss1 ux10 eo10
%-------------------------to save them for future use with INQMEHOM1.m.-----
Ak=kron\bigl(\text{real}\bigl(B1\bigr)+\text{li}\ast \text{imag}\bigl(B1\bigr),A1\bigr)+kron\bigl(\text{real}\bigl(B2\bigr)+\text{li}\ast \text{imag}\bigl(B2\bigr),A2\bigr);
Abark=eo10Unp1\times Ul\times mLvecc10uxt\ast Ak;
tD1=kron\bigl(\text{real}\bigl(E1\bigr)+\text{li}\ast \text{imag}\bigl(E1\bigr),C1\bigr);
tD2=kron\bigl(\text{real}\bigl(E2\bigr)+\text{li}\ast \text{imag}\bigl(E2\bigr),C2\bigr);
veccG=vecc(G);
veccYo=vecc(Yo);
veccXo=vecc(Xo);
TG=eo10\ast veccG;
%-------------------------Set counter k for the file INQMEHOM1.m.m.---------
k=0;
%-------------------------Split initial value to vector vecc-------------------
%------------------------(w,r,t columns).-------------------------------------
X0=vecc(X);
tf\text{in}=1;
if (flpseudos~=0)
    X0=[X0;0];
    tfin=pseudosfin;
end
signRd=1;
%--------------------------Save input and constant data to file gqmesreg.----
clear gqmesreg
%--------------------------Save gqmesreg ux vecc10ux eo10 Ul\times mLvecc10uxt Ak Abark tD1
D1 tD2 D2 veccG tG Aa anpluss1 k gam veccXo veccYo Inn Ip Inp n
p np flagfrechet flagproj flpseudos signRd flchom flcarith
%------------------------Set and save optimization FSO\text{OLVE} parameter-------
%--------------------------to file options data.-------------------------------
options=optimset\bigl('\text{LargeScale}',\text{largescale},'\text{Jacobian}',\text{Jacobian},
\text{LineSearchType}',\text{linesearch},'\text{TolFun}',\text{efinal},'\text{MaxIter}',1000\ast \text{np},
'\text{Display}',\text{display solver});
save options data options
%---------------------------Inner-Loop: Continuous Homotopy.-----------
nFLOPS0=flops;
tCPU=cputime;
[epsilon,X,nsteps,stepmin]=RK45GQMEHOM1\bigl('\text{INGQMEHOM1}',0,tfi\text{n},X0,
\text{e,n,p,gam,pdef,flh,flpseudos,maxnsteps,minstep,flagcor,displayode});
tCPU=cputime-tCPU;
nFLOPS=flops-nFLOPS0;
%-----------------------------Form from solution epsilon(s)=1 the solution X.----
[rx cr]=size(X);
plX=X;
plepsilon=epsilon;
X1=invvec\bigl(\text{real}(X_{(\text{rx}:\text{rx},1:}\text{np}))'+\text{li}\ast \text{imag}(X_{(\text{rx}:\text{rx},1:}\text{np}))',n,p\bigr);
if (flagproj==0)
    LX1=Aa\ast vecc(Xi)+anpluss1;
else
    LXi=1;
end
Appendix C: The GQME-Toolbox


% Evaluate polyXa=f(epsilon) and solution--------------------------------------  
% X(epsilon=1).----------------------------------------------------------------  
for i=1:cr  
    repolyXa(i,:) = polyfit(epsilon, real(X(:,i:i)), 5);  
    reXa(i,1) = polyval(repolyXa(i,:),1);  
    impolyXa(i,:) = polyfit(epsilon, imag(X(:,i:i)), 5);  
    imXa(i,1) = polyval(impolyXa(i,:),1);  
end  

polyXa = [repolyXa; impolyXa];  
Xa = reXa + li*imXa;  
X = invecc(Xa(1:np,:),n,p);  
if (flagproj==0)  
    LX = Aa * veccc(X) + anpplus1;  
else  
    LX = 1;  
end  


% Take best estimate of solution X(epsilon(s)=1).-----------------------------  
if (norm(error)<norm(errori))  
    X=Xi;  
    LX=LXi;  
    error=errori;  
end  

% Scale back solution X.------------------------------------------------------  
X = u10.*X;  
if (LX~=-0)  
    X=(1/LX) * X;  
end  

error = abs(A1*X*B1+A2*X*B2+C1*X*D1*X*E1+C2*X*D2*X*E2+G);  

% If error not acceptable and solution----------------------------------------  
% is not at infinity solve F(veccc(X))=0--------------------------------------  
% with initial condition-------------------------------------------------------  
% the best estimate of solution X(epsilon(s)=1).------------------------------  
if ((norm(error)>efinal) & (LX~=-0) & (nsteps<maxnsteps) & (stepmin>=minstep))  
    [X,fvalX,exitflagX,outputX,jacobianX]=fzero( 'FGQMEHOM1' , [real(X) imag(X)] , options);  
    X=X(1:n,1:p)+li*X(1:n,p+1:2*p);  
    error = abs(A1*X*B1+A2*X*B2+C1*X*D1*X*E1+C2*X*D2*X*E2+G);  
end  

% Compute the Jacobian of GME at the solution-------------------------------  
% X(epsilon(s)=1).-------------------------------------------------------------  
XD1=X*D1;  
XD2=X*D2;  
D1X=D1*X;  
D1XT=real(D1X)'+li*imag(D1X)';  
D2X=D2*X;  
D2XT=real(D2X)'+li*imag(D2X)';  
JacobianX = Ak+t*D1* (kron(Ip, XD1) + kron(D1XT, Inn)) + t*D2* (kron(Ip, XD2) + kron(D2XT, Inn));  
if ((norm(error)>efinal) & (LX~=-1) & (nsteps>maxnsteps) & (stepmin>minstep))
disp('Warning: Solution obtained may be a solution at infinity.')

disp('For verification check if:');
disp('a) L(X)=0');
disp('b) nsteps>maxnsteps');
disp('c) stepmin>minstep');
disp('d) error is not acceptable');
end
if (LX==0)
disp('Warning: Solution obtained is a solution at infinity.');
end
plpsos=plesion;
if (flpsos==0)
    plpsos='empty';
else
    plesion=plX(:,np+1);
end
if (sum(plotdisplay)==sum('on'))
    figure;
    plot3(real(plX),plepsion,imag(plX)) ;grid;title('Homotopy paths of elements of X(epsilon).');
    xlabel('re(X(epsilon)) elements');ylabel('epsilon');zlabel('Im(X(epsilon)) elements');
    plot3(real(plX),plpsos,imag(plX)) ;grid;title('Homotopy paths of elements of [X(s);epsilon(s)].');
    xlabel('[re(X(s));epsilon(s)]; elements');ylabel('s');zlabel('[Im(X(s));epsilon(s)] elements');
else
    plot3(real(plX),plepsion,imag(plX)) ;grid;title('Homotopy paths of elements of X(epsilon).');
    xlabel('re(X(epsilon)) elements');ylabel('epsilon');zlabel('Im(X(epsilon)) elements');
end
end
%------------------------END OF THE PROGRAM---------------------------

%------------------------VASILIOS A. TSACHOURIDIS-----------------------
%---------------------------------16-11-2000---------------------------------
**RK45GQMEHOM1** Solve differential equations, higher order method.
**RK45GQMEHOM1** integrates a system of ordinary differential equations using 4th and 5th order Runge-Kutta formulas.

\[ [T,Y,nsteps,stepmin] = RK45GQMEHOM1(F, T0, Tfinal, Y0, TOL, n, p, gam, def, flh, flpseudos, maxnsteps, minstep, flagcor,displayode), \]
integrates the system of ordinary differential equations described by the M-file F.M, over the interval T0 to Tfinal, with initial conditions Y0. Tolerance TOL displays status while the integration proceeds.

**INPUT:**
- **F** - String containing name of user-supplied problem description.
  Call: yprime = fun(t,y) where F = 'fun'.
- **t** - Time (scalar).
- **y** - Solution column-vector.
- **yprime** - Returned derivative column-vector;
  \[ yprime(i) = \frac{dy(i)}{dt}. \]
- **t0** - Initial value of t.
- **tfinal** - Final value of t.
- **y0** - Initial value column-vector.
- **tol** - The desired accuracy. (Default: tol = 1.e-6).
- **n,p** - Dimensions of the solution Q=invvecc(Y): n x p.
- **gam** - when n=p: gam>0 (gam<0) if X>=0 (X<=0) solution required.
- **pdef** - For Positive (Negative) semidefinite pdef=1
  - For NOT-Positive (NOT-Negative) definite pdef=-1
- **flh** - Flag for ingqmeshom.m
- **flpseudos** - flpseudos>0 when pseudo-arc length parameterization is used.
- **maxnsteps** - Maximum number of integration steps allowed.
- **minstep** - Minimum integration step allowed.
- **flagcor** - Flag for corrector on each step (On when 0, Off when <>0).
- **displayode** - displays the step and epsilon on each iteration if 'on' and does not display if 'off'.

**OUTPUT:**
- **T** - Returned integration time points (column-vector).
- **Y** - Returned solution, one solution column-vector per tout-value.
- **nsteps** - Number of integration steps used in the ODE routine RK45GQMEHOM1.m.
- **stepmin** - Minimum integration step used in the integration ODE routine RK45GQMEHOM1.m.
- **maxnsteps** and **minstep** are used together in
The result can be displayed by: plot(tout, yout).

For more details see PhD Thesis:

Vassilios A. Tsachouridis.

"A Unified Framework for the Numerical Solution &
Analysis of Generalized Algebraic Quadratic Matrix
Equations with Engineering and Scientific Applications:
Theory & Software Design and Implementation"

Department of Engineering,
University of Leicester, 2002

\begin{verbatim}
function [tout, yout, nsteps, stepmin] = RK45_GQME_HOM1(ypfun, t0, tfinal, y0, tol, n, p, gam, pdef, flh, flpseudos, maxnsteps, minstep, flflagc, displayode);
% The Fehlberg coefficients:
alpha = [1/4 3/8 12/13 1 1/2];
beta = [ [ 1 0 0 0 0 0 ]/4
[ 3 9 0 0 0 0 ]/32
[ 1932 -7200 7296 0 0 0 ]/2197
[ 8341 -32832 29440 -845 0 0 ]/4104
[-6080 10400 -28352 9295 -5643 0 ]/20520 ];
gamma = [ [392880 0 395364 385735 -1371249
277020]/7618050
[-2090 0 22528 21970 -15048 -27360]/752400 ];
pow = 1/5;
% Initialization
normercond=tol+1;
t = t0;
hmax = (final - t)/10;
h = hmax/10;
hl=hmax+1;
stepmin=hl;
y = y0(:);
f = zeros(length(y),6);
chunk = 128;
tout = zeros(chunk,1);
yout = zeros(chunk,length(y));
k = 1;
tout(k) = t;
yout(k,:) = y.';
if flagpseudos=0
ynpplus1=y(n*p+1);
end
flp=0;
if flflagc=0
load('optionsdata', 'options');
end
% The main loop.
while (fl=1) & (t + h > t)
if ((flagpseudos=0) & (t + h > tfinal)), h = tfinal - t; end
if (sum(displayode) == sum('on'))
if flagpseudos=0
fprintf(2,'[epsilon step]=[e %e
','[t h]);
else
end
\end{verbatim}
if ynpplus1<0
    disp('Error: epsilon(s)<0');
    break;
end
if t > tfinal
    disp('Arc-length larger than expected.'
end
fprintf(2,'[epsilon s step]=[%e %e %e]
[end
end

% Compute the slopes.
    temp = feval(ypfun, t, y, flh);
    f(:,1) = temp(:);
    for j = 1:5
    temp = feval(ypfun, t+alpha(j)*h, y+h*beta(:,j), flh);
        f(:,j+1) = temp(:);
    end

% Ensure the right direction for the solution when n=p
% and y+h*f*gamma(:,1) is not >=0 or <=0.
if n==p
    if pdef=-1
        %-----------> = 0  Not-Negative definite and <= Not-positive definite.
        eigyl = real(eig(invec(y(l:n*p),n,p)));
        fgammay = f*gamma(:,1);
        eigyl = real(eig(invec(fgamma(1:n*p),n,p)));
        if Q>=0 direction.
            if (gam>0)
                if (min(eigsl)<0) & ((-max(eigyl)/min(eigsl))<h) & ((-max(eigyl)/min(eigsl))>0)
                    h=-max(eigyl)/min(eigsl);
                end
            end
        % For Q<0 direction.
            if (gam<0)
                if (max(eigsl)>0) & ((-min(eigyl)/max(eigsl))<h) & ((-min(eigyl)/max(eigsl))>0)
                    h=-min(eigyl)/max(eigsl);
                end
            end
    end
end
if pdef==1
    %-----------> = 0  Positive semidefinite and <= Negative semidefinite.
    eigyl = real(eig(invec(y(l:n*p),n,p)));
    fgammay = f*gamma(:,1);
    eigyl = real(eig(invec(fgamma(1:n*p),n,p)));
    if Q>=0 direction.
        if (gam>0)
            if (min(eigsl)<0) & ((-min(eigyl)/min(eigsl))<h) & ((-min(eigyl)/min(eigsl))>0)
                h=-min(eigyl)/min(eigsl);
            end
        end
    % For Q<0 direction.
        if (gam<0)
            if (max(eigsl)>0) & ((-max(eigyl)/max(eigsl))<h) & ((-max(eigyl)/max(eigsl))>0)
                h=-max(eigyl)/max(eigsl);
            end
        end
    end
end
Appendix C

Estimate the error and the acceptable error.
\[
\delta = \text{norm}(h\cdot f\cdot \text{gamma}(;2), 'inf');
\]
\[
\tau = \text{tol} \cdot \max(\text{norm}(y, 'inf'), 1.0);
\]
Update the solution only if the error is acceptable.
\[
\text{if} (\delta \leq \tau)
\]
\[
t = t + h;
\]
\[
y = y + h\cdot f\cdot \text{gamma}(;1);
\]
Correct the ODE solution to path tracking if required.
\[
\text{if} (\text{flagcor} == 0)
\]
\[
\text{if} \ f\text{lpseudo} == 0
\]
\[
\epsilon = y(n*p+1);
\]
\[
\text{else}
\]
\[
\epsilon = t;
\]
\[
\text{end}
\]
\[
X = y(1:n*p);
\]
\[
\text{load}('\text{storepg}', 'A1', 'B1', 'A2', 'B2', 'C1', 'D1', 'E1',
'D2', 'E2', 'G', 'ux10', 'eo10');
\]
\[
\text{load}('ggeomestoreg', 'veccl0ux', 'Aa', 'anpplus1',
'veccko', 'vecyo', 'flagproj');
\]
\[
\text{if} (\text{flagproj} == 0)
\]
\[
\text{LX} = Aa \cdot X + anpplus1;
\]
\[
\text{else}
\]
\[
\text{LX} = 1;
\]
\[
Aa = \text{zeros}(1,n*p);
\]
\[
\text{end}
\]
\[
\text{Xvecc10ux} = X \cdot \text{veccl0ux};
\]
\[
\text{Xux} = \text{invvec}(\text{Xvecc10ux}, n, p);
\]
\[
fveccXvecc10ux = eo10 \cdot \text{vecc}(A1 \cdot (Xux \cdot LX) \cdot B1 + A2 \cdot (Xux \cdot LX) \cdot B2 + C1 \cdot Xux \cdot
D1 \cdot Xux \cdot E1 + C2 \cdot Xux \cdot D2 \cdot Xux \cdot E2 + G \cdot (LX^2));
\]
\[
\text{if} (\text{flh} == 0)
\]
\[
H = \epsilon \cdot fveccXvecc10ux - gam \cdot (1 - \epsilon) \cdot (\text{vecyo}. \cdot X \cdot LX - vecyo \cdot (LX A2));
\]
\[
\text{else}
\]
\[
H = \epsilon \cdot fveccXvecc10ux - gam \cdot (1 - \epsilon) \cdot (vecyo. \cdot X \cdot X - vecyo \cdot (LX A2));
\]
\[
\text{end}
\]
\[
\text{end}
\]
\[
\text{end}
\]
\[
k = k + 1;
\]
\[
\text{if} \ k > \text{length(tout)}
\]
\[
tout = [tout; \text{zeros}(\text{chunk}, 1)];
\]
yout = [yout; zeros(chunk,length(y))];
end
tout(k) = t;
yout(k,:) = y.';
end
% Update the step size.
if (delta ~= 0.0)
    h = min([hmax 0.8*h*(tau/delta)^pow h1]);
    h1=hmax+1;
% Stop the maximum number of steps and minimum step exceeded.
    nsteps=k;
    stepmin=min(stepmin,h);
    if ((h<minstep)&&(k>maxnsteps))
        t=tfinal;
        disp('Minimum step less than specified and maximum
number of steps exceeded.');
    end
end
end
if fflpseudos==0
    ynpplus1=y(n*p+1);
    if ynpplus1>=1
        flps=1;
    end
end
if (sum(displayode)==sum('on'))
    fprintf(2,'%epsilon s step=[%e %e %e]
',ynpplus1 t h));
end
else
    ynpplus1=1;
end
end
if (t < tfinal) && (flps==1)||(ynpplus1<1) && (flps==1))
    disp('Singularity likely.');
end
tout = tout(1:k);
yout = yout(1:k,:);
%-----------------------------------------------END OF THE PROGRAM-----------------------------------------------
%-----------------------------------------------VASSILIOS A. TSACHOURIDIS-----------------------------------------------
%---------------------------------------------------------------------------------------------------
% Forms the derivative vector ordinary differential equation, of the Homotopy continuation, used for the solution of the scaled and projective transformed General Quadratic Matrix Equation (GQME):

% (10.^e0)*vec([A1((10.^ux)*X*LX)*B1+A2((10.^ux)*X*LX)*B2+
+C1((10.^ux)*X*D1((10.^ux)*X)*E1+
+C2((10.^ux)*X)*D2((10.^ux)*X)*E2+G*(L*X^2)])]=0.

% Where, * denotes the Hadamard product.

% This file is used as input to the differential equation solvers, RK45GQMEHOM1.m. See RK45GQMEHOM1.m m-file.
% Computation of derivatives are based on Jacobians via Frechet derivatives.
% The user has the option to select the dX(epsilon)/epsilon computation for two kinds of homotopies:

% 1. \( H(X(\epsilon), \epsilon) = \epsilon f(\text{vec}(10.^{ux}X(\epsilon))) - \gamma (1-\epsilon)(Y o * X(\epsilon) - X o) \)

% 2. \( H(X(\epsilon), \epsilon) = \epsilon f(\text{vec}(10.^{ux}X(\epsilon))) - \gamma (1-\epsilon)(Y o * X(\epsilon) - X o) \)

% Or when pseudo-arc length parameterization \( s \) is used:

% 1. \( H(X(\epsilon(s)), \epsilon(s)) = \epsilon(s)f(\text{vec}(10.^{ux}X(\epsilon(s))) \)

% 2. \( H(X(\epsilon(s)), \epsilon(s)) = \epsilon(s)f(\text{vec}(10.^{ux}X(\epsilon(s))) \)

% For more details see m-file GQMEHOM1.m (Type, help GQMEHOM1.m).

% For details see m-file GQMEHOM1.m (Type, help GQMEHOM1.m).

% For more details see PhD Thesis:

% Vassilios A. Tsachouridis.


% Department of Engineering,
% University of Leicester, 2002

function [Xp]=INGQMEHOM1(epsilon,X,flh);
'D2', 'E2', 'G', 'ux10');
load('gqmesstoresg', 'ux', 'veccl0ux', 'eo10',
    'eo10Unpx1XUlmx1veccl0uxt', 'Abark', 'tD1', 'tD2', 'tG', 'Aa',
    'anplusplus', 'k', 'gam', 'veccXo', 'veccYo', 'Inn', 'Ip', 'Inp',
    'n', 'p', 'flagproj', 'flpseudos', 'signRd', 'flchom',
    'fclarith');
if flpseudos==0
    spseudos=epsilon;
    epsilon=X(n*p+1);
    X=X(1:n*p);
end
if flagproj==0
    LX=Aa*X+anplusplus;
else
    LX=1;
    Aa=zeros(1,n*p);
end
dX L X d vec cX = (Inp * LX + X * Aa) ;
dX o L X d vec cX = diag (veccYo) * (Inp * LX + X * Aa) ;
dLX2dvecCX = 2 * tG * LX * Aa ;
dLoLX2dvecCX = 2 * veccXo * LX * Aa ;
veccl0ux = X .* veccl0ux ;
veccl0ux = invvec (veccl0ux, n, p) ;
tXuXt = Xux * Dl ;
tXuXD = Xux * D2 ;
tDl Xux = Dl * Xux ;
tD2 Xux = D2 * Xux ;
end
if (flih == 0)
    dH d vec cX = epsilon * dveccl0uxt * (1 - epsilon) * (dX L X d vec cX - dLX2dvecCX) ;
else
    dH d vec cX = epsilon * dveccl0uxt * (dLX2dvecCX - dLoLX2dvecCX) ;
end
Xp = - pinv (dH d vec cX) * dH d epsilon ;
if flpseudos==0
    if fclarith
        % If pseudo-arc length parameterization is used.
        Xp = Xp + er ;
    else
        % If flchom==0
        % If H is complex analytic and complex arithmetic is used.
        Xp = Xp ;
    else
        % If flchom==0
        % If H is complex analytic and real arithmetic is used.
        Xp = Xp ;
    end
end
Appendix C

The GQME-Toolbox

\[dP_{\text{vec}X} = \text{real}(dH_{\text{vec}X}) - \text{imag}(dH_{\text{vec}X})\]
\[= \text{real}(dH_{\text{depsilon}}) + \text{imag}(dH_{\text{depsilon}})\]

\[dP_{\text{psuedo}} = \text{real}(dH_{\text{epsilon}}) - \text{imag}(dH_{\text{epsilon}})\]

\[\text{for } i = 1:2n*p\]
\[\text{signRd} = \text{signRd} \times \text{sign}(Rr(i,i))\]

\[\text{signXp} = (-1)^{(2n*p)} \times \text{signRd}\]

\[\% \text{Select initial direction and save it in the file signXpo.}\]

\[\text{if } (\text{spseudo} = 0)\]
\[\% \text{Select initial direction such that}\]
\[\% \text{epsilon}(\text{spseudo}) > 0\]
\[\text{Signes} = \text{signXp} \times \text{Qr}(2n*p,2n*p+1)\]

\[\text{if } (\text{Signes} < 0)\]
\[\text{signXpo} = -\text{sign}(\text{Signes})\]
\[\text{else}\]
\[\text{signXpo} = \text{sign}(\text{Signes})\]
\[\text{end}\]

\[\text{save signXpofile signXpo}\]

\[\text{load('signXpofile', 'signXpo')};\]

\[\text{if } (\text{signXp} \times \text{signXpo} < 0)\]
\[\text{signXp} = -\text{signXp}\]

\[\% \text{END OF THE PROGRAM}\]

\%-----------------------------------------------VASSILIOS A. TSACHOURIDIS-----------------------------------------------

\%--------------------------16-11-2000-----------------------------------------------
Appendix C

The GQME-Toolbox 419

%**************************************************************%  
%**************************************************************%  
%* M-File by *  
%* V. A. TSACHOURIDIS *  
%* 05-06-2000 *  
%* All rights reserved. *  
%**************************************************************%  
%**************************************************************%

%Compute the scaling factors ux and eo for the scaled General
%Quadratic Matrix Equation (GQME):

%  (10.^eo) * [vec(Am1((10.^ux) *X)Bm1)+Am2((10.^ux) *X)Bm2
%  +Cm1((10.^ux) *X)Dm1((10.^ux) *X)Em1
%  +Cm2((10.^ux) *X)Dm2((10.^ux) *X)Em2+Gm)] = 0.
%
% * and o denote the Hadamard and Kronecker product
% respectively
% .^ denote element to element matrix operator with base 10
% and exponent ux.
% Am1,Bm1,Am2,Bm2,Cm1,Dm1,Em1,Cm2,Dm2,Em2,Gm, are the
% respective magnitudes of the coefficients,
% A1,B1,A2,B2,C1,D1,E1,C2,D2,E2,G of the non-scaled GQME
% A1Xb1+A2Xb2+C1Xd1+D1Xd2+E1+G = 0.
% The method is based in the minimization of the individual
%magnitudes and relative magnitude variations of the exponents
%of the coefficients in the equation above.
%When a coefficient is a complex (not real) matrix, minimization
%of its magnitude result in minimization of both real and
%imaginary parts of the matrix since for a complex matrix
%A=real(A)+imag(A)i, it is (from the Pythagorean theorem),
%Am>=real(A), Am>=imag(A).
%Zero coefficient exponents can be scaled assuming they are
%equal to very small number eps provided by the user, or they
%can be removed from the minimization. In the later case the
%results will be different, and symmetry if required may not
%be preserved.
%The user must declare which of the two cases above will take
%place with eps.
%The scaled equation with the coefficients in magnitude format
%can be written as:
%
%a)  (Abar)vec(Xhat)+(Bbar)vec(Xhat' o Xhat)+Cbar=0
%    for Optimal Scaling.
%    Where, Abar: ltxxp, Bbar: ltxnp^2 ,Cbar: ltxp, Xhat: nxp.
%b)  A1hatXhatB1hat+A2hatXhatB2hat+C1hatXhatD1hatXhatE1hat
%    +C2hatXhatD2hatXhatE2hat+Ghat=0
%    for Sub-optimal Scaling.
%    Where, A1hat,A2hat,Clhat,C2hat:txn,
%    B1hat,B2hat,E1hat,E2hat: px1, D1hat,D2hat: pxn,
%    Ghat: tx1, Xhat: nxp.
%
The original solution to the NON-scaled equation (ux=0,oe=0)
is recovered by X=(10.^ux)*Xhat.
%  
%
%**************************************************************%  
%M-FILES (SUBROUTINES) USED WITH THIS PROGRAM: =  
%**************************************************************%
Appendix C

The GQME-Toolbox 420

---

\% vecc.m
\% invvecc.m
\% svec.m
\% insvec.m
\% kronsum.m
\% skronsum.m

\% PROGRAM INPUTS:

\% A1,B1,A2,B2,C1,D1,E1,C2,D2,E2,G. Are the non-scaled GQME equation coefficients.
\% flagscale=0 if scaling required.
\% flagscale<>0 if scaling not required. In this case all outputs returned as string 'empty', and ux=0lx1.
\% flagv=0 if both minimization of the individual magnitudes and relative magnitude variations of the exponents of the coefficients required.
\% flagv<>0 if only minimization of the individual magnitudes of the exponents of the coefficients required.
\% eps=0 if zero coefficient exponents are not scaled; no approximation to zero take place.
\% eps<>0 if zero coefficient exponents are scaled; zero is approximated by eps.
\% flagopt=0 for optimal scaling.
\% In this case A1hat,B1hat,A2hat,B2hat,C1hat,D1hat,E1hat, C2hat,D2hat,E2hat,Ghat are returned as string 'empty'.
\% flagopt<>0 for constrained (sub-optimal) scaling, when all elements of ux are equal and the same hold for the elements of eo.
\% flagsym=0 if square symmetric ux and invvec(eo) are not required.
\% flagsym<>0 if square symmetric ux and invvec(eo) required.
\% sux, seo are the symbolic representation of the X scale factor ux and the equations scale factor eo respectively. sux & seo are given as a symbolic input matrices as follows:
\% If flagv=0 then sux and seo are any symbolic (type-real) quantities with the same dimensions as ux and eo respectively.
\% Hence sux is defined as sux=[suxij], i=1,...,n , j=1,...,p, with suxij=sym('suxij','real').
\% Similarly seo is defined as seo=[seoi], i=1,...,lt, with seoi=sym('seoi','real').
\% Note that when the data are complex numbers ux & eo are still real,
\% since the method consider the magnitudes of the complex
% numbers.
% If flagv<>0 then sux and seo are any symbolic quantities.

%%%%%%%%%%%%%%%%%%%%%%%%
PROGRAM OUTPUTS:=
%%%%%%%%%%%%%%%%%%%%%%%%
% - Abar,Bbar,Cbar,ux,eo,Sm,A1hat,B1hat,A2hat,B2hat,
% C1hat,D1hat,Elhat,C2hat,D2hat,E2hat,Ghat.
% - ux is the variables scale factor with dimension nxp (when
% no scaling required is returned as zero array 0nxp).
% - eo is the equations scale factor with dimension t1x1
% (when no scaling required is returned as zero vector
% 0t1x1).
% - S is the value of the objective function before scaling.
% - Sm is the minimum of the objective function obtained.
% - If Sm>S then no scaling take place (no minimum obtained),
% and ux=0nxp, eo=0t1x1.
% - tCPUscale: The computer CPU time required for scaling.
% - nFLOPSscale: The The number of FLOPS required for
% scaling.
%%%%%%%%%%%%%%%%%%%%%%%%
%For more details see PhD Thesis:
% Vassilicos A. Tsachouridis.
% "A Unified Framework for the Numerical Solution &
% Analysis of Generalized Algebraic Quadratic Matrix
% Equations with Engineering and Scientific Applications:
% Theory & Software Design and Implementation"
% Department of Engineering,
% University of Leicester, 2002
%Give the Command:
% [Abar,Bbar,Cbar,ux,eo,S,Sm,A1hat,B1hat,A2hat,B2hat,C1hat,
% D1hat,Elhat,C2hat,D2hat,E2hat,Ghat,tCPUscale,nFLOPSscale]=
% SGQME(A1,B1,A2,B2,C1,D1,E1,C2,D2,E2,G,flagscale,flagv,eps,
% flagopt,flagsym,sux,seo);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [Abar,Bbar,Cbar,ux,eo,S,Sm,A1hat,B1hat,A2hat,B2hat,C1hat,
 D1hat,Elhat,C2hat,D2hat,E2hat,Ghat,tCPUscale,nFLOPSscale]=SGQME(A1,B1
 ,A2,B2,C1,D1,E1,C2,D2,E2,G,flagscale,flagv,eps,flagopt,flagsym,
sux,seo);
nFLOPSscale0=flops;
tCPUscale=cputime;
[nA1,mA1]=size(A1);
[nB1,mB1]=size(B1);
\[ n_{A2}, m_{A2} = \text{size}(A2); \]
\[ n_{B2}, m_{B2} = \text{size}(B2); \]
\[ n_{C1}, m_{C1} = \text{size}(C1); \]
\[ n_{D1}, m_{D1} = \text{size}(D1); \]
\[ n_{E1}, m_{E1} = \text{size}(E1); \]
\[ n_{C2}, m_{C2} = \text{size}(C2); \]
\[ n_{D2}, m_{D2} = \text{size}(D2); \]
\[ n_{E2}, m_{E2} = \text{size}(E2); \]
\[ n_{G}, m_{G} = \text{size}(G); \]
\[ n_{sux}, m_{sux} = \text{size}(sux); \]
\[ n_{seo}, m_{seo} = \text{size}(seo); \]

if \((n_{A1} = n_{A2}) \mid (n_{A2} = n_{C1}) \mid (n_{C1} = n_{C2}) \mid (m_{A1} = m_{A2}) \mid (m_{A2} = m_{C1}) \mid (m_{C1} = m_{C2})\)
\[ \text{disp('A1, A2, C1 and C2 must have the same dimensions txn.'; break end; if } \]
\[(n_{B1} = n_{B2}) \mid (n_{B2} = n_{E1}) \mid (n_{E1} = n_{E2}) \mid (m_{B1} = m_{B2}) \mid (m_{B2} = m_{E1}) \mid (m_{E1} = m_{E2})\]
\[ \text{disp('B1, B2, E1 and E2 must have the same dimensions px1.'; break end; if } \]
\[(n_{D1} = n_{D2}) \mid (m_{D1} = m_{D2}) \mid (n_{D1} = n_{B1}) \mid (m_{D1} = m_{A1}) \mid (n_{D2} = n_{B2}) \mid (m_{D2} = m_{A2})\]
\[ \text{disp('D1 and D2 must the same dimension and the same number of rows with B1, B2, E1, E2 and the same number of columns with A1, A2, C1, C2; dimension pxn.'; break end; if } \]
\[(n_{A1} = n_{G}) \mid (m_{B1} = m_{G})\]
\[ \text{disp('G must have the same number of rows with A1, A2, C1, C2 and the same number of columns with B1, B2, E1, E2; dimension txl.'; break end; if } \]
\[(\text{flagv=0})\]
\[ \text{if } \neg ((n_{sux} = n_{A1}) \& (m_{sux} = n_{B1})) \]
\[ \text{disp('The symbolic sux must have the same dimensions with u; dimension nxp.'; break; end; if } \]
\[ \neg ((n_{seo} = n_{A1} \ast m_{B1}) \& \neg (m_{seo} = 1)) \]
\[ \text{disp('The symbolic seo must have the same dimensions with eo; dimension tlx1.'; break; end; end; t=n_{A1}; n=m_{A1}; p=n_{B1}; l=m_{B1}; lt=l \ast t; np=n+p; np2=np \ast 2; \]

%-------------------------------------------------No scaling required.--------------------------------------------------
Abar='empty';
Bbar='empty';
Cbar='empty';
Alhat='empty';
B1hat='empty';
A2hat='empty';
B2hat='empty';
C1hat='empty';
D1hat='empty';
E1hat='empty';
C2hat='empty';
E2hat='empty';
D2hat='empty';
Ghat='empty';
ux=zeros(n,p);
eo=zeros(lt,1);
S='empty';
Sm='empty';

%---------------------------------------------------Scaling required.--------------------------------------------------
if (flagscale==0)
%-------------------------------Determine presence of linear-------------------------------
%-------------------------------and quadratic terms.-------------------------------
omegal=1;
omega2=1;
    if ((norm(A1,'fro')==0) | (norm(B1,'fro')==0))
        omegal=0;
    end
    if ((norm(A2,'fro')==0) | (norm(B2,'fro')==0))
        omega2=0;
    end
omega=omegal+omega2;
ksi1=1;
ksi2=1;
    if ((norm(C1,'fro')==0) | (norm(D1,'fro')==0) | (norm(E1,'fro')==0))
        ksi1=0;
    end
    if ((norm(C2,'fro')==0) | (norm(D2,'fro')==0) | (norm(E2,'fro')==0))
        ksi2=0;
    end
ksi=ksi1+ksi2;
%-------------------------------Determine presence of constant term.-------------------
cc=1;
    if (norm(G,'fro')==0)
        cc=0;
    end
    if (omega+ksi==0) & (cc==0)
disp('Infinite number of solutions. Solution Set the Euclidian space nxp.');
break;
    end
    if (omega+ksi==0) & (cc==0)
disp('No solution exist.');
break;
    end
%-------------------------------Continue.-------------------------------------------------
It=eye(t,t);
In=eye(n,n);
Ip=eye(p,p);
Inp=eye(np,np);
Il=eye(lt,lt);
Inlp=eye(np,lt);
iltnp=eye(lt,np);
Ulx1=ones(lt,1);
Appendix C

The GQME-Toolbox 424

\[
\begin{align*}
U_{lxl} &= \text{ones}(1,lt); \\
Unpxl &= \text{ones}(np,1); \\
U_{lxnp} &= \text{ones}(1,np); \\
U_{lxl} &= \text{ones}(1,np2); \\
Unxp &= \text{ones}(n,p); \\
Unx &= \text{ones}(n,n); \\
Upxp &= \text{ones}(p,p); \\
U_{t}x &= \text{ones}(t,1); \\
Unpxn &= \text{ones}(np,np); \\
Unpxlt &= \text{ones}(np,lt); \\
ULtxn &= \text{ones}(lt,np); \\
ULtnpxl &= \text{ones}(lt*np,1); \\
\text{if} \ (\text{flagopt}==0) \\
\text{sux} &= \text{sux}(1,1)*Unxp; \\
\end{align*}
\]

\%-----------------------------------------------If zero coefficient exponents are scaled.---------------------------------------------

\text{if} \ (\text{eps}==0) \\
\text{for} \ i=1:t \\
\text{for} \ j=1:n \\
\text{if} \ (\text{abs}(A_{1}(i,j))==0) \\
A_{1}(i,j) = \text{eps}; \\
\text{end} \\
\text{if} \ (\text{abs}(A_{2}(i,j))==0) \\
A_{2}(i,j) = \text{eps}; \\
\text{end} \\
\text{if} \ (\text{abs}(C_{1}(i,j))==0) \\
C_{1}(i,j) = \text{eps}; \\
\text{end} \\
\text{if} \ (\text{abs}(C_{2}(i,j))==0) \\
C_{2}(i,j) = \text{eps}; \\
\text{end} \\
\text{end} \\
\text{end} \\
\text{for} \ i=1:t \\
\text{for} \ j=1:p \\
\text{if} \ (\text{abs}(B_{1}(i,j))==0) \\
B_{1}(i,j) = \text{eps}; \\
\text{end} \\
\text{if} \ (\text{abs}(B_{2}(i,j))==0) \\
B_{2}(i,j) = \text{eps}; \\
\text{end} \\
\text{if} \ (\text{abs}(E_{1}(i,j))==0) \\
E_{1}(i,j) = \text{eps}; \\
\text{end} \\
\text{if} \ (\text{abs}(E_{2}(i,j))==0) \\
E_{2}(i,j) = \text{eps}; \\
\text{end} \\
\text{end} \\
\text{end} \\
\text{for} \ i=1:p \\
\text{for} \ j=1:n \\
\text{if} \ (\text{abs}(D_{1}(i,j))==0) \\
D_{1}(i,j) = \text{eps}; \\
\text{end} \\
\text{if} \ (\text{abs}(D_{2}(i,j))==0) \\
D_{2}(i,j) = \text{eps}; \\
\text{end} \\
\text{end} \\
\text{end} \\
\text{for} \ i=1:t
\]
for j=1:n
    if (abs(A1(i,j))==0)
        SignA1(i,j)=0;
    end
    if (abs(A2(i,j))==0)
        SignA2(i,j)=0;
    end
    if (abs(C1(i,j))==0)
        SignC1(i,j)=0;
    end
    if (abs(C2(i,j))==0)
        SignC2(i,j)=0;
    end
end
for i=1:t
    for j=1:n
        if (abs(A1(i,j))==0)
            SignA1(i,j)=0;
        end
        if (abs(A2(i,j))==0)
            SignA2(i,j)=0;
        end
        if (abs(C1(i,j))==0)
            SignC1(i,j)=0;
        end
        if (abs(C2(i,j))==0)
            SignC2(i,j)=0;
        end
    end
end
for i=1:p
    for j=1:1
        if (abs(B1(i,j))==0)
            SignB1(i,j)=0;
        end
        if (abs(B2(i,j))==0)
            SignB2(i,j)=0;
        end
        if (abs(E1(i,j))==0)
            SignE1(i,j)=0;
        end
        if (abs(E2(i,j))==0)
            SignE2(i,j)=0;
        end
    end
end
for i=1:p
    for j=1:n
        if (abs(D1(i,j))==0)
            SignD1(i,j)=0;
        end
        if (abs(D2(i,j))==0)
            SignD2(i,j)=0;
        end
    end
end
for i=1:t
    for j=1:1
        if (abs(G(i,j))==0)
            SignG(i,j)=0;
        end
    end
end
for i=1:t
    for j=1:n
        if (abs(LA1(i,j))==inf)
            LA1(i,j)=0;
        end
        if (abs(LA2(i,j))==inf)
            LA2(i,j)=0;
        end
        if (abs(LC1(i,j))==inf)
            LC1(i,j)=0;
        end
        if (abs(LC2(i,j))==inf)
            LC2(i,j)=0;
        end
    end
end
for i=1:p
    for j=1:1
        if (abs(LB1(i,j))==inf)
            LB1(i,j)=0;
        end
        if (abs(LB2(i,j))==inf)
            LB2(i,j)=0;
        end
        if (abs(LE1(i,j))==inf)
            LE1(i,j)=0;
        end
        if (abs(LE2(i,j))==inf)
            LE2(i,j)=0;
        end
    end
end
for i=1:p
    for j=1:n
        if (abs(LD1(i,j))==inf)
            LD1(i,j)=0;
        end
        if (abs(LD2(i,j))==inf)
            LD2(i,j)=0;
        end
    end
end
for i=1:t
    for j=1:1
if (abs(LG(i,j))==inf)
    LG(i,j)=0;
end
end
LBA1=kronsum(LB1',LA1);
LBA2=kronsum(LB2',LA2);
LE1C1=kronsum(LE1',LC1);
LE2C2=kronsum(LE2',LC2);
veccLD1=vecc(LD1);
veccLD2=vecc(LD2);
veccLD1t=vecc(LD1');
veccLD2t=vecc(LD2');
veccLG=vecc(LG);
Unpx1veccLD1=Unpx1*vecc(LD1)';
Unpx1veccLD2=Unpx1*vecc(LD2)';
KrU1xnpLE1C1=kron(U1xnp,LE1C1);
KrU1xnpLE2C2=kron(U1xnp,LE2C2);
SgnB1A1=kron(real(SignB1)'+1i*imag(SignB1)',SignA1);
SgnB2A2=kron(real(SignB2)'+1i*imag(SignB2)',SignA2);
SgnE1C1=kron(real(SignE1)'+1i*imag(SignE1)',SignC1);
SgnE2C2=kron(real(SignE2)'+1i*imag(SignE2)',SignC2);
SgnD1=Unpx1*(real(vecc(SignD1))'+1i*imag(vecc(SignD1))');
SgnD2=Unpx1*(real(vecc(SignD2))'+1i*imag(vecc(SignD2))');
SgnE1D1C1=(Ultxl*(real(vecc(SignD1))'+1i*imag(vecc(SignD1))')).*kron(U1xnp,SgnE1C1);
SgnE2D2C2=(Ultxl*(real(vecc(SignD2))'+1i*imag(vecc(SignD2))')).*kron(U1xnp,SgnE2C2);
SgnE=vecc(SgnD);
%---------------------------------Compute the value of the objective-------------------%---------------------------------function before scaling.-----------------------------------
Sin=[omega1*LBA1 omega2*LB2A2
ks1*(Ultxl*vecc(Unpx1veccLD1)'+KrU1xnpLE1C1
ks2*(Ultxl*vecc(Unpx1veccLD2)'+KrU1xnpLE2C2) cc*veccLG];
Siv=0;
if (flagv==0)
kSin=0;
for i=1:lt
    for j=1:2*(np+np2)+1
        if (~(Sin(i,j)==0))
            kSin=kSin+1;
            Sinv(1,kSin)=Sin(i,j);
        end
    end
end
for i=1:kSin-1
    for j=i+1:kSin
        Siv=Siv+(Sinv(1,i)-Sinv(1,j))^2;
    end
end
if (np2>100)
    S=normest(Sin,'fro')^2+Siv;
else
    S=norm(Sin,'fro')^2+Siv;
end
%-----------------------------------Minimization of coefficient-------------------%-----------------------------------variation required.-----------------------------------
if (flagv==0)
sveccuxt=svecc(sux)';
SLlp=Ultxl*sveccuxt+seo*U1xnp+LB1A1;
\[ S_{L2p} = U_{ltx1} \cdot svecct + \text{sec} + U_{lnp} + LB2A2; \]
\[ S_{Q1p} = U_{ltx1} \cdot svec((\text{kronsum}(sux',sux) + U_{npx1vecLD1})' \cdot s\text{ec} + U_{lnp} + K \times U_{lnpLE1C1}; \]
\[ S_{Q2p} = U_{ltx1} \cdot svec((\text{kronsum}(sux',sux) + U_{npx1vecLD2})' \cdot s\text{ec} + U_{lnp} + K \times U_{lnpLE2C2}; \]
\[ SCp = s\text{ec} + \text{vec}c \times LG; \]
\[ S_{L1m} = - U_{ltx1} \cdot svec(t + \text{sec} + U_{lnp} + LB1A1; \]
\[ S_{L2m} = - U_{ltx1} \cdot svec(t + \text{sec} + U_{lnp} + LB2A2; \]
\[ S_{Q1m} = U_{ltx1} \cdot svec((\text{kronsum}(sux',sux) + U_{npx1vecLD1})' - \text{sec} + U_{npx2} \times Krt1U_{npx1vecLE1C1}; \]
\[ S_{Q2m} = U_{ltx1} \cdot svec((\text{kronsum}(sux',sux) + U_{npx1vecLD2})' - \text{sec} + U_{npx2} \times Krt1U_{npx1vecLE2C2}; \]
\[ V_{THETA v lp} = [\omega_{l} \times S_{L1p} \quad \omega_{2} \times S_{L2p} \quad \kappa_{l} \times S_{Q1p} \quad \kappa_{2} \times S_{Q2p} \quad cc \times SCp]; \]
\[ V_{THETA v lm} = [\omega_{l} \times S_{L1m} \quad \omega_{2} \times S_{L2m} \quad \kappa_{l} \times S_{Q1m} \quad \kappa_{2} \times S_{Q2m} \quad cc \times SCm]; \]
\[ \%-----------------------------------------------Form compact row vectors THETAvlp-------\]
\[ \%-----------------------------------------------and THETAvlm with the non-zero-------------\]
\[ \%-----------------------------------------------elements of VTHETAvlp and VTHETAvlm-------\]
\[ \%-----------------------------------------------respectively.---------------------------------------------------------\]
\[ k_t = 0; \]
\[ \text{for } i = 1:lt \]
\[ \quad \text{for } j = 1:2 \times (np + np2) + 1 \]
\[ \quad \quad \text{if } (V_{THETA v lp}(i,j) = 0) \]
\[ \quad \quad \quad k_t = k_t + 1; \]
\[ \quad \quad \quad \text{THETAvlp}(1,k_t) = V_{THETA v lp}(i,j); \]
\[ \quad \quad \quad \text{THETAvlm}(1,k_t) = V_{THETA v lm}(i,j); \]
\[ \quad \text{end} \]
\[ \text{end} \]
\[ \text{end} \]
\[ TJvuxp = 0 \times svecct; \]
\[ TJvuxm = 0 \times svecct; \]
\[ TJveop = 0 \times \text{sec}'; \]
\[ TJveom = 0 \times \text{sec}'; \]
\[ \text{for } i = 1:k_t - 1 \]
\[ \quad \text{for } j = i+1:k_t \]
\[ \quad \quad TJvuxp = TJvuxp + simple(jacobian((THETAvlp(1,i) - THETAvlp(1,j))^2, svecct)); \]
\[ \quad \quad TJveop = TJveop + simple(jacobian((THETAvlp(1,i) - THETAvlp(1,j))^2, \text{sec}')); \]
\[ \quad \quad TJvuxm = TJvuxm + simple(jacobian((THETAvlm(1,i) - THETAvlm(1,j))^2, svecct)); \]
\[ \quad \quad TJveom = TJveom + simple(jacobian((THETAvlm(1,i) - THETAvlm(1,j))^2, \text{sec}')); \]
\[ \text{end} \]
\[ \text{end} \]
\[ TJvuxp = 0.5 \times TJvuxp; \]
\[ TJveop = 0.5 \times TJveop; \]
\[ TJvuxm = 0.5 \times TJvuxm; \]
\[ TJveom = 0.5 \times TJveom; \]
\[ \text{Coeff11} = \text{simple}(\text{jacobian}(TJvuxp, svecct)); \]
\[ \text{Coeff12} = \text{simple}(\text{jacobian}(TJvuxp, \text{sec}')); \]
\[ \text{Coeff21} = \text{simple}(\text{jacobian}(TJveop, svecct)); \]
\[ \text{Coeff22} = \text{simple}(\text{jacobian}(TJveop, \text{sec}')); \]
\[ \text{CoeffCa} = 0.5 \times (TJvuxp - TJvuxm); \]
\[ \text{CoeffCb} = 0.5 \times (TJveop - TJveom); \]
\[ \text{THETAv} = \text{eval(Coeff11)} \quad \text{eval(Coeff12)} \quad \text{eval(Coeff21)} \quad \text{eval(Coeff22)}]; \]
\[ \text{LAMDAv} = -\text{eval(CoeffCa)} \quad \text{eval(CoeffCb)}]; \]
\[ \%-----------------------------------------------Minimization of coefficient--------------------------\]
\[ \%-----------------------------------------------variation not required.-------------------------------------\]
\[ \text{LAMDAv} = \text{zeros(np+lt,1);} \]
Appendix C

THETA = zeros(np + lt, np + lt);
end
THETA11 = omega1 * Inp;
THETA12 = omega2 * Unpxlt;
THETA21 = omega2 * Ultxn;
THETA22 = omega2 * np + lt;
THETA = [THETA11 THETA12; THETA21 THETA22];

LAMDA = -
[(omega1 * LB1A1' + omega2 * LB2A2') * Ultx1; (omega1 * LB1A1 + omega2 * LB2A2) * Unpx1];

THETA11 = 2 * ksi * lt * (Unpxnp + np + np);
THETA12 = 2 * ksi * np + Unpxlt;
THETA21 = 2 * ksi * np + Ultxn;
THETA22 = 2 * ksi * np + lt;

THETAQ = [THETAQ11 THETAQ12; THETAQ21 THETAQ22];

LAMDAQ11 = (omega1 * LB1A1' + omega2 * LB2A2') * Ultx1;
LAMDAQ12 = (omega1 * LB1A1 + omega2 * LB2A2) * Unpx1;

THETAQ11 = 2 * ksi * lt * (Unpxnp + np + np);
THETAQ12 = 2 * ksi * np + Unpxlt;
THETAQ21 = 2 * ksi * np + Ultxn;
THETAQ22 = 2 * ksi * np + lt;

THETAQ = [THETAQ11 THETAQ12; THETAQ21 THETAQ22];

LAMDAQa = [LAMDAQ11 LAMDAQa2] * vec(LD1t + ksi2 * vec(LD2t));
LAMDAQb = [LAMDAQ11 LAMDAQb2] * vec(ksi1 * LE1C1' + ksi2 * LE2C2');

LAMDAQa = kron(Ulx1, LAMDAQa1) * vec(ksi1 * LE1C1' + ksi2 * LE2C2');
LAMDAQb = kron(eye(lt, lt), np * Ultxnp) * vec(LD1t + ksi2 * LE2C2');

LAMDAQa = [LAMDAQa1 LAMDAQa2];
LAMDAQb = [LAMDAQb1 LAMDAQb2];

THETAC = [zeros(np, np + lt); zeros(lt, np) * vec(LD1t)];
LAMDAC = [zeros(np, 1): vec(LD1t)];

LAMDA = LAMDA + LAMDAQ + LAMDAC;

THETA = THETA + LAMDA;

if (flagopt = 0)
    uxoe = pinv(THETA) * LAMDA;
    ux = invecc(uxoe(1:np), n, p);
    eo = invecc(uxoe(np + 1:np + lt), t, 1);
    %-----------------------------------------------Symmetry if feasible required.------------------------
    if (n = p) & (p = l) & (l = t) & (flagsym = 0)
        for i = 1:n
            for j = 1:i
                ux(j, i) = ux(i, j);
            end
        end
        for i = 1:t
            for j = 1:i
                eo(j, i) = eo(i, j);
            end
        end
    end
    eo = vecc(eo);
else
    %-----------------------------------------------A sub-optimal solution required.------------------------
    vth = [Ulxnp * THETA(1:np, 1:np) * Unpxl
           Ulxnp * THETA(1:np, np + 1:np + lt) * Ultxl;
           Ulxlt * THETA(np + 1:np + lt, 1:np)
           * Unpxl
           Ulxlt * THETA(np + 1:np + lt, np + 1:np + lt) * Ultx1];
    vve = [Ulxnp * LAMDA(1:np); Ulxlt * LAMDA(np + 1:np + lt)];
    vuxeoe = pinv(vth) * vve;
    uxbar = vuxeoe(1, 1);
    eobar = vuxeoe(2, 1);
    ux = uxbar * Unpx;
    eo = eobar * Ultx1;
    A1hat = (10^uxbar) * A1;
    B1hat = (10^eobar) * B1;
A2hat=(10^uxbar)*A2;
B2hat=(10^eobar)*B2;
C1hat=C1;
D1hat=(10^2*(uxbar+eobar))*D1;
E1hat=E1;
C2hat=C2;
D2hat=(10^2*(uxbar+eobar))*D2;
E2hat=E2;
Ghat=(10^eobar)*G;
end
SLuxeo1=U1tx1*vecc(ux)'+eo*U1xnp+LB1A1;
SLuxeo2=U1tx1*vecc(ux)'+eo*U1xnp+LB2A2;
SQuxeo1=U1tx1*vecc(kronsum(ux',ux)+Unpx1vecLDL1)'+eo*U1xnp2+KrU
1xnpLE1C1;
SQuxeo2=U1tx1*vecc(kronsum(ux',ux)+Unpx1vecLDL2)'+eo*U1xnp2+KrU
1xnpLE2C2;
SCeo=eo+vccLG;
Abar=SignB1A1.*(10.^SLuxeo1)+SignB2A2.*(10.^SLuxeo2);
Bbar=SignE1D1C1.*(10.^SQuxeo1)+SignE2D2C2.*(10.^SQuxeo2);
Cbar=SignG.*(10.^SCeo);
%----------------------------------------Compute the minimum obtained.--------------------------
Smv=0;
So=[omegal*SLuxeo1 omega2*SLuxeo2 ksi1*SQuxeo1 ksi2*SQuxeo2
cc*SCeo];
if (flagv==0)
ks=0;
for i=1:lt
 for j=1:2*(np+np2)+1
 if (So(i,j)==0)
  ks=ks+1;
  Sov(1,ks)=So(i,j);
 end
 end
for i=1:ks-1
 for j=i+1:ks
  Smv=Smv+(Sov(1,i)-Sov(1,j))^2;
 end
 end
if (np2>100)
 Sm=normest(So,'fro')^2+Smv;
else
 Sm=norm(So,'fro')^2+Smv;
end
end
if (flagscale==0)
 if (S<Sm)
  disp('No minimum obtained. ');
 end
end
tCPUscale=cputime-tCPUscale;
nFLOPSscale=flops-nFLOPSscale0;
%----------------------------------------END OF THE PROGRAM----------------------------------------
%----------------------------------------VASSILIOS A. TSACHOURIDIS----------------------------------------
%----------------------------------------06-05-2000----------------------------------------
Appendix C

The GQME-Toolbox 431

%All rights reserved.
%Calculation of the solution at epsilon=1 (F(X)=0).
%See GQMEHOM1.m

%For more details see PhD Thesis:
%Vassilios A. Tsachouridis.

"A Unified Framework for the Numerical Solution &
Analysis of Generalized Algebraic Quadratic Matrix
Equations with Engineering and Scientific Applications:
Theory & Software Design and Implementation"

Department of Engineering,
University of Leicester, 2002

---------------------------------------------------------------------------------
function [Xs,JXs]=FQGMEHOM1(X);
load('gqmesetresg', 'Ak', 'tD1', 'D1', 'tD2', 'D2', 'veccG',
'Ip', 'n', 'p', 'np');
X=X(:,1:p)+i*X(:,p+1:2*p);
veccX=vecc(X);
XD1=X*D1;
XD2=X*D2;
D1X=D1*X;
D1Xt=real(D1X)+i*imag(D1X);
D2X=D2*X;
D2Xt=real(D2X)+i*imag(D2X);
fvectX=Ak*veccX+(tD1*kron(Ip,XD1)+tD2*kron(Ip,XD2))
*veccX+veccG ;
Xs=[real(fveccX) imag(fveccX)];
%-----------------------------------------------Jacobian JF(X)------------------------------
if nargout>1
dfveccXdveccX=Ak+tD1*(kron(Ip,XD1)+kron(D1Xt,Ip))
+tD2*(kron(Ip
,XD2)+kron(D2Xt,Ip)));
JXs=[[real(dfveccXdveccX);zeros(np,np)]
[zeros(np,np);imag(dfveccXdveccX)]];
end
%-----------------------------------------------END OF THE PROGRAM------------------------
%-----------------------------------------------VASSILIOS A. TSACHOURIDIS-------------------
%-----------------------------------------------31-03-2000--------------------------------
function \([Xs, JXs] = \text{FGQMEODE}(X)\);
load('dataodef', 'epsilon', 'flh');
'D2', 'E2', 'G', 'ux10');
load('gqmesotestg', 'ux', 'vecc10ux', 'eo10',
eo10Unpx1UXmx1vecc10uxt', 'Abark', 'tD1', 'tD2', 'tG', 'Aa',
anplussl', 'k', 'gam', 'veccXo', 'veccYo', 'Inn', 'Ip', 'Inp',
n', 'p', 'np', 'flagproj', 'fplseudos', 'signRd', 'flchom',
'flcarith');

\[ X = X(:,1) + i \cdot X(:,2) \]
if flagproj==0
\[ L_X = Aa \cdot \text{vecc}(X) + \text{anplussl} \]
else
\[ \text{LX} = Aa \cdot \text{vecc}(X) + \text{anplussl} \]
end

dYoXLDveccX=diag(veccYo) * (Inp * LX + X * Aa);
dXLDveccX=(Inp * LX + X * Aa);
dtGLX2dveccX=2 * tG * LX + Aa;
dXcLX2dveccX=2 * veccXo * LX + Aa;
vecc10ux=X.*vecc10ux;
Xux=inversec(Xvecc10ux,n,p);
tXuxD1=Xux*D1;
tXuxD2=Xux*D2;
tD1Xux=D1*Xux;

tD1Xuxt=real(tD1Xux)'+i*imag(tD1Xux)';
tD2Xux=D2*Xux;

tD2Xuxt=real(tD2Xuxt)'+i*imag(tD2Xuxt)';

fveccXvecc10uxx=real10.*vecc(A1*(Xux* LX) * B1+ A2* (Xux* LX) * B2+ C1*Xux*
D1*Xux*E1+C2*Xux*D2*Xux*E2+G*(LX^2));
if (flh==0)

\[ \text{veccH} = \epsilon \cdot \text{fveccXvecc10uxx-gam*(1-}\epsilon) \cdot (\text{veccYo} \cdot X \cdot LX - \text{veccXo} \cdot (LX^2)) \]
else
\[ \text{veccH} = \epsilon \cdot \text{fveccXvecc10uxx-gam*(1-}\epsilon) \cdot (\text{veccYo} \cdot X \cdot X - \text{veccXo} \cdot (LX^2)) \]
end
\[ Xs = [\text{real(veccH)} \quad \text{imag(veccH)}] \]
%------------------------------------------------------------------------------
\( \text{Jacobian } JH(X) \) ---
if nargout>1
dfveccXvecc10uxdveccX=A_bark*dXLXdveccX+e_010Unpx1XU1mx1vecc10uxt
.*(tD1*(kron(Ip,tXuxD1)+kron(tD1Xuxt,Inn))+tD2*(kron(Ip,tXuxD2)
+kron(tD2Xuxt,Inn))))+dtGLX2dveccX;
if (flh==0)
  dhdveccX=epsilon*dfveccXvecc10uxdveccX-gam*(1-
  epsilon)*(dyoXLXdveccX-dXoLX2dveccX);
else
  dhdveccX=epsilon*dfveccXvecc10uxdveccX-gam*(1-
  epsilon)*(diag(2*veccYo.*X)-dXoLX2dveccX);
end
JXs=[[real(dhdveccX);zeros(np,np)]
[zeros(np,np);imag(dhdveccX)]];
end
clear dataodef;
clear storepg;
clear gqmoestoresq;
%------------------END OF THE PROGRAM---------------------
%----------------------------------------VASSILIOS A. TSACHOURIDIS-------------------------------
%--------------------------------------------------------------------------31-03-2000---------------------------------
Appendix C

The GQME-Toolbox 434

% All rights reserved.
% Calculation of the vecr(A) of a matrix A, vrA.
% Give the command:
% [vrA]=vecr(A);

function [vrA]=vecr(A)

[n,m]=size(A);
vrA=zeros(n*m,1);
k=0;
for i=1:n;
    for j=1:m;
        k=k+j;
        vrA(k,1)=A(i,j);
    end
end
end
%-------------------END OF THE PROGRAM-------------------
%-------------------VASSILIOS A. TSACHOURIDIS-----------------
%-------------------31-03-2000------------------------------
% All rights reserved.
% Calculation of the vec(A) of a matrix A, vrA.
% For more details see PhD Thesis:
% Vassilios A. Tsachouridis.
% "A Unified Framework for the Numerical Solution &
% Analysis of Generalized Algebraic Quadratic Matrix
% Equations with Engineering and Scientific Applications:
% Theory & Software Design and Implementation"
% Department of Engineering,
% University of Leicester, 2002
% Give the command:
% [vcA]=vec(A);
function [vcA]=vec(A)
[vcA]=vec(real(A)')+i*vec(imag(A)');
%------------------------------------------------END OF THE PROGRAM-------------------
%--------------------------------------------VASSILIOS A. TSACHOURIDIS---------------------------------
%----------------------------------------------31-03-2000---------------------------------------
%All rights reserved.
%Calculation of matrix nxm A from its vecr(A), vrA.
%Give the command:
% [A]=invecr(vrA,n,m);
function [A]=invecr(vrA,n,m)
[vn,nm]=size(vrA);
if ~(n*m==vn)
    disp('The rows of vec(A) must be equal with the product n*m.');
    break
end
A=zeros(n,m);
k=0;
for i=1:n;
    for j=1:m;
        k=k+1;
        A(i,j)=vrA(k,1);
    end
end
%-----------------------------------------------END OF THE PROGRAM-----------------------------------------------
%-----------------------------------------------VASSILIOS A. TSACHOURIDIS-----------------------------------------------
%-----------------------------------------------31-03-2000-----------------------------------------------
%All rights reserved.
%Calculation of matrix nxm A from its vec(A), vcA.

%For more details see PhD Thesis:
%Vassilios A. Tsachouridis.

"A Unified Framework for the Numerical Solution & Analysis of Generalized Algebraic Quadratic Matrix Equations with Engineering and Scientific Applications:
Theory & Software Design and Implementation"

Department of Engineering,
University of Leicester, 2002

%Give the command:
%A = invvec(vcA,n,m);

function [A] = invvec(vcA,n,m)
[A] = invvecr(real(vcA),m,n)'+i1*invvecr(imag(vcA),m,n)';

%-----------------------------END OF THE PROGRAM-----------------------------
%---------------------------VASSILIOS A. TSACHOURIDIS-----------------------
%-----------------------------31-03-2000--------------------------------------
%All rights reserved.
%Calculation of the Kronecker Sum (A) of two matrices A1,A2.
%For more details see PhD Thesis:
% Vassilios A. Tsachouridis.
% "A Unified Framework for the Numerical Solution &
% Analysis of Generalized Algebraic Quadratic Matrix
% Equations with Engineering and Scientific Applications:
% Theory & Software Design and Implementation"
% Department of Engineering,
% University of Leicester, 2002

%Give the command:
[A]=kronsum(A1,A2);
function [A]=kronsum(A1,A2);
[n1,m1]=size(A1);
[n2,m2]=size(A2);
A=zeros(n1*n2,m1*m2);
i=0:n2:n1*n2;
j=0:m2:m1*m2;
[a11,a1]=size(i);
[a22,a2]=size(j);
r=1;r1=1;
c=1;c1=1;
while (r<=a1-1) & (r1<=n1)
    while (c<=a2-1) & (c1<=m1)
        k=i(1,r);k1=i(1,r+1);l=j(1,c);l1=j(1,c+1);
        A(k+1:k1,l+1:l1)=A1(r1,c1)+A2;
        c=c+1;c1=c1+1;
    end
    r=r+1;r1=r1+1;
    c=1;c1=1;
end
%--------------------END OF THE PROGRAM-------
%--------------------VASSILIOS A. TSACHOURIDIS-----------------------------------------------31-03-2000--------------------------------------
%All rights reserved.
%Calculation of the symbolic svecr(A) of a matrix A, svrA.
%Give the command:
% [svrA]=svecr(A);
function [svrA]=svecr(A)
A=A;
[n,m]=size(A);
svrA=A(1,1)*zeros(n*m,1);
k=0;
for i=1:n;
 for j=1:m;
 k=k+1;
 svrA(k,1)=A(i,j);
 end
end
%-----------------------------------------------END OF THE PROGRAM-----------------------------------------------
%-----------------------------------------------VASSILOS A. TSACHOURIDIS-----------------------------------------------
%-----------------------------------------------31-03-2000-----------------------------------------------
% All rights reserved.
% Calculation of the symbolic svecc(A) of a matrix A, svcA.
% Give the command:
% [svcA]=svecc(A);

function [svcA]=svecc(A)
[svcA]=svecr(real(A)')+i*svecr(imag(A)');

%--------------------------------END OF THE PROGRAM--------------------------------------
%--------------------------------VASSILIOS A. TSACHOURIDIS-----------------------------------
%--------------------------------31-03-2000----------------------------------------------------------
% All rights reserved.
% Calculation of symbolic matrix nxm A from its symbolic
% svecr(A), svrA.
% Give the command:
% [A]=insvecr(svrA,n,m);
function [A]=insvecr(svrA,n,m)
[vn,nm]=size(svrA);
if ~ (n*m==vn)
disp('The rows of vec(A) must be equal with the product n*m.');
break
end
k=0;
for i=1:n;
for j=1:m;
k=k+1;
A(i,j)=svrA(k,1);
end
end
%-----------------------------END OF THE PROGRAM-------------------------------
%-----------------------------VASSILIOS A. TSACHOURIDIS-----------------------------
%-----------------------------31-03-2000---------------------------------------------
% All rights reserved.
% Calculation of symbolic matrix nxm A from its svecc(A), svcA.
% Give the command:
% [A]=insvecc(svcA,n,m);

function [A]=insvecc(svcA,n,m)
[A]=insvecr(real(vcA),m,n)'+1i*insvecr(imag(vcA),m,n)';

%------------------------------------------------END OF THE PROGRAM----------------------------------------
%------------------------------------------------VASSILIOS A. TSACHOURIDIS-------------------------------------
%------------------------------------------------31-03-2000------------------------------------------------
function [A] = skronsum(sA1, sA2);

% Calculation of the symbolic Kronecker Sum (A) of two symbolic matrices sA1, sA2.
% Give the command:
% [A] = skronsum(sA1, sA2);

A1 = sA1;
A2 = sA2;

[n1, m1] = size(A1);
[n2, m2] = size(A2);

A = A1(1, 1) * zeros(n1*n2, m1*m2);
i = 0:n2:n1*n2;
j = 0:m2:m1*m2;

[A1, a1] = size(i);
[a22, a2] = size(j);

r = 1; r1 = 1;
c = 1; c1 = 1;
while (r <= a1 - 1) & (r1 <= n1)
while (c <= a2 - 1) & (c1 <= m1)
    k = i(1, r); k1 = i(1, r + 1);
    l = j(1, c); l1 = j(1, c + 1);
    A(k + 1: k1, l + 1: l1) = A1(r1, c1) + A2;
    c = c + 1; c1 = c1 + 1;
end
r = r + 1; r1 = r1 + 1;
c = 1; c1 = 1;
end

% --------------------------- END OF THE PROGRAM ---------------------------
% --------------------------- VASSILIOS A. TSACHOURIDIS ---------------------
% --------------------------- 31-03-2000 -----------------------------------
Appendix C

M-File by
V. A. TSACHOURIDIS
17-06-2001
All rights reserved.

%Perform Numerical Analysis to the
%General Quadratic Matrix Equation (GQME):

% A1XB1+A2XB2+C1XD1XE1+C2XD2XE2+G=0.

% Where A1,A2,C1,C2: nxn, B1,B2,E1,E2:pxp, D1,D2:pxn, G:nxp,
% X:nxp, complex matrices.

%Give the command:

[CnX,Eprfmax,Eprfmaxf1,Erb,Erbmax,Erbmaxf1,NS,NSf1]=
NAQGME(X,A1,aA1,A2,aA2,B1,aB1,B2,aB2,C1,aC1,C2,aC2,D1,
 aD1,D2,aD2,E1,aE1,E2,aE2,G,aG);

%OUTPUTS:

- CnX: the condition number.
- Eprfmax: the upper bound of the practical relative
  forward error of the solution.
- Eprfmaxf1: the upper bound of the practical relative
  forward error of the solution, including rounding errors.
- Erb: the actual relative backward error.
- Erbmax: the upper bound of the relative backward error
  of the solution.
- Erbmaxf1: the upper bound of the relative backward error
  of the solution, including rounding errors.
- NS: index for the numerical stability.
  NS=1: numerically stable. NS=0: numerically unstable.
- NSf1: index for the numerical stability
  including rounding errors.
  NSf1=1: numerically stable. NS=0: numerically unstable.

%INPUTS:

- X,A1,aA1,A2,aA2,B1,aB1,B2,aB2,C1,aC1,C2,aC2,
  D1,aD1,D2,aD2,E1,aE1,E2,aE2,G,aG: Equations' solution,
  data matrices and their respective perturbations indexes

For more details see PhD Thesis:

Vassilios A. Tsachouridis.

Department of Engineering,
University of Leicester, 2002

function
[CnX, Eprfmax, Eprfmaxf1, Erb, Erbmax, Erbmaxf1, NS, NSf1] = NAGQME(X, A1, aA1, aA2, aB1, aB2, aB2, C1, aC1, C2, aC2, D1, aD1, D2, aD2, E1, aE1, E2, aE2, G, aG);
% Check the dimension of the input matrices.

[nA1, mA1] = size(A1);
[nA2, mA2] = size(A2);
[nB1, mB1] = size(B1);
[nB2, mB2] = size(B2);
[nC1, mC1] = size(C1);
[nC2, mC2] = size(C2);
[nD1, mD1] = size(D1);
[nD2, mD2] = size(D2);
[nE1, mE1] = size(E1);
[nE2, mE2] = size(E2);
[nG, mG] = size(G);
[n, p] = size(X);
[nA1, mA1] = size(aA1);
[nA2, mA2] = size(aA2);
[nB1, mB1] = size(aB1);
[nB2, mB2] = size(aB2);
[nC1, mC1] = size(aC1);
[nC2, mC2] = size(aC2);
[nD1, mD1] = size(aD1);
[nD2, mD2] = size(aD2);
[nE1, mE1] = size(aE1);
[nE2, mE2] = size(aE2);
[nG, mG] = size(aG);
if (nA1*mA1~1) | (nA2*mA2~1) | (nB1*mB1~1) | (nB2*mB2~1) | (nC1*mC1~1) | (nC2*mC2~1) | (nD1*mD1~1) | (nD2*mD2~1) | (nE1*mE1~1) | (nE2*mE2~1) | (nG*mG~1)
disp('epsilon, aepsilon, gama, aagama, aA1, aA2, aB1, aB2, aC1, aC2, aD1, aD2, aE1, aE2, aG, aE2ux, aE10eo, aAa, anplus1, aanplus1, aYo, aXo and flh, must be scalars.');
break
end
if (nA1~nA2) | (nA2~nC1) | (nC1~nC2) | (mA1~mA2) | (mA2~mC1) | (mC1~mC2) disp('A1, A2, C1, and C2, must have the same dimensions nxn.');
break
end
if (nB1~nB2) | (nB2~nE1) | (nE1~nE2) | (mB1~mB2) | (mB2~mE1) | (mE1~mE2) disp('B1, B2, E1, and E2, must have the same dimensions pxp.');
break
end
if (nD1~nD2) | (mD1~mD2)
disp('D1 and D2, must have the same dimensions pxn.'); break
if (nA1~nG) | (mB1~mG)
    disp('G must have the same number of rows with A1, A2, C1, C2 and the same number of columns with B1, B2, E1, E2; dimension nxp.'); break
end
if (n~nG) | (p~mG)
    disp('G and X, must have the same dimensions nxp.'); break
end
%------------------- Perform the Computations. ----------------
%For standard Double IEEE arithmetic ur=2^-53. If different
%ur should changed appropriately.
ur=2^-53;
c=4;
gcn=c*n*ur/(1-n*ur);
gcp=c*p*ur/(1-p*ur);
np=n*p;
In=eye(n,n);
Ip=eye(p,p);
Inp=eye(np,np);
Unxn=ones(n,n);
Upxp=ones(p,p);
Unxn=ones(n,p);
Upxn=ones(p,n);
Unpxl=ones(np,1);

Xt=real(X) +li*imag(X);
tveccX=real(vec(X)) +li*imag(vec(X));
A1t=real(A1)';
A2t=real(A2)';
B1t=real(B1)';
B2t=real(B2)';
C1t=real(C1)';
C2t=real(C2)';
D1t=real(D1)';
D2t=real(D2)';
E1t=real(E1)';
E2t=real(E2)';
aA=kron([aA1 aA2],Unxn);
aB=kron([aB1 aB2],Upxp);
aC=kron([aC1 aC2],Unxn);
aD=kron([aD1 aD2],Upxn);
aE=kron([aE1 aE2],Upxp);
KdA1=kron(B1t*Xt,In);
KdA2=kron(B2t*Xt,In);
KdA=[KdA1 KdA2];
KdB1=kron(Ip,A1*X);
KdB2=kron(Ip,A2*X);
KdB=[KdB1 KdB2];
KdC1=kron(E1t*Xt*D1t*Xt,In);
KdC2=kron(E2t*Xt*D2t*Xt,In);
KdC=[KdC1 KdC2];
KdD1=kron(E1t,C1)*kron(Xt,X);
KdD2=kron(E2t,C2)*kron(Xt,X);
KdD=[KdD1 KdD2];
KdE1=kron(Ip,C1*X*D1*X);
KdE2=kron(Ip,C2*X*D2*X);
KdE=[KdE1 KdE2];
Appendix C

The GQM E-Toolbox

**KdG=**\(\text{Inp} \);  
\(\text{KdX} = \text{kron}(\text{B1t} \cdot \text{A1} + \text{kron}(\text{B2t} \cdot \text{A2}) + \text{kron}((\text{E1t} \cdot \text{C1}) \cdot \text{kron}(\text{Xt} \cdot \text{D1} \cdot \text{In}) + \text{kron}(\text{B1t} \cdot \text{C1}) \cdot \text{kron}(\text{Ip} \cdot \text{X} \cdot \text{D1}) + \text{kron}(\text{E2t} \cdot \text{C2}) \cdot \text{kron}(\text{Xt} \cdot \text{D2t} \cdot \text{In}) + \text{kron}(\text{E2t} \cdot \text{C2}) \cdot \text{kron}(\text{Ip} \cdot \text{X} \cdot \text{D2}); \)

\(\text{HX} = \text{A1} \cdot \text{X} \cdot \text{B1} + \text{A2} \cdot \text{X} \cdot \text{B2} + \text{C1} \cdot \text{X} \cdot \text{D1} \cdot \text{E1} + \text{C2} \cdot \text{X} \cdot \text{D2} \cdot \text{X} \cdot \text{E2} + \text{G}; \)

\(\text{flHmax1} = \text{gcn} \cdot \text{normest}(\text{A1}, \text{fro'}) + \text{normest}(\text{X}, \text{fro'}) + \text{normest}(\text{B1}, \text{fro'}); \)

\(\text{flHmax2} = \text{gcn} \cdot \text{normest}(\text{A2}, \text{fro'}) + \text{normest}(\text{X}, \text{fro'}) + \text{normest}(\text{B2}, \text{fro'}); \)

\(\text{flHmax} = \text{flHmax1} + \text{flHmax2} + \text{ur} \cdot \text{normest}(\text{G}, \text{fro'}); \)

\(\text{KX} = \\text{kron}(\text{Upxn}, \text{aA}) \cdot \text{KdA} \cdot \text{kron}(\text{Unxp}, \text{aB}) \cdot \text{KdB} \cdot \text{kron}(\text{Upxn}, \text{aC}) \cdot \text{KdC} \cdot \text{kron}(\text{Unxp}, \text{aD}) \cdot \text{KdD} \cdot \text{kron}(\text{Unxp}, \text{aE}) \cdot \text{KdE} \cdot \text{aG} \cdot \text{KdG}; \)

\(\text{CnX} = \text{normest}(\text{inv} \cdot (\text{KdX} \cdot \text{KX}), \text{fro'}) / \text{normest}(\text{X}, \text{fro'}); \)

\(\text{Er} = \text{normest}(\text{pinv}(\text{KX}) \cdot \text{vecc}(\text{HX}), \text{fro'}); \)

\(\text{Erbmaxl} = \text{normest}(\text{pinv}(\text{KX}), \text{fro'}) \cdot \text{normest}(\text{vecc}(\text{HX}), \text{fro'}); \)

\(\text{Erbmax} = \text{normest}(\text{pinv}(\text{KX}), \text{fro'}) \cdot \text{normest}(\text{vecc}(\text{HX}), \text{fro'}); \)

\(\text{Eprfmax} = \text{normest}(\text{HX}, \text{fro'}) \text{fro'} + \text{flHmax} \cdot \text{normest}(\text{KdX}, \text{fro'}); \)

\(\text{NS} = (\text{Eprfmax} \leq \text{CnX} \cdot \text{Erbmax}) ; \)

\(\text{NSfl} = (\text{Eprfmax} \leq \text{CnX} \cdot \text{Erbmax}) ; \)

%-------------------END OF THE PROGRAM-------------------------

%------------------VASSILIOS A. TSACHOURIDIS------------------

%------------------------------------17-06-2001------------------------
Driver for NAGQME.m

For more details see PhD Thesis:

Vassilios A. Tsachouridis.

"A Unified Framework for the Numerical Solution &
Analysis of Generalized Algebraic Quadratic Matrix
Equations with Engineering and Scientific Applications:
Theory & Software Design and Implementation"

Department of Engineering,
University of Leicester, 2002

\[
\begin{align*}
\text{aA1} &= \text{norm}(A1, 'fro') ; \\
\text{aA2} &= \text{norm}(A2, 'fro') ; \\
\text{aB1} &= \text{norm}(B1, 'fro') ; \\
\text{aB2} &= \text{norm}(B2, 'fro') ; \\
\text{aC1} &= \text{norm}(C1, 'fro') ; \\
\text{aC2} &= \text{norm}(C2, 'fro') ; \\
\text{aD1} &= \text{norm}(D1, 'fro') ; \\
\text{aD2} &= \text{norm}(D2, 'fro') ; \\
\text{aE1} &= \text{norm}(E1, 'fro') ; \\
\text{aE2} &= \text{norm}(E2, 'fro') ; \\
\text{aG} &= \text{norm}(G, 'fro') ; \\
\end{align*}
\]

\[
\left[ \text{CnX}, \text{Eprfmax}, \text{Eprfmaxf1}, \text{Erb}, \text{Erbmax}, \text{Erbmaxf1}, \text{NS}, \text{NSf1} \right] = \text{NAGQME}(X, \text{Al}, \text{aAl}, \text{aA2}, \text{aB1}, \text{aB2}, \text{aC1}, \text{aC2}, \text{aDl}, \text{aD2}, \text{aE1}, \text{aE2}, \text{aE2}, \text{aG}) ; \\
\text{disp}(' [\text{CnX}, \text{Eprfmax}, \text{Eprfmaxf1}, \text{Erb}, \text{Erbmax}, \text{Erbmaxf1}, \text{NS}, \text{NSf1}] = ') ; \\
\left[ \text{CnX}, \text{Eprfmax}, \text{Eprfmaxf1}, \text{Erb}, \text{Erbmax}, \text{Erbmaxf1}, \text{NS}, \text{NSf1} \right] \text{1}
\]

\%-----------------------------------END OF THE PROGRAM-----------------------------------
\%-----------------------------------VASSILIOS A. TSACHOURIDIS-----------------------------------
\%-----------------------------------17-06-2001-----------------------------------------------
C.3 Modified Versions of Major Functions of GQME-Toolbox

This section provides modified versions of GQMEHOM1.m, INGQMEHOM1.m, RK45GQMEHOM1.m, FGQMEODE.m and FGQMEHOM.m. These are GQMEHOM11.m, INGQMEHOM11.m, RK45GQMEHOM11.m, FGQMEODE11.m and FGQMEHOM11.m respectively.

The new feature of these functions is that the intermediate store of data is avoided and the relevant data are now inputs to the above functions. Doing this saves on computation time from open, read and write data to files.

Note that, the numerical examples of this thesis have been studied with the original routines of section C.2. In order to test the numerical examples with the modified routines in this section, the only change in drivers (see appendix D) is the use of GQMEHOM11.m instead of GQMEHOM1.m.

The software code for the above is given in the subsequent pages.
Appendix C

The GQME-Toolbox

%******************************************************************************************
%******************************************************************************************
%* M-File by
%*     V. A. TSACHOURIDIS
%*     20-12-2001
%* All rights reserved.
%******************************************************************************************
%******************************************************************************************

% Finds using a Homotopic Continuation the solution of the General Quadratic Matrix Equation (GQME):
%  A1XB1+A2XB2+C1XD1XE1+C2XD2XE2+G=0.
% Where A1, A2, C1, C2: nxn, B1, B2, E1, E2: pxp, D1, D2: pxn, G: nxp, X: nxp, complex matrices.
% Give the command:
% [error, X, JacobianX, LX, ux, polyXa, plX, epsilon, plpseudos, nsteps, stepmin, tCPU, nFLOPS, tCPUscale, nFLOPSscale]=GQMEHOMll
% (A1, B1, A2, B2, C1, D1, E1, C2, D2, E2, G, A1o, B1o, A2o, B2o, C1o, D1o, E1o, C2o, D2o, E2o, Go, flagscale, flagv, eps, flagopt, flagsym, sux, se, fproj, Aa, annplus1, gam, pdef, Xo, Yo, SignXoYo, step, e, efinal, flh, flchom, flcarith, fplpseudos, pseudosfin, maxnsteps, minstep, flagcor, largescale, jacobiangqme, linesearch, % displaysolver, display ode, plotdisplay);
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %M-FILES (SUBROUTINES) USED WITH THIS PROGRAM:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% - SGQME.m
% - INQMEHOMll.m
% - RKQMEHOMll.m
% - FGQMEHOMll.m
% - FGQMEODE11.m
% - vecc.m
% - invecc.m
% %To see details of the above m-files and the m-files/functions used by them type, help "filename".m, and to view them open the respective m-files by a tex editor.
%In addition, build in MATLAB (version: 5.3.0.10183 (R11)) functions and m-files from the "Optimization Toolbox" and the "Symbolic Math. Toolbox", are used.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%OUTPUTS:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% - error: the absolute value of the residual error of the solution.
% - X: the solution.
% - JacobianX: the jacobian of the solution veccX(l).
% - LX=AaX+anpplus1
%   LX>1 when a projective transformation take place.
%   LX=1 when a projective transformation does not take
Appendix C The GQME-Toolbox 451

\%
\% place.
\% LX=0 when the solution obtained is of GQME at infinity.
\%
\% - ux, eo: the variable and equation scaling factors respectively.
\% When no scaling used ux and eo are returned as zero arrays.
\% See m-file SGQME.m (type help SGQME.m) for details.
\%
\% - polyXa=[repolyXa; impolyXa]: Matrix containing on each row
\% the coefficients of the polynomial approximation to
\% Homotopy paths of vecc(LX*X(epsilon)./(10.^ux)).
\% The first np rows are with respect to the real part of
\% vecc(LX*X(epsilon)./(10.^ux)), while the last np rows are
\% with respect to the imagine part of
\% vecc(LX*X(epsilon)./(10.^ux)).
\%
\% - plX, plepsilon, plpseudos: The ODE output (points of the
\% Homotopy Paths).
\%
\% - nsteps: Number of integration steps used in the ODE
\% routine RK45GQMEHOM11.m.
\%
\% - stepmin: Minimum integration step used in the integration
\% ODE routine RK45GQMEHOM11.m.
\%
\% - tCPU: The computer CPU time required for tracing the
\% homotopy paths when step=1.
\%
\% - nFLOPS: The number of FLOPS required for tracing the
\% homotopy paths when step=1.
\%
\% - tCPUscale: The computer CPU time required for scaling.
\%
\% - nFLOPSscale: The The number of FLOPS required for
\% scaling.
\%
\%============================================================================

\% INPUTS:
\%============================================================================

\% - A1o,B1o,A2o,B2o,C1o,D1o,E1o,C2o,D2o,E2o,Qo:
\% Initial matrices such that
\% A1oXB1o+A2oXB2o+C1oXD1oXE1o+C2oXD2oXE2o+Go=0.
\%
\% - l>step>0: Fixed step of the discrete Homotopy, from which
\% A1,B1,A2,B2,C1,D1,E1,C2,D2,E2,Q, are obtained gradually
\% from A1o,B1o,A2o,B2o,C1o,D1o,E1o,C2o,D2o,E2o,Qo, as
\% D1=0, step, . . , 1.
\%
\% - Xo,Yo:nxp:Initial conditions.
\% When n=p=m=1 and pdef=+-1: Yo,Xo>=0 if gam>0, Yo>=0,
\% Xo<=0 if
\% gam<0.
\%
\% >=(<=) means, Not-negative (Not-positive) definite if
\% pdef=1.
\% >=(<=) means, negative (positive) definite if pdef=1.
\% If pdef=0: no specific kind of solution required.
\% Also Xo, must have nonzero elements when flh<>0.
\% Yo<=0, always.
\% Xo, Yo, in the general case are choosen as random.
Appendix C

- **SignXoYo**: \( \text{nxp. Determines when } flh<>0, \text{ if } \)
- \( X_{ij}(0)=(X_{oij}/Y_{oij})^{0.5}, \) or \( X_{ij}(0)=-(X_{oij}/Y_{oij})^{0.5}. \)
- Hence \( \text{SignXoYoij=1 for } X_{ij}(0)=(X_{oij}/Y_{oij})^{0.5} \) and
- \( \text{SignXoYoij=-1 for } X_{ij}(0)=-(X_{oij}/Y_{oij})^{0.5}. \)

- **sux, seo, flagscale, flagv, eps, flagopt, flagsym**: See m-file SGQME.m (Type, help SGQME.m).

- **flagproj**: 
  0, if the projective transformation \( (\hat{F},LX,np+1) \) is used for the solution.
  <>0, when the projective transformation \( (\hat{F},LX,np+1) \) is not used for the solution.
  Where \( LX=A_{avec}(X)+anpplusl. \)
  \( Aa=[a1 a2 ... anp] \) and \( anpplus1 \), are random complex numbers in general.
  When flagproj=0 then the solution to the easy problem (ODE initial condition) is automatically fixed by the program as:

\[
\text{vec}(X(0)) = anpplusl(\text{pinv}(\text{Inp}-\text{vecc}(Xo./Yo)*Aa)\text{vecc}(Xo./Yo), \text{ when } flh=0.
\]

\[
\text{vec}(X(0)) = anpplusl(\text{pinv}(\text{Inp}-\text{vecc}(\text{SignXoYo})*\text{vecc}(Xo./Yo).^0.5)*Aa)\text{vecc}(\text{SignXoYo})*\text{vecc}(Xo./Yo).^0.5), \text{ when } flh>0.
\]

(Where ./ denotes element by element division)

Note that in this case symmetry and special structures of \( Xo, Yo \) given by the user may be lost so the user must give a \( Xo \) such that \( \text{vec}(X(0)) \) is of the desired kind.

Thesis and m-file INGQMEHOM11.m (Type, help INGQMEHOM11.m).

- **flh** (See m-file INGQMEHOM11.m (Type, help INGQMEHOM11.m)): The user has the option to select the \( dX(\epsilon)/d\epsilon \) computation for two kinds of homotopies:

  1. \( H(X(\epsilon),\epsilon)=\epsilon f(\text{vec}(X(\epsilon)))\) - \( -\gamma(1-\epsilon) (Yo*X(\epsilon)*X(\epsilon)-Xo) \), when \( flh<>0. \)

  2. \( H(X(\epsilon),\epsilon)=\epsilon f(\text{vec}(X(\epsilon)))\) - \( -\gamma(1-\epsilon) (Yo*X(\epsilon)-Xo) \), when \( flh=0. \)

Or when pseudo-arc length parameterization \( s \) is used:

  1. \( H(X(\epsilon(s)),\epsilon(s))=\epsilon s f(\text{vec}(X(\epsilon(s))))\) - \( -\gamma(1-\epsilon(s)) (Yo*X(\epsilon(s))*X(\epsilon(s))-Xo) \), when \( flh<>0. \)

  2. \( H(X(\epsilon(s)),\epsilon(s))=\epsilon s f(\text{vec}(X(\epsilon(s))))\) - \( -\gamma(1-\epsilon(s)) (Yo*X(\epsilon(s))-Xo) \), when \( flh=0. \)

- **flchom=0**: If \( H \) is a complex homotopy. flchom<>0 if \( H \) is a real homotopy.
Appendix C

- flcarith=0: If complex arithmetic is used. flcarith<>0 if real arithmetic is used.

- flpseudos<>0: If pseudoarc-length parameterization is used, and flpseudos=0 otherwise. For the case flpseudos<>0, it is assumed that \( H(x(\epsilon(s),\epsilon(s)) \) is transversal to zero, and more important that \( H^{-1}(0) \) is bounded, with no solutions at infinity. Otherwise the algorithm will no work in general.

- pseudosfin: Estimated pseudo-arc length when such parameterization is used.

- maxnsteps: Maximum number of integration steps allowed in the ODE routine RK45GQMEHOM11.m.

- minstep: Minimum integration step allowed in the integration ODE routine RK45GQMEHOM1.m.

Note: maxnsteps & minstep are used together in combination for truncate very long paths assumed that they divergent to infinity. This is helpful when there are infinite number of solutions at infinity. Hence Projective Transformation may not be able of computing such solutions.

- flagcor: Flag for corrector in each step of the ODE RK45GQMEHOM11.m, (On when 0, Off when <>0).

- e: is the zero tolerance for the path tracking.

- efinal: is the zero tolerance for the solution obtained.

- largescale: It is given as string 'on' or 'off' when the local solver at epsilon uses a large or a medium scale optimization algorithm. For more details see MATLAB m-file FSOLVE.m.

- jacobiangqme: It is given as string 'on' when the local solver at epsilon uses the jacobian explicit formulas, or 'off' otherwise.

- linesearch: Line search type for the local solver at epsilon. It is given as a string 'quadcubic' or 'cubicpoly'. When jacobiangqme='on' then Linesearch='quadcubic' and when jacobiangqme='off' then Linesearch='cubicpoly', are preferable. linesearch is actived only when largescale='off'.

- displaysolver: It is given as string 'iter' if display of results, of the local solver at epsilon, on each iteration required, or 'off' otherwise.

The local solver at epsilon, is a subspace trust region method, based on an interior-reflective Newton method of the FSOLVE.m when largescale='on'. When largescale='off' the local solver at epsilon is the Gauss-Newton solver of the FSOLVE.m. For details type: help fsolve and help optimset and view.
% m-file FGQMEHOM11.m.
% - displayode: It is given as string 'on' if display of results, of the ODE solver on each step required, or 'off' otherwise.
% - plotdisplay: It is given as string 'on' for plotting the 3-D graph (epsilon,realX(epsilon),imagX(epsilon)), or 'off' otherwise.
% For tracking the homotopy paths the RK45GQMEHOM11.m ODE solver is used.
% Computations can take place in complex or real arithmetic in MALAB (version: 5.3.0.10183 (R11)).
% For more details see PhD Thesis:
% Vassilios A. Tsachouridis.
% Department of Engineering,
% University of Leicester, 2002

function [error, X, JacobianX, LX, LX, ux, eo, polyXa, plX, pi epsilon, plpseudos, ns te, stepmin, tCPU, nFLOPS, tCPUscale, nFLOPSscale] = GQMEH0M11(Al, Bl, A2, B2, Cl, Dl, El, Cl, Dl, El, C2, D2, E2, G, Alo, Blo, A2o, B2o, C2o, D2o, E2 o, Go, flagscale, flagv, eps, flagopt, flagsym, sux, seo, flagproj, Aa, applus1, gam, pdef, Xo, Yo, SignXoYo, step, e, efinal, flh, flchom, flcarit h, flpseudos, pseudosfin, maxnsteps, minstep, flagcor, largescale, j actobiangqme, linesearch, displaysolver, displayode, plotdisplay);
% -----------------------------------------------------------
% Check the dimension of the input matrices.
% -----------------------------------------------------------
[nA1, mA1] = size(Al);
[nB1, mB1] = size(B1);
[nA2, mA2] = size(A2);
[nB2, mB2] = size(B2);
[nC1, mC1] = size(C1);
[nD1, mD1] = size(D1);
[nE1, mE1] = size(E1);
[nC2, mC2] = size(C2);
[nD2, mD2] = size(D2);
[nE2, mE2] = size(E2);
[nG, mG] = size(G);
[nAlo, mAlo] = size(Alo);
[nBlo, mBlo] = size(Blo);
[nA2o, mA2o] = size(A2o);
[nB2o, mB2o] = size(B2o);
[nClo, mClo] = size(Clo);
[nDlo, mDlo] = size(Dlo);
[nElo, mElo] = size(Elo);
[nC2o, mC2o] = size(C2o);
[nD2o, mD2o] = size(D2o);
[nE2o, mE2o] = size(E2o);
[nGo, mGo] = size(Go);
[nXo, mXo] = size(Xo);
Appendix C

The GQME-Toolbox 455

```matlab
[nYo,mYo]=size(Yo);
[nSignXoYo,mSignXoYo]=size(SignXoYo);
if
(nA1==nA2)|(nA2==nC1)|(nC1==nC2)|(mA1==mA2)|(mA2==mC1)|(mC1==mC2)
disp('A1, A1o, A2, A2o, C1, Cl0, and C2, C2o must have the same dimensions nxn.');
break
end
if
(nBl==nB2)|(nB2==nEl)|(nEl==nE2)|(mBl==mB2)|(mB2==mEl)|(mEl==mE2)
disp('B1, B1o, B2, B2o, E1, E1o, and E2, E2o must have the same dimensions pxp.');
break
end
if (nDl==nD2)|(mDl==mD2)|(nDlo==nDlo)|(mD2o==mD2o)
disp('D1, D1o and D2, D2o must have the same dimensions pxn.');
break
end
if (nAl==nG)|(mBl==mG)|(nG==nGo)|(mG==mGo)
disp('G, Go must have the same number of rows with A1, A2, C1, C2 and the same number of columns with B1, B2, E1, E2; dimension nxp.');
break
end
if (nXo==nG)|(mXo==mG)|(nYo==nG)|(mYo==mG)
disp('Xo and Yo must have the same dimensions with G; dimension nxp.');
break
end
if
(nSignXoYo==nXo)|(mSignXoYo==mXo)|(nSignXoYo==nYo)|(mSignXoYo==mYo)
disp('SignXoYo must have the same dimensions with Xo, Yo; dimension nxp.');
break
end
for i=1:nSignXoYo
  for j=1:mSignXoYo
    if (SignXoYo(i,j)==1)&&(SignXoYo(i,j)==-1)
      stopsign=1;
    end
  end
end
stoppersign=0;
if (stoppersign==1)
disp('SignXoYo must have elements +1 or -1.');
break
end
if (nA1==mA1)
disp('A1, A1o, A2, A2o, C1, Cl0, C2, C2o and Ao must have the same dimensions nxn.');
break
end
if (nBl==mBl)
disp('B1, B1o, B2, B2o, E1, E1o, E2, E2o and Bo must have the same dimensions pxp.');
break
end
if (step>1)|(step==0)
```

disp('step must be 0<step<=1.');
break
end
if (maxnsteps<=0)
    disp('maxnsteps must be maxnsteps>0. ');
    break
end
if (minstep<0)
    disp('minstep must be minstep>=0 . ');
    break
end
imeqdat l=norm (imag (A1), 'fro') + norm (imag (A2), 'fro') + norm (imag (B1), 'fro') + norm (imag (B2), 'fro') + norm (imag (C1), 'fro') + norm (imag (C2), 'fro');
imeqdat2=norm (imag (D1), 'fro') + norm (imag (D2), 'fro') + norm (imag (E1), 'fro') + norm (imag (E2), 'fro') + norm (imag (G), 'fro');
imeqdat3=norm (imag (Alo), 'fro') + norm (imag (A2o), 'fro') + norm (imag (Blo), 'fro') + norm (imag (B2o), 'fro') + norm (imag (C1o), 'fro') + norm (imag (C2o), 'fro');
imeqdat4=norm (imag (Dlo), 'fro') + norm (imag (D2o), 'fro') + norm (imag (Elo), 'fro') + norm (imag (E2o), 'fro') + norm (imag (Go), 'fro');
imeqdat5=norm (imag (Xo), 'fro') + norm (imag (Yo), 'fro') + norm (imag (Xo), 'fro') + norm (imag (Yo), 'fro') + norm (imag (Anpplusl), 'fro') + norm (imag (SignXoYo), 'fro') + norm (imag (gam), 'fro');
imeqdat=imeqdat1+imeqdat2+imeqdat3+imeqdat4+imeqdat5;
if (imeqdat~0)&(flchom~0)
disp('H is a complex homotopy. Give flchom=0. ');
break
end
if (sum(largescale)~0) & (sum(1 'off'))
disp(' largescale must be the string 'on' or 'off ' ' ');
    break
end
if (sum(jacobiangqme)~0) & (sum('off '))
disp(' jacobiangqme must be the string 'on' or 'off ' ' ');
    break
end
if (sum(linesearch)~0) & (sum('quadcubic') & (sum('cubicpoly'))
disp('linesearch must be the string 'quadcubic' or 'cubicpoly' '');
    break
end
if (sum(displaysolver)~0) & (sum('iter'))
disp('displaysolver must be the string 'iter' or 'off' ' ' ');
    break
end
if (sum(displayode)~0) & (sum('on'))
disp('displayode must be the string 'on' or 'off' ' ' ');
    break
end
if (sum(plotdisplay)~0) & (sum('on'))
disp('plotdisplay must be the string 'on' or 'off' ' ' ');
    break
end
m=nA1;
n=mAl;
p=nBl;
l=mBl;
np=n*p;
lm=l*m;
Im=eye(m,m);
In=eye(n,n);
Inn=In;
Ip=eye(p,p);
Inp=eye(np,np);
I2np=eye(np,np);

% ---------------------------Set initial value.---------------------------
if (flagproj==0)
  if (anpplusl==0)
    disp('anpplusl must be nonzero.');
    break
  end
  if (flh==0)
    X=invecc(anpplusl*pinv(Inp-
    vecc(Xo./Yo).*Aa)*vecc(Xo./Yo),n,p);
  else
    if (nnz(Xo)<n*p)
      disp('For homotopy map 1. Xo must have nonzero elements.');
      break
    end
    if (nnz(Yo)<n*p)  
      disp('For homotopy map 1. Yo must have nonzero elements. 
    break
    end
    X=invecc(anpplusl*pinv(Inp-
    (vecc(SignXoYo).*vecc(Xo./Yo).^0.5)*Aa)*vecc(Xo./Yo).^0.5),n,p);
  end
else
  if (flh==0)
    X=Xo./Yo;
  else
    if (nnz(Xo)<n*p)
      disp('For homotopy map 1. Xo must have nonzero elements.');
      break
    end
    if (nnz(Yo)<n*p)
      disp('For homotopy map 1. Yo must have nonzero elements. 
    break
    end
    X=SignXoYo.*(Xo./Yo).^0.5);
end
end
if (n==p) & (m==l)
eX=eig(X);
for h=1:n
  if (real(eX(h))<0) & (gam>0) & (pdef==1)
    disp('Xo and gam must be consisted.');
    break
  end
  if (real(eX(h))>0) & (gam<0) & (pdef==1)
    disp('Xo and gam must be consisted.');
  end
end
break
end
end
end
Alf=A1;
Blf=Bl;
A2f=A2;
B2f=B2;
C1f=C1;
D1f=D1;
Elf=El;
C2f=C2;
D2f=D2;
E2f=E2;
Gf=G;

%--------------------Check if initial value is a solution.-----
error=abs (Alf *X*Blf+A2f*X*B2f+C1f*X*D1f*X*Elf+C2f*X*D2f*X*E2f+Gf);
if (norm(error) <= e)
    polyXa=0;
else
%--------------------Outer-Loop: Discrete Homotopy -----------------
    if (step == 1)
        D1=1;
        flagend=1;
    else
        D1=0;
        flagend=0;
    end
while (D1 <= 1) & (flagend <= 1)
    fprintf(2,' [Discrete homotopy for data step]=%e\n',D1);
    Al=Al0+D1*(Alf-Al0);
    Bl=Bl0+D1*(Blf-Bl0);
    A2=A20+D1*(A2f-A20);
    B2=B20+D1*(B2f-B20);
    C1=C10+D1*(C1f-C10);
    D1=D10+D1*(D1f-D10);
    El=El0+D1*(Elf-El0);
    C2=C20+D1*(C2f-C20);
    D2=D20+D1*(D2f-D20);
    E2=E20+D1*(E2f-E20);
    G=Go+D1*(Gf-Go);
    \n    %--------------------Scale equation.-----------------------------
    \[Abar, Bbar, Cbar, ux, eo, S, Sm, Alhat, Blhat, A2hat, B2hat, Clhat, D1hat, Elhat, C2hat, D2hat, E2hat, Ghat, tCPUscale, nFLOPSscale\] = SGQME(Al,Bl,A2,B2,Cl,D1,El,C2,D2,E2,G,flagscale,flagv,eps,flagopt,flagsym,sux,seo);
    ux10=10.*ux;
    vecccux=vecc(ux);
    veccc10ux=10.*(vecccux);
    U1mx1=ones(lm,1);
    U1mx1vecc10uxt=U1mx1*vecc10ux';
    diagvecc10ux=diag(vecc10ux);
    eo10=10.*eo;
    U1xnp=ones(1,np);
    eo10Unpx1=eo10*U1xnp;
    eo10Unpx1U1mx1vecc10uxt=eo10Unpx1.*U1mx1vecc10uxt;
    \n    %------------------------Evaluate constant terms of-----------
    %-----------------------------H(X,epsilon(s)) to save them--------
    %-----------------------------for future use with INGQMEHOM1.m---------
Appendix C

The GQM E-Toolbox 459

\[ A_k = \text{kron}(\text{real}(B_1) + \text{i} \times \text{imag}(B_1)) + \text{kron}(\text{real}(B_2) + \text{i} \times \text{imag}(B_2)), A_2) \]
\[ A_{1k} = \text{kron}(\text{real}(E_1) + \text{i} \times \text{imag}(E_1)) + C_1 \]
\[ A_{2k} = \text{kron}(\text{real}(E_2) + \text{i} \times \text{imag}(E_2)), C_2) \]
\[ \text{veccG} = \text{vecc}(G); \]
\[ \text{veccY} = \text{vecc}(Y); \]
\[ \text{veccX} = \text{vecc}(X); \]
\[ t_0 = \text{eo10} \ast \text{veccG}; \]

%-------------------------------Set counter k for the file INGQMEHOM1.m.---
%-------------------------------Split initial value to vector---------------------
%-------------------------------vecc (w,r,t columns).-------------------------

X0 = \text{vecc}(X);

if (flpseudos == 0)
    X0 = [X0; 0];
endif

\[ \text{epsilon}, X, nsteps, stepmin] = \text{RK45GQMEHOM11}(0, t\text{fin}, X0, e, n, p, n, p, \text{gam}, pdef, phi, flpseudos, maxnsteps, minstep, flagcor, displayode, A1, B1, A2, B2, C1, D1, E1, C2, D2, E2, G, \text{ux}, \text{ux10}, \text{vecc10ux}, \text{eo10}, \text{eo10Unpx1UXUlmxl veccl0ux}, Abark, tD1, tD2, tG, Aa, anpplus1, k, veccX, veccY, Inn, Ip, Ip, flagproj, signRd, flagplus, flchom, flcarith, options); \]
\[ tCPU = \text{cputime} - tCPU; \]
\[ \text{nFLOPS0} = \text{flops}; \]
\[ \text{polyXa} = \text{[repolya; impolyxa]}; \]
\[ \text{Xa} = \text{reXa} + \text{i} \times \text{imXa}; \]
\[ \text{X} = \text{invecc}(Xa(1:np,:), n, p); \]

%-------------------------------Form from solution epsilon(s)=1 the solution X.---
%[rx cr]=size(X);

plX = X;
pepsilon = epsilon;
Xi = \text{invecc}(\text{real}(X(:, rx, 1:np)) + \text{i} \times \text{imag}(X(:, rx, 1:np)), n, p);

if (flagproj == 0)
    LX = Aa * vecc(Xi) + anpplus1;
else
    LX = 1;
end

errori = abs(invecc(eo10, n, p) \times (A1 \times ((ux10 \times Xi) \times LX) \times B1 + A2 \times ((ux10 \times Xi)) \times B2 + C1 \times ((ux10 \times Xi)) \times C2 \times ((ux10 \times Xi)) \times D2 \times ((ux10 \times Xi)) \times E2 + G \times (LXi^2)));

%----------Evaluate polyxa=f (epsilon) and solution----------------
%----------X(epsilon=1).----------------------------------------

for i = 1:cr
    repolyxa(i, :) = polyfit(epsilon, real(X(:, i:i)), 5);
    rexa(i, :) = polyval(repolyxa(i, :), 1);
    impolyxa(i, :) = polyfit(epsilon, imag(X(:, i:i)), 5);
    imxa(i, :) = polyval(impolyxa(i, :), 1);
end

polyxa = [rexa; imxa];
Xa = rexa + li \times imxa;
X = invecc(Xa(1:np,:), n, p);
if (flagproj == 0)
    LX = Aa \times vecc(X) + anpplus1;
else
    LX = 1;
end

error=abs(invecc(eol0,n,p).*(A1*((uxl0.*X)*LX)*B1+A2*((uxl0.*X)*LX)*B2+C1*(uxl0.*X)*D1*X*E1+C2*(uxl0.*X)*D2*X*E2+G))

%----------Take best estimate of solution X(epsilon(s)=1).----
if (norm(errori)<norm(error))
    X=Xi;
    LX=LXi;
    error=errori;
end

%----------Scale back solution X.-----------------------------
X=uxl0.*X;
if (LX~=0)
    X=(1/LX)*X;
end

error=abs(A1*X*B1+A2*X*B2+C1*X*D1*X*E1+C2*X*D2*X*E2+G);

%----------If error not acceptable and solution------------------
%----------is not at infinity solve F(vec(X))=0------------------
%----------(using local convergence solver) with-----------------
%----------initial condition the best estimate-----------------
%----------of solution X(epsilon(s)=1).---------------------
if ((norm(error)>efinal) & (LX~=0) & (nsteps<=maxnsteps) & (stepmin>=minstep))
    [X,fvalX,exitflagX,outputX,jacobianX]=fSolve('FGQMEHOM11',[real(X) imag(X)],options,Ak, tD1, D1, tD2, D2, vecG, Inn, Ip, n, p, np);
    X=X(1:n,1:p)+li*X(1:n,p+1:2*p);
    error=abs(A1*X*B1+A2*X*B2+C1*X*D1*X*E1+C2*X*D2*X*E2+G);
end

%-----------Compute the Jacobian of GQME at the-----------------
%-----------solution X(epsilon(s)=1).----------------------------
XD1=X*D1;
XD2=X*D2;
D1X=D1*X;
D1Xt=real(D1X)'+li*imag(D1X)';
D2X=D2*X;
D2Xt=real(D2X)'+li*imag(D2X)';
JacobianX=Ak+tD1*(kron(Ip,XD1)+kron(D1Xt,Inn))+tD2*(kron(Ip,XD2)+kron(D2Xt,Inn));
if ((norm(error)>efinal) & (LX~=1) & (nsteps>maxnsteps) & (stepmin<minstep))
    disp('Warning: Solution obtained may be a solution at
    infinity.')
    disp('For verification check if:
    (a) L(X)=0
    (b) nsteps>maxnsteps
    (c) stepmin<minstep
    (d) error is not acceptable');
end
if (LX==0)
    disp('Warning: Solution obtained is a solution at
    infinity.');
end
plpseudos=plepsilon;
if (flpseudos==0)
    plpseudos='empty';
else
    plepsilon=plX(:,np+1);
end
if (sum(plotdisplay)==sum('on'))
%-----------Plot the trajectory Homotopy paths.-----------------
figure;
if (flpseudos~=0)
plot3(real(plX),plepsilon,imag(plX));grid;title('Homotopy paths of elements of X(epsilon).');
xlabel('re(X(epsilon)) elements');ylabel('epsilon');zlabel('Im(X(epsilon)) elements');
plot3(real(plX),plpseudos,imag(plX));grid;title('Homotopy paths of elements of [X(s);epsilon(s)].');
xlabel('[re(X(s));epsilon(s)]; elements');ylabel('s');zlabel('[Im(X(s));epsilon(s)] elements');
else
plot3(real(plX),plepsilon,imag(plX));grid;title('Homotopy paths of elements of X(epsilon).');
xlabel('re(X(epsilon)) elements');ylabel('epsilon');zlabel('Im(X(epsilon)) elements');
end
end
%----------Check flags and repeat loop if necessary.-----------
Dl=Dl+step;
if Dl>=1
flagend=flagend+1;
Dl=1;
end
end
%-------------------END OF THE PROGRAM---------------------
% Forms the derivative vector ordinary differential equation, % of the Homotopy continuation, used for the solution % of the scaled and projective transformed General Quadratic % Matrix Equation (GQME):

\[(10. \text{Aeo}) \cdot \text{vec} [A1((10. \text{ux}) \cdot X \cdot LX)] + A2((10. \text{ux}) \cdot X \cdot LX) B2 + C1((10. \text{Aux}) \cdot X) D1((10. \text{Aux}) \cdot X) E1 + C2((10. \text{ux}) \cdot X) D2((10. \text{ux}) \cdot X) E2 + G \cdot (LX^2)] = 0.\]

Where, \(^*\) denotes the Hadamard product.

This file is used as input to the differential equation solvers, RK45GQMEHOM11.m. See RK45GQMEHOM11.m m-file.

Computation of derivatives are based on Jacobians via Frechet derivatives.

The user has the option to select the \(dX(\epsilon)/d\epsilon\) computation for two kinds of homotopies:

1. \(H(X(\epsilon), \epsilon) = \epsilon f(\text{vec}((10. \text{Aux}) \cdot X(\epsilon))) - \gamma(1-\epsilon)(Y_o \cdot X(\epsilon) - X_o)\)

2. \(H(X(\epsilon), \epsilon) = \epsilon f(\text{vec}((10. \text{Aux}) \cdot X(\epsilon))) - \gamma(1-\epsilon)(Y_o - X_o)\)

Or when pseudo-arc length parameterization \(s\) is used:

1. \(H(X(\epsilon(s)), \epsilon(s)) = \epsilon(s) f(\text{vec}((10. \text{ux}) \cdot X(\epsilon(s)))) - \gamma(1-\epsilon(s))(Y_o \cdot X(\epsilon(s)) - X_o)\)

2. \(H(X(\epsilon(s)), \epsilon(s)) = \epsilon(s) f(\text{vec}((10. \text{ux}) \cdot X(\epsilon(s)))) - \gamma(1-\epsilon(s))(Y_o \cdot X(\epsilon(s)) - X_o)\)

For details see m-file GQMEHOM11.m (type help GQMEHOM11.m)

For more details see PhD Thesis:

Vassilios A. Tsachouridis.


Department of Engineering,
University of Leicester, 2002

function 
[Xp,signXp]=INGQMEHOM11(epsilon,X,f,A1,B1,A2,B2,C1,D1,E1,C2,D2,E2,G,ux,ux10,vec10ux,so10,so10unpx1xulmx1veccl0uxt,Abark,tD
1, tD2, tG, Aa, anpplus1, k, gam, vecceXo, vecceYo, Inn, Ip, Inp, n, p, flagpro
j, flpseudos, signRd, flchom, flcarith, signXp);
if flpseudos==0
spseudo=epsilon;
epsilon=X(n*p+1);
X=X(1:n*p);
end
if flagproj==0
LX=Aa*X+anpplus1;
else
LX=1;
Aa=zeros(1,n*p);
end
dXLXdveccX= (I np * LX+X*Aa)  ;
dYoXLXdveccX=diag (veccYo) * (Inp*LX+X*Aa) ;
dtGLX2dveccX=2*tG*LX*Aa ;
dXoLX2dveccX=2*vecceXo*LX*Aa ;
Xvecc10ux=X.*veccl0ux;
Xux=invecc(Xvecc10ux,n,p);
tXuxD1=Xux*D1;
tXuxD2=Xux*D2;
tD1Xux=Xux;
tD1Xuxt=real(tD1Xux)'+li*imag(tD1Xux)';
tD2Xuxt=real(tD2Xuxt)'+li*imag(tD2Xuxt)';
veccXvecclOux=eo10.*vecc(A1*(Xux*LX))*B1+A2*(Xux*LX)*B2+C1*Xux*
D1*Xux+B1+C2*Xux*B2*Xux*E2+G*(LX^2) ;
dfveccXvecc10uxdveccX=Abark*dXLDveccX+eo10UnpxlUXUlm1vecc10uxt
.* (tD1*(kron (Ip, tXuxD1)+kron (tD1Xuxt, Inn)) +tD2*(kron (Ip, tXuxD2)
+kron(tD2Xuxt, Inn))) +dtGLX2dveccX;
if (flh=0)
dHdveccX=epsilon*dfveccXvecclOuxdveccX-gam*(1-
epsilon) *(dYoXLXdveccX-dXoLX2dveccX) , *
dHdepsilon=fveccXvecclOux+gam* (veccYo. *X*LX-veccXo* (LXA2 )) ;
else
dHdveccX=epsilon*dfveccXvecclOuxdveccX-gam*(1-
epsilon) *(diag (2*veccYo. *X) -dXoLX2dveccX) ;
dHdpsilon=fveccXvecc10ux+gam* (veccYo. *X.*X-veccXo*(LX^2)) ;
end
Xp=-pinv (dHdveccX) *dHdpsilon;
%---------------------Xp=- (dHdveccXA#) *dHdpsilon via QR.--------
% [Q,R]=qr(dHdveccX);
% Xp=R\ (R' \ (dHdveccX'*(-dHdpsilon)));
% r=-dHdpsilon-dHdveccX*Xp;
% er=R\ (R' \ (dHdveccX'*r));
% Xp=Xp+er;
%---------------------If pseudo-arc length parameterization-----
%---------------------------------------------------------------
if flpseudos==0
%---------------------If H is complex analytic and complex-------
%---------------------------------------------------------------
if flchom==0
%---------------------If H is complex analytic and real---------
%---------------------------------------------------------------
dPdveccX=[real (dHdveccX) -imag (dHdveccX)];img (dHdveccX)
real (dHdveccX));
dPdpsudos=[real (dHdpsilon);imag (dHdpsilon)];
dPdpseudos=[dPdveccX dPdpsudos];
\[ [Q_r, R_r] = qr(dPdpseudos') \]
for \( i = 1:2n*p \)
\[ \text{signRd} = \text{signRd} \times \text{sign} (R_r(i,i)) \]
end
\[ \text{signXp} = ((-1) \times (2n*p)) \times \text{signRd} \]
if \( \text{spseudo} = 0 \)
%-----------------------Select initial direction such that--------
%-----------------------epsilon (spseudo) > 0.------------------------
Signes = signXp \times Q_r(2n*p, 2n*p + 1);
\[ \text{signXp} = \text{sign} (\text{Signes}) \]
end
\[ \text{signXpo} = -\text{sign} (\text{Signes}) \]
end
if \( \text{signXp} \times \text{signXpo} < 0 \)
\[ \text{signXp} = -\text{signXp} \]
end
\[ X_p = \text{signXp} \times [Q_r(:, n*p + l) + li*Q_r(n*p + l:2n*p + l); Q_r(2n*p + l/2n*p + l)] \]
\[ dHdspseudo = [dHdvecX dHdepsilon] \]
\[ [Q_r, R_r] = qr(dHdspseudo') \]
for \( i = 1:n*p \)
\[ \text{signRd} = \text{signRd} \times \text{sign} (R_r(i,i)) \]
end
\[ \text{signXp} = ((-1) \times (n*p)) \times \text{signRd} \]
if \( \text{spseudo} = 0 \)
%-----------------------Select initial direction such that--------
%-----------------------epsilon (spseudo) > 0.------------------------
Signes = signXp \times Q_r(n*p, n*p + 1);
\[ \text{signXp} = \text{sign} (\text{Signes}) \]
end
\[ \text{signXpo} = -\text{sign} (\text{Signes}) \]
end
if \( \text{signXp} \times \text{signXpo} < 0 \)
\[ \text{signXp} = -\text{signXp} \]
end
\[ X_p = \text{signXp} \times Q_r(:, n*p + 1) \]
end

%-----------------------END OF THE PROGRAM-----------------------
%-----------------------VASSILIOS A. TSACHOURIDIS-----------------------
%-----------------------20-12-2001-----------------------
**Appendix C** The GQME-Toolbox 465

```plaintext
%**************************************************************************
%**************************************************************************
% M-File by
% V. A. TSACHOURIDIS
% 20-12-2001
% All rights reserved.
%**************************************************************************
%**************************************************************************

**RK45GQMEHOM11** Solve differential equations, higher order method.

**RK45GQMEHOM11** integrates a system of ordinary differential equations using 4th and 5th order Runge-Kutta formulas.

```
[tout,yout,nsteps,stepmin] = RK45GQMEHOM11(t0, tfinal, t0,
  tol, n, p, gam, def, flh, flpseudos, maxnsteps, minstep,
  flagcor, displayode, A1, B1, A2, B2, C1, D1, E1, C2, D2, E2, G, ux, uxt10,
  vecclux, eol10, eol10Unpx1XU1mx1vecc10uxt, Abark, tD1, tD2, tG, Aa,
  anplus1, k, vecxcO, vecyo, Inn, Ip, flagsproj, signRd, flchom,
  floarith),
```

integrates the system of ordinary differential equations described by the M-file INGQMEHOM11.m.

to the interval T0 to Tfinal, with initial conditions Y0.
Tolerance TOL displays status while the integration proceeds.

**INPUT:**
t0 - Initial value of t.
tfinal - Final value of t.
yO - Initial value column-vector.
tol - The desired accuracy. (Default: tol = 1.e-6).
n, p - Dimensions of the solution Q=invvecc(Y): nxp.
gam - when n=p: gam>0 (gam<0) if X>=0 (X<=0) solution required.
pdef - For Positive (Negative) semidefinite pdef=1
  - For NOT-Positive (NOT-Negative) definite pdef=-1
  - For Any other case pdef<>+1,-1
flh - Flag for INGQMEHOM11.m
flpseudos - flpseudos>0 when pseudo-arc length parameterization is used.
maxnsteps - Maximum number of integration steps allowed.
minstep - Minimum integration step allowed.
maxsteps and minstep are used together in combination.
flagcor - Flag for corrector on each step (On when 0, Off when <>0).
displayode - displays the step and epsilon on each iteration if 'on' and does not display if 'off'.

For the rest of inputs see GQMEHOM11.m

**OUTPUT:**
tout - Returned integration time points (column-vector).
Yout - Returned solution, one solution column-vector per tout-value.
nsteps - Number of integration steps used in the ODE routine RK45GQMEHOM11.m.
stepmin - Minimum integration step used in the integration ODE routine RK45GQMEHOM11.m.
```
% The result can be displayed by: plot(tout, yout).

% For more details see PhD Thesis:
% Vassilios A. Tsachouridis.
% "A Unified Framework for the Numerical Solution &
% Analysis of Generalized Algebraic Quadratic Matrix
% Equations with Engineering and Scientific Applications:
% Theory & Software Design and Implementation"
% Department of Engineering,
% University of Leicester, 2002

function [tout, yout, nsteps, stepmin] = RK45GQMEHOM11(t0, tfinal, y0, tol, n, p, np, gam, pdef, flh, flpseudos, maxnsteps, minstep, flagcor, displayode, A1, B1, A2, B2, C1, D1, E1, C2, D2, E2, G, u, ux10, vecclOux, eolO, eolOUnpx; XUlmxlvecclOux, Abark, tD1, tD2, tG, Aa, anplusl, k, vecclox, veccloy, In n, Ip, Inp, flagproj, signRd, flchom, flcarith, options);

% The Fehlberg coefficients:
alpha = [1/4 3/8 12/13 1 1/2];
beta = [1 0 0 0 0] / 4
beta = [3 9 0 0 0] / 32
beta = [1932 -7200 7296 0 0 0 1/2197
beta = [8321 -32800 29440 -845 0 0 1/4104
beta = [-6080 41040 -28042 9295 -845 0 1/20520];

beta = [902880 0 3953664 3855735 -1371249
beta = [2090 0 22528 21970 -15048

pow = 1/5;

% Initialization
normerond = tol + 1;
t = t0;

hmax = (tfinal - t)/10;
h = hmax/10;
hl = hmax + 1;

stepmin = hl;
y = y0(:);

chunk = 128;
tout = zeros(chunk, 1);
yout = zeros(chunk, length(y));
k = 1;
tout(k) = t;
yout(k, :) = y.';
if flpseudos == 0
ynpplusl = y(n*p+1);

flps = 0;

if flagcor == 0
load('optionsdata', 'options');
end

% The main loop.
while (flps == 1) & (t + h > t)
if ((flpseudos == 0) & (t + h > tfinal)), h = tfinal - t; end
if (sum(displayode) == sum('on'))
if flpseudos == 0
fprintf(2, 'epsilon step' = [e e]n, [t h]);
else
    if ynpplusl<0
        disp('Error: epsilon(s)<0');
        break;
    end
    if t > tfinal
        disp('Arc-length larger than expected.');
    end
    fprintf(2,'[epsilon s step][%e %e %e]\n',ynpplusl t h);
end
% Compute the slopes.
signXpo=0;
[temp,signXpo] =
ingqmehomll(t,y,flh,A1,B1,A2,B2,C1,D1,E1,C2,D2,E2,G,ux,ux10,veccl0ux,eo10,eo10Unpx1UXU1mx1vecc10uxt,Abark,tD1,tD2,ta1,anpplus1,k,gam,veccXo,veccYo,Inn,Ip,Inp,n,p,flagproj,flpseudos,signRd,flchom,flcarith,signXpo);
f(:,1) = temp(:,);
for j  =1:5
    [temp,signXpo]=
ingqmehomll(t+alpha(j) *h,y+h*f*beta(:,j),flh,A1,B1,A2,B2,C1,D1,E1,C2,D2,E2,G,ux,ux10,veccl0ux,eo10,eo10Unpx1UXU1mx1vecc10uxt,Abark,tD1,tD2,ta1,anpplus1,k,gam,veccXo,veccYo,Inn,Ip,Inp,n,p,flagproj,flpseudos,signRd,flchom,flcarith,signXpo);
f(:,j+1) = temp(:);
end
% Ensure the right direction for the solution when n=p
% and y+h*f*gamma(:,1) is not >=0 or <=0.
if n==p
    if pdef==-1
        %----->=0 Positive semidefinite and <= Negative semidefinite.
eigy=real(eig(invecc(y(1:n*p),n,p)));
fgamma=f*gamma(:,1);
eigsl=real(eig(invecc(fgamma(1:n*p),n,p)));
        % For Q>=0 direction.
        if (gam>0)
            if (min(eigsl)<0) & ((-max(eigy)/min(eigsl))<h) &((-
                max(eigy)/min(eigsl))>0)
                h=-max(eigy)/min(eigsl);
            end
        end
        % For Q<=0 direction.
        if (gam<0)
            if (max(eigsl)>0) & ((-min(eigy)/max(eigsl))<h) &((-
                min(eigy)/max(eigsl))>0)
                h=-min(eigy)/max(eigsl);
            end
        end
    end
    if pdef==1
        %--------0 Positive semidefinite and <= Negative semidefinite.
eigy=real(eig(invecc(y(1:n*p),n,p)));
fgamma=f*gamma(:,1);
eigsl=real(eig(invecc(fgamma(1:n*p),n,p)));
        % For Q>=0 direction.
        if (gam>0)
            if (min(eigsl)<0) & ((-max(eigy)/min(eigsl))<h) &((-
                max(eigy)/min(eigsl))>0)
                h=-min(eigy)/max(eigsl);
            end
        end
% For Q<=0 direction.
if (gam<0)
    if (max(eigsl)>0) & ((-max(eigy)/max(eigsl))<h) & ((-max(eigy)/max(eigsl))>0)
        h=-max(eigy)/max(eigsl);
    end
end
end

% Estimate the error and the acceptable error.
    delta = norm(h*f*gamma(:,2),'inf');
    tau = tol*max(norm(y,'inf'),1.0);
% Update the solution only if the error is acceptable.
if (delta <= tau)
    t = t + h;
    y = y + h*f*gamma(:,1);
% Correct the ODE solution to path tracking if required.
if (flagcor==0)
    if flpseudos==0
        epsilon=y(n*p+1);
    else
        epsilon=t;
    end
    X=y(l:n*p);
    if (flagproj==0)
        LX=Aa*X+anpplus1;
    else
        LX=1;
        Aa=zeros(1,n*p);
    end
    Xveccl0ux=X.*veccl0ux;
    Xux=invvec(Xveccl0ux,n,p);
    fvecclXveccl0ux=eo10.*veccl(A1*(Xux*LX)*B1+A2*(Xux*LX)*B2+C1*Xux*
        D1*Xux*E1+C2*Xux*D2*Xux*E2+G*(LX^2));
    if (flh==0)
        H=epsilon*fvecclXveccl0ux-gam*(1-
        epsilon)*(veccYo.*X.*X-veccXo*(LX^2));
    else
        H=epsilon*fvecclXveccl0ux-gam*(1-
        epsilon)*(veccYo.*X.*X-veccXo*(LX^2));
    end
    normercond=norm(H,'fro')/norm(X,'fro');
end
if (normercond>tol)
    [X,fvalX,exitflagX,outputX,jacobianX]=fsolve('FGQMEODE1',
        [real(X) imag(X)],options,epsilon,flh,
        A1,B1,A2,B2,C1,D1,E1,C2,D2,E2,G,ux,ux10,vecc10ux,eqo,eq10,
        vecc10ux,eo10,eo10Unpx1
        XUXMXVECC10UX,AARK,TD1,TD2,TDG,AA,ANPLUS1,K,GAM,VECCXO,VECCYO,
        INN,IP,INP,N,P,NP,FLAGPROJ,FLPSUEOS,SIGNRD,FICHOM,FLCARITH);
    X=X(:,1)+i*X(:,2);
    Xveccl0ux=X.*veccl0ux;
    Xux=invvec(Xveccl0ux,n,p);
    fvecclXveccl0ux=eo10.*veccl(A1*(Xux*LX)*B1+A2*(Xux*LX)*B2+C1*Xux*
        D1*Xux*E1+C2*Xux*D2*Xux*E2+G*(LX^2));
    if (flagproj==0)
        LX=Aa*X+anpplus1;
    end
end
if (flh==0)
    H=epsilon*fvecclXveccl0ux-gam*(1-
    epsilon)*(veccYo.*X.*X-veccXo*(LX^2));
else

H = epsilon * fvecX * veccl0ux * gam * (1 - epsilon) * (veccXo * X * vecXo * (LX^2));
end
end
end
k = k + 1;
if k > length(tout)
    tout = [tout; zeros(chunk,1)];
yout = [yout; zeros(chunk,length(y))];
end
tout(k) = t;
yout(k,:) = y.';
end

% Update the step size.
if (delta ~= 0.0)
    h = min([hmax 0.8*h*(tau/delta)^pow h1]);
    h1 = hmax + 1;
    % Stop the maximum number of steps and the minimum step
    % exceeded.
    nsteps = k;
    stepmin = min(stepmin, h);
    if ((h < minstep) & (k > maxnsteps))
        t = tfinal;
        disp('Minimum step less than specified and maximum
number of steps exceeded.');
    end
end
if flpseudos == 0
    ynpplusl = y(n*p + 1);
    if ynpplusl > 1
        flps = 1;
    if (sum(displayode) == sum('on'))
        fprintf(2,'[epsilon s step]=[%e %e %e]
', [ynpplusl t h]);
    end
else
    ynpplusl = 1;
end
end
if ((t < tfinal) & (flps == 1)) | ((ynpplusl < 1) & (flps == 1))
    disp('Singularity likely.');
end
tout = tout(1:k);
yout = yout(1:k,:);
%----------------------------------END OF THE PROGRAM---------------------------------
%----------------------------------VASSILIOS A. TSACHOURIDIS---------------------------------
%----------------------------------20-12-2001---------------------------------------------
% Calculation of the solution X(epsilon) of H(X(epsilon), epsilon)=0.
% See GQMEHOM11.m

% For more details see PhD Thesis:
% Vassilios A. Tsachouridis.
% "A Unified Framework for the Numerical Solution &
% Analysis of Generalized Algebraic Quadratic Matrix
% Equations with Engineering and Scientific Applications:
% Theory & Software Design and Implementation"
% Department of Engineering,
% University of Leicester, 2002

function [Xs, JXs] = FGQMEODE11(X, epsilon, flh, A1, B1, A2, B2, C1, D1, E1, C2, D2, E2,
G, uX, uXl0, veccl Oux, eol0, eol0UnpxlXUlmxlveccl Ouxt, Abark, tD1, tD2,
tA, anplplus1, k, gam, vecxo, vecYo, Inn, Ip, Inp, n, p, np, flagproj,
flpseudos, signRd, flchom, flcarith)

X = X(:,1) + li*X(:,2);
if flagproj = 0
LX = Aa * vecx (X) + anplplus1;
else
LX = 1;
Aa = zeros (1, n*p);
end

dYo = LX * vecx (X) * diag (vecYo) *(Inp * LX + X * Aa);
dXoLX = 2 * vecYo * LX + X * Aa;
dtGLX = 2 * tG * LX + X * Aa;
dXoLX2 = 2 * vecYo * LX + X * Aa;

vecx10ux = X * veccl Oux;
Xux = invvecx (vecx10ux, n, p);
tXuxD1 = Xux * D1;
tXuxD2 = Xux * D2;
tD1Xux = D1 * Xux;
tD2Xux = real (tD1Xux) + li*imag (tD1Xux);

vecx10ux = X * vecx10ux + gam * (1-epsilon) * (vecYo * X * LX - vecx10ux);

if (flh <= 0)
vecxH = epsilon * vecx10ux - gam * (1-epsilon) * (vecYo * X * LX - vecx10ux);
else
vecxH = epsilon * vecx10ux - gam * (1-epsilon) * (vecYo * X * LX - vecx10ux);
end

Xs = [real (vecxH) imag (vecxH)];

%---------------------------------------------Jacobian JH(X)---------------------------------------------
if nargout > 1
dvecx10ux = Abark * dvecx10ux + epsilon * vecx10ux + gam * (1-epsilon) * (vecYo * X * LX - vecx10ux);
end

X=[real (vecxH) imag (vecxH)];
else
    dHdveccX = epsilon*dfveccXvecc10uXdveccX-gam*(1-epsilon)*(diag(2*veccYo.*X)-dXoLX2dveccX);
end
JXs = [ [ real(dHdveccX); zeros(np,np) ]
        [ zeros(np,np); imag(dHdveccX) ] ];
%
%---------------------END OF THE PROGRAM-----------------------------
%---------------------VASSILIOS A. TSACHOURIDIS---------------------
%---------------------20-12-2001----------------------------------
Appendix C
The GQME-Toolbox 472

% All rights reserved.
% Calculation of the solution at epsilon=1 (F(X)=0).
% See GQMEHOM11.m

% For more details see PhD Thesis:
%  Vassilios A. Tsachouridis.
% "A Unified Framework for the Numerical Solution &
% Analysis of Generalized Algebraic Quadratic Matrix
% Equations with Engineering and Scientific Applications:
% Theory & Software Design and Implementation"
% Department of Engineering,
% University of Leicester, 2002

function
[Xs,Xs]=FGQMEHOM11(X,Ak,tD1,D1,tD2,D2,veccG,Inn,Ip,n,p,np);
X=X(:,1:p)+li*X(:,p+1:2*p);
veccX=vecc(X);
XD1=X*D1;
XD2=X*D2;
D1X=D1*X;
D1Xt=real(D1X)+li*imag(D1X);
D2X=D2*X;
D2Xt=real(D2X)+li*imag(D2X);
fveccX=Ak*veccX+(tD1*kron(Ip,XD1)+tD2*kron(Ip,XD2))*veccX+veccG;
Xs=[real(fveccX) imag(fveccX)];
%--------------------------------------Jacobian JF(X)-----------------------------
if nargsout>1
dfveccXdveccX=Ak+tD1*(kron(Ip,XD1)+kron(D1Xt,Inn))+tD2*(kron(Ip
, XD2)+kron(D2Xt,Inn));
JXs=[[real(dfveccXdveccX);zeros(np,np)]
[zeros(np,np);imag(dfveccXdveccX)]
end
%--------------------------------------END OF THE PROGRAM-------------------------
%--------------------------------------VASSILIOS A. TSACHOURIDIS--------------------
%--------------------------------------20-12-2001---------------------------------
Appendix D

Drivers for the Numerical Examples

D.1 Introduction

In this appendix, the drivers (m-files) for the computation of the numerical examples in chapters 6-7 are given.

D.2 Drivers

The software code for these drivers is given in the subsequent pages.
% Driver: Example61.m
% Examples 1, 2: well conditioned,
% Examples 3, 4: ill conditioned with the increase of k.
k = 4; s = 1; example = 4;
% Example 1.
if (example == 1)
    Aa = diag([-10^(-k) -2 -3*10^(-k)]);
    Da = diag([10^(-k) 1 10^(-k)]);
    Ca = diag([3*10^(-k) 5 7*10^(-k)]);
end
if (example == 2)
% Example 2.
    Aa = diag([10^(-k) 2*10^(-k) 3*10^(-k)]);
    Da = diag([10^(-k) 10^(-k) 10^(-k)]);
    Ca = diag([10^(-k) 1 10^(-k)]);
end
if (example == 3)
% Example 3.
    Aa = diag([10^(-k) 2 3*10^(-k)]);
    Da = diag([10^(-k) 1 10^(-k)]);
    Ca = diag([10^(-k) 4*10^(-2k) 8*10^(-k)]);
end
if (example == 4)
% Example 4.
    Aa = diag([-10^(-k) -2 -3*10^(-k)]);
    Da = diag([10^(-k) 1 10^(-k)]);
    Ca = diag([3*10^(-k) 5 7*10^(-k)]);
end
At = kron(eye(s), s, Aa);
Dt = kron(eye(s), s, Da);
Ct = kron(eye(s), s, Ca);
[n, m] = size(At);
u = ones(n, 1);
U = eye(n, n) - 0.2 / norm(u)^2 * u * u';
A = U * At * U;
D = U * Dt * U;
C = U * Ct * U;

% Exact solution Y.
Yt = zeros(n, n);
for i = 1:n
    Yt(i, i) = (At(i, i) + sqrt(At(i, i)^2 - Dt(i, i) * Ct(i, i))) / (-Dt(i, i));
end
Y = U * Yt * U;
A1 = 0.1 * A';
B1 = 10 * eye(n, n);
A2 = 5 * eye(n, n);
B2 = 0.2 * A;
C1 = 0.4 * eye(n, n);
D1 = 2 * D;
E1 = 0.5 * eye(n, n);
C2 = 0.6 * eye(n, n);
D2 = 0.5 * D;
E2 = 2 * eye(n, n);
G = C;
Alo = 0 * A1;
B1o = 0 * B1;
A2o = 0 * A2;
B2o = 0 * B2;
C1o = 0 * C1;
D1o = 0 * D1;
E1o = 0 * E1;
Appendix D

Drivers for the Numerical Examples 475

\[ C_20 = 0 \cdot C_2; \]
\[ D_20 = 0 \cdot D_2; \]
\[ E_20 = 0 \cdot E_2; \]
\[ \text{Go} = 0 \cdot \text{G}; \]
\[ \text{flagproj} = 1; \text{flagscale} = 1; \text{flagv} = 1; \epsilon = 0; \text{flagopt} = 0; \text{flagsym} = 0; \]
\[ \text{sux} = \text{sym('sux1', 'real')} \cdot \text{sym('sux2', 'real')} \]
\[ \text{sym('sux3', 'real')} \cdot \text{sym('sux21', 'real')} \cdot \text{sym('sux22', 'real')} \]
\[ \text{sym('sux33', 'real')}; \]
\[ \text{se0} = \text{sym('se01', 'real')} \cdot \text{sym('se021', 'real')} \cdot \text{sym('se031', 'real')} \cdot \text{sym('se041', 'real')} \cdot \text{sym('se051', 'real')} \cdot \text{sym('se061', 'real')} \cdot \text{sym('se071', 'real')} \cdot \text{sym('se081', 'real')} \cdot \text{sym('se091', 'real')}; \]
\[ \text{to} = \text{clock}; \]
\[ \text{tscale} = \text{etime}(\text{clock}, \text{to}); \]
\[ \text{gam} = 1; \text{eps} = 1e-12; \text{efinal} = 1e-20; \text{flh} = 0; \text{flagcor} = 1; \text{pdef} = 0; \text{step} = 1; \]
\[ \text{maxsteps} = \text{realmax}; \text{minstep} = \text{realmin}; \]
\[ \text{largescale} = \text{'on'}; \text{jacobianqme} = \text{'on'}; \text{linesearch} = \text{'cubicpoly'}; \]
\[ \text{displaysolver} = \text{'off'}; \text{displayode} = \text{'off'}; \text{plotdisplay} = \text{'off'}; \]
\[ \text{Yo} = \text{ones}(n,n); \text{Xo} = (1e-0) * \text{eye}(n,n); \text{SignXoYo} = \text{ones}(n,n); \]
\[ \text{Aa} = [8.3850e-001 5.6807e-001 3.7041e-001 \ldots 7.0274e-001 5.4657e-001 4.4488e-001 \ldots 6.9457e-001 6.2131e-001 7.9482e-001]; \]
\[ \text{anplusl} = \text{norm}(\text{Aa}); \]
\[ \text{flchom} = 1; \text{flcarith} = 0; \]
\[ \text{psuedosfin} = 0; \]
\[ \text{to} = \text{clock}; \]
\[ \text{tc} = \text{etime}(\text{clock}, \text{to}); \]
\[ \text{teq} = \text{tc} - \text{tscale}; \]
\[ \text{nat} \]

Save example61nspSignXoYo Xo Yo X error Y JacobianX LX ux eo S Sm plX plesislon plesiodus nsteps nFLOPS nFLOPSscale stepmin tcomputation teCPU teCPUscale NA flagproj flagscale flagv eps flagopt flagsym Aa anplusl gam e efnal flh flagcor pdef step maxsteps mindstep flchom flcarith plesiodus plesiosfin largescale jacobiangmelinearsearch displaysolver displayode plotdisplay
% Driver: Example62.m
% Examples 1, 2: well conditioned,
% Examples 3, 4: ill conditioned with the increase of k.
% k=4; s=1; example=4;
% Example 1.
if (example==1)
Aa=diag([-10^(-k) -2 -3*10^(-k)]);
Da=-diag([10^(-k) 1 10^(-k)]);
Ca=diag([3*10^(-k) 5 7*10^(-k)]);
end
if (example==2)
% Example 2.
Aa=diag([10^(-k) 2*10^(-k) 3*10^(-k)]);
Da=-diag([10^(-k) 10^(-k) 10^(-k)]);
Ca=diag([10^(-k) 1 10^(-k)]);
end
if (example==3)
% Example 3.
Aa=diag([10^(-k) 2 3*10^(-k)]);
Da=-diag([10^(-k) 1 10^(-k)]);
Ca=diag([10^(-k) 4*10^(-2*k) 8*10^(-k)]);
end
if (example==4)
% Example 4.
Aa=diag([-10^(-k) -2 -3*10^(-k)]);
Da=-diag([10^(-k) 1 10^(-k)]);
Ca=diag([3*10^(-k) 5 7*10^(-k)]);
end
At=kron(eye(s,s),Aa);
Dt=kron(eye(s,s),Da);
Ct=kron(eye(s,s),Ca);
[n,m]=size(At);
u=ones(n,1);
U=eye(n,n)-(2/norm(u)^2)*u*u';
A=U*At*U;
D=U*Dt*U;
C=U*Ct*U;

% Exact solution Y.
Yt=zeros(n,n);
for i=1:n
Yt(i,i)=(At(i,i)+sqrt(At(i,i)^2-Dt(i,i)*Ct(i,i)))/(-Dt(i,i));
end
Y=U*Yt*U;
A1=0.1*A';
B1=10*eye(n,n);
A2=5*eye(n,n);
B2=0.2*A;
C1=0.4*eye(n,n);
D1=2*D;
E1=0.5*eye(n,n);
C2=0.6*eye(n,n);
D2=0.5*D;
E2=2*eye(n,n);
G=C;
A10=0*A1;
B10=0*B1;
A20=0*A2;
B20=0*B2;
C10=0*C1;
D10=0*D1;
E10=0*E1;
Appendix D Drivers for the Numerical Examples 477

C2o=0*C2;
D2o=0*D2;
E2o=0*E2;
Go=0*G;
flagproj=1;flagscale=1;flagv=1;eps=0;flagopt=0;flagsym=0;
sux=['sux1l','real'] sym('sux12','real')
sym('sux13','real');sym('sux21','real') sym('sux22','real')
sym('sux23','real');sym('sux31','real') sym('sux32','real')
sym('sux33','real');
se0=['s01l','real'];sym('s021','real') sym('s031','real');sym('s041','real') sym('s051','real') sym('s061','real') sym('s071','real');sym('s081','real') sym('s091','real');
to=clock;

Abar,Ebar,Cbar,ux,eo,S,Sm,Alhat,B1hat,A2hat,B2hat,C1hat,D1hat,E1hat,C2hat,D2hat,E2hat,Ghat,tCPUscale,nFLOPSscale=SQME(A1,B1,A2,B1,C1,D1,E1,C2,D2,E2,G,Alo,Blo,A2o,B2o,Clo,D1o,E1o,C2o,D2o,E2o,
Go,flagscale,flagv,eps,flagopt,flagsym,sux,se0);
tscale=etime(clock,to);
gam=1.0;e=1e-12;efinal=1e-20;flh=0;flagcor=1;pdef=0;step=1;
maxnstep=maxmin;minstep=maxmin;
largesscale='on';jacobianqme='on';linesearch='cubicpoly';
diplaysolver='off';diplayode='off';plotdisplay='off';
Yo=ones(n,n);Xo=(1e-0)*eye(n,n)/SignXoYo;
annplu1=norm(Aa);
flchom=1;flcarith=0;
flpseudos=0;
pseudosfin=100;
to=clock;

%[error,X,JacobianX,LX,ux,eo,polyXa,p1X,plepsilon,p1pseudos,nstep,
stepmin,tCPU,nFLOPS,tCPUscale,nFLOPSscale]=GQMEHOM1(A1,B1,A2
,B2,C1,D1,E1,C2,D2,E2,G,Alo,Blo,A2o,B2o,Clo,D1o,E1o,C2o,D2o,E2o
,Go,flagscale,flagv,eps,flagopt,flagsym,sux,se0,flagproj,Aa,annplu1,gam,pdef,Xo,Yo,SignXoYo,step,e,efinal,flh,flchom,flcarith,
pseudosfin,maxnstep,minstep,flagcor,largesscale,jacobianqme,linesearch,displaysolver,displayode,plotdisplay);

%[error,X,JacobianX,LX,ux,eo,polyXa,p1X,plepsilon,p1pseudos,nstep,
stepmin,tCPU,nFLOPS,tCPUscale,nFLOPSscale]=GQMEHOM1(A1,B1
,A2,B2,C1,D1,E1,C2,D2,E2,G,Alo,Blo,A2o,B2o,Clo,D1o,E1o,C2o,D2o
,E2o,Go,flagscale,flagv,eps,flagopt,flagsym,sux,se0,flagproj,Aa,annplu1,gam,pdef,Xo,Yo,SignXoYo,step,e,efinal,flh,flchom,flcarith,
pseudosfin,maxnstep,minstep,flagcor,largesscale,jacobianqme,linesearch,displaysolver,displayode,plotdisplay

%error,X,JacobianX,LX,ux,eo,polyXa,p1X,plepsilon,p1pseudos,nstep
stepmin,tCPU,nFLOPS,tCPUscale,nFLOPSscale]=GQMEHOM1(A1,B1
,A2,B2,C1,D1,E1,C2,D2,E2,G,Alo,Blo,A2o,B2o,Clo,D1o,E1o,C2o,D2o
,E2o,Go,flagscale,flagv,eps,flagopt,flagsym,sux,se0,flagproj,Aa,annplu1,gam,pdef,Xo,Yo,SignXoYo,step,e,efinal,flh,flchom,flcarith,
pseudosfin,maxnstep,minstep,flagcor,largesscale,jacobianqme,linesearch,displaysolver,displayode,plotdisplay

tcomputation=etime(clock,to);
teq=tcomputation-tscale;
natest
NA=[CnX Erfmaxmax Epfmaxfl Erb Efbmax Erbmaxfl NS NSfl];

%nsnp
%snpl
%svnp

save example62nsnp SignXoYo Xo Yo X error Y JacobianX LX ux eo
S Sm plX plepsilon plpseudos nsteps nFLOPS nFLOPSscale stepmin
tcomputation tCPU tCPUscale NA flagproj flagscale flagv eps
flagopt flagsym Aa anplu1 gam e efinal fh flagcor pdef step
maxnstep minstep flchom flcarith flpseudos pseudosfin
largesscale jacobianqme linesearch displaysolver displayode
plotdisplay
% Driver: Example63.m

de=0;
A=[-de 1 0 0;-1 -de 0 0 0 de 1;0 0 -1 de];
B=[1;1;1;1];
C=B';
E=diag([1 1 1 1]);
Q=1;
R=1;
S=0;
A1=0.1*(A'-C'*S*inv(R)*B');
B1=10*E;
A2=5*E';
B2=0.2*(A-B*inv(R)*S*C);
C1=0.4*E;
D1=-2*B*inv(R)*B';
E1=0.5*E;
C2=0.6*C1;
D2=0.5*D1;
E2=2*E1;
G=C'*Q*S*inv(R)*S';
Alo=0*A';
Blo=0*B1;
A2o=0*A1;
B2o=0*B2;
Clo=0*C1;
Dlo=0*D1;
Elo=0*E1;
C2o=0*C2;
D2o=0*D2;
E2o=0*E2;
G0=0*G;
sux=sym('s','real')*eye(4,4);seo=sym('seo','real')*ones(16,1);
flagproj=1;eps=0;flagscale=1;flagv=1;flagopt=0;flagsym=0;
to=clock;
[Abar,Bbar,Cbar,ux,eo,S,Sm,Aihat,Bihat,B2hat,C1hat,D1hat,
 E1hat,C2hat,D2hat,E2hat,Ghat,tcFUscale,nFLOPsScale]=SGQME(A1,B1,
A2,B2,C1,D1,E1,C2,D2,E2,G,flagscale,flagv,eps,flagopt,flagsym,
sux,seo);
tscale=etime(clock,to);
save dfile ux eo
gam=1;e=1e-6;efinal=1e-20;flh=0;flchom=0;flcarith=0;
flagcor=1;
flpseudos=0;
pseudosfin=1;
maxnsteps=realmax;minstep=1e-60;
largescale='on';jacobianqme='on';linesearch='cubicpoly';
display solver='off';displayode='off';plotdisplay='off';
step=1;pdef=0;
Yo=ones(4,4);
Xo=1e-8*[1 0 0 0;0 1 0 0;0 0 1 0;0 0 0 1];
SignXoYo=[1 1 1 1;1 1 1 1;1 1 1 1;1 1 1 1];
Aa=rand(1,16);
anpplusl=norm(Aa);
to=clock;
[error,X,JacobianX,LX,ux,eo,polyXa,plX,plepsilon,plpseudos,nste,
 ps,stepmin,tcCPU,nFLOPs,tcFUscale,nFLOPsScale]=GQMEHOM1(A1,B1,A2,
B2,C1,D1,E1,C2,D2,E2,G,A1o,Blo,A2o,B2o,Clo,Dlo,E1o,C2o,D2o,E2o,
Go,flagscale,flagv,eps,flagopt,flagsym,sux,seo,flagproj,Aa,anp
 plus1,gam,pdef,Xo,Yo,SignXoYo,step,e,efinal,flh,flchom,flcarith
,flpseudos,pseudosfin,maxnsteps,minstep,flagcor,largescale,jaco
bianqme,linesearch,display solver,displayode,plotdisplay);
Appendix D Drivers for the Numerical Examples

% [error, X, JacobianX, LX, ux, eo, polyXa, plX, plepsilon, plpseudo, nst
% eps, stepmin, tCPU, nFLOPS, tCPUscale, nFLOPSscale] = GMEHOM11(A1, B1
% A2, B2, C1, D1, E1, C2, D2, E2, G, A1o, B1o, A2o, B2o, C1o, D1o, E1o, C2o, D2o
% E2o, Go, flagscale, flagv, eps, flagopt, flagsym, sux, seo, flagproj, A
% Aa, anaplus1, gam, pdef, Xo, Yo, SignXoYo, step, e, efinal, fih, flchom, fl
	tcomputation = etime(clock, to);

teq = tcomputation - tscale;

% natest
NA = [CnX Eprfmax Eprfmaxfl Erb Erbmax Erbmaxfl NS NSf1];
% nsnp
% snp
save example63nsnp SignXoYo Xo Yo X error JacobianX LX ux eo S
Sm plX plepsilon plpseudo nsteps nFLOPS nFLOPSscale stepmin
tcomputation teq tCPU tCPUscale NA flagproj flagscale flagv eps
flagopt flagsym Aa anaplus1 gam e efinal fih flagcor pdef step
maxnsteps minstep flchom flcarith flpseudo pseudosfin
largescale jacobiangqmne linesearch display solver displayode
plotdisplay
%Driver: Example64.m
clear all;
A = [-0.161 -6.004 -0.58822 -9.9835 -0.4073 -3.9820];
eye(7,7);zeros(1,8));
B = [0 0.0064 0.00235 0.0713 1.0002 0.1045 0.9955];
E1 = 0.001 * ([0 0 0 0 0 0 11 132 18];zeros(1,8));
E2 = [0 0 1];
vita = l;e-
Rl = E1;''*E1;
R2 = vita * E2;*E2;
A1 = 0.4 * A ';
B1 = 2.5 * eye(8,8);
A2 = 5 * eye(8,8);
B2 = 0.2 * A ';
C1 = 0.2 * eye(8,8);
D1 = -B * inv(R2) * B';
E1 = 5 * eye(8,8);
load mdata
C2 = m;
D2 = -0.2 * D1;
E2 = 5 * C2';
G = R1;
Alo = 0 * A ';
Blo = 0 * B1;
Azo = 0 * A1;
Bzo = 0 * B2;
Clo = 0 * C1;
Dlo = 0 * D1;
Elo = 0 * E1;
Czo = 0 * C2;
Dzo = 0 * D2;
Ezo = 0 * E2;
Go = 0 * G;
sux = sym('su', 'real') * eye(8, 8); seo = sym('se', 'real') * eye(8, 8);
flagproj = 1; flagscale = 1; flagv = 1; eps = 0; flagopt = 1; flagsym = 0;
to = clock;
[Abar, Bbar, Cbar, ux, eo, S, Sm, Alhat, B1hat, A2hat, B2hat, Clhat, D1hat,
E1hat, C2hat, D2hat, E2hat, Ghat, tCPUscale, nFLOPs scale] = SGQME
(A1, B1,
A2, B2, C1, D1, E1, C2, D2, E2, G, flagscale, flagv, eps, flagopt, flagsym,
sux, seo);
tscale = etime(clock, to);
gam = 1; e = le-12; efinal = le-20; flh = 0; f1chom = 0; f1carith = 0;
flagcon = 1; pdef = 0; step = 1;
flpseudos = 0;
pseudofin = 1;
maxnstep = realmax; minnstep = realmin;
largescale ' on'; jacobiangqm = 'on'; linesearch 'quadr cubic';
displaySolver 'off'; displayode 'off'; plotdisplay 'off'
Y = ones(8, 8); Xo = le-
16*eye(8,8); step = 1; pdef = 0; SignXoYo = ones(8, 8);
Aa = zeros(1, 64);
anpplus1 = 1;
[error, X, JacobianX, LX, ux, eo, polyXa, plX, plesil, plpseudos, nst,
stepmin, tCPU, nFLOPS, tCPUscale, nFLOPs scale] = GQMEHOM1
(A1, B1, A2,
B2, C1, D1, E1, C2, D2, E2, G, A1o, B1o, A2o, B2o, C1o, D1o, E1o, C2o,
D2o, E2o, Go, flagscale, flagv, eps, flagopt, flagsym, sux, seo, flagproj, Aa,
anpplus1, gam, pdef, Xo, Yo, SignXoYo, step, e, efinal, flh, f1chom, f1carith,
f1pseudos, pseudofin, maxnstep, minnstep, flagcon, largescale, jacob,
Appendix D

Drivers for the Numerical Examples 481

\% , A2 , B2 , C1 , D1 , E1 , C2 , D2 , E2 , G , A10 , B10 , A20 , B20 , C10 , D10 , E10 , C20 , D20
\%, E20 , G , flagscale , flagv , eps , flagopt , flagsym , sux , seo , flagproj , A
\% , anplusplus , gam , pdef , Xo , Yo , SignXoYo , step , e , efinal , flh , fchom , fl
\% , carith , flpseudos , pseudosfin , maxnsteps , minstep , flagcor , largesca
\% , jacobianqme , linesearch , displaysolver , displayode , plotdisplay
\% y);

tcomputation = etime (clock , t0);
teq = tcomputation - tscale;
natest
NA = [CnX Epfmax Ephmaxfl Erb Erbmax Erbmaxfl NS NSfl];
save example64e12 SignXoYo Xo Yo X error JacobianX LX ux eo S
Sm plX epsilon nsteps nFLOPS nFLOPSscale stepmin tcomputation
teq tCPU tCPUscale NA flagproj flagscale flagv eps flagopt
flagym Aa anplusplus gam e efinal flh flagcor pdef step
maxnsteps minstep fchom flcarith flpseudos pseudosfin
largescale jacobianqme linesearch displaysolver displayode
plotdisplay m
% Driver: Example71.m
% gam=1+1i, flh=1, flagproj=1 or flagproj=0, flagscale=1 or %flagscale=0.
% Yo=[1;1];Xo=1e-8*[1;1];
% Since the easy problem has $2^{(n\times p)}=2^{(2\times 1)}=4$, solutions.
% Each one of these leads to a finite solution or to a solution % at infinity.
% Hence, we have 4 homotopy paths in the Euclidean complex space.
% The 4 solutions (start points of the easy problem) can be % determined via SignXo as follows:
% SignXoYo=[SignXoYo01;SignXoYo02], where:
%
% Solution No. | SignXoYo01 | SignXoYo02
%---------------|----------|------------------|
% 1            | 1        | 1
% 2            | 1        | -1
% 3            | -1       | 1
% 4            | -1       | -1

% From the above:
% SignXoYo=[1;1], SignXoYo=[-1;-1], SignXoYo=[-1;1], % SignXoYo=[1;-1] (all 4 finite real roots)
% real homotopy not work, solutions paths break to complex for % 0<epsilon<1.
clear all;
a=2;
s1=1;
s2=2;
q1=1;
q2=1/4;
A1=[-a 0;0 -a];
B1=1;
A2=[-1 0;0 -1];
B2=a;
C1=[1/5 0;0 0];
D1=[s1 2*s2];
E1=5;
C2=[0 0;0 1/2];
D2=[2*s1 s2];
E2=2;
G=[-q1;-q2];
A1o=0*A1;
B1o=0*B1;
A2o=0*A2;
B2o=0*B2;
C1o=0*C1;
D1o=0*D1;
E1o=0*E1;
C2o=0*C2;
D2o=0*D2;
E2o=0*E2;
Go=0*G;
flagproj=1;flagscale=0;flagv=0;eps=0;flagopt=0;flagsym=0;
sux=[sym('sux1','real');sym('sux21','real')];
seo=[sym('seo1','real');sym('seo21','real')];
Aa=[9.5013e-001 2.3114e-001];
anplug=6.0684e-001;
gam=1+1i;e=1e-14;efinal=1e-20;flh=1;flagcor=1;pdef=0;step=1;
maxsteps=realmax;minstep=realmin;
flchom=0;fcarith=0;
flpseudos=0;pseudosfin=1;
large-scale='on'; jacobianqme='off'; linesearch='cubicpoly';
displaysolver='off'; displayode='off'; plotdisplay='off';
to=clock;
[Abar, Bbar, Cbar, ux, eo, S, Sm, Alhat, B1hat, A2hat, B2hat, C1hat, D1hat, 
B1hat, C2hat, D2hat, E2hat, Ghat, tCFUscale, nFLOPSscale]=SGQM(B1, B2, 
C1, D1, E1, C2, D2, E2, G, flagscale, flagv, eps, flagopt, flagsym, 
sux, seo);
tscale=etime(clock, to);
% save file ux eo;
for k=0:3;
Yo=ones(2,1); Xo=1e-16*ones(2,1);
% k=0, ..., (2^4-1)=0, 2, 3 (4 different combinations)
SignXoYo=-invvec(bitget(k, [1:2])', 2, 1);
for i=1:2
if SignXoYo(i,1)>=0
SignXoYo(i,1)=1;
end
end
fprintf(2, ' [Solution No.]=%i
', k+1);
to=clock;
[error, X, JacobianX, Lx, ux, eo, polyXa, plX, plepsilon, plpseudos, nsteps, 
stepmin, tCPU, nFLOPS, tCFUscale, nFLOPSscale]=GQEHMOM1(A1, B1, A2, 
B2, C1, D1, E1, C2, D2, E2, G, A1o, B1o, A2o, B2o, C1o, D1o, E1o, C2o, D2o, 
E2o, Go, flagscale, flagv, eps, flagopt, flagsym, sux, seo, flagproj, A, anp, 
plus1, gam, pdef, Xo, Yo, SignXoYo, step, e, efinal, flh, flichom, flearith, 
pleps, pseudosfin, nmaxsteps, minstep, flagcor, largescale, jacobiqme, 
linesearch, displaysolver, displayode, plotdisplay); % error, X, JacobianX, Lx, ux, eo, polyXa, plX, plepsilon, plpseudos, nsteps, 
stepmin, tCPU, nFLOPS, tCFUscale, nFLOPSscale]=GQEHMOM11(A1, B1, 
A2, B2, C1, D1, E1, C2, D2, E2, G, A1o, B1o, A2o, B2o, C1o, D1o, E1o, C2o, D2o, 
E2o, Go, flagscale, flagv, eps, flagopt, flagsym, sux, seo, flagproj, A 
% = anpplus1, gam, pdef, Xo, Yo, SignXoYo, step, e, efinal, flh, flichom, flearith, 
% pleps, pseudosfin, nmaxsteps, minstep, flagcor, largescale, jacobiqme, 
linesearch, displaysolver, displayode, plotdisplay); %
tcomputation(k+1:k+1,:) = etime(clock, to);
% nates
NA(k+1:k+1, :)=Cnx Eprfmax Eprfma xl Erb Erbmax Erbmaxfl NS 
NSfl);
tcpu(k+1:k+1, :)=tCPU;
tcpuScale(k+1:k+1, :)=tCFUscale;
teq(k+1:k+1,:) = tcomputation(k+1:k+1, 1:1)-tscale;
[nplX, mplX] = size(plX);
er(k+1:k+1,:) = vecx(error)';
X(k+1:k+1,:) = real(vecx(X))'+li*imag(vecx(X))';
Xo(k+1:k+1,:) = real(vecx(Xo))'+li*imag(vecx(Xo))';
Yo(k+1:k+1,:) = real(vecx(Yo))'+li*imag(vecx(Yo))';
SignXoYo(k+1:k+1,:) = vecx(SignXoYo)';
sizeXJ(k+1:k+1, 1:2) = size(JacobianX);
JX(k+1:k+1,:) = real(vecx(JacobianX))'+li*imag'(vecx(JacobianX));
Lx(k+1:k+1, 1:1) = LX;
uxx(k+1:k+1,:) = vecx(ux)';
se0(k+1:k+1,:) = eo';
plx(k+1:k+1, nplX*mplX) = real(vecx(plX)') + li*imag(vecx(plX)');
sizeplx(k+1:k+1,:) = size(plX);
plep(k+1:k+1, nplX) = plepsilon';
if (flpseundos==0)
plpse(k+1:k+1, nplX) = plpseudos';
else
plpse = plpseudos;
end
nsteps(k+1:k+1,:) = steps;
steps(k+1:k+1,:) = stepmin;
nfLOPS(k+1:k+1,:) = nFLOPS;
nfLOPSscale(k+1:k+1,:) = nFLOPSscale;
end
%n SNP
%n SP
%n SNP
%n SVN
%n SP
%n SVP
save example71svn p signXoYo xo yo x er JX Lx uxx eoo S Sm plx
size plx plep plpse nosteps nfLOPS nfLOPSscale stepmi
tcomputation teq tcpu tcpuscale NA flagproj flagscale flagv eps
flagopt flagsym Aa anpluse1 gme e final flh flagcor pdef step
maxnsteps minstep f3chom f3carith f3pseu f3ps f3psfin
large scale jacobiangqme linesearch displaysolver displayode
plottdisplay Xs float Xs
Appendix D

% Driver: Example72.m
% gam=1+1i, flh=0, flagproj=1 or flagproj=0, flagscale=1 or flagscale=0.
% Yo=[1 1;1 1]; Xo=1e-8*[1 1;1 1];
% Since the easy problem has 2^(n=2^l=2^16=16, solutions.
% Each one of these leads to a finite solution or to a solution at infinity.
% Hence, we have 16 homotopy paths in the Euclidean complex space.
% The 16 solutions (start points of the easy problem) can be determined via SignXo as follows:
% SignXoYo=[SignXoYo11 SignXoYo12;SignXoYo21 SignXoYo22], where:
% Solution No. SignXoYo11 SignXoYo12 SignXoYo21 SignXoYo22
% 1 1 1 1 1
% 2 1 1 1 1
% 3 1 1 1 1
% 4 1 1 1 1
% 5 1 1 1 1
% 6 1 1 1 1
% 7 1 1 1 1
% 8 1 1 1 1
% 9 1 1 1 1
% 10 1 1 1 1
% 11 1 -1 -1 -1
% 12 1 -1 -1 -1
% 13 1 -1 -1 -1
% 14 1 -1 -1 -1
% 15 1 -1 -1 -1
% 13 1 -1 -1 -1
% From the above:
% SignXoYo=[1 1;1 1], SignXoYo=[-1 -1;1 -1], SignXoYo=[-1 1;1 -1], SignXoYo=[1 -1;1 1] (all 4 finite roots)
% For the other 12 possible combinations of +1,-1 in SignXo, the solutions are at infinity.
% The solutions at infinity computed with flagproj=0, and flagscale=0 or flagscale=1.
clear all;
A1=3*eye(2,2);
B1=0.1*eye(2,2);
A2=0.7*eye(2,2);
B2=eye(2,2);
C1=0.4*eye(2,2);
D1=10*eye(2,2);
E1=0.1*eye(2,2);
C2=0.6*eye(2,2);
D2=5*eye(2,2);
E2=0.2*eye(2,2);
G=[-8 -12 -18 -26];
Alo=0*A1;
Blo=0*B1;
A2o=0*A2;
B2o=0*B2;
C1o=0*C1;
D1o=0*D1;
E1o=0*E1;
C2o=0*C2;
D2o=0*D2;
E2o=0*E2;
Go=0*G;
flagproj=0;flascale=0;flav=1;eps=0;flaopt=0;flasym=0;
sux=[sym('sux1','real')
sym('sux2','real') sym('sux21','real') sym('sux22','real')];
seо=[sym('seо1','real') sym('seо2','real') sym('seо31','real') sym('seо41','real')]
Aa=[3.5287e-001 8.1317e-001 9.8613e-003 1.3889e-001];
anplusl=2.0277e-001i;
gam=1+l;j;e=1e-6;eфинаl=1e-16;flh=1;flагcor=1;пdеf=0;step=1;
maxnsteps=realmax:minstep=1e-60;
flчом=0;flкарTHE=0;
fl псевд通告=0;psевдосфп=1;
lаргешcale=1;
jaсобиангкме='оф';линеарhар='кубикполи';
dисплалвр=оф;дисплонд=оф;
то=clock,
[Abar,Bbar,Cbar,ux,oe,S,Sm,Ahat,Bhat,A2hat,B2hat,A1hat,D1hat,
B1hat,B2hat,Dhat,Ehat,Ghat,тCPUscale,nFLOPSscale]=SGQMEHOM1(A1,B1
,A2,B2,C1,D1,E1,C2,D2,E2,G,Alo,Bio,A2o,B2o,C1o,D1o,Elo,C2o,D2o,E2o
,Go,flаскаглв,flась,eps,flаопт,flасим,sux,seо,flагр,flagproj,Aa,ан
plusl,гам,пдеф,ux,ео,SignXoYo,step,e,ефинаl,flh,flчом,flкарTHE,
плсевд通告,псевдосфп,макснепст,мипшет,flагкор,ларгешcale,jaсобиа
гкме,линеарhар,дисплалвр,дисплонд,плотдиспл]

for k=0:15;
Yo=ones(2,2);Xo=1e-0*ones(2,2);
ifr=0,...,16 different combinations)
SignXoYo=инvecс(bitget(k,[1:4]),2,2);
for i=1:2
    for j=1:2
        if SignXoYo(i,j)==0
            SignXoYo(i,j)=1;
        end
    end
end
fprintf(2,'[Solution No.]=%i\n',k+1);
tо=clock,
[ошибка,Х,jaсобианХ,lx,ux,о,polYХ,plX,плсилон,плсевд通告,нстепн,
тCPU,tCPUscale,nFLOPSscale]=SGQMEHOM1(A1,B1,A2
,B2,C1,D1,E1,C2,D2,E2,G,Alo,Bio,A2o,B2o,C1o,D1o,Elo,C2o,D2o,E2o
,Go,flаскаглв,flась,eps,flаопт,flасим,sux,seо,flагр,flagproj,Aa,ан
plusl,гам,пдеф,ux,ео,SignXoYo,step,e,ефинаl,flh,flчом,flкарTHE,
плсевд通告,псевдосфп,макснепст,мипшет,flагкор,ларгешcale,jaсобиа
гкме,линеарhар,дисплалвр,дисплонд,плотдиспл]

for k+1:k+1,:)=etime(cloсk,tо);
nатест
NA(k+1:k+1,1:8)=[CnX Eпрфмакс Eпрфмакс1 Erb Erbmax Erbmaxfl NS
NSfl1];
tcpu(k+1:k+1,:)=tCPU;
tcpusсale(k+1:k+1,:)=tCPUscale;
tвq(k+1:k+1,:)=tвcomputation(k+1:k+1,1:1):tscale;
[nplX,mplX]=size(plX);
er(k+1:k+1,:)=vecс(ошибка)';
x(k+1:k+1,:)=real(veсс(Х))'+1ili*image(veсс(Х))';
xo(k+1:k+1,:)=real(veсc(Хо))'+1ili*image(veсc(Хo))';
yp(k+1:k+1,:)=real(veсc(Уo))'+1ili*image(veсc(Уo))';
signXoYo(k+1:k+1,:)=veсc(SignXoYo)';
sizeJX(k+1:k+1,1:2)=size(JacobianX);
JX(k+1:k+1,:)=real(vec(JacobianX))+li*imag(vec(JacobianX));
LX(k+1:k+1,1:1)=LX;
uxx(k+1:k+1,:)=vec(ux);
eoo(k+1:k+1,:)=eo';
plx(k+1:k+1,1:npLX*mplX)=real(vec(plX))'+li*imag(vec(plX))';
sizeplx(k+1:k+1,:)=size(plX);
plep(k+1:k+1,1:npLX)=plepsilon';
if (fplseu=0)
    plpse(k+1:k+1,1:npLX)=plpseu;
else
    plpse=plpseu;
end
nosteps(k+1:k+1,:)=nsteps;
stepmi(k+1:k+1,:)=stepmin;
fLOPS(k+1:k+1,:)=fLOPS;
fLOPSscale(k+1:k+1,:)=fLOPSscale;
end
%nsnp
%snp
%svnp
%sp
%svp

%Symbolic computation of the system's solutions.
The solutions are Xs or equivalently Xsfloat.
x11= sym('x11'); x12= sym('x12'); x21= sym('x21'); x22= sym('x22');
[x11s,x12s,x21s,x22s]=solve(x11*x11+x12*x21+x11-
8,x11*x12+x12*x22+x12-12,x21*x11+x22*x21+x21-
18,x21*x12+x22*x22+x22-26,'x11','x12','x21','x22');
Xs=[x11s x12s x21s x22s];
%Conversion to float double precision number.
x11=double(x11s);
x12=double(x12s);
x21=double(x21s);
x22=double(x22s);
Xsfloat=[x11 x12 x21 x22];
save example72sp signxoyo xo yo x er JX Lx uxx eoo S Sm plx
sizeplx piep plpse nosteps nfLOPS nfLOPSscale stepmi
tcomputation tech tcpuscale na flagproj flagscale flagv eps
flagopt flagsym Aa anplus1 gama eefinal flh flagcor pdef step
maxnsteps minstep fchom fcarith fipseu fipseu scale fipseu
large scale jacobian gm linesearch display solver display ode
plot display Xs Xsfloat
% Driver: Example73.m

clear all;
A = [17.6 1.28 2.89; 1.28 0.824 0.413; 2.89 0.413 0.725];
B = [7.66 2.45 2.1; 0.23 1.04 1.04; 0.6 0.756 0.658];
Q = [121 18.9 15.9; 2.7 0.145 11.9; 3.64 15.5];
A1 = B;
B1 = 0.4 * eye(3,3);
A2 = 0.6 * B;
B2 = eye(3,3);
C1 = 0.4 * A;
D1 = 10 * eye(3,3);
E1 = 0.1 * eye(3,3);
C2 = 0.6 * A;
D2 = 5 * eye(3,3);
E2 = 0.2 * eye(3,3);
G = Q;
Alo = 0 * A1;
Blo = 0 * B1;
A2o = 0 * A2;
B2o = 0 * B2;
Clo = 0 * C1;
Dlo = 0 * D1;
Elo = 0 * E1;
C2o = 0 * C2;
D2o = 0 * D2;
E2o = 0 * E2;
Go = 0 * G;
flagproj = 1; flagscale = 0; flagv = 0; flagopt = 0; flagsym = 0;
A1'' = 0 * A';
sux = [sym('sux11', 'real'), sym('sux12', 'real')]
sym('sux13', 'real'), sym('sux21', 'real'), sym('sux22', 'real')
sym('sux23', 'real'), sym('sux31', 'real'), sym('sux32', 'real')
sym('sux33', 'real');
se0 = [sym('se01', 'real'), sym('se021', 'real'), sym('se031', 'real')]
sym('se041', 'real'), sym('se051', 'real'), sym('se061', 'real')
sym('se071', 'real'), sym('se081', 'real'), sym('se091', 'real');
gam = -1 - 1; e = 10^-6; efinal = 10^-20; flh = 0; flagcor = 1; pdef = 0; step = 1;
maxsteps = realmax; minstep = realmin;
flchom = 0; flcarith = 0;
flpseudos = 0; flpseudosfin = 1;
largescale = 'on'; jacobianqme = 'off'; linesearch = 'cubico poly';
displaysolver = 'off'; displayode = 'off'; plotdisplay = 'off';
to = clock;
[tAbar, Ebar, Cbar, UX, EO, S, Sm, Alhat, Blhat, A2hat, B2hat, Clhat, D1hat, E1hat, C2hat, D2hat, E2hat, Ghat, tCPUscale, nFLOPSscale] = SGQME(A1, B1, A2, B2, C1, D1, E1, C2, D2, E2, G, flagscale, flagv, eps, flagopt, flagsym, sux, se0);
tscale = etime(clock, to);
Yo = ones(3,3); Xo = 0*ones(3,3); SignXoYo = ones(3,3);
Aa = [8.3850e-001 5.6807e-001 3.7041e-001; 7.0274e-001 5.4657e-001 4.4488e-001; 6.9457e-001 6.2131e-001 7.9482e-001];
ansplus1 = norm(Aa);
[error, X, JacobianX, LX, UX, EO, POlyXa, plX, plepsilon, plpseudos, nste, ps, stepmin, tCPU, nFLOPS, tCPUscale, nFLOPScale] = GQEMOM1(A1, B1, A2, B2, C1, D1, E1, C2, D2, E2, G, Alho, B1ho, A2o, B2o, C1o, D1o, E1o, C2o, D2o, E2o, Go, flagscale, flagv, eps, flagopt, flagsym, sux, se0, flagproj, Aa, anplus1, gam, pdef, Xo, Yo, SignXoYo, step, e, efinal, flh, flchom, flcarith, flpseudos, pseudosfin, maxnsteps, minstep, flagcor, largescale, jacobianqme, linesolversolver, displayode, plotdisplay);
% [error, X, JacobianX, LX, UX, EO, POlyXa, plX, plepsilon, plpseudos, nste,
% eps, stepmin, tCPU, nFLOPS, tCPUscale, nFLOPSscale\] = GQMEHOM11 (A1, B1 \\
% , A2, B2, C1, D1, E1, C2, D2, E2, G, A1o, B1o, A2o, B2o, C1o, D1o, E1o, C2o, D2o \\
% , E2o, Go, flagscale, flagv, eps, flagopt, flagsym, sux, se0, flagproj, A \\
% a, anplus1, gam, pdef, Xo, Yo, SignXoYo, step, e, efinal, flh, flchom, fl \\
% carith, fipseudos, pseudosfin, maxnsteps, minstep, flagcor, largesca \\
% le, jacobiangqme, linesearch, display solver, displayode, plotdisplay \\
% y); \\
\text{natest} \\
\text{NA} = [CnX Eprfmax Eprfmax1 Erb Erbmax Erbmaxfl NS NSfl]; \\
\text{save example73e6snphgam} \text{b SignXoYo Xo Yo X error JacobianX LX ux} \\
\text{eo S m pLX ples} \text{sion nsteps nFLOPS nFLOPSscale stepmin tCPU} \\
\text{tCPUscale NA flagproj flagscale flagv eps flagopt flagsym Aa} \\
anpplus1 gam e efinal flh flagcor pdef step maxnsteps minstep \\
flchom flicarith flpseudos pseudosfin largesca jacobiangqme \\
linesearch displaysolver displayode plotdisplay
%Driver: Example74.m
% a14*x1+a15*x2+a11*x1^2+a12*x2^2+x1*a13*a16=0
% a24*x1+a25*x2+a21*x1^2+a22*x2^2+x2*a23*a26=0
%Yo=[1;1];Xo=le-8*[1;1], flh=1, flagproj=0,
%flagscale=0 or flagscale=1 and flafv=0 or flagv=1 and
%flagopt=0, flagsym=0.
%Since the easy problem has 2^(nxp)=2^(2x1)=4, solutions.
%Each one of these leads to a finite solution or to a solution
%at infinity.
%Hence, we have 4 homotopy paths in the Euclidean complex
space.
The 4 solutions (start points of the easy problem) can be
determined via SignXo as follows:
%SignXoYo=[SignXoYo11;SignXoYo21], where:

%Solution No. SignXoYo11 SignXoYo21
% 1  1   -1
% 2  1   1
% 3 -1   -1
% 4 -1   1

%From the above:
%SignXoYo=[1;1], (one complex root at infinity)
%SignXoYo=[-1;-1], (one complex root essentially near infinity)
%SignXoYo=[1;-1], (two opposite real roots)
a11=-0.229242874000584e-3;a12=0.239288083622779e-14;
a13=-0.273493390049709e+2;a14=-0.553727730639579e+4;
a15=0.276995297192016e+7;a16=0.142521110826190e+2;
a21=-0.719432423708819e-4;a22=0.239288083622779e-14;
a23=-0.273494828914557e+2;a24=0.553727730565810e+4;
a25=-0.276995297192090e+7;a26=0.141753439263839e+2;
A1=[a14 0;a24 0];
B1=1;
A2=[0 a15;0 a25];
B2=1;
C1=[a11 -a12;a21 -a22];
D1=[1 -1];
E1=1;
C2=[0 a11+a12+a13 0 a21+a22+a23];
D2=[1 0];
E2=1;
G=[a16 a26];
A0=0*A1;
B0=0*B1;
A0=0*A1;
B0=0*B2;
C0=0*C1;
D0=0*D1;
E0=0*E1;
C0=0*C2;
D0=0*D2;
E0=0*E2;
G=0*G;
flagproj=0;flagscale=0;flagv=1;eps=0;flagopt=0;flagsym=0;
sux=[sym('sux1', 'real');sym('sux21', 'real')];
seq=[sym('seq1', 'real');sym('seq2', 'real')];
gam=1+i;e=le-6;efinal=le-50;flh=1;flchom=0;flcarith=0;
maxsteps=realmax;minstep=le-60;
flagcor=1;pdef=0;step=1;
flpseudos=0;pseudosfin=1;
largescale='on';jacobiangqme='off';linesearch='cubicpoly';
Appendix D Drivers for the Numerical Examples  491

```matlab
Aa=[3.6775e-001 6.2080e-001];
anplus1=norm(Aa);
to=clock;
[Abar,Bbar,Cbar,ux,eo,S,Sm,Alhat,B1hat,Al2hat,B2hat,C1hat,D1hat,
E1hat,D2hat,E2hat,St,tfCPUscale,tfFLOPSScale]=SQME(A1,B1
,A2,B2,C1,D1,E1,C2,D2,E2,G,flagscale,flagv,eps,flagopt,flagsym,
sux,seo);
tscale=etime(clock,to);
%save d file ux eo;
for k=0:3;
Yo=ones(2,1);Xo=1e-8*ones(2,1);
%k=0, ..., (2^4-1)=0,2,3 (4 different combinations)
SignXoYo=-invvec(bitget(k,[1:2])',2,1);
for i=1:2
if SignXoYo(i,1)<>0
SignXoYo(i,1)=1;
end
end
fprintf(2,'[Solution No.]=%i
',k+1);
to=clock;
[error,X,JacobianX,LX,ux,eo,polyXa,plX,plepsilon,plpseudo,nstep,
step,tfCPU,nFLOPS,tfCPUscale,nFLOPSscale]=GQMEHOM1(A1,B1,A2
,B2,C1,D1,E1,C2,D2,E2,G,A1o,B1o,A2o,B2o,C1o,D1o,E1o,C2o,D2o,E2o
,flagscale,flagv,eps,flagopt,flagsym,sux,seo,flagproj,A,a,
anplus1,gam,pdef,Xo,Yo,SignXoYo,step,e,efinal,flag,flagopt,
flag.sym,flag proj,flag solver,plotdisplay,sdisplay);%
%error,X,JacobianX,LX,ux,eo,polyXa,plX,plepsilon,plpseudo,nst
%eps,step,tfCPU,nFLOPS,tfCPUscale,nFLOPSscale]=GQMEHOM1(A1,B1
%A2,B2,C1,D1,E1,C2,D2,E2,G,A1o,B1o,A2o,B2o,C1o,D1o,E1o,C2o,D2o
%E2o,flagscale,flagv,eps,flagopt,flagsym,sux,seo,flagproj,A
%a,anplus1,gam,pdef,Xo,Yo,SignXoYo,step,e,efinal,flag,flagopt,
%flag.sym,flag proj,flag solver,plotdisplay,sdisplay);%
tcpu(k+1:k+1,:)=tfCPU;
tcpuScale(k+1:k+1,:)=tfCPUscale;
teq(k+1:k+1,:)=tcpu(k+1:k+1,:)-tscale;
[npIX,mlPIX]=size(plX);
er(k+1:k+1,:)=vec(error)';
x(k+1:k+1,:)='real(vec(X))'+li*imag(vec(X))';
xo(k+1:k+1,:)='real(vec(Xo))'+li*imag(vec(Xo))';
yo(k+1:k+1,:)='real(vec(Yo))'+li*imag(vec(Yo))';
SignXoYo(k+1:k+1,:)='vec(SignXoYo)';
sizeJX(k+1:k+1,1:2)=size(JacobianX);
JX(k+1:k+1,:)='real(vec(JacobianX))'+li*imag(vec(JacobianX))';
Lx(k+1:k+1,1:1)='LX';
uxx(k+1:k+1,:)='vec(ux)';
eo(k+1:k+1,:)='eo';
plx(k+1:k+1,1:mlPIX*mlPIX)=real(vec(plX))'+li*imag(vec(plX))';
sizePlx(k+1:k+1,:)='size(plX)';
plpse(k+1:k+1,1:mlPIX)=plpseudo';
if (flpseduos<>0)
plpse(k+1:k+1,1:mlPIX)=plpseudo';
else
plpse=plpseudo;
```
end
nsteps(k+1:k+1,:) = nsteps;
stepmi(k+1:k+1,:) = stepmin;
fLOPS(k+1:k+1,:) = nFLOPS;
fLOPSscale(k+1:k+1,:) = nFLOPSscale;
end

save example74sp signXoYo xo yo x er JX Lx uxx eoo S Sm plx
sizeplx plep plpse nsteps nFLOPS nFLOPSscale stepmi
tcomputation teq tcpu tcpuscale NA flagproj flagscale flagv eps
flagopt flagsym Aa anplus1 gam e efinal flh flagcor pdef step
maxnsteps minstep flchom flcarith flpseudos pseudosfin
largescale jacobiangqme linesearch displaysolver displayode
plotdisplay
% Driver: Example75.m
% Lorenz attractor equilibrium points.
% SignXoYo=
% [1;1;1]: X=[6*2^(1/2);6*2^(1/2);27]
% [-1;1;1]: X=[-6*2^(1/2);-6*2^(1/2);27]
% [-1;-1;1]: X=[0;0;0]
% [1;-1;1], [1;-1;-1], [-1;1;-1], [-1;-1;-1], [1;1;-1]: X at infinity

clear;
A1=0.4*[-10 10 0;28 -1 0;0 0 -(8/3)];
B1=1;
A2=[-10 10 0;28 -1 0;0 0 -(8/3)];
B2=0.6;
C1=[0 0;1 0 0;0 0 0];
D1=[0 0 0];
E1=1;
C2=[0 0 0;0 0 0;1 0 0);
D2=0.*C2;
E2=0.*E2;
G=[0 0 0];
Alo=0.*A1;
Blo=0.*B1;
A2o=0.*A1;
B2o=0.*B2;
C1o=0.*C1;
D1o=0.*D1;
E1o=0.*E1;
C2o=0.*C2;
D2o=0.*D2;
E2o=0.*E2;
Glo=0.*G;
flagproj=0;flagscale=1;flagv=1;eps=0;flagopt=0;flagsym=0;
sux=[sym('sux11','real');sym('sux21','real');sym('sux31','real')];
seo=[sym('seo11','real');sym('seo21','real');sym('seo31','real')];
gam=1+i;e=1e-8;efinal=1e-20;flh=1;flchom=0;flcarith=0;
maxsteps=realmax;minstep=realmin;
flagcor=1;pdef=0;step=1;
flagpseudos=0;pseudosfin=1;
largescale='on';jacobianqme='off';linesearch='cubicpoly';
displaysolver='off';displayode='off';plotdisplay='off';
Aa=[2.6775e-001 3.2080e-001 9.2670e-001];
anplu1=0.54;
to=clock;
[Abar,Bbar,Cbar,ux,eo,S,Sm,A1hat,B1hat,A2hat,B2hat,C1hat,D1hat,
E1hat,C2hat,D2hat,E2hat,Ghat,tCPUscale,nFLOPSscale]=SGQME(A1,B1,
A2,B2,C1,D1,E1,C2,D2,E2,G,flagv,eps,flagopt,flagsym,
sux,seo);
tscale=etime(clock,to);
for k=0:7;
Yo=ones(3,1);Xo=1e-0*ones(3,1);
% k=0,...,(2^3-1)=0,...,7 (8 different combinations)
SignXoYo=-invvec(bitget(k,[1:3]'),3,1);
for i=1:3
  if SignXoYo(i,1)==0
    SignXoYo(i,1)=1;
  end
end
fprintf(2,'[Solution No.]=%i\n',k+1);
to=clock;
[error, X, JacobianX, LX, ux, eo, polyXa, pX, ps, nsteps, tCPU, nFLOPS, tCPU_scale, nFLOPS_scale] = GQM_EHOM1 (A1, B1, A2, B2, C1, D1, E1, C2, D2, E2, G, Alo, B1o, Ao2, B2o, Cl0, D10, E10, C2o, D2o, E2o, Go, flagscale, flagv, eps, flagopt, flagsym, sux, seo, flagproj, Aa, anplusl, gam, pdef, Xo, Yo, SignXoYo, step, e, efinal, flh, flchom, flcarith, fpseudos, pseudosfin, maxnsteps, minstep, flagcor, largescale, jacobianX, linesearch, displaysolver, displayode, plotdisplay);

% [error, X, JacobianX, LX, ux, eo, polyXa, pX, ps, nsteps, tCPU, nFLOPS, tCPU_scale, nFLOPS_scale] = GQM_EHOM1 (A1, B1, A2, B2, C1, D1, E1, C2, D2, E2, G, Alo, B1o, Ao2, B2o, Cl0, D10, E10, C2o, D2o, E2o, Go, flagscale, flagv, eps, flagopt, flagsym, sux, seo, flagproj, Aa, anplusl, gam, pdef, Xo, Yo, SignXoYo, step, e, efinal, flh, flchom, flcarith, fpseudos, pseudosfin, maxnsteps, minstep, flagcor, largescale, jacobianX, linesearch, displaysolver, displayode, plotdisplay);

tcomputation (k+1:k+1,:) = etime (clock, to);

n = test

NA (k+1:k+1,1:8) = [CnX Epmax Erpmax fl Erb Erbmax Erbmaxfl NS NSFfl];

tcpu (k+1:k+1,:) = tCPU;

tcupscale (k+1:k+1,:) = tCPU_scale;

teq (k+1,:) = tcomputation (k+1:k+1,1:1) - tscale;

[npX, mpX] = size (pLX);

er (k+1,:) = vecc (error)';

x (k+1,:) = real (vecc (X))' + li * imag (vecc (X))';

xo (k+1,:) = real (vecc (Xo))' + li * imag (vecc (Xo))';

yo (k+1,:) = real (vecc (Yo))' + li * imag (vecc (Yo))';

signXoYo (k+1,:) = vecc (SignXoYo)';

sizeX (k+1,:) = size (jacobianX);

JX (k+1,:) = real (vecc (jacobianX))' + li * imag (vecc (jacobianX))';

LX (k+1,:) = LX;

uxx (k+1,:) = vecc (ux)';

eo (k+1,:) = eo';

plX (k+1,:) = real (vecc (pLX))' + li * imag (vecc (pLX))';

sizeplX (k+1,:) = size (pLX);

plep (k+1,:) = plepseudos';

if (fpseudos == 0)
    plpse (k+1,:) = plpseudos';
else
    plpse = plpseudos;
end

nsteps (k+1,:) = nsteps;

stepmi (k+1,:) = stepmin;

nFLOPS (k+1,:) = nFLOPS;

nFLOPS_scale (k+1,:) = nFLOPS_scale;

end

Symbolic computation of the system's solutions.

The solutions pairs are the elements in each row of vector Xs or equivalently X.

x1 = sym ('x1'); x2 = sym ('x2'); x3 = sym ('x3');

[x1s, x2s, x3s] = solve (-10*x1+10*x2, 28*x1-x2-x1*x3,- (8/3)*x3+x1*x2, 'x1', 'x2', 'x3');

Xs = [x1s x2s x3s];

Conversion to float double precision number.

x1 = double (x1s);

x2 = double (x2s);

x3 = double (x3s);

Xsfloaat = [x1 x2 x3];

% np

% sp
save example75e6nsp signXoYo xo yo x er JX Lx uxx eco S Sm plx
sizeplx plep plpse nosteps nfLOPS nfLOPSscale stepmi
tcomputation teq tcpu tcpuscale NA flagproj flagscale flagv eps
flagopt flagsym Aa anaplus1 gam e sfinal flh flagcor pdef step
maxnsteps minstep flchom flcarith flpseudos pseudosfin
largescale jacobiangqs linesearch displaysolver displayode
plotdisplay Xsfloat Xs