Robust Pole Assignment by Output Feedback Using Optimization Methods

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

by

Myungho Oh, MSc (U.S.A)
Department of Engineering
University of Leicester
Leicester LE1 7RH
United Kingdom

March 1993
To my mother
who has prayed for me
one thousand and ninety five days and nights,
and would never be forgotten, this work is dedicated.
Acknowledgements

I would like to thank Dr. Spurgeon for her supervision and help throughout the research and writing of this thesis. I am greatly indebted to Dr. Da-Wei Gu for his many discussions and valuable suggestions during the research. I wish to express my sincere appreciation to Professor Postlethwaite; but for his encouragement this work would not have been finished. In addition, particular thanks are due to Professor Park, Chan Mo who has encouraged me by sending e-mails every week for 3 years.
Statement of Originality

The accompanying thesis submitted for the degree of Doctor of Philosophy entitled Robust Pole Assignment by Output Feedback Using Optimization Methods is based on work conducted by the author in the Department of Engineering of the University of Leicester mainly during the period between April 1991 and March 1993. All the work recorded in this thesis is original unless otherwise acknowledged in the text or by references. None of the work has been submitted for any degree in this or any other University.

March 23, 1993

[Signature]

Myungho Oh
Lariat Logan applies bang-bang control for eigenvalue placement.

IEEE Control Systems, December 1992
Abstract

A robust output feedback pole assignment method, which seeks to achieve a robust solution in the sense that the assigned poles are as insensitive as possible to perturbations in the system parameters, is studied. In particular, this work is concerned with pole assignment in a specified region rather than assignment to exact positions, whereby the freedom to obtain a robust solution may be realized. The robust output feedback pole assignment problem is formulated as an optimization problem with a special structure in matrix form. Efficient optimization methods and numerical algorithms for solving such a problem are proposed by introducing a concept of the derivative of a matrix valued function. The homotopy method, which is known as a globally convergent method, is applied to solve the robust output feedback pole assignment problem to overcome possible difficulties with the choice of feasible starting point. A new algorithm based on the homotopy approach for solving the pole assignment problem is proposed. Numerical examples of the robust pole assignment problem demonstrate how the homotopy algorithm globally converges to optimal solutions regardless of initial starting points with an appropriately defined homotopy mapping.

The proposed algorithms are illustrated using an aircraft case study. It is seen that the controllers obtained using robust pole assignment methods yield the robust flight control and maintain the closed-loop system properties closer to the nominal ones. They are shown to be more robust than those obtained by an alternative direct pole assignment method which is frequently used to develop aircraft control
strategies without attempting to optimize any robustness criterion. Indeed, the robust output feedback pole assignment method proposed in this study is a method which can be applied for control system design to achieve one important design objective, robustness.
Contents

Abstract i

1 Introduction 1

2 Design of Control Systems 7
2.1 Introduction ......................................................... 7
2.2 Background to Control Systems .......................... 10
    2.2.1 Classifications of Control Systems .................. 10
    2.2.2 History of Feedback Control Systems .......... 11
2.3 Feedback Control Systems Design ..................... 15
    2.3.1 Advantage of Feedback Control Systems ........ 15
    2.3.2 Control System Design Process .................... 16
    2.3.3 Design Objectives ....................................... 20
    2.3.4 Performance Specifications ......................... 26
    2.3.5 Objective Functions in Mathematical Models .... 28
2.4 Concluding Remarks ........................................... 29

3 Pole Assignment Methods 30
3.1 Introduction ....................................................... 30
3.2 Classification of Output Feedback Pole Assignment Methods . 33
6.2.1 Concept of Homotopy ............................................. 103
6.2.2 Imbedding Methods .............................................. 105
6.3 Homotopy Algorithms .............................................. 109
6.3.1 Existing Algorithms ............................................. 109
6.3.2 Proposed Algorithms for Solving Typical Optimization
  Problems ................................................................. 120
6.3.3 The Application of the Proposed Algorithms to Robust Pole
  Assignment ............................................................. 124
6.4 Concluding Remark ................................................ 142

7 Case Study - Pole Assignment for an Aircraft Control System Design 143
  7.1 Introduction .......................................................... 143
  7.2 Purposes of Flight Control Systems ......................... 146
  7.3 Mathematical Model .............................................. 147
  7.4 Eigenstructure Assignment ...................................... 149
  7.5 Numerical Example .............................................. 151
  7.6 Concluding Remarks ............................................ 159

8 Conclusion and Further Research 189

References 195
Statement of Originality

The accompanying thesis submitted for the degree of Doctor of Philosophy entitled Robust Pole Assignment by Output Feedback Using Optimization Methods is based on work conducted by the author in the Department of Engineering of the University of Leicester mainly during the period between April 1991 and March 1993. All the work recorded in this thesis is original unless otherwise acknowledged in the text or by references. None of the work has been submitted for any degree in this or any other University.

Myungho Oh

March 23, 1993
Out of Control

Lariat Logan applies bang-bang control for eigenvalue placement.

IEEE Control Systems, December 1992
Chapter 1

Introduction

Control system design involves the choice of control system structure and the selection of suitable parameters to achieve specified design objectives effectively and efficiently. A control system may be designed in terms of an open-loop or a closed-loop control system. An open-loop control system is one in which the control action is independent of the output. A closed-loop control system is one in which the control action is dependent on the output. Though an open-loop control system may be much simpler and more economical, it is less applicable in practice where unexpected disturbances, parameter variations and uncertainties in the system's dynamic are inevitable. Therefore, a closed-loop control system, that is a feedback control system, is more practical. The basic purpose of a feedback control system design is to obtain a suitable control system which is stable, as minimally sensitive to perturbations in the system parameters as possible, may suppress the effect of undesirable disturbances and achieves a minimum steady-state error for input commands. However, the presence of model uncertainties and/or parameter uncertainties may not allow the achievement of such feedback properties satisfactorily without choosing appropriate control system components to perform specified design objectives. It is well known that eigenvalues and associated eigenvectors
play an important role yielding the dynamical characteristics of a given system. The eigenvalue will determine the time-domain characteristic of a mode whilst the eigenvector determines the shape of that mode. By choosing ideal entries within a given eigenvector the contribution of a given state to a particular mode may be prescribed. The pole configuration is directly related to the stability, steady state accuracy and satisfactory transient response of the system. To ensure satisfactory transient response of the system, poles must be located at suitable positions.

The pole assignment problem may be subdivided into static feedback pole assignment problems and dynamic feedback pole assignment problems. It may be further classified into pole assignment by full state feedback and pole assignment by output feedback, depending upon whether it is based on complete state information or incomplete state information. The case of output feedback pole assignment where the measured output only is used by the controller is more practical. However, there are some restrictions on pole assignability. Besides the controllability and observability assumptions, a dimensionality requirement is also essential. That is, the number of inputs plus the number of outputs must not be less than the dimension of the state variable in order to guarantee that poles are assignable arbitrarily close to those desired. Although the location of the poles may characterize the stability of the underlying system, an arbitrary choice of closed-loop poles may result in a poor controller design. The control effort required to move open-loop poles by feedback depends upon the distances through which they have to be moved. Moreover, uncertainties of model, parameter and/or structure may affect the locations of poles, or the assigned values may not necessarily be the real desired values because the model itself may not represent the underlying real system appropriately. A control system needs to be designed so as to remain stable in spite of such uncertain environment. The closed-loop eigenvalues must be as insensitive as possible to perturbations in the system parameters. A robust pole assignment concept is thus called for. Moreover, robust pole assignment in a specified region rather than to exact positions is
seen to be more realistic. Methods which design output feedback control schemes for exact pole placement may achieve the design objectives for some nominal linear system representation only if these assigned closed-loop poles vary significantly with system changes. In fact, poles assigned to an insensitive location within a region may not move further away from the position than a pole assigned to an exact but sensitive location. A proper choice of robustness measurement is crucial for formulating a robust pole assignment problem. Though it may be measured by several different ways, the condition number of the closed-loop system matrix may be one reasonable criterion to evaluate the system robustness since it indicates the maximum effect of the perturbations in the system matrix and thus it may measure the sensitivity of the eigenvalues of the matrix to variations in its entries.

The robust pole assignment problem may be formulated as an optimization problem in terms of a scalar valued objective function with matrix arguments and matrix valued constraints with matrix arguments. However, because of the special structure of the constraints, the typical nonlinear optimization techniques may have some difficulties including the evaluation of the Jacobian of the constraints. Some efficient methods thus need to be developed. An appropriate optimization method using matrix valued functions will solve the robust pole assignment problem with a set of constraints of matrix valued functions. Most optimization algorithms to solve continuous nonlinear programming problem involve solving a set of Kuhn-Tucker type equations which produce stationary points for optimal solution. Most of such algorithms have a common drawback in that global convergence is not guaranteed; they may not converge to a feasible and/or an optimal solution if the chosen starting points are not sufficiently close to that solution. The feasibility of the initial starting point may also affect the efficiency of the algorithm. Selection of a suitable starting point for the robust output feedback pole assignment problem has proved difficult using such classical optimization algorithms. Though there are some methods to reach a feasible solution starting from an infeasible one or some rules to choose
It is well known that if a suitable homotopy mapping can be established for a given nonlinear programming problem, the homotopy methods will provide an elegant theoretical framework to alleviate the problems associated with starting point selection. The homotopy method for optimization, also referred to as a parametric optimization method, pathfollowing method or continuation method, is a branch of mathematical programming. The method may be effectively applied to solve engineering problems in which the data in the model depend on parameters whose variations may be controlled by a system designer or exogenously. It has been used to find a solution to a given system by systematically varying an appropriately defined artificial parameter or parameters. This idea is applied to solve constrained optimization problems by suitable manipulation of the active constraint set corresponding to the parameter variations. It has the advantage of global convergence and is efficient for solving ill-conditioned optimization problems. Though any appropriate optimization method may be applicable to solve control system design problems, more effective methods may be required to solve the robust output feedback pole assignment problem formulated in this research.

Chapter Two presents some of background material for control systems which includes discussion of the basic concepts and the history of feedback control. The control system design process consists of five phases; Formulation of the problem, Construction of a mathematical model, Deriving a solution, Analysis and synthesis, Implementation. A description of the design objective includes concepts of stability, robustness and robust stability. Performance specifications are described in terms of frequency domain and time domain.

In Chapter Three, output feedback pole assignment methods are classified into four categories; direct eigenstructure assignment, parametric eigenstructure assignment, robust eigenstructure assignment and perturbation analysis. Some basic
properties relating to pole assignability are reviewed. A representative selection of the algorithms based on direct eigenstructure assignment methods and robust eigenstructure assignment methods are discussed. A selection of test problems, both for well-conditioned and ill-conditioned problems, is used to compare the abilities of the existing algorithms.

Chapter Four will present a measurement of robustness in terms of the condition number. Some simple examples demonstrate how the condition number for a given matrix relates to the sensitivity of the parameter variations to the matrix. It will be seen that the condition number will indeed measure the sensitivity of the eigenvalues of the matrix to variations in its entries. The robust output pole assignment problem is formulated as an optimization problem with a special structure whereby the objective function is described by a scalar valued function of matrix arguments and the constraints are a set of matrix valued functions. Necessary and sufficient conditions for existence of feasible solutions for this problem are derived.

Chapter Five will develop efficient optimization methods for solving problems with matrix valued constraints which will effectively solve robust pole assignment problems. A concept of the derivative of a matrix valued function is proposed and some useful formulae to calculate the derivatives are derived. Two numerical algorithms are suggested. The one will be applied for solving problems with smooth objective function and constraints, and the other for solving non-smooth problems. Numerical examples and comparisons to the other existing algorithms are summarized.

In Chapter Six, the homotopy method, which is also known as a parametric optimization problem, is extended to solve the robust pole assignment problem. A basic concept of homotopy, several different types of homotopy mapping and their properties are reviewed. Kuhn-Tucker equations are derived in the form of the parametric optimization problem. Some of the existing algorithms including
initial value methods, nonlinear complementarity problems are summarized. A new
algorithm based on the homotopy approach is proposed. The homotopy method is
used for solving the pole assignment problem by defining a matrix homotopy func-
tion. A Taylor series expansion of a matrix valued function is considered to convert
the matrix valued optimization problem into a quadratic programming structure.
Some associated formulae are derived. Numerical examples of the robust pole as-
scription problem will demonstrate how the homotopy algorithm globally converges
to optimal solutions effectively regardless of initial starting points.

Chapter Seven provides a case study of pole assignment for an aircraft control
system. Basic goals of flight control system design are to maintain specified equi-
librium states of aircraft motions and to adjust pilot control deficiencies. Though
aircraft motion involves highly nonlinear equations, they can be linearized for small
deviations from a stationary flight conditions such as constant altitude and velocity,
and small angle of attack etc. A decoupled linearized model of the longitudinal and
lateral motion of a light aircraft is used in this case study. The case study will
demonstrate how the robust pole assignment methods and algorithms proposed in
this research may be efficiently applied for solving control system design problems.
Three different controllers are designed by assigning closed-loop poles to prescribed
positions using different pole assignment methods. Design 1 is obtained using the
modal assignment method which is a direct pole assignment method. Designs 2
and 3 are obtained using robust pole assignment methods. It will be seen that
controllers designed using the algorithms presented here yield more robust flight
control.

Chapter Eight presents the conclusions of this thesis and suggests further re-
search work.
Chapter 2

Design of Control Systems

2.1 Introduction

Control system designers or analysts must be aware of the state-of-the-art concepts of systems theory in addition to control theory. They are supposed to be not only the qualified engineers but also the wise systems analysts. In order to design a control system successfully, the underlying system must be thoroughly understood and analyzed.

A system may be defined in many different ways. For example, (1) A system is a set or assemblage of things connected, associated, or interdependent, so as to form a complex unity; a whole composed of parts in orderly arrangement according to some scheme or plan; rarely applied to a simple or small assemblage of things [181]. (2) A system is an arrangement or collection of objects that operate together for a common purpose. The objects may include machines, human, physical and biological entities. Everything excluded from the system itself is considered to be part of the system's environment [160].

The first definition above is restricted to a hardware system while the second
one includes more general concepts. In the general sense, a system consists of entities, sets of relations and some deliberate goals. A system may not necessarily be physical, but it can be either conceptual or abstract. In any case, a system is a set of components operated to accomplish a common set of goals. Thus, by definition, unicellular animals cannot be systems but even a small insect may be considered as a system. A human society is a large system. The solar system is a larger system.

A systems approach, sometimes used as a synonym of systems analysis, systems engineering, or systems science is usually defined as the study and use of systems by scientific and mathematical methods. This definition seems to be somehow ambiguous. The mathematics has existed since human history began. So called scientific may be meaningful in a particular era only. Indeed, the paradigm of today may not necessarily pertain tomorrow. Systems are made up of sets of components which work together to perform the overall objective or goal of the whole. The systems approach is a method of treating these total systems and their components as a whole. In fact, the essential feature of a systems approach is the concept of treating and analyzing the problems as a whole rather than piecemeal to improve the performance of the underlying system.

The process of such a systems approach may be classified in many different ways depending upon the fields of study or the analyst’s viewpoint. The Encyclopedia America [62] classifies the steps in the system approach as follows.

- Formulate the problem.
- Identify and describe the components of the system and their interrelationships.
- Develop mathematical or logical models.
- Analyze system performance and study alternative means for accomplishing
objectives in terms of criteria such as cost, size, effectiveness, and risk.

- Select the best system on the basis of the specific criteria.
- Build or implement the physical or abstract system that has been selected.

Other different classifications may be found in [1], [162]. Note that each of these phases is not independent of each other, but continuously interacts with the others. They may be proceeded in the above order but not necessarily be terminated in the same order. At any step, there may be feedback to a previous step until the study is completed.

This chapter reviews the history of feedback control systems. Major events in the historical development of control systems are summarized in a table. Control system design processes are broken down into five phases: (1) Problem formulation, (2) Construction of a mathematical model, (3) Deriving a solution, (4) Analysis and synthesis and (5) Implementation. Each phase is discussed in detail. Descriptions of design objectives and design specification are the main parts of this chapter since they are very important factors for system designs. In particular, design objectives of stability, robustness and robust stability are emphasized. Performance specifications are considered in terms of frequency domain and time domain. Brief definitions of each specification are given. Three frequently appearing types of objective functions in the mathematical formulation of control system design problems are introduced; LQG-design, $H^2$-design and $H^\infty$-design.
2.2 Background to Control Systems

2.2.1 Classifications of Control Systems

The British Standard Institution (BSI: [31], Part I, item No.10) defines a control system as an arrangement of elements (amplifiers, converters, human operators, etc) interconnected and interacting in such a way as to maintain or to affect

![Diagram of major classes of systems](image)

Figure 2.1: Major classes of system (Brogan [32], pp13)
(Dashed lines indicate the similar subdivisions to the others shown on the same level.)
in a prescribed manner, some conditions of a body, process or machine which forms part of the system.

A control system is an arrangement or interconnection of physical components forming system configurations which will provide a desired system response. Though not all systems may necessarily be physical systems, the control system is concerned mainly with physical ones. If there is no human intervention in the process of maintaining a satisfactory relationship between the input and the output of a system, it is called an automatic control system. Since in modern control theory, most of the cases are concerned with automatic control, in this thesis, control systems will refer to automatic control systems. Control systems can be classified in many different ways. Brogan [32] introduces the family tree which illustrates the major control system classifications as shown in Fig. 2-1.

Since feedback control systems are the main concern of this research, they will now be discussed in more detail.

### 2.2.3 History of Feedback Control Systems

Mayr [151] has investigated the history of early feedback control devices in his book, *The Origins of Feedback Control*. Though it is reported ([24], pp1) that the word *feedback* seems to have been used first in 1920, Mayr traces back the feedback control devices invented during the Hellenistic period (323 - 31 BC). According to Mayr, the water clock, which may be one of the earliest feedback devices known, was invented by Ktesibios who lived in Alexandria and served as mechanician under King Ptolemy II Philadelphus (285-274 BC). A float valve is designed to control the flow rate in the water clock as a feedback mechanism. Philon’s oil lamp (about 250 BC) is a feedback device to maintain a constant liquid level in the lamp by controlling a supply of liquid from the reservoir. Heron (1st century AD) invented several
float regulators, including an automatic wine dispenser, to maintain constant liquid levels in the devices. Mayr explains why these devices have feedback mechanism. Many other interesting examples can be found in his book. A thermostatic furnace invented by a Dutchman, Cornelis Drebbel in 1690, is the first feedback system invented independently of ancient models [90]. The first industrial application of a feedback control device is known as James Watt’s centrifugal flying-ball governor invented in 1788 to control the speed of a steam engine. Some other historical surveys of the feedback control systems may be found in [24], [54], [74]. A selected historical development of feedback control systems is summarized in the following table based on these references.
### Table 1. Major events of historical development in control system

<table>
<thead>
<tr>
<th>Year</th>
<th>Contributor</th>
<th>Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>285-247BC</td>
<td>Ktesibios, Alexandria</td>
<td>Water clock</td>
</tr>
<tr>
<td>about 250 BC</td>
<td>Philon, Byzantium</td>
<td>Oil lamp</td>
</tr>
<tr>
<td>1st century BC</td>
<td>Heron, Alexandria</td>
<td>Automatic wine dispenser/Floating regulator</td>
</tr>
<tr>
<td>12th century</td>
<td>?, China</td>
<td>Drinking-straw regulator</td>
</tr>
<tr>
<td>1620</td>
<td>Cornelis Drebbel, Holland</td>
<td>Temperature regulator</td>
</tr>
<tr>
<td>1680</td>
<td>Becher, Germany</td>
<td>Temperature regulator</td>
</tr>
<tr>
<td>1681</td>
<td>Dennis Papin, France</td>
<td>The first pressure regulator for steam boiler(safety valve)</td>
</tr>
<tr>
<td>1745</td>
<td>Edmund Lee, Britain</td>
<td>Fan-tail(self-regulating wind machine)</td>
</tr>
<tr>
<td>1758</td>
<td>James Brindley, Britain</td>
<td>Steam engine regulator</td>
</tr>
<tr>
<td>1765</td>
<td>Polzunov, Russia</td>
<td>Water level float regulator</td>
</tr>
<tr>
<td>1784</td>
<td>Sutton Thomas Wood, Britain</td>
<td>Steam boiler float level regulator</td>
</tr>
<tr>
<td>1785</td>
<td>Robert Hilton, Britain</td>
<td>Speed sensing device</td>
</tr>
<tr>
<td>1787</td>
<td>Thomas Mead, Britain</td>
<td>Windmill speed regulator</td>
</tr>
<tr>
<td>1788-1800</td>
<td>James Watt, Britain</td>
<td>Steam engine with centrifugal governor</td>
</tr>
<tr>
<td>1789</td>
<td>Stephen Hooper, Britain</td>
<td>Roller reefing sails</td>
</tr>
<tr>
<td>1790</td>
<td>Périer brothers, France</td>
<td>Speed regulator</td>
</tr>
<tr>
<td>1793</td>
<td>Abraham-Louis Brequet, Swit.</td>
<td>Pendule sympathique</td>
</tr>
<tr>
<td>1839</td>
<td>Jacob D. Custer, America</td>
<td>Shaft governor</td>
</tr>
<tr>
<td>1840</td>
<td>Benjamin Hick, Britain</td>
<td>Speed-reference governors</td>
</tr>
<tr>
<td>1840/51</td>
<td>Siemens brothers, German</td>
<td>Chronometric governor</td>
</tr>
</tbody>
</table>
Table 1 (continued)

<table>
<thead>
<tr>
<th>Year</th>
<th>Contributor</th>
<th>Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>1862</td>
<td>Foucault, France</td>
<td>Friction governor</td>
</tr>
<tr>
<td>1868</td>
<td>Maxwell, Scotland</td>
<td>The first systematic study of the stability of feedback control</td>
</tr>
<tr>
<td>1876</td>
<td>Vyshnegradskii, Russia</td>
<td>Mathematical theory of regulator</td>
</tr>
<tr>
<td>1877</td>
<td>Routh, Canada</td>
<td>Criteria for stability</td>
</tr>
<tr>
<td>1882</td>
<td>Hartnell, Britain</td>
<td>Steady-state design of governors</td>
</tr>
<tr>
<td>1892</td>
<td>Lyapunov, Russia</td>
<td>Stability of motion</td>
</tr>
<tr>
<td>1893/94</td>
<td>Stodola, Hungary</td>
<td>Time constant analysis</td>
</tr>
<tr>
<td>1895</td>
<td>Tolle, Germany</td>
<td>Graphical method of design</td>
</tr>
<tr>
<td>1895</td>
<td>Hurwitz</td>
<td>First use of stability conditions in a practical control system design</td>
</tr>
<tr>
<td>1910</td>
<td>Bateman, America</td>
<td>Use of Laplace transform for control</td>
</tr>
<tr>
<td>1922</td>
<td>Minorsky, Russia</td>
<td>Automatic controllers for steering ships and stability criterion</td>
</tr>
<tr>
<td>1927</td>
<td>Black, America</td>
<td>The earliest proposed solution to the robust control problem</td>
</tr>
<tr>
<td>1932</td>
<td>Nyquist, America</td>
<td>Graphical plot of the loop frequency response to determine stability</td>
</tr>
<tr>
<td>1934</td>
<td>Hazen, America</td>
<td>First precise analytical approach to design of closed-loop control system</td>
</tr>
<tr>
<td>1927-1945</td>
<td>Bode, America</td>
<td>Methodology of feedback amplifier design</td>
</tr>
<tr>
<td>1948</td>
<td>Evans, America</td>
<td>Root-locus method</td>
</tr>
<tr>
<td>1950s</td>
<td>Bellman, Kalman; Pontryagin, Russia</td>
<td>State-space approach</td>
</tr>
<tr>
<td>1964</td>
<td>George Devol, America</td>
<td>First industrial robot design</td>
</tr>
</tbody>
</table>
2.3 Feedback Control Systems Design

2.3.1 Advantage of Feedback Control Systems

Feedback control system or Closed-loop control system has been variously defined. For example, BSI[31], Part 1, item No. 2002 defines the feedback(closed-loop, monitored) control system as:

Closed-loop control system or monitored control system is a control system which processes monitoring feedback, the deviation signal formed as a result of this feedback being used to control the action of a final control element in such a way to tend to reduce the deviation to zero.

Though an open-loop control system may be much simpler and more economical, it is less applicable in practice when unexpected disturbances or parameter variations, or uncertainties in the system’s dynamic are present. Most textbooks have given the advantages of feedback control as follows.

i) To make the system output follow the specified input function automatically and to ensure the stability characteristics.

ii) To reduce the sensitivity of system performance to variations of parameter value or modelling errors.

iii) To achieve insensitive system performance and improve the system capacity to attenuate noise and unwanted disturbances.

iv) To ensure the desired transient and steady-state response of the system.
Control system design is the arrangement of the control system structure and selection of suitable components and parameters to achieve specified design objectives effectively and efficiently. In the system theory it is understood that, in general, effective is *doing the right thing*, while efficiency is *doing the thing right*. Thus, if a system is designed effectively, the desired goal of the system, or its output will be achieved *optimally*. If it is designed efficiently, the input to achieve desired goals will be utilized *optimally*. Here the *input* does not necessarily imply the input signal only, but all the manipulated inputs to the system. Control *system design* is concerned with a control system to be designed, while control *system analysis* is concerned with the existing system which is *already designed*. Control system design will be based on control system analysis for both the current system or the system to be designed. These two are interdependent and carried out by a kind of feedback process. It may be more appropriate to combine these two phases into one and call it a *control system design based on the control system analysis*, which will be called *control system design by analysis*. The control system design process may be classified in a similar way as the steps in systems approach discussed in Section 2.3. In this discussion, it will be broken down as follows. Other different classifications may be found in [46], [62]

- Formulation of the problem
- Construction of a mathematical model
- Deriving a solution
- Analysis and synthesis
- Implement the system
Problem Formulation

Before solving a problem, the problem itself and its components must be identified and formulated as precisely as possible. The components of the problem include the design objectives/performance specifications, constraints, alternative courses of action(input) and possible outcomes(output) of the choice of the alternatives, etc. Rules and relationships between these components need to be constructed. So called measures of effectiveness(MOE) which consist of the importance of objectives and the efficiency of courses of action must also be defined. Some knowledge of the Theory of Measurement(for example, see [231]) will be crucial to define such an MOE. The process of problem formulation is very important since if it has not been correctly formulated, all the remaining processes will be meaningless.

Modelling

A model may be defined as a simplified or idealized description or conception of a particular system, situation, or process that is put forward as a basis for calculations, predictions, or further investigation [181]. Ackoff and Sasieni(1968) define the model in the following sense. A model is a simplified representation or abstraction of reality, which will be able to predict and explain phenomena with a high degree of accuracy. The McGraw-Hill Encyclopedia of Science and Technology [160] classifies the model as (1)Mental model, (2)Iconic model, (3)Linguistic model, (4)Scale model, analog model or prototype and (5)Mathematical model or computer simulation model. Although in control system design, models may not be necessarily restricted to a particular type, in general, mathematical models are the main concern. A mathematical model is expressed in terms of mathematical symbols and expressions to describe the properties of the underlying system. The modelling phase is concerned with finding the most general conceptual framework in which a scientific theory or a technological problem can be posted without losing the essential feature of the theory or the problem under consideration. From a
system designer's point of view, the model must be constructed to represent the real system as accurately as possible but must be tractable. These objectives may be contradictory. If the model is too complicated, a solution may never be found or the process may be too expensive in terms of time and money. If a model is too simplified to represent a given phenomenon adequately, a solution derived from the model will be meaningless. Thus, to be tractable, assumptions need to be simplified and approximations will be made, without significant loss of accuracy. The degree of accuracy and the tractability will depend upon the design objectives and the availability of solution methods. Models should be continually adjusted during all design processes.

Deriving a Solution

After formulating a suitable mathematical model for the system under study, a solution to this model needs to be derived. There are several different types of procedures for deriving a solution from the model; such as analytic procedures, numerical procedures and simulation. An analytic procedure, sometimes called a deductive method, is a solution method which proceeds directly to the explicit closed-form solution in terms of the control variable. A numerical procedure or iterative procedure starts with an initial solution and then iterates upon the process to generate an improved solution. Simulation may be used for many purposes. One of its important purposes in control system design will be the case when a solution is derived analytically or numerically, it may be meaningful for a particular instant, say, terminal state or steady state but not the intermediate states or the transient states. In some cases, it may not be possible to construct a model for the problem under study or it cannot be solved by the above methods. In these cases experimental optimisation/designs procedure may be used which can be carried out by means of simultaneous experimental design or sequential design method. More detailed discussion may be found in [1].
**Analysis and Synthesis**

Even if the system under consideration has been modelled and an *optimal solution* has been somehow found, the control system design problem has not been solved yet since a model is simply a partial representation of the real system. In general, an optimal solution is meaningful only when the controlled variables and/or all established relationships and the properties among the components of the system remain unchanged or at least have not changed significantly. In fact, a control system may be designed for either an existing system and/or a new system. Control system analysis and synthesis are the tools for achieving design objectives for both case. A distinction between analysis, design and synthesis is made as follows by Tou [233]:

> By *analysis* it means that the control system has been designed and what is regarded is an investigation of its performance under specified conditions. *Design* is often used in the sense that a control system satisfies the statement of the problem must be found. *Synthesis* is reserved for the more ideal situation, where there is a clear procedure, usually entirely mathematical for going from the problem statement to its solution.

Control system analysis is a process of investigating the rules of an existing known system and calculating its performance under specified condition or its response to a given input excitation. If a control system has been designed, it will be interpreted as an existing system. Control system analysis is concerned with the performance of a given system with respect to change of inputs or disturbances. Control system synthesis is the process of defining a system to generate some specified input-output relationships. Synthesis involves a mathematical procedure for finding solution from its problem statement. Control system design methods are usually classified as (1) design by analysis and (2) design by synthesis, where the former is accomplished by modifying the characteristics of an existing or standard system configuration,
and the latter is accomplished by defining the form of the system directly from its specifications. Thus indeed, the control system design is concerned with how the performance of the systems can be improved with or without changing the system when the performance is unsatisfactory, which will be carried out by means of analysis or synthesis.

Implementation

Since the objective of system analysis is to improve the performance of the underlying system, and furthermore, during any step of the systems approach, each process may be continually adjusted, the system designer's or system analyst's study is not completed until the desired goal is achieved. The architect, the designer of a building may be the only one who knows well the essential characteristics of the building he has designed, and knows what problems may arise during the construction. He should participate in the construction work during the whole process. The system analyst/designer will be in a similar situation. For her/him, the implementation process is also important as a continuous interplay between the other steps. Some of the crucial items in the control system design process will be discussed in the subsequent sections.

2.3.3 Design Objectives

The basic goal of control system design is to meet the design objectives and performance specifications of the system to be designed. The objectives can be anything, whichever is considered as appropriate. The following are some of those design objectives which are frequently considered in the literature.

- Stability
- Robustness
For a system designer, it will be crucial that each objective should be analyzed and compromised in order to achieve the overall system objectives rather than any individual characteristic alone. Since some of these objectives may be interrelated or indeed conflicting, the choice of appropriate weights or the goals will depend on the designer's preferences, which unfortunately, may be subjective. Each objective listed above will be discussed briefly in the following.

2.3.3.1 Stability

Willems [253] defines stability in two different classes. The first one is stability of free systems in which there is no input or, equivalently, of forced systems with a given input, which were originally discussed by Lyapunov in 1893, and the second one is stability of forced systems with various inputs.

Definition 2.1: Stability of a free system([253], pp3)

Let $x_0$ be an equilibrium state of the free dynamic system

$$\dot{x} = f(x, t), \text{ with } f(x_0, t) = 0 \text{ for all } t. \quad (2.1)$$

The equilibrium state $x_0$, or the equilibrium solution $x(t) = x_0$ is called stable if for any given $t_0$ and positive $\epsilon$, there exists a positive $\delta(t, t_0)$ such that

$$\|x_0 - x_0\| < \delta \text{ implies } \|x(t, x_0, x_0)\| < \epsilon, \forall t \geq t_0. \quad (2.2)$$

In words, the equilibrium state of the system is called stable if when the system is
perturbed from the equilibrium state subsequent motions remains in a correspond­
ingly small neighbourhood of the equilibrium state.

Definition 2.2: Stability of forced systems([253], pp11)

Consider the forced dynamic system

\[
\begin{align*}
\dot{x} &= f(x, t; u(t)) \\
y &= g(x, t; u(t))
\end{align*}
\]  

Let \( f(x_e, t; 0) = 0 \) for all \( t \), so that \( x_e \) is an equilibrium state of system (2.3) for zero-input. Then \( x_e \) is called input-output stable if for any positive \( N \) there exists a positive number \( M \) such that \( \|u(t)\| < N \forall t \geq t_0 \) implies \( \|g(x(t; x_e, t_0), t; u(t))\| < M \forall t \geq t_0 \).

That is, a dynamic system is called input-output stable if for any bounded input a bounded output is produced regardless of the initial state. It is well known that the Nyquist stability criterion can be generalized to multivariable system to investigate the closed-loop stability of a system, and the generalized Nyquist diagram can be used as an indicator of closed-loop stability [153], [154].

### 2.3.3 Robustness

Robustness is one of the most important design objectives. Since the plant dynamics are not usually completely known, it may not be possible to describe the real system (or the original system) accurately. Even if the real system is completely known, unmodelled dynamical elements from linearization and/or simplification may significantly affect the performance of the system under study. In the literature related to robust control system theory, these types of uncertainties are usually classified as model uncertainty, parameter uncertainty and structure uncertainty. Model uncertainty is caused by inaccuracies of the parameter values and/or the model structure to describe the underlying system. Parameter uncertainty is caused by perturbations of the parameters while structure uncertainty is due to imperfect model
structure. Structure uncertainty may be classified further into structured uncertainties and unstructured uncertainties. If the structure of the error matrix and perturbations to each individual matrix are known, it is called structured uncertainty. If only the bound of the error matrix norm is known, it is unstructured uncertainty.

Robustness of the system is something to be achieved by ‘minimizing the sensitivity of the system performances’ with respect to its variation. In the Oxford English Dictionary [181], the following are cited for describing the concept of robustness.

- What is important is the recognition of common features in the set of outcomes; these are the inductive inferences which may be classed as forecasts. We say that the system is robust in respect to a particular set of outcomes.

- A concept both more vague and much more difficult to ensure is termed robustness. A robust algorithm is one which in practice usually yields the global minimum or a good local minimum of any function of even a large number of variables from a poor initial approximation.

In control theory, it has been used in many different ways. For example, a state feedback law, designed to stabilize the reduced-order system model is robust in the sense that it will stabilize the actual full-order system model for a sufficiently small perturbation to the state dynamic provided the neglected fast modes are asymptotically stable([133], pp143). It is also used in the sense of covering large parameter variations and providing good gain and phase margins [134], maintaining adequate stability margins or other performance levels in spite of model errors or deliberate over-simplifications [32], or the ability of a system to remain stable despite model inaccuracies and parameter variations [184]. In terms of eigenvalue, a system is said to be robust if the closed-loop eigenvalues are as insensitive as possible to
Several methods to measure the robustness of a system are suggested in the literature. Pang et al [184] define the normality indicator which may provide a quantitative measure of departure from normality of a given matrix to use as an indicator for the robustness. Postlethwaite et al [193] introduce two measures; Insensitivity to parameter variations and/or unmodelled dynamics and Insensitivity to external disturbances to indicate robustness of the system. Kautsky et al [121] use the condition number of the closed-loop system matrix to measure the system robustness.

2.3.3.3 Robust Stability

A control system must be designed so that it remains stable regardless of plant parameter variations, modelling errors or unmodelled dynamics. Doyle et al ([56], pp9 & pp51) define robust stability in the following sense.

Let \( \mathcal{P} \) be the family of all perturbed plants \( \tilde{P} \). A controller \( C \) is robust with respect to a certain characteristic of the system if this characteristic holds for every plant in \( \mathcal{P} \). A controller \( C \) provides robust stability if it provides the prescribed characteristic for every plant in \( \mathcal{P} \).

In the frequency domain, the robustness of the closed-loop stability under the plant parameter variations can be measured by the stability margin. Since stability requires that all eigenvalues lie in the left-half of the complex plane, a stability degree defined as the minimum distance of the eigenvalues from the imaginary axis given by ([150], pp84)

\[
d_s = - \min_i \text{Re}\{\lambda_i[A + BKC]\}
\]  

(2.4)
may be used as a measure of stability robustness in the time domain. Other definitions and various robust stability measures and test methods of robust stability have been suggested in the literature [129], [136], [153], [237].

2.3.3.4 Steady-state accuracy

The steady-state is the equilibrium state attained such that the conditions of each component of the system do not change with respect to time unless the system is exited by an input or an external disturbance. Steady-state accuracy will be obtained when the error signal approaches a sufficiently small value for a long period of time.

2.3.3.5 Fast transient response

Transient response of a control system involves the responses of the system prior to reaching a steady-state. For a good system design, it is desirable that the transient response be sufficiently fast and damped. For a second order system, it is understood, in general, that for a desirable transient response, the damping ratio will be between 0.4 and 0.8. Beyond these ranges, there will be an excessive overshoot or a too slow response to an input, respectively. Therefore, no excessive overshoot for abrupt inputs, an acceptable level of oscillation in an acceptable frequency range, and a satisfactory speed of response and settling time will be the main concerns.

2.3.3.6 Satisfactory frequency response

The frequency response is the steady-state response of a system to a sinusoidal input signal. Satisfactory frequency response means satisfactory bandwidth, limits on maximum input-to-output magnification, frequency at which this magnification occurs, and proper gain and phase margins. A large bandwidth corresponds to
the poles that are far to the left of the imaginary axis in the complex plane and produce a fast rise time, while a small bandwidth yields the opposite effect. An optimum bandwidth is one which transmits the signals unattenuated but filters out the unwanted noises. The values of the gain and phase margins bound the behaviour of the closed-loop system near the resonant frequency. Experience has shown that for satisfactory transient response, the phase margin will usually be required to be between $30^\circ$ and $80^\circ$, and the gain margin be greater than 6 db.

### 2.3.4 Performance Specifications

For each design objective above the performance must be specified. Such specifications, called performance specifications or system design specifications, need not necessarily be measured by the cardinal utilities. They may be defined qualitatively. They may not always be required to be rigid but may be somewhat conceptual. However, they should be able to indicate whether the desired behaviours of a system are achieved. Unfortunately, the terminology performance is sometimes used in different ways, for example, in the narrow sense where interest is confined to reference input tracking, disturbance rejection and noise rejection [153], etc. However, in this thesis, performance means the MOE (measure of effectiveness) for the specified design objectives, which may include stability, robustness and robust stability, etc., as discussed in the previous sections. Performance specifications are the specified requirements of these design objectives. There are various ways of defining the performance specifications depending upon the methods for choosing appropriate controller parameters for the underlying control system design. It is generally agreed that there are basically two design methods; frequency domain methods and time domain methods. Thus it may be reasonable to classify the performance specifications as frequency domain specifications and time domain specifications as in [53].
2.3.4.1 Frequency Domain Specifications

Frequency domain specifications may include the following quantitative measurements.

(1) **Gain margin**: Gain margin is the reciprocal of the magnitude of the open-loop transfer function at the phase crossover frequency $\omega$ where the phase angle is -180°. That is, gain margin = $1/|G(j\omega)|$. Gain margin measures a relative stability in the sense that how much the gain can be increased before the stable system becomes unstable, or it must be decreased to make the unstable system to be stable.

(2) **Phase margin**: Phase margin is defined as 180° plus the phase angle $\phi$ of the open-loop transfer function at the gain crossover frequency. It measures a relative stability of the system in terms of phase lag at the gain crossover frequency to bring the system to the boundary of instability.

(3) **Delay time**: Delay time is the speed of response of a signal measured by $\gamma$, where $\gamma = \arg H(j\omega)$, and $H$ is the closed-loop transfer function.

(4) **Bandwidth**: The range of frequencies of the input over which the system(filter, amplifier etc.) will respond satisfactorily, often the points at which the performance has changed by 3 dB from a mean level, or the half-power points.

(5) **Cutoff rate**: The frequency rate at which the magnitude ratio decreases beyond the cutoff frequency.

(6) **Resonance peak**: The maximum value of the magnitude of the closed-loop frequency response.

(7) **Resonance frequency**: The frequency at which a resonance peak occurs.

2.3.4.2 Time Domain Specifications

Time domain specifications will define the measurements for the following items, each of which will have a steady-state and a transient component.
(1) **Overshoot**: Maximum difference between the transient and steady-state solutions for a unit-step function input.

(2) **Delay time**: The time required for the response to reach 50% of its final value the very first time.

(3) **Rise time**: The time required for the response to rise from 10 to 90%, 5 to 95% or 0 to 100% of its final value.

(4) **Settling time**: The time required for the response to reach and remain within a range of specified percentage of its final value.

Any satisfactorily designed system must satisfy these design objectives and design specifications as much as possible. However, unfortunately, it may not always be possible because some of the objectives/specifications may conflict with each other, or may not necessarily be precisely measurable by an additive quantity. Multiobjective optimization theory may be applied to find a compromise solution for this case.

### 2.3.5 Objective Functions in Mathematical Models

Constructing appropriate objective function(s), i.e., the performance measures in an appropriate mathematical form, is a crucial phase in formulating the mathematical model. The chosen form is closely related to the problem of accuracy and tractability. In control system design, frequently appearing types of objective functions are as follows.

1. \[ J = \int_0^\infty (x^T Rx + u^T Ru)dt \quad \text{(LQG-design)} \quad (2.5) \]
2. \[ J = \int_{-\infty}^{\infty} ||E(j\omega)||^2 d\omega \quad \text{(H^2 - design)} \quad (2.6) \]
3. \[ J = \sup_{\nu} ||E(j\omega)|| \quad \text{(H^\infty - design)} \quad (2.7) \]

The choice of objective function will depend on a quantitative measure of each performance specification and the solution techniques to be applied. If some of
the measures are based on qualitative concepts rather than quantitative ones, the
objective function may be described in terms of, namely, utility functions, linguistic
variables or fuzzy set concept etc. If more than one objective function has been
considered, the problem needs to be transformed into a multiobjective optimization
problem. Objective functions may also be reformulated during the design process.

2.4 Concluding Remarks

The systems approach is a method which can be applied to study and design a sys­
tem more scientifically and systematically. It has been broadly applied in the fields
of natural science and engineering as well as social science. The essential feature of
the systems approach is treating the underlying system as a whole rather than as
each piecemeal of it. Control system design process based on the systems approach
has been classified into five phases and each of them has been discussed briefly.
Though all of the phases are important, the most crucial step is problem formula­
tion. Since no model can be perfect, i.e., a model is not a full representation of the
real system but simply a partial representation of it, a good control system design
must be able to reduce sensitivity to the internal parameter variations, external
disturbances and/or unmodelled dynamics. Concepts of robustness/robust stabil­
ity which may provide the most reasonable measurement of the underlying system
performances, and other design objectives have been introduced. Performance spec­
cifications based on frequency domain and time domain methods are discussed. The
material discussed in this chapter is that which may provide the background for the
theme of this thesis. Pole assignment in desired locations has been understood to
play an important role for achieving control system design objectives, which will be
the subject of the next chapter.
Chapter 3

Pole Assignment Methods

3.1 Introduction

The basic goal of control system design is to achieve the design objectives and performance specifications required for the system. The eigenstructure (eigenvalues and eigenvectors) plays an important role in determining the dynamical characteristics of a system. The eigenvalue determines the time-domain characteristic of a mode; the rate of decay or rise of the system response. The eigenvector determines the shape of this response. Indeed the pole configuration is directly related to the stability, steady state accuracy and satisfactory transient response of the system. To ensure satisfactory transient response of the system, the closed-loop poles must be located at suitable positions if the open-loop plant cannot be altered by the designer to improve its performance. The location of the poles in the complex plane entirely characterizes the stability of linear time invariant systems. Thus, it is desired to find suitable feedback gains such that the poles of the closed-loop system matrix can be assigned to desired locations. This is the motivation for the pole assignment problem. Pole assignment problems can be broadly classified into static feedback...
pole assignment problems and dynamic feedback pole assignment problems whereby this research considers only the former case. There are basically two different types of static feedback pole assignment problems; pole assignment by full state feedback and pole assignment by output feedback. Full state feedback pole assignment assigns the closed-loop system poles to desired locations based on the availability of complete state information, while output feedback pole assignment is based on incomplete state information and feeding back the measured output only.

Consider the continuous linear time-invariant system

\[
\begin{align*}
\dot{x} &= Ax + Bu \\
y &= Cx
\end{align*}
\]

where \( x \in \mathbb{R}^n, u \in \mathbb{R}^m, y \in \mathbb{R}^p \) represent the state, input and output vectors and \( A, B, C \) are compatibly dimensioned constant matrices. It is well known [255] that, for state feedback control, an arbitrary set of poles may be assigned by means of an appropriate state feedback if the system is controllable. However, for output feedback control, the controllability assumption alone is not sufficient to guarantee the pole assignability. Davison [48] has shown that if rank \( C = p < n \), and if \( (A, B) \) are controllable, then \( p \) eigenvalues of the closed-loop system matrix may be assigned arbitrarily close to those desired. Furthermore, Davison and Wang [51] extended this result to show that if the triple \( (A, B, C) \) is both controllable and observable with \( B \) and \( C \) of full rank, then for almost all \( (B, C) \) pairs, \( \min\{n, m+p-1\} \) eigenvalues are assignable arbitrarily close to those desired by output feedback. However, if the inequality \( m + p > n \) does not hold, such algorithms cannot assign the remaining \( n - (m + p - 1) \) poles, though Brash and Pearson [29] have shown that, for a controllable and observable system, it is always possible to obtain an augmented system satisfying this inequality by adjoining a dynamic compensator of suitable order. Ahmari and Vaceau [4], Kimura [127], Seraji [218],[219], Munro and Novin-Hirbod [171], Sambandan and Chandrasekharan [210], Chen and Hsu [40], and Han [109] introduce some methods or algorithms in the context of dynamic compensator
for assigning the poles without the dimensionality requirement. Nevertheless the approach seems to be less practical in optimization theory point of view because increasing dimensionality of the argumented matrices associated, as well as their increased sparsities may degrade the computational efficiency. Fletcher et al [70] have presented some necessary and sufficient conditions for eigenstructure assignment relaxing the dimensionality condition \( m + p > n \). Other different method has also been proposed by Chu et al [45]. They develop a pole assignment algorithm which may be used to approximately assign all desired poles by optimizing the trade-off between robustness of the system and the accuracy of the locations of assigned poles. However, the distance between desired and actual poles assigned by the algorithm may be, for some case, unconstrained [45], even unstable [226]. Ho and Fletcher [111] have developed some conditions for an upper bound of the norm of the gain matrix to minimize the error bound of eigenvalue perturbations using the perturbation theory. Chen et al [41] develop an algorithm to determine the output feedback matrix required to assign \( \max(p, m) - 1 \) poles and to provide suitable locations for the remaining unassigned poles.

Despite these various approaches, no one proven approach is yet available and the problem of pole assignment by output feedback may still be considered open. Most of the existing algorithms have some limitations. For example, the eigenvalues to be assigned need to be distinct or the dimensions of the system matrices are required to satisfy dimensionality constraints, eg., \( p > m, m + p > n \), etc. These conditions will be discussed in more detail later.

In this chapter, the classification of output feedback pole assignment methods is discussed; the methods are classified into four categories and some of the literature surveys on these subjects are carried out. Some basic properties relating to pole assignability are then reviewed. The relationships between controllability and pole assignability are detailed. A representative selection of the algorithms is discussed.
This includes the mathematical formulation of the problems and fundamental concepts of the algorithms behind the selected methods. A selection of test problems is used to assess their relative strengths and weaknesses. Several examples, both for well-conditioned and ill-conditioned problems will demonstrate abilities of the algorithms.

3.2 Classification of Output Feedback Pole Assignment Methods

Pole assignment by output feedback control can be described formally as follows.

Consider the continuous linear time-invariant system given by (3.1) where \( x \in \mathbb{R}^n \), \( u \in \mathbb{R}^m \), \( y \in \mathbb{R}^r \) represent the state, input and output vectors and \( A, B, C \) are compatibly dimensioned constant matrices. It is desired to find an output feedback gain matrix \( K \in \mathbb{R}^{m \times p} \) which specifies a linear constant control law \( u = Ky \) such that the poles in the closed-loop dynamic

\[
\dot{x} = (A + BKC)x
\]

have prescribed locations. Moreover, these poles are to be assigned to optimize some appropriately defined performance assessment. The design of the output feedback controller solving this problem is called the pole assignment problem (or eigenstructure assignment problem) by output feedback control. Output feedback pole assignment problems have been studied by many authors for a number of years. These types of problems may be classified into the following four categories [175], [229].

- Direct eigenstructure assignment
- Parametric eigenstructure assignment
Robust eigenstructure assignment

Singular perturbation analysis

The direct eigenstructure assignment approach involves the choice of an eigenstructure which is as close as possible to that prescribed by applying some of the geometric properties of the space on which the eigenstructure is defined. The transient response of the system (3.2) depends not only upon the eigenvalues or poles but also upon the eigenvectors. The eigenvalue will determine the time-domain characteristic of a mode while the eigenvector determines the shape of that mode. By choosing ideal entries within a given eigenvector the contribution of a given state to a particular mode may be prescribed [15], [50], [51], [65], [68], [72], [127], [128], [138], [155], [168], [194], [228].

The parametric eigenstructure assignment is based on the fact that the closed-loop eigenvalues do not uniquely define a closed-loop system and a parametric expression for the controller gain matrix is explicitly determined by the associated eigenvectors [63], [182].

Robust eigenstructure assignment involves choosing a feedback such that the assigned poles are as insensitive as possible to the perturbations in the system matrices. Thus, in addition to the method of direct and/or parametric eigenstructure assignment, optimization of an appropriately defined measure of system performance subject to the given constraints is considered [45], [96], [107], [111], [179].

Singular perturbation analysis seeks to assign an appropriate eigenstructure to the closed-loop system of interest using singular perturbation theory. In the literature, several different approaches are used based on this concept. The first [27],[28] defines a tracking error between the plant output vector and command input vector, and an associated transformed state variable representation. A diagonalized closed-loop transfer function is then obtained so that the tracking behaviour of the
system is non-interacting. This is used to determine the controller matrices or gain parameters. The second is, by transforming the plant models to the singular linear systems, and compared to the reduced-order design model for sufficiently small perturbation. Controller parameters are then chosen so that the closed-loop model will have specified properties [79], [118], [133], [209]. Such approaches have been used for output feedback pole assignment problems in [69], [120], [183], etc.

Some fundamental theorems appearing in the early literature relating to the pole assignment problem will now be reviewed. These theorems state the pole assignability properties and related conditions.

### 3.3 Pole Assignment Properties

A fundamental theorem of the pole assignability for single input state feedback systems is stated as follows by Kalman et al.

Theorem 3.1 ([117], pp49):

Let the pair \( \{A, b\} \) be completely controllable, and let

\[
\rho(\lambda) = \lambda^n + \beta_1 \lambda^{n-1} + \cdots + \beta_n
\]

be an arbitrary polynomial, where \( n = \text{dim}[A] \). Then there is a vector \( k \) such that \( \chi_{A-kA} = \rho(\lambda) \), where the characteristic polynomial \( \chi_A \) of a square matrix \( A \) is the polynomial \( \det(\lambda I - A) \).

It is reported by Kalman et al that this theorem was first obtained, in 1959 by J. E. Bertran using the root-locus method, and also, R.W. Bass, in 1961, independently formulated and proved the theorem based on linear algebra in unpublished lecture notes. Similar results appeared in [203], [207]. Kalman et al interpret the Theorem 3.1 as follows.
The possibility of constructing an arbitrarily good control law is limited only by the controllability properties of the plant. If the plant is finite dimensional, linear, smooth, and constant, there is no limitation whatever once complete controllability is present. The theorem says that in a single-input, completely-controllable, constant system, the characteristic roots can be chosen at will if we allow feedback. This shows that feedback is enormously important, the dynamics of a plant may be altered by means of feedback in a completely arbitrary fashion, subject only to the requirement that we have an accurate representation of the physical behavior of the plant as a linear constant system. It is not necessary to build a plant so that it has inherently good dynamical characteristics; the desired characteristics can be achieved later, artificially, by means of feedback. As a result, one has great freedom in the design and physical construction of the plant.

Wonham [255] has generalized the above property by the following theorem for multi-input systems.

**Theorem 3.2:** The pair \((A, B)\) is controllable if and only if, for every choice of the set \(\Lambda = \{\lambda_1, \ldots, \lambda_n\}\), there is a matrix \(K\) such that \(A + BK\) has \(\Lambda\) for its set of eigenvalues.

Since in practice, particularly for large order systems, the complete state observation is undesirable and is not often practical, the question of pole assignability based upon incomplete state observation and feeding back the measured output only, instead of full-state feedback, was raised and studied by some authors [29], [147]. Davison was the first to show the following result.

**Theorem 3.3** [48]:

If \((A, B)\) are controllable, if \(C\) has rank \(p \leq n\) and if \(A\) has eigenvalues that are distinct or are repeated such that the eigenvalues of each Jordan block of the Jordan canonical form of \(A\) are distinct, then a linear feedback of output variables \(u = Ky\),
where $K$ is a constant gain matrix, can always be found so that $p$ eigenvalues of the closed-loop matrix $A + BK$ are arbitrarily close (but not necessarily equal) to $p$ preassigned values, where the preassigned values are chosen so that any complex numbers appearing do so in complex conjugate pairs.

Davison and Chatterjee [49] and Sridhar and Lindorff [230] extend the above result by showing $\max(m, p)$ eigenvalues are assignable almost arbitrarily, where $m$ is the rank of $B$. Davison and Wang further introduce the following theorem.

**Theorem 3.4** [51]:
Given $(A, B, C)$ controllable and observable with $B, C$ of full rank, then for almost all $(B, C)$ pairs, there exists an output gain matrix $K$ so that $A + BK$ has $\min(n, m + p - 1)$ eigenvalues assigned arbitrarily close to $\min(n, m + p - 1)$ specified symmetric eigenvalues.

Kimura [127] has shown that if the system is both controllable and observable and if $n \leq m + p - 1$, then an almost arbitrary set of distinct closed-loop poles is assignable by output feedback. When the dimensionality requirement does not hold, Kimura further introduces the following theorem of pole assignability condition by a dynamic compensator.

**Theorem 3.5** [127]:
If the system is both controllable and observable, then a dynamic compensator of order $r = n - m - p + 1$ can assign almost arbitrary poles for the overall closed-loop system provided that the poles to be assigned are all distinct.

Srinathkumar [228] has derived the following property for pole assignability by output feedback gain in the context of eigenvectors as well as eigenvalues.

**Theorem 3.6** [228]:
If the system is both controllable and observable, with $B$ and $C$ of full rank, then $\max(m, p)$ closed-loop eigenvalues can be assigned and $\max(m, p)$ eigenvectors or reciprocal vectors (by duality) can be partially assigned with $\min(m, p)$ entries in
3.4 Pole Assignment Algorithms

There have been hundreds of papers concerning pole assignment methods appearing in the literature. Though each has its particular features, these algorithms and methods may be broadly classified into four categories as discussed in Section 3.2. It is not a purpose of this research to discuss all these algorithms. Only a few selective algorithms will be reviewed to understand the general concept of pole assignment algorithms and their advantages and restrictions. Unfortunately, it is not possible to choose any one 'optimal' algorithm to introduce the underlying ideas. No algorithm is proven to be the best one; the efficiencies are clearly problem dependent. In this section two different types of pole assignment algorithm will be reviewed. The first algorithm is based on the so-called direct eigenstructure assignment methodology. The second one is based on the robust eigenstructure assignment approach.

3.4.1 Direct Eigenstructure Assignment Methods

Kimura [127], [128] formulated the pole assignment problem by output feedback as the choice of eigenvectors $x_i$ and $y_j$ with respect to the desired eigenvalue $\lambda_j$ such that

\begin{align}
(A + BKC)x_i &= \lambda_i x_i, \quad i = 1, ..., n \\
y_j^T(A + BKC) &= \lambda_j y_j^T, \quad j = 1, ..., n \\
\end{align}

where

\begin{align}
\mathcal{F}(\lambda_j) &= \{x_j \in \mathbb{C}^n : (\lambda_j I - A)x_j \in \mathcal{R}(B)\} \\
\mathcal{G}(\lambda_j) &= \{y_j \in \mathbb{C}^{m^*} : y_j^T(\lambda_j I - A) \in \mathcal{R}(C)^T\}
\end{align}

Each vector arbitrarily chosen.

Most pole assignment algorithms are based on the properties introduced above. Pole assignment algorithms will now be reviewed.
A, B and C are defined in 3.1

He has shown that a set of eigenvalues \( \Lambda = (\lambda_1, ..., \lambda_n) \) is assignable if and only if there exist vectors \( x_j \in \mathcal{F}(\lambda_j) \) and \( y_j \in \mathcal{G}(\lambda_j), j = 1, ..., n \) such that

i) \( \{ x_j, j = 1, ..., n \} \) are linearly independent in \( \mathbb{C}^n \) and \( \lambda_j = \lambda_j^H \) implies \( x_j = x_j^H \).

ii) \( \{ y_j, j = 1, ..., n \} \) are linearly independent in \( \mathbb{C}^n \) and \( \lambda_j = \lambda_j^H \) implies \( y_j = y_j^H \).

iii) \( y_j^T x_j = 0, \forall i \neq j \).

Fletcher [65], [68], Miminis [167], Misra and Patel [168], Srinathkumar [228] and others used a similar direct design methodology to assign the desired poles. These algorithms adopt the methods of unitary transformations, plain rotations, and/or deflation. Before introducing an algorithm based on this direct methodology, the deflation method which provides the basis for the algorithm will be first reviewed.

3.4.1.1 Deflation Method

Any \( n \times n \) matrix \( A \) can be reduced to a lower dimensional matrix which will preserve some property or certain special structure. For example, \( A \) can be reduced to tridiagonal form or reduced to an \( (n - 1) \times (n - 1) \) matrix such that the \( n - 1 \) eigenvalue/eigenvector pairs of the reduced matrix are a subset of the eigenvalues/eigenvectors of \( A \). The deflation method can be described more fully as follows [186], [252].

For a given \( n \times n \) matrix \( A \), suppose one of the eigenvalues \( \lambda_1 \) and its corresponding eigenvector \( x_1 \) are known with \( \| x_1 \|_2 = 1 \). Consider any unitary matrix \( U \) whose first column is the eigenvector \( x_1 \), i.e., \( U e_1 = x_1 \), thus \( U = (x_1, U_1) \). Consider the unitary similarity transformation of \( A \) induced by \( U \) such that

\[
U^H A U = (x_1, U_1)^H A ((x_1, U_1)) = (U^H \lambda_1 x_1, U^H A U_1) = \begin{bmatrix} \lambda_1 e_1 & * \\ 0 & U_1^H A U_1 \end{bmatrix}
\]
Suppose \((x_2, \lambda_2)\) is an eigenpair of \(A_1\), i.e., \(A_1x_2 = \lambda_2 x_2\). Then, by the property above, \(U_1 \begin{bmatrix} 0 \\ x_2 \end{bmatrix}\) is an eigenvector of \(A\) with respect to the eigenvalue \(\lambda_2\). Continuing this process, the following can be generated.

\[
U_j^H A_j U_j = \begin{bmatrix} \lambda_j & * \\ 0 & A_{j-1} \end{bmatrix}, \quad j = 1, \ldots, n - 1
\]

Thus after \(n\) steps, the following will be obtained:

\[
U_n^H U_{n-1}^H \cdots U_2^H A U_1 \cdots U_2 U_1 = \begin{bmatrix} \lambda_1 & \cdots & * \\ 0 & \cdots & \cdots \\ & \cdots & \cdots \\ 0 & 0 & \cdots & A_n \end{bmatrix}
\]

(3.8)

In practice, since the eigenvector \(x_j\) is unknown, only the vector \(\hat{x}_j = x_j \cos \eta + y_j \sin \eta\), \(y_j^T x_j = 0\), \(\|y_j\| = 1\), where \(\eta\) is the angle between \((x_j, \hat{x}_j)\) is known, the orthogonal matrix \(U_j\) such that \(U_j c_j = \hat{x}_j\), and associated similarity transformation

\[
U_j^H A_j U_j = \begin{bmatrix} \beta & c^H \\ c & A_{j-1} \end{bmatrix}
\]

(3.9)

can be considered. However, if \(\eta = \text{angle} < x_j, \hat{x}_j >\) is small enough, it is known that([186], pp86).

\[
\|c\| < \|A - \beta\| \eta + o(\eta^2).
\]

That is, \(d\) is negligible, where

\[
Q^H U_j^H A_j U_j Q = \begin{bmatrix} \beta & d^H \\ d & A_{j-1} \end{bmatrix}
\]

(3.11)

which provides an unitary similarity transformation algorithm. Pole assignment by the method of deflation is based upon the fact that an eigenvalue problem for an
$n \times n$ matrix can be reduced sequentially to an $(n - 1) \times (n - 1)$ matrix without changing the eigenvalues of the system by unitary similarity transformations. For example, for a given $n \times n$ matrix $A$, let $x_j$ be such that $Ax_j = \lambda_j x_j, \|x_j\| = 1$. Then a matrix $U \in C^{n \times (n-1)}$ exists such that $(x_j, U)$ is unitary. Since $A x_j = \lambda_j x_j, A(x_j, U) = (\lambda_j x_j, AU)$,

$$(x_j, U)^H A(x_j, U) = \begin{bmatrix} x_j^H \\ U^H \end{bmatrix} \begin{bmatrix} \lambda_j x_j \\ AU \end{bmatrix} = \begin{bmatrix} \lambda_j x_j^H x_j & x_j^H AU \\ \lambda_j U^H x_j & U^H AU \end{bmatrix}$$

Also, since $x_j^H x_j = 1$ and the columns of $U$ are orthogonal to $x_j$, we have $U^H x_j = 0$. Let $x_j = x_1, U^H AU = A_2$, and $x_j^H AU = h_i$. Then, $(x_j, U)^H A(x_j, U) = \begin{bmatrix} \lambda_1 & h_i \\ 0 & A_2 \end{bmatrix}$. Furthermore, it can be shown that $\sigma(A) = \{\lambda_1, \cdots, \lambda_n\} = \{\lambda_1, \sigma(A_2)\}$ and the eigenvalues of $A$ are preserved in the reduced matrix $A_2$. A direct pole assignment algorithm for output feedback developed by Miminis [167] using the deflation method is now considered.

### 3.4.1.2 Direct Algorithm

This algorithm assigns $\min\{n, m+p-1\}$ closed-loop poles close to those specified. To start the algorithm, it is necessary that $p \geq m$ initially, otherwise, a transposed system, $A^T - C^T K^T B^T$ should be considered instead. It is first necessary to define the concept of a strong transmission zero.

**Definition 3.1** [167]:

A number $\lambda \in \mathbb{C}$ is a strong transmission zero of the system (3.1) if and only if $G(\lambda) = 0$ or equivalently,

$$\{\forall \mu \neq 0, \begin{bmatrix} A - \lambda I & B \\ C & 0 \end{bmatrix} \begin{bmatrix} x \\ \mu \end{bmatrix} = 0, \text{ with } x = (\lambda I - A)^{-1} B \mu\} \quad (3.12)$$

where $G(\lambda) = C(\lambda I - A)^{-1} B$.

With this definition, the following theorem provides the information regarding pole
assignability.

**Theorem 3.7 [167]:**

\( \lambda \in \mathbb{C} \) cannot be assigned by output feedback if and only if \( \lambda \) is a strong transmission zero.

The main steps of the algorithm are summarized as follows.

**Algorithm 3.1:**

1. **Step 1:** Compute a feasible eigenvector \( x_1 \) such that \((A_1 - B_1K_1C_1)x_1 = \lambda_1 x_1, \|x_1\| = 1\), where \((A_1 - B_1K_1C_1)x_1 = A - BKC, x_1 \in \mathcal{N}(U_1^T(A_1 - \lambda I))\) where \(\mathcal{N}(\cdot)\) denotes the null space of \(\cdot\).

2. **Step 2:** Allocate the eigenpair \((\lambda_1, x_1)\) to \(A_1 - B_1K_1C_1\) as follows.
   
   i) Consider a unitary transformation of \(C_1\) such that \(V_i^H C_1 x_i = \delta_1 e_i\).
   
   ii) Consider the QR decomposition of \(B_1\) where \(B_1 = (U_0, U_1)(Z_{B1}^T, 0)^T\), as defined in (3.23).

   iii) Consider \(((A_1 - B_1K_1C_1)x_1 = \lambda_1 x_1\).

   iv) \(Z_{B1}k_1 = U_0^H (A_1 - \lambda_1 I)x_1 / \delta_1\).

3. **Step 3:** Compute the unitary transformation \(Q_1 = (x_1, Q_1)\). Thus \(Q_1 e_1 = x_1, Q_1^H x_1 = e_1\).

4. **Step 4:** Perform the unitary similarity transformation

\[
Q_1^H (A_1 - B_1K_1C_1)Q_1 = \begin{bmatrix}
\lambda_1 & * \\
0 & A_2 - B_2K_2C_2
\end{bmatrix},
\]

where \(A_2 = \hat{Q}_1^H A_1 \hat{Q}_1 - \hat{Q}_1^H B_1k_1 c_1^H, B_2 = \hat{Q}_1^H B_2, V_1^H C_1 \hat{Q}_1 = \begin{bmatrix} c_1^H \\ C_3 \end{bmatrix}\).

5. **Step 5:** Continue the allocation with \(A_2 - B_2K_2C_2\).

Some examples will be tested by the algorithm in Section 3.6.
Robust pole assignment algorithms based on both full-state feedback [3], [17], [25], [35], [61], [119], [121]), and output feedback [45], [96], [107], [111], [177], [179]) have been extensively proposed in the literature. Since most of these algorithms are basically based on the projection theorem and coordinate descent methods, these will first be discussed.

### 3.4.2.1 Projection Theorem

The projection theorem is one of the most important optimization principle since it provides when and how an optimal solution in terms of a minimum distance from a point to a subspace of a given space exists and can be obtained.

**Theorem 3.8** ([148], pp 50):

Let $X$ be a pre-Hilbert space, $M$ a subspace of $X$, and $x$ an arbitrary vector in $X$. If there exists a vector $m_0 \in M$ such that $\|x - m_0\| \leq \|x - m\|$ for any $m \in M$, then $m_0$ is unique. Furthermore, $m_0 \in M$ is a unique minimizing vector in $M$ if and only if the error vector $x - m_0$ is orthogonal to $M$.

**Theorem 3.9:** The classical Projection Theorem([148], pp 50)

Let $H$ be a Hilbert space and $M$ a closed subspace of $H$. Corresponding to any vector $x \in H$, there exists a unique vector $m_0 \in M$ such that $\|x - m_0\| \leq \|x - m\|$ for any $m \in M$. Furthermore, $m_0 \in M$ is the unique minimizing vector if and only if $x - m_0$ is orthogonal to $M$.

### 3.4.2.2 Coordinate Descent Method

It is not clear why the coordinate descent method has been frequently used for developing robust pole assignment algorithms since it has never been tightly justified.
in the literature. It will depend upon the problems under consideration but in general, the convergence rate of algorithms based on this method may not necessarily be favourable when compared with the Newton type methods or other steepest descent methods (see for example, [149], pp 229). Nevertheless it has been used by many authors perhaps because it is easy to implement and easy to visualize its process like, for example, the so-called hill climbing method. The coordinate descent method is a variation of the steepest descent method but belongs to an entirely different class of algorithm.

Descent directions along each coordinate $x_i$, with all other components fixed, are found by solving the following optimization problem.

$$\min_{x_i} f(x_1, \ldots, x_n), \text{ where } f : \mathbb{R}^n \to \mathbb{R}^1, \ f \in C^1. \quad (3.13)$$

That is, for a given point $x^0 = (x_1, \ldots, x_n)$, a new improved solution $x^{k+1}$ is sought by changing a single component $x_i$. A local minimum of $f$ will then be found by sequentially minimizing with respect to the remaining components. It is well known that this method has both local and global convergence properties [149]. However, the method should be applied with care. The proof of convergence of the coordinate descent method is based on the assumption that a search along any coordinate direction will yield a unique minimum point. Thus, if the minimum point is not unique along each coordinate direction then the algorithm may fail to generate a direction of improvement for the function under consideration.

### 3.4.2.3 Robust Pole Assignment Algorithm

The robust full-state feedback pole assignment algorithm by Kautsky et al [121] is seen to provide good results [176], [179]. A related output feedback pole assignment algorithm based on the state feedback algorithm has also been developed. This algorithm will first be presented.
3.4.2.3.1 Robust Pole Assignment Algorithm for State Feedback

(1) Problem Formulation

Consider

\[ \dot{x} = Ax + Bu \]  

(3.14)

where \( A \in \mathbb{R}^{nxn} \), \( B \in \mathbb{R}^{nxm} \), \( x \in \mathbb{R}^n \), \( u \in \mathbb{R}^m \)

The problem is to find \( X \) such that

\[
\min K_2 = \| X \|_2 \| X^{-1} \|_2 \\
\text{s.t.} \quad (A + BK)X = X \Lambda 
\]

(3.15)\hspace{1cm}(3.16)

where \( \Lambda \) is an \( n \times n \) (block) diagonal matrix whose (block) diagonal elements are the eigenvalues to be assigned.

A necessary and sufficient condition for existence of a feedback gain matrix \( K \) and eigenvector matrix \( X \) is stated by the following theorem.

**Theorem 3.10** [121]:

Given \( A \) and \( X \) non-singular, then there exists feedback gain matrix \( K \) and a solution to (3.16) if and only if

\[
U_B^T (AX - X\Lambda) = 0 
\]

(3.17)

where \( B = (U_0, U_1) \begin{bmatrix} Z_B \\ 0 \end{bmatrix} \), \( U = (U_0, U_1) \) orthogonal, \( Z_B \) nonsingular

(3.18)

The feedback gain is given by

\[
K = Z^{-1}U_0^T (X\Lambda X^{-1} - A) 
\]

(3.19)

(2) Algorithm

Kautsky et al [121] introduce three methods for solving the robust feedback pole assignment problem. One of the methods, namely method 1 is as follows.
Algorithm 3.2:

Step 1. Define $X_j = (x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_n)$

Step 2. Perform QR decomposition such that $X_j = (Q_j, q_j) \begin{bmatrix} R_j \\ 0^T \end{bmatrix}$

Step 3. Solve $\min \|X^{-1}\|_F = \| \begin{bmatrix} R_j^{-1}Q_j^T S_j \\ I_m \end{bmatrix} \rho_j \omega_j \|_2$, where $x_j = S_j \omega_j$, $\rho_j = 1/(q_j^T S_j \omega_j)$, $\| \omega_j \|_2 = 1$.

3.1: Find unitary matrix $\tilde{P}_j$ such that $S_j^T q_j = \tilde{P}_j (\sigma_j e_m) = \sigma_j (P_j p_j) e_m = \sigma_j p_j$

3.2: Solve $\min \| H_1 P_j W_j + H_1 P_j \|_2$, where $\begin{bmatrix} R_j^{-1}Q_j^T S_j \\ I_m \end{bmatrix}$, where $w_j = \rho_j P_j^T \omega_j$

Step 4. Update $x_j = S_j \omega_j = S_j (P_j w_j + p_j)/\rho_j \sigma_j$, where $\rho_j^2 = (w_j^T w_j + 1)/\sigma_j^2$.

Numerical examples are solved using this algorithm in Section 3.6.

3.4.2.3.2 Robust Pole Assignment Algorithm for Output Feedback

To solve output feedback pole assignment problem given in (3.1), Chu et al. (1985) derived a necessary and sufficient condition to find a real matrix $K$ and a nonsingular matrix $X$ satisfying

$$ (A + BK) \bar{x}_j = \lambda_j x_j, \quad j = 1, \ldots, n \quad (3.20) $$

$$ U_i^T (AX - XL) = 0 \quad (3.21) $$

$$ (X^{-1}A - XL^{-1}) V_i = 0 \quad \text{or} \quad (Y^T A - LY^T) V_i = 0, \quad (3.22) $$

where $B = (U_0, U_1) \begin{bmatrix} Z_B \\ 0 \end{bmatrix}$, $C^T = (V_0, V_1) \begin{bmatrix} Z_C \\ 0^T \end{bmatrix}$

$$ U = (U_0, U_1), \quad V = (V_0, V_1) \text{ orthogonal} $$

$Z_B, Z_C$ nonsingular, $Y = [(X^{-1})^T]_{\text{normalized}}$

$$ \lambda = \text{diag}(\lambda_1, \ldots, \lambda_n) $$
They formulate the robust output pole assignment problem as follows.

(1). Problem Formulation

Consider the problem described in (3.1). It is desired to find $X$ and $K$ such that

$$\min \|DX^{-1}\|_F^2$$

s.t. $$(A + BK)X = XA$$

where $D = \text{diag}(d_1, \ldots, d_n)$, $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n)$, $d_j(j = 1, \ldots, n)$: weighting factors, $\lambda_j(j = 1, \ldots, n)$: poles to be assigned.

Or equivalently,

$$\min F_1(X) = \|DX^{-1}\|_F^2$$

s.t. $F_2(Y_j) = \|T_j^T y_j\|_2/\|y_j\|_2^2 = 0$

$$x_j \in \mathcal{S}_j$$

where $Y = (y_1, \ldots, y_n) = [(X^{-1})^T]_{\text{normalized}}$

A penalty function is then considered to relax the constraint $F_2 = 0$ such that

$$\min F = F_1 + F_2 = \|DX^{-1}\|_F^2 + \sum \omega_j \|T_j^T y_j\|_2^2$$

(2). Algorithm

Chu et al's algorithm includes the following steps.

Algorithm 3.3:

Step 1. Find QR decomposition of $B$ and $C$ as in (3.23).

Step 2. Construct orthonormal bases $S_j, \mathcal{S}_j$ and $T_j, \hat{T}_j$ as in (3.24)-(3.25) by

$$[U_j^T(A - \lambda_j I)^T = (\mathcal{S}_j, S_j) \begin{bmatrix} R_j \\ 0 \end{bmatrix}, \quad [V_j^T(A^T - \lambda_j I)^T = (\hat{T}_j, T_j) \begin{bmatrix} K_j \\ 0 \end{bmatrix}$$
Step 3. Find $M = A + BKC$ such that $MX = XA$, $K = ZBUT'(XA^{-1} - A)VZ^{-1}$.

The original paper describes the design philosophy, yet does not provide a precise numerical implementation. In particular, the numerical method used to minimize $F = F_1 + F_2$ by choosing $x_j \in S_j$ is not described. Since $F_1$ and $F_2$ are coupled by $X$ and $Y$ such that $Y^T = X^{-1}$, if $X$ is nonsingular, and neither $X$ and $Y$ can be expressed in terms of each others, no standard minimization technique can be directly applicable. Slade [226] has further developed an algorithm based on the above original algorithm. Numerical results obtained by the Chu et al./Slade algorithm are compared with Mininis algorithm to evaluate their efficiencies in section 3.6. Evaluation of any algorithm is not a simple matter because it involves many factors to be considered. An evaluation based on one factor only, say, accuracy or least computation time, etc., will be inappropriate. To evaluate any software package it is necessary to understand the general concept of the algorithm and the performance evaluation methods employed. This will now be explored.

3.5 Basis for Evaluation of Algorithms

3.5.1 Concept of Algorithm

The algorithm is defined in many different ways. In the Oxford English Dictionary [181], algorithm is defined as: a process, or set of rules, usually are expressed in algebraic notation, now used especially in computing, machine translation and linguistics. Also, it is cited as: Algorithm, used by computer programmers to designate the numerical or algebraic notations which express a given sequence of computer operations, define a programme or routine conceived to solve a given type of problem. Trakhtenbrot et al [232] defines it as a list of instructions specifying
a sequence of operations which will give the answer to any problem of given type. By the Dictionary of Computing [52], an algorithm is a prescribed set of well-defined rules or instructions for the solution of a problem in a finite number of steps. Horowitz et al [113] define it more specifically as follows. An algorithm is a finite set of instructions which, if followed, accomplish a particular task. In addition every algorithm must satisfy the following criteria:

i) input: there are zero or more quantities which are externally supplied;

ii) output: at least one quantity is produced;

iii) definiteness: each instruction must be clear and unambiguous;

iv) finiteness: if we trace out the instructions of an algorithm, then for all cases the algorithm will terminate after a finite number of steps;

v) effectiveness: every instruction must be sufficiently basic that it can in principle be carried out by a person using only pencil and paper. It is not enough that each operation be definite as in (iii), but it must also be feasible.

Thus an algorithm is a finite ordered sequence of operations, which precisely specifies how given elements of a set of inputs have to be transformed into elements of a set of outputs. More mathematically rigorous definitions can be found elsewhere in [23], [149], [190], [257]. For example, An Algorithm is a point-to-set mapping such that given a closed subset \( X \) of a Banach space \( B \), to construct points in \( X \) which have property \( A \) [140], [190].

Thus, more precisely, by algorithm A at a given point \( x \in X \), a sequence \( \{x_1, x_2, \ldots\} \), where \( x_{k+1} \in A(x_k), \forall k \) can be generated. Note that an algorithm is a point-to-set mapping rather than a point-to-point mapping. Therefore, not likewise any point-to-point mapping which may generate the same sequence, an algorithm defined by point-to-set mapping may not generate the same sequence though they converge, if
it is the case, to the same point. In practice, different computer programs to solve
the same problem may not generate the same results even though they start at the
same point. However, it may not be serious as long as they converge to an optimal
solution.

3.5.2 Closedness of Algorithm

Closedness of an algorithm for point-to-set mapping is a generalized concept of con­tinuity of a function for point-to-point mapping. It can be defined more rigorously
as follows.

Definition 3.2: Closed map ([257], pp 88)

A point-to-set map $A : X \to Y$ is closed at $x \in X$ if $x_k \to x$, $x_k \in X$ and
$y_k \to y$, $y_k \in Y$ imply $y \in A(x)$.

Closedness of an algorithm is the key for global convergence of the sequence gen­erated by the algorithm. The following global convergence theorem provides technical
conditions that ensure convergence of the algorithm mapping.

3.5.2.1 Convergence Properties and Rate of Convergence

Theorem 3.10: Global convergence theorem ([257], pp 91)

Let a point-to-set map $A : X \to X$ be an algorithm to seek a minimum, which
given a point $x_0 \in X$, generates the sequence $(x_k)_{k=0}^{\infty}$. Let a solution set
$\Omega \subset X$ be
given, and suppose

1. All points $x_k$ are in a compact subset of $X$.

2. There is a continuous function $Z : X \to \mathbb{R}^1$ such that

i) if $x \not\in \Omega$, then $Z(y) < Z(x)$, $\forall y \in A(x)$

ii) if $x \in \Omega$, then $Z(y) \leq Z(x)$, $\forall y \in A(x)$
iii) the map $A$ is closed at $x$ if $x \not\in \Omega$.

Then the limit of any convergent subsequence is a solution.  

Even when an algorithm mapping is not closed, a sequence generated by the algorithm may converge but not necessarily to an optimal solution. This is the reason why an algorithm is required to be closed.

### 3.5.3 Performance Evaluation of Algorithms

It cannot be concluded that an algorithm is good simply because it is closed and converges to an optimal solution. There are many other factors which affect the performance of an algorithm. For example, even if a sequence generated by the algorithm converges to an optimal solution, the speed of convergence or number of iterations to reach that solution with a required accuracy is also important. No optimization algorithm may have the credit of the best algorithm for solving a given optimization problem. An algorithm which solves a particular problem successfully and efficiently may not necessarily solve other problems with the same degree of satisfaction. It is quite often the case that when the same algorithm is applied to solve very similar problems, it performs differently. There is no general rule for what evaluation criteria should be employed. It may be advisable to include as many performance measures as possible. For example, robustness, reliability, CPU time, number of function evaluations, number of problems solved, accuracy of solution obtained, convergence rate, etc. Even if an algorithm is developed into some optimization codes, the relative performance of these codes may depend on the computer and compiler used, machine language routines, machine work loads at a given instant of time, desired degree of accuracy, choice of problems and performance measures, skill and experience of the users etc.

Schittkowski [214] includes several factors as performance criteria to evaluate
nonlinear programming codes, such as, accuracy, efficiency, reliability, global convergence, performance for solving degenerate problems, ill-conditioned problems and indefinite problems, sensitivity to the variation of the problems and starting point, ease of use, etc. Some examples of how to measure these performance criteria are as follows.

1. Accuracy:
   Accuracy is measured by the geometric mean of
   i) the absolute objective function values
   ii) the sum of the constraint violations
   iii) Euclidean norms of the Kuhn-Tucker point or the arithmetic mean of the numbers of exact digits

2. Efficiency
   Efficiency is measured by execution time and number of function and gradient evaluations

3. Reliability
   Number of unsuccessful test runs, the average objective function values and sums of constraint violations of the non-successful solutions are considered to compare successful and unsuccessful of algorithm.

4. Global convergence
   The percentage of global convergences, geometric mean of objective function values, geometric mean of sum of constraint violations are considered.

5. Performance for solving degenerate problems
   If at least one of the active constraints is redundant at the optimal solution, i.e., if at least one of the Lagrangian multipliers associated with the equality constraints vanishes, it is called a degenerate problem. The criteria to be evaluated are
i) Are there any difficulties for solving degenerate problems?

ii) Does the algorithm take any advantage of redundant constraints?

iii) Does degeneracy influence the final accuracy of solution?

6. Performance for solving ill-conditioned problems

It is well known that the local convergence of NLP code depends heavily on the condition number of the Hessian matrix at the optimal point. In general, to solve an ill-conditioned problem, a longer calculation time and more function or gradient evaluations are required. Thus a good algorithm/code must be efficient for solving ill-conditioned problems.

7. Performance for solving indefinite problems

If the Hessian of Lagrangian function happens to be positive definite at the stationary point, it will be rather lucky. But in general, this is not the case. Thus an algorithm can be evaluated for its performance by observing whether an indefinite Hessian of the Lagrangian leads to any numerical difficulties, affects a different final accuracy, or decreases the efficiency.

8. Sensitivity to slight variations of the problems

One optimization code may be more sensitive to slight variations of the objective function or constraints than another. In fact performance of an algorithm may be influenced by the special structure of the objective function or of the constraints.

9. Sensitivity to position of the starting point

If an algorithm has a globally convergent property, a starting point should not affect successfully generating an optimal solution. However, if this is not the case, the starting point may seriously affect reaching an optimal solution. To evaluate the sensitivity to the position of the starting point for a given code, Schittowski suggested a concept of relative efficiency. This is measured
by the result obtained from a starting point close to the solution and one far away from the solution, then divide the last results by the first to obtain the relative accuracy.

10. Ease of use

Schittkowski suggested the following items to evaluate the ease of use of an optimization code.

i) Quality of documentation

ii) Provision of problem data and function

iii) Program organization

iv) Sensitivity to input parameters

The numerical results obtained by the direct method and the robust assignment method previously discussed will now be compared in the context of the evaluation criterion discussed in this section.

3.6 Numerical Examples

Some numerical examples are tested to compare the existing feedback pole assignment algorithms. Particular attention is paid to the relative performance of these algorithms which attempt to optimize some robustness criteria and those which just aim to assign poles to prescribed locations.

Example 3.1 [121]:

Consider the following problem with dimension $n = 4$, $m = 2$.

$$ A = \begin{bmatrix} 1.3800 & -0.2077 & 6.7150 & -5.6760 \\ -0.5814 & -4.2900 & 0.0 & 0.6750 \\ 1.0670 & 4.2730 & -6.6540 & 5.8930 \\ 0.0480 & 4.2730 & 1.3430 & -2.1040 \end{bmatrix}, \quad B = \begin{bmatrix} 0.0000 & 0.0000 \\ 5.6790 & 0.0000 \\ 1.1360 & -3.1460 \\ 1.1360 & 0.0000 \end{bmatrix} $$
where the open-loop poles are \((1.99095, -5.05657, -8.66589, 0.06350)\) and the poles to be assigned are \((-0.2, -1.0, -5.0566, -8.6659)\). The results obtained by Kautsky et al.'s algorithm during 500 sweeps are as follows.

The feedback gain matrix obtained at sweep 1:

\[
\begin{bmatrix}
-0.0709213216 & 0.0878183739 & -0.5473124267 & 0.4190353467 \\
1.4728922602 & 0.0938925515 & 1.1466935101 & -0.4367901051 \\
\end{bmatrix}
\]

The poles assigned are \((-0.20000, -1.00000, -5.05660, -8.66590)\). The condition numbers obtained at the 1st, 5th, 10th, 20th, 50th and 100th sweeps are 5.182565, 3.678122, 3.514640, 3.477320, 3.472327, 3.472492, respectively. The feedback gain matrices obtained at the 200th and 500th sweeps are both

\[
\begin{bmatrix}
0.0837229027 & -0.0530275509 & -0.0107110144 & -0.0334233027 \\
1.2507109585 & 0.2774205206 & 0.9228289422 & -0.2524611347 \\
\end{bmatrix}
\]

with condition numbers 3.47249219. After 200 sweeps, the feedback gain matrices and the condition numbers have not changed up to 10 significant digits.

**Example 3.2** [166]:

Consider the following system,

\[
A = \begin{bmatrix}
2 & 4 & 6 & 7 & 8 & 9 \\
1 & 3 & 2 & 6 & 4 & 7 \\
1 & 4 & 6 & 7 & 8 & 9 \\
0 & 7 & 4 & 2 & 4 & 5 \\
0 & 0 & 6 & 7 & 8 & 9 \\
0 & 0 & 0 & 0 & 5 & 8
\end{bmatrix}, \quad B = \begin{bmatrix}
10^{-4} & 10^2 & 10^2 \\
0 & 10^{-4} & 10^2 \\
0 & 0 & 10^{-4} \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix},
\]

where the open-loop poles are \((23.11154, -4.42995, -0.35921, 4.65063, 4.65063, 1.37636)\) and the poles to be assigned are \((1, 2, 4, 5, 7, 8)\). Note that this is not a stable pole assignment but is used simply to demonstrate how the algorithm works. This is an ill-conditioned problem as will be seen later. A direct state feedback pole assignment algorithm proposed by Miminis and Paige (1988) assigns the
poles to \((1, 4, 7, -0.876704, 5.06053 \times 10^4 \pm i1.23907 \times 10^6)\), with the feedback gain matrix

\[
K = \begin{bmatrix}
9.99998 \times 10^3 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.99999 \times 10^6 & -1.21067 \times 10^3 & -1.53738 \times 10^3 & 1.31735 \times 10^3 & 0.0 \\
1.0 \times 10^2 & 1.21067 \times 10^3 & 1.4667 \times 10^3 & 1.4 \times 10^3 & 2.6667 \times 10^3 & 3.3767 \times 10^3
\end{bmatrix}
\]

They explain why their algorithm assigns not all poles to the desired locations as follows.

... Despite the accuracy in \(K\), the computed eigenvalues of \(A - BK\) were found to be \((1, 4, 7, -0.876704, 5.06053 \times 10^4 \pm i1.23907 \times 10^6)\). This is so because the condition number of the eigenproblem of \(A - BK\) computed by MATLAB was found to be “infinite”, meaning that it was greater than \(2^{57} \approx 10^{17}\).

Nevertheless, Kautsky et al’s algorithm produces much better results after 5 sweeps as follows. The feedback gain matrix is

\[
K = \begin{bmatrix}
-130.01391745 & 988.69524643 & 1796.80614930 & -137.31038499 & 1018.73690019 & 2003.7264558 \\
0.0859850470 & -0.2036160392 & -0.3364297161 & 0.267983175 & 0.0603151733 & -0.0700017009 \\
-0.0257840681 & -0.0830722082 & -0.1612857235 & -0.2610678062 & -0.3331038301 & -0.4975147044
\end{bmatrix}
\]

with condition number \(\kappa_2(X) = 264.245\) or \(\kappa(A - BK) = 1936.743\).

Results obtained from several more algorithms for feedback pole assignment in the literature are compared with Kautsky et al’s robust feedback pole assignment algorithm. It has been seen that their robust feedback pole assignment algorithm is as efficient as the other algorithms both in accuracy and efficiency. Though the problem seems to be slow convergence rate near to an optimal solution as seen in Example 3.1, it may not be serious since, in practice, the algorithm generates a robust solution in a few sweeps, in general. In fact, as far as the accuracy of assigned poles and robustness measured in terms of condition number are concerned, no more superior full-state feedback algorithm was found except perhaps the one proposed in Chapter 5.
Some test results by Chu et al.'s robust output feedback pole assignment algorithm and Miminis output feedback pole assignment algorithm based on a direct eigenstructure assignment approach are summarized as follows.

Example 3.3 [65]:
Consider the following problem with dimension $n = 4$, $m = 3$, $p = 3$.

\[
A = \begin{bmatrix}
0 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 \\
-1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\end{bmatrix}, \quad B = \begin{bmatrix}
0 & 0 & 1 \\
1 & 0 & 0 \\
0 & 0 & 0 \\
0 & 1 & 0 \\
\end{bmatrix}, \quad C = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

where the open-loop poles are $(0, 0, 1.618, -0.618)$, and poles to be assigned are $(-1, -2, -3, -4)$. This is the case where $n = A < m + p = 6$ and it is not a well conditioned problem as will be seen below. Chu et al. [1985] assign the poles at $(-0.973, -2.226, -2.918, -3.80)$ with a condition number 778.2. It should be noted that the distance between desired and actual poles is unconstrained in this problem formulation.

For this example, by Miminis algorithm, poles are assigned at $(0, -1.0, -2.99999, -3.99999)$, that is only 3 poles are assignable with a condition number 61.143, and the feedback gain is

\[
\begin{bmatrix}
20.49999970976 & -20.49999970976 & 0. \\
0. & 0. & 0. \\
8.99999931310 & -8.49999931310 & 0. \\
\end{bmatrix}
\]

Example 3.4 ([121], adapted):
Slade [226] considers the following problem with dimension $n = 4$, $m = 2$, $p = 3$.

\[
A = \begin{bmatrix}
1.3800 & -0.2077 & 6.7150 & -5.6760 \\
-0.5814 & -4.2900 & 0.0 & 0.6750 \\
1.0670 & 4.2730 & -6.6540 & 5.8930 \\
0.0480 & 4.2730 & 1.3430 & -2.1040 \\
\end{bmatrix}
\]
where the open-loop poles are \((1.99095, -5.05657, -8.66589, 0.06350)\), and poles to be assigned are \((-0.20, -1.0, -5.0566, -8.6659)\). Slade, using the modified algorithm of Chu et al, assigns the poles at \((-0.1995, -1.0004, -5.0566, -8.6654)\) with a condition number 11.308 and feedback gain matrix obtained as

\[
K = \begin{bmatrix}
-0.1483 & -0.1890 & 0.1214 \\
-0.0862 & 0.3830 & 0.4389
\end{bmatrix}
\]

With this example, several different cases are tested by the Miminis algorithm and the results are as follows.

**Case 1:**
To assign the poles at \((-1.0, -0.2, -5.0566, -8.6659)\), the assigned poles are
\((-5.056002361942, -0.200002123603, -8.665899158183, -0.999996519309)\), with a condition number 51.192 and the output feedback gain matrix

\[
\begin{bmatrix}
0.995047790101 & 0.043991603909 & -0.886643164562 \\
0.541029807322 & -0.121344438505 & -0.77043957995
\end{bmatrix}
\]

**Case 2:**
To assign the poles at \((-1.0, -5.0566, -0.2, -8.6659)\), the same poles as Case 1, but with larger condition number 104.293 are obtained.

**Case 3:**
To assign the poles at \((0.0, -0.001, -1.0, -4.0)\), the poles assigned are
\((-4.00000270007, -0.0004992924122, -0.000499924122, -0.999999093228)\) with condition number 10834.609 and feedback gain matrix

\[
\begin{bmatrix}
-0.489655125641 & 0.021206924198 & 0.104011028941 \\
0.306990428909 & -0.1473657924857 & -0.58112536679
\end{bmatrix}
\]
Case 4:
To assign the poles at \((-0.99999, -5.056, -1, -8.6659)\), obtained the poles of
\((-5.056005328606, -8.665898153773, -0.999993051615, -0.999993051615)\) with con­
dition number of 26873.396 and feedback gain matrix

\[
\begin{bmatrix}
2.288880056918 & -0.095560483502 & -2.50772290567 \\
1.104853108409 & -0.178178228622 & -1.468130926083
\end{bmatrix}
\]

As seen above, for some cases, the Mininis algorithm produces closed-loop ma­tri­ces with reasonable magnitude of condition numbers. However, if the priority(order) of assigning the given poles is altered, a different result is obtained. In particular, as shown in cases 3 and 4, if some of the eigenvalues to be assigned are not sufficiently distinct, the resulting closed-loop matrix turns out very ill-conditioned and even the algorithm fail to assign poles arbitrarily close to the desired ones.

The reason for obtaining such an unreliable result seems to be mainly because the algorithm does not attempt to optimize any performance assessment, except the angle between \(H_0\) and \(W_0\), where \(CT = (H_0, H_1)\), \(Z_1\), and \([U^H(A_1 - \lambda I)]^T = (H_0, H_1)\), is minimized which does not guarantee to achieve an optimal solution, but simply one of the feasible solutions.

On the other hand, when a matrix is reduced to a lower dimensional one by sequentially applying the deflation method, none of the eigenvalues of the deflated matrices are exactly the same as the original ones (see example in [73], pp112). Moreover, when some eigenvalues to be assigned are too close together, the associated eigenvectors may be very close to each other. Hence, the reduced matrix may not necessarily be the exact representation of the original matrix in the sense that the eigenstructure is preserved. For this reason, the algorithm based on the deflation method may produce unsatisfactory results as shown in case 3 and case 4.
of example 3.

There are some other restrictions of this algorithm. This algorithm can assign the poles only up to \( \min\{n, m + p - 1\} \). Furthermore, a pole cannot be assigned if it is a strong transmission zero of the system. Another restriction of the algorithm is that the algorithm stops when the corresponding input distribution matrix including those in the transposed system becomes square. In this case a modification of the original algorithm as presented in the literature is required. It is necessary to solve the system of equations \((A_i - \lambda_i I_i) = BK_C\) with respect to \(K\), instead of continuing the original algorithm. It is also necessary to modify the algorithm for the assignment of complex poles. The most significant drawbacks of this algorithms are:

i) There is no guarantee that the poles will be assigned with favourable condition numbers.

ii) It is not possible to control \(n - (m + p - 1)\) poles if \(n > m + p - 1\).

Though the full-state feedback pole assignment algorithm proposed by Kautsky et al [121] is seen to provide satisfactory results, the results obtained by Chu et al's output feedback pole assignment algorithm are, in general, not completely satisfactory because if the condition numbers are favourable, the results do not satisfy the constraints and vice versa. The problem with the algorithm may be, as indicated earlier, the case when the eigenvector matrix \(X\) is singular where the objective function is undefined. The original model constructed is in the form of

\[
\min \|X^{-1}\|_F^k + \sum \omega_j \|F_j y_j\|_2^2
\]

or

\[
\min \|X^{-1}\|_F^k \quad s.t \quad F_p = \sum \|F_j y_j\|_2^2 = 0
\]

Note that it is obvious that (3.34) has a trivial solution \(y_j = 0, \forall j\), for which \(X^{-1}\) is not defined. However, the model does not exclude this case which may lead
to an incorrect result. Moreover, it is not known if such a point exists elsewhere theoretically. Another possible problem with the algorithm based on such a penalty function method is that the Hessian of the penalty function becomes increasingly ill-conditioned as the penalty parameter approaches zero. Consequently, the algorithm may not guarantee an optimal solution.

3.7 Concluding Remarks

In this chapter, output feedback pole assignment methods and some selected algorithms have been reviewed. Some properties relating to pole assignability are surveyed. For full state feedback control, if the system is controllable, an arbitrary set of poles may be assigned by selecting an appropriate state feedback. However, for output feedback control, beyond the controllability assumption, an observability assumption and some dimensionality requirements are essential to guarantee the pole assignability. Some basic concepts of algorithm and related evaluation criteria are reviewed. A direct pole assignment algorithm and a robust pole assignment algorithm are compared using some numerical examples to evaluate their efficiencies. Examples show that, though the efficiency measured in terms of execution time was not significantly different in each method, the robust pole assignment algorithm produces much better controller design than the other feedback pole assignment algorithms which do not attempt to achieve any robust solution. Indeed, an arbitrary choice of closed-loop poles may result in a poor controller design. Having demonstrated the need for the incorporation of appropriate robustness criteria, subsequent chapters will further explore the concept of effective and efficient robust pole assignment.
Chapter 4

A Framework for Robust Pole Assignment by Output Feedback

4.1 Introduction

As stated in the previous chapter, the location of the poles in the complex plane entirely characterizes the stability of a linear time invariant system. However, an arbitrary choice of closed-loop poles may result in a poor controller design. In other words, only placing the poles in desired locations for the nominal system is not sufficient for designing a satisfactory system because uncertainties of model, parameter and/or structure may affect the pole locations. Thus it is required to assign the poles in some optimal way. The concept of robustness which is discussed in Chapter Two is such a criterion that must be considered. A system needs to be designed so as to remain stable in spite of model inaccuracies and parameter variations, or so that the closed-loop eigenvalues are as insensitive as possible to perturbations in the system parameters. Pole assignment with the incorporation of a robustness criterion is called the robust pole assignment problem. In particular, this
is concerned with robust pole assignment in a specified region rather than assigning to exact positions. Pole assignment in a specified region can be argued to be a more realistic problem from the practical point of view. Methods which design output feedback control schemes for exact pole placement may achieve the design objectives for some nominal linear system representation only. These closed-loop poles will vary with system changes. In practical terms a pole assigned to an exact but sensitive location may move further away from its precise location than a pole assigned to an insensitive location within a chosen band. Although the problem of pole assignment by output feedback has been studied for a number of years, very few authors have considered the problem of pole assignment in prescribed regions using output feedback.

The ideas of assigning poles to prescribed regions were originally suggested in terms of state feedback. See, for example, Anderson and Moore [14], Kawasaki and Shimemura [122], Heger and Frank [108], Huang and Lee [112], Lee and Juang [142], Furuta and Kim [77], Wittenmark, Evans and Soh [256], Abdul-Wahab and Zohdy [2], Shieh et al [222], Rachid [197], and Chow [42]. Most of these studies simply aim to assign the poles in some specified region, but have not attempted to optimize any performance criterion for the poles assigned. In other words, what they have done is to find a ‘feasible solution’ which may not be ‘optimal’. For example, Rachid [197] derives sufficient conditions for pole assignment within a specified circle for perturbed systems. This does not incorporate any performance assessment. Others consider pole assignment within a region which may be specified by a circle, a rectangle, a parabola, or a hyperbola. This is accomplished for an LQ-problem by solving Riccati equations. References to the assignment of poles to prescribed regions by output feedback can be found in Sirisena and Choi [225], Chen et al [41], Champetier and Magni [38], and Haddad and Bernstein [106]. The algorithm suggested by Sirisena et al [225] is based on forming the characteristic equation and modifying the feedback gain sequentially so that all poles to be assigned are moved
toward the prescribed region. Chen et al. [41] suggest an algorithm to assign exactly a $\max(m,p)$ poles, when $m + p < n$. A feedback gain producing acceptable locations for the remaining poles is found by calculating the coefficients of the residual polynomial corresponding to those unassigned poles, and a root locus plot is used to establish the loci of these poles. Champetier et al. [38] propose another polynomial method to assign some poles exactly and locate the others in a specified region of the complex plane. This is achieved in a similar way to Chen et al. [41]. Haddad et al. [106] develop a design procedure that combines linear-quadratic optimal control with regional pole assignment. They introduce some necessary and sufficient conditions for characterizing output-feedback controllers with bounded performance and regional pole constraints.

In this chapter, a robustness measurement in terms of condition number is discussed. An appropriately defined condition number of the closed-loop system matrix is proven to be a reasonable criterion to evaluate system robustness. Some properties of condition number and numerical examples are introduced to demonstrate how the condition number of a given matrix measures the sensitivity of its eigenvalues with respect to the variations in the entries of the matrix. It is then shown that the robust output feedback pole assignment problem can be formulated as an optimization problem which can be solved effectively by applying a typical nonlinear programming technique with some modification. This formulation enables the incorporation of appropriate robustness criteria. A mathematical model is formulated for the given problem and the necessary and sufficient conditions for the existence of feasible solutions in terms of the left/right eigenvector matrix and the feedback gain matrix are given. The mathematical formulation introduced and the conditions for existence of solutions are flexible so that they are applicable both for full-state feedback and output feedback pole assignment problems. That is, it is shown that a full-state feedback pole assignment problem can be solved as a special case of output feedback pole assignment problem by the proposed model.
4.2 Measure of Robustness

In the formulation of an optimization problem for control system design, an appropriate choice of objective function is very important. It must be able to measure the system's performance as accurately as possible, but it should also be mathematically tractable. In matrix theory, it is well known that an appropriately defined condition number yields the maximum effect of the perturbations in the data within the matrix. Thus, in order to improve the robustness in the sense that the assigned poles are as insensitive as possible to perturbations in the closed-loop system, it will be reasonable to define a measure of effectiveness by a condition number of the closed-loop system matrix

\[ (A + BKC). \]  

(4.1)

Based on the definition of Wilkinson ([252], pp86), the condition number of (4.1) may be defined by several ways as follows.

\[ \kappa_1(X) = \|X\|_1\|X^{-1}\|_1 \]  

(4.2)

\[ \kappa_2(X) = \|X\|_2\|X^{-1}\|_2 \]  

(4.3)

\[ \kappa_F(X) = \|X\|_F\|X^{-1}\|_F \]  

(4.4)

\[ \kappa_p(X) = \|X\|_p\|X^{-1}\|_p \]  

(4.5)

or \[ \kappa_\infty(X) = \|X\|_\infty\|X^{-1}\|_\infty, \]  

(4.6)

where \( X \) is the right eigenvector matrix of (4.1) and the corresponding matrix norms are defined as

\[ \|X\|_1 = \max_j \sum_i |x_{ij}| \]  

(1-norm)  

(4.7)

\[ \|X\|_2 = (\text{maximum eigenvalue of } X^TX)^{\frac{1}{2}} \]  

(spectral norm, bound norm or 2-norm)  

(4.8)

(4.9)
CHAPTER 4. ROBUST POLE ASSIGNMENT

\[ ||X||_p = \left( \sum_{i,j} |x_{ij}|^p \right)^{\frac{1}{p}} \]  \hspace{1cm} (4.10)

(Euclidean norm, Frobenius norm, Schur norm, or Hilbert-Schmidt norm)

\[ ||X||_p = \left( \sum_{i,j} |x_{ij}|^p \right)^{\frac{1}{p}}, 1 \leq p \leq 2 \hspace{1cm} (a \text{ special case of H"{o}lder norm; } p \geq 1) \]  \hspace{1cm} (4.11)

\[ ||X||_\infty = \max_j \sum_i |x_{ij}| \hspace{1cm} (\infty-\text{norm}). \]  \hspace{1cm} (4.12)

Matrix norm of $X$ satisfies the following axioms ([252], pp55).

1. $||X|| > 0$ unless $X = 0$  \hspace{1cm} (4.13)
2. $||kX|| = |k||X||$ for any complex scalar $k$  \hspace{1cm} (4.14)
3. $||X + Y|| \leq ||X|| + ||Y||$  \hspace{1cm} (4.15)
4. $||XY|| \leq ||X|| \cdot ||Y||$  \hspace{1cm} (4.16)

Other matrix norms which satisfy the above relations can also be used to define a condition number. A different definition of the condition number of a given matrix $A$ may be found in Lancaster et al ([141], pp385) and Golub et al ([92], pp79) as follows.

\[ \kappa(A) = ||A|| \cdot ||A^{-1}||. \]  \hspace{1cm} (4.17)

The following simple examples demonstrate how the condition number (4.17) relates to the system robustness.

Consider the following linear system of equations

\[ A_1x_1 = b_1, \]  \hspace{1cm} (4.18)
\[ A_2x_2 = b_2, \hspace{1cm} \text{where} \]  \hspace{1cm} (4.19)

\[
A_1 = \begin{bmatrix}
0.110 & 0.440 \\
0.220 & 0.881
\end{bmatrix}, \quad A_2 = \begin{bmatrix}
0.750 & -0.610 \\
0.660 & 0.790
\end{bmatrix}, \quad b_1 = \begin{bmatrix}
1.0 \\
2.0
\end{bmatrix}, \quad b_2 = \begin{bmatrix}
1.0 \\
2.0
\end{bmatrix}.
\]

The solution to these equations can be easily found as

\[
x_1 = \begin{bmatrix}
9.02090909 \\
0.00000000
\end{bmatrix}, \quad x_2 = \begin{bmatrix}
2.01989750 \\
0.84413627
\end{bmatrix}, \hspace{1cm} \text{respectively}
\]
In case there are some perturbations in $A_i$ or $b_i$, for example,

$$A_1 + \delta A_1 = \begin{bmatrix} 0.110 & 0.440 \\ 0.221 & 0.881 \end{bmatrix}, \text{ where } \delta A_1 = \begin{bmatrix} 0 & 0 \\ 0.001 & 0 \end{bmatrix},$$

or

$$b_1 + \delta b_1 = \begin{bmatrix} 1.0001 \\ 1.9999 \end{bmatrix}, \text{ where } \delta b_1 = \begin{bmatrix} 0.0001 \\ -0.0001 \end{bmatrix},$$

respectively, the solution $x_1$ to (4.18) corresponding to each perturbation will then be perturbed as

$$\begin{bmatrix} -3.03030303 \\ 10.29181818 \end{bmatrix}, \text{ or } \begin{bmatrix} 3.03030303 \\ -0.30000000 \end{bmatrix},$$

respectively. (4.20)

Likewise, if $A_2$ or $b_2$ is perturbed by the same amounts above, the new solutions $x_2$ to (4.19) will be

$$\begin{bmatrix} 2.01860005 \\ 0.84261482 \end{bmatrix}, \text{ or } \begin{bmatrix} 2.01991559 \\ 0.84399457 \end{bmatrix},$$

respectively. (4.21)

Note that small perturbations of the elements in $A_1$ and $b_1$ affect the solutions by as much as 135 %, while the perturbations in $A_2, b_2$ affect them at most 0.06 %. The condition numbers of $A_1$ and $A_2$ defined by (4.17) with induced 2-norm are 9366.0089 and 1.0663, respectively. The larger condition number of the given matrix corresponds to the larger perturbation in its solution. It can be shown [92] that

for the linear systems of equations in the form of (4.18), (4.19),

$$\frac{||\delta x||}{||x||} \leq ||A|| ||A^{-1}|| \frac{||\delta b||}{||b||} \text{ or } \frac{||\delta x||}{||x + \delta x||} \leq ||A|| ||A^{-1}|| \frac{||\delta A||}{||A||}.$$  (4.22)

That is, the relative change in the exact solution is bounded above by $||A|| ||A^{-1}|| \times$ relative perturbation in the data, and thus the quantity $||A|| ||A^{-1}||$ indicates the maximum effect of the perturbations in the system matrices.

Consider now the condition numbers defined in (4.2) to (4.6) which are of primary importance in this thesis. Bauer and Fike [22] and Wilkinson ([252], p87) show that the overall sensitivity of the eigenvalues of a given matrix $A$ is dependent upon the
size of $\kappa(X)$, where $X$ is the right eigenvector matrix of $A$. This is demonstrated in the following example.

Consider the following two $3 \times 3$ matrices,

\[
A_1 = \begin{bmatrix}
-20.0 & 0 & 0 \\
0 & -6.8392 & -30.2486 \\
0 & -16.5702 & -39.0207
\end{bmatrix}
\]

and

\[
A_2 = \begin{bmatrix}
-40.0 & 290.0 & 900.59 \\
0 & -89.3785 & -30.3445 \\
0 & 161.0 & 52.8712
\end{bmatrix}
\]

The eigenvalues of the above matrices are $(-20.000000, -24.900877, -44.368452)$, and $(-40.000000, -31.4174229, -5.0898845)$ respectively, while their condition numbers as defined in (4.3) are 2.207 and 1105.622 respectively. Suppose all the off-diagonal elements of these two matrices are perturbed by 1%. The corresponding eigenvalues will be perturbed to $(-20.000000, -25.093533, -44.175796)$ and $(-40.000000, -38.989710, 2.482403)$, respectively. Note that the matrix with the larger condition number experiences the larger perturbation in its eigenvalues. If all of the off diagonal elements of $A_1$, the well-conditioned matrix, are perturbed by 200% its eigenvalues remain in the stable locations of $(-20.000000, -8.199419, -61.069909)$. Note that the ill-conditioned matrix $A_2$ produces unstable eigenvalues following a 1% perturbation in its off diagonal elements. The condition number of a given matrix is seen to measure the sensitivity of its eigenvalues with respect to the variations in the entries of the matrix. This concept may be extended to analyze the robustness of the closed-loop feedback system matrix $(A + BKC)$ with respect to variations in its data, i.e., $\kappa(X)$, the condition number of $(A + BKC)$ may be one reasonable criterion to evaluate system robustness. Let the objective be to minimize the condition number of the closed-loop system matrix $(A + BKC)$.
The following properties for matrix norm will be useful.

**Lemma 4.1**: Unitary invariance of the matrix norms ([186], pp14)

\[ \|QAR\|_2 = \|A\|_2 \]
\[ \|QAR\|_F = \|A\|_F \]

for all \( A \in \mathbb{R}^{n \times m} \), if and only if \( Q \) and \( R \) are orthogonal (unitary for complex) matrices, respectively, i.e.,

\[ QQ^T = Q^TQ = I_n \]
\[ RR^T = R^TR = I_m. \]

Note that above property holds only for 2-norm and Frobenius norm.

Based on the above lemma the following property can easily be verified.

**Lemma 4.2**: The condition number defined in terms of the 2-norm or the Frobenius norm is invariant with respect to unitary similarity transformation, i.e.,

\[ \|X\|\|X^{-1}\| = \|TXTT^{-1}\|\|(TXT^{-1})^{-1}\| \] for all orthogonal matrix \( T \).

**Lemma 4.3**: Condition number of a complex matrix

Let \( X \) be an \( n \times n \) complex matrix such that

\[ X = \begin{pmatrix} a_{11} + i b_{11} & a_{21} + i b_{21} & \cdots & a_{n1} + i b_{n1} \\ a_{12} + i b_{12} & a_{22} + i b_{22} & \cdots & a_{n2} + i b_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} + i b_{1n} & a_{2n} + i b_{2n} & \cdots & a_{nn} + i b_{nn} \end{pmatrix} \]

For all orthogonal matrix \( T \).
where \( n_c \leq \frac{n}{2} \) is the number of pairs of complex vectors in \( X \).

Let

\[
X_r = (x_1^{re}, x_1^{im}, x_2^{re}, x_2^{im}, \ldots, x_{n_c}^{re}, x_{n_c}^{im}, x_{n_c+1}, \ldots, x_n) \tag{4.30}
\]

be the real representation of complex matrix \( X \). Then

\[
\kappa(X) \leq \kappa(X_r)\kappa(E), \tag{4.31}
\]

where

\[
E = \text{diag} \left\{ \begin{pmatrix} 1 & 1 \\ i & -i \end{pmatrix}, \ldots, \begin{pmatrix} 1 & 1 \\ i & -i \end{pmatrix}, 1, \ldots, 1 \right\}, \tag{4.32}
\]

where there are \( n_c \) 2 \( \times \) 2 block matrices and \((n - 2n_c)\) 1's.

Furthermore,

\[
\kappa_2(X) \leq \sqrt{2} \kappa_2(X_r). \tag{4.33}
\]

**Proof:**

\[
X = \begin{pmatrix} x_1^{re} + ix_1^{im}, x_2^{re} + ix_2^{im}, x_3^{re} + ix_3^{im}, \ldots, x_{n_c}^{re} + ix_{n_c}^{im}, x_{n_c+1}, x_{n_c+2}, \ldots, x_n \end{pmatrix}
\]

\[
= X_r \begin{pmatrix} 1 & 1 & 0 & 0 & \cdots & 0 \\ i & -i & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 1 & \cdots & 0 \\ 0 & 0 & i & -i & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 \\ 0 & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 \end{pmatrix} = X_rE.
\]
By an axiom of the matrix norm (4.16) and from a property of the matrix inverse, the following hold:

\[ \|X\| = \|X_cE\| \leq \|X_c\|\|E\| \]  \hspace{1cm} (4.34)

and

\[ (X_cE)^{-1} = E^{-1}X_c^{-1}. \]  \hspace{1cm} (4.35)

Hence,

\[ \kappa(X) = \kappa(X_cE) = \|X_cE\|\|(X_cE)^{-1}\| = \|X_c\|\|E^{-1}X_c^{-1}\| \]

\[ \leq \|X_c\|\|E\|\|E^{-1}\|\|X_c^{-1}\| = \kappa(\|X_c\|\kappa(\|E\|)). \]

It can also easily be verified that \( \kappa_2(E) \) is 1 if \( 2n_c = n \) or \( \sqrt{2} \) if \( 2n_c < n \) which imply \( \kappa_2(X) \leq \sqrt{2} \kappa_2(X_c) \).

This property will be useful for developing robust pole assignment algorithms in the later chapter.

### 4.3 Mathematical Model

Consider the continuous linear time-invariant system

\[ \dot{x} = Ax + Bu \] \hspace{1cm} (4.36)

\[ y = Cx \] \hspace{1cm} (4.37)

where \( x \in \mathbb{R}^n, u \in \mathbb{R}^m, y \in \mathbb{R}^p \) represent the state, input and output vectors and \( A, B, C \) are compatibly dimensioned constant matrices. It is desired to find an output feedback gain matrix \( K \in \mathbb{R}^{m\times p} \) which specifies a linear constant control law \( u = Ky \) such that the poles in the closed-loop dynamic

\[ \dot{x} = (A + BK)Cx \] \hspace{1cm} (4.38)

are in a specified region. Moreover, these poles are to be assigned to optimize an appropriately defined performance function. A mathematical model for this...
problem can be constructed as follows.

Let

\[(\lambda_1 \pm i\sigma_1, \lambda_2 \pm i\sigma_2, \ldots, \lambda_{n_c} \pm i\sigma_{n_c}, \lambda_{2n_c+1}, \ldots, \lambda_n)\] (4.39)

be the poles to be assigned, where \(n_c \leq \frac{n}{2}\) is the number of pairs of complex poles.

Suppose the specified region of poles to be assigned are constrained by upper and lower bounds such that

\[\lambda_k \leq \lambda_k \leq \bar{\lambda}_k, \quad k = 1, \ldots, n_c, 2n_c + 1, \ldots, n, \quad \text{and} \]
\[\sigma_j \leq \sigma_j \leq \bar{\sigma}_j, \quad j = 1, \ldots, n_c. \]

(4.40) \hspace{1cm} (4.41)

Let

\[\Lambda = \text{diag}[\lambda_1, \ldots, \lambda_{n_c}, \lambda_{2n_c+1}, \ldots, \lambda_n], \]

where

\[\lambda_k = \begin{bmatrix} \lambda_k & \sigma_k \\ -\sigma_k & \lambda_k \end{bmatrix}, \quad k = 1, \ldots, n_c \]

(4.42) \hspace{1cm} (4.43)

and \(X\) is an \(n \times n\) matrix of corresponding eigenvectors of the closed-loop system matrix defined as

\[X = [x^1_1, x^2_1, \ldots, x^1_{n_c}, x^2_{n_c}, x^1_{n_c+1}, \ldots, x^1_n]. \]

(4.44)

Let \(f(X), \ f : R^{nxn} \rightarrow R, \ X \in R^{nxn}\) be the objective function which appropriately represents the measure of robustness of the system under consideration. Then one possible problem formulation can be written as:

**Problem 4.1.** [176], [179]:

\[
\begin{align*}
\min & \quad f(X) \\
\text{s.t} & \quad (A + BK)X = XL \\
& \quad \Lambda \in \Omega \quad (4.45) \hspace{1cm} (4.46) \hspace{1cm} (4.47)
\end{align*}
\]

where \(\Omega = \{\{\lambda_k, \sigma_j\} : \lambda_k \leq \lambda_k \leq \bar{\lambda}_k, \sigma_j \leq \sigma_j \leq \bar{\sigma}_j, \quad k = 1, \ldots, n_c, 2n_c + 1, \ldots, n, \quad j = 1, \ldots, n_c\}. \)
4.4 Existence of Solution

Necessary and sufficient conditions for the existence of the feasible solutions $X$ and the corresponding feedback gain $K$ to the Problem 4.1 are given by the following theorem, which is a generalization of the result introduced in [45].

Assume that, without loss of generality, $B$ is of full column rank and $C$ full row rank. Let $B$ and $C$ in Problem 4.1 be decomposed by

$$B = QB = [Q_0, Q_1] \begin{bmatrix} \hat{B}_1 \\ 0 \end{bmatrix}$$

$$C = CV^T = [\hat{C}_1, 0] \begin{bmatrix} V_0^T \\ V_1^T \end{bmatrix}$$

where $Q$ and $V$ are orthogonal, $\hat{B}_1, \hat{C}_1$ are nonsingular matrices, respectively.

**Theorem 4.1:**

There exists a feasible solution $(X, \Lambda, K)$ to Problem 4.1 with $X$ nonsingular if and only if there exist $X, Y$ and $\Lambda$ satisfying

$$XY - I = 0$$

$$\lambda_{q-1} = \lambda_{2j}, j = 1, ..., \eta_e$$

$$\sigma_j \leq \sigma_i \leq \sigma_{j}, j = 1, ..., \eta_e$$

$$\lambda_i \leq \lambda_i \leq \hat{\lambda}_i, i = 1, ..., n.$$  

and, either

$$Q_1^T (X\Lambda - AX) = 0$$

$$Q_0^T (X\Lambda Y - A)V_1 = 0$$

or

$$Q_1^T (X\Lambda Y - A)V_0 = 0$$

$$(\Lambda Y - YA)V_1 = 0$$

The feedback gain of feasible solutions can be obtained by

$$K = \hat{B}_1^{-1}Q_0^T(X\Lambda Y - A)V_0\hat{C}_1^{-1}.$$  

(4.57)
Proof:

Since $X$ satisfies $(A + BK)X = XA$ for $A \in \Omega$,

$$X\Lambda X^{-1} - A = BK = Q\hat{B}K\hat{\mathcal{C}}V^T$$  \hspace{1cm} (4.58)

Pre and post multiplying both sides of (4.58) by $Q^T$ and $V$, respectively, and denoting $X^{-1} = Y$,

$$Q^T(XAY - A)V = \hat{B}K\hat{\mathcal{C}} = \begin{bmatrix} \hat{B}_1 \\ 0 \end{bmatrix} K[\hat{C}_1,0] = \begin{bmatrix} \hat{B}_1K \\ 0 \end{bmatrix} [\hat{C}_1,0] = \begin{bmatrix} \hat{B}_1K\hat{C}_1 \\ 0 \end{bmatrix}.$$  \hspace{1cm} (4.59)

Note that

$$Q^T(XAY - A)V = \begin{bmatrix} Q_0^T(XAY - A)V_0 \\ Q_1^T(XAY - A)V_1 \end{bmatrix} = \begin{bmatrix} \hat{B}_1K\hat{C}_1 \\ 0 \end{bmatrix}.$$  \hspace{1cm} (4.60)

Thus,

$$Q_0^T(XAY - A)V_0 = 0$$  \hspace{1cm} (4.59)

$$Q_1^T(XAY - A)V_1 = 0$$  \hspace{1cm} (4.60)

$$Q_1^T(XAY - A)V_1 = 0$$  \hspace{1cm} (4.61)

$$K = \hat{B}_1^{-1}Q_0^T(XAY - A)V_0\hat{C}_1.$$  \hspace{1cm} (4.62)

It is clear that (4.59),(4.60),(4.61) can be combined by either

$$\begin{cases} Q_0^T(XAY - A)V_0 = 0 \\ Q_1^T(XAY - A)V_1 = 0 \end{cases}$$  \hspace{1cm} (4.62)

or

$$\begin{cases} Q_1^T(XAY - A)V_1 = 0 \\ (XAY - A)V_1 = 0 \end{cases}$$  \hspace{1cm} (4.63)

Equivalently, post and pre multiplying the left-hand sides of the second equations in (4.62) and (4.63) by $X$ and $Y$, respectively,

$$\begin{cases} Q_0^T(XA - AX) = 0 \\ Q_0^T(XAY - A)V_1 = 0 \end{cases}$$  \hspace{1cm} \text{or} \hspace{1cm} \begin{cases} Q_1^T(XAY - A)V_0 = 0 \\ (AY - YA)V_1 = 0 \end{cases}$$
Note that equations (4.53) and (4.54) provide upper and lower bounds on the desired real and imaginary parts of the poles. If \( x_i \in X \) and \( y_i \in Y \) are chosen so that 
\[
y^T x_i = 1, i = 1, \ldots, n,
\]
it can be identified that \( Y = X^{-1} \) will be the left eigenvector matrix of the closed-loop system matrix with respect to a given \( \Lambda \). It will now be shown that the real representation of complex matrix in the form of (4.44) also satisfies the equation (4.46) when \( n_c \leq 2 \) pairs of complex poles are going to be assigned.

Suppose \( (\lambda_k \pm i\sigma_k) \) are a pair of complex pole to be assigned and \( (x_k \pm i\eta_k) \) are the corresponding eigenvectors. Then,
\[
(A + BKC)(x_k \pm i\eta_k) = (\lambda_k \pm i\sigma_k)(x_k \pm i\eta_k), \quad k = 1, \ldots, n_c, \quad (4.64)
\]
which yields
\[
(A + BKC)x_k \pm i(A + BKC)\eta_k = \lambda_k x_k - \sigma_k \eta_k \pm i(\sigma_k x_k + \lambda_k \eta_k). \quad (4.65)
\]
Equating real and imaginary parts in the equation (4.65), gives the following:
\[
(A + BKC)x_k - \lambda_k x_k + \sigma_k \eta_k = 0 \quad (4.66)
\]
\[
(A + BKC)\eta_k - \sigma_k x_k - \lambda_k \eta_k = 0 \quad (4.67)
\]
which can be written in the form
\[
(A + BKC)(x_k, \eta_k) = (x_k, \eta_k) \begin{pmatrix} \lambda_k & \sigma_k \\ -\sigma_k & \lambda_k \end{pmatrix}, \quad k = 1, \ldots, n_c. \quad (4.68)
\]
Note that the complex conjugate pair of eigenvalue/eigenvector produces the same set of equations (4.66)-(4.67). Thus the complex matrix in the form of (4.44) is indeed the eigenvector matrix associated with the given eigenvalues. This ends the proof of the theorem 6. \( \square \)

By the above theorem, the following corollary can easily be derived.
Corollary 4.1:
Consider a full-state feedback pole assignment problem such that
\[
\begin{align*}
\min f(X) \\
\text{s.t.} \\
(A + BK)X = XA \\
\Lambda \in \Omega
\end{align*}
\]  
(4.69) 
(4.70) 
(4.71)
where \( \Omega \) is as defined in (4.49). Then \((X, \Lambda, K)\) is a feasible solution to the problem if and only if \(X, Y\) and \(\Lambda\) satisfy
\[
Q_f^T(X\Lambda - AX) = 0 \quad (4.72)
\]
(4.51) - (4.54).
The feedback gain can be obtained by
\[
K = \hat{B}_1^{-1}Q_0^T(X\Lambda Y - A). \quad (4.73)
\]
Thus a full-state feedback pole assignment problem can be solved as a special case of output feedback pole assignment problem by the proposed model.

Problem 4.2:
\[
\begin{align*}
\min \quad & F(X,Y) \\
\text{s.t.} \\
G_1(X, Y, \Lambda) &= Q_f^T(X\Lambda Y - A) = 0 \quad (4.75) \\
G_2(X, Y, \Lambda) &= Q_0^T(X\Lambda Y - A)V_1 = 0 \quad (4.76) \\
G_3(X, Y) &= XY - I = 0 \quad (4.77) \\
\Lambda &\in \Omega \quad (4.78)
\end{align*}
\]
In any field of science where a mathematical systems approach is applied, it is
often preferable to consider a dual system or transformed system instead of the original one. It may be easier or more tractable to manipulate the transformed system in preference to the original one. In optimization theory, duality concepts are broadly applied. In control theory, a system matrix may be transformed to some particular desired form, such as Jordan-form, Hessenberg form or upper triangular form and the original problem hence reformulated as a more tractable one with all the properties of the original problem preserved. It is therefore useful to consider under what conditions the original pole assignment problem formulated in this chapter can be transformed to an equivalent problem.

Consider a transformed system where \( \tilde{A} = TAT^T, \tilde{B} = TB, \tilde{C} = CT^T \) and \( \tilde{X} = TX \), where \( T \) is any orthogonal matrix of appropriate dimension to formulate the following problem.

**Problem 4.3:**

\[
\begin{align*}
\min f(\tilde{X}) \\
\text{s.t.} & \quad (\tilde{A} + \tilde{B} \Lambda \tilde{C})\tilde{X} = \tilde{X}\Lambda \\
\Lambda & \in \Omega
\end{align*}
\]

The following theorem may easily be established.

**Theorem 4.2:**

\((\tilde{X}, \Lambda, K)\) is a feasible solution to Problem 4.3 if and only if \((X, \Lambda, K)\) is a feasible solution to Problem 4.1. In other words, a feasible solution to a transformed system can be obtained by the same \( \Lambda, K \) of the original system together with an orthogonal transformation of the eigenvector matrix \( X \) induced by \( T \).

**Proof**

Suppose \((\tilde{X}, \Lambda, K)\) is a feasible solution to the transformed system. Then, it must
satisfy the constraint (4.80). Thus,

\[
TX \Lambda = \tilde{X} \Lambda \\
= (A + \tilde{B}K\tilde{C})\tilde{X} \\
= (TAT^T + TBKCT^T)TX \\
= T(A + BKC)X.
\]

And hence, \((A + BKC)X = X\Lambda\) which implies \((X, \Lambda, K)\) is a feasible solution to Problem 4.1.

Conversely, consider any feasible solution \((X, \Lambda, K)\) to Problem 4.1. Then

\[
X \Lambda = (A + BKC)X \Rightarrow T^T \tilde{X} \Lambda = (T^TA\tilde{T} + T^TBK\tilde{C}T)T^T\tilde{X} \\
= T^T(A + BKC)\tilde{X}.
\]

It follows that \((A + BKC)\tilde{X} = X\Lambda\) which implies \((\tilde{X}, \Lambda, K)\) is a feasible solution to the transformed system. 

The following theorem states that a pole assignment problem may be transformed equivalently into other forms by orthogonal transformation with appropriately chosen objective functions.

**Theorem 4.3:**

Problem 4.1 is equivalent to Problem 3 if the objective function is defined in terms of the 2-norm or the Frobenius norm.

**Proof:**

By Theorem 4.2, \((\tilde{X}, \Lambda, K)\) is a feasible solution to Problem 4.2 if and only if \((X, \Lambda, K)\) is a feasible solution to Problem 4.1. By definition, they must also satisfy the constraints in Problem 4.3. This proves feasibility. To show the objective function values of the original system and of the transformed system are identical, it is necessary to show the matrix norms \(||X||\) and \(||TX||\), \(||X^{-1}||\) and \(||(TX)^{-1}||\), induced by the 2-norm or the Frobenius norm, are identical. If \(Q\) and \(R\) in Lemma
4.1 are chosen as $T$ and $I$, respectively, it is obvious that the above are true. It will be demonstrated with examples in a later chapter that a pole assignment problem is indeed invariant under orthogonal transformations of $A, B, C$ and $X$, respectively.

The objective function $f(X)$ in Problem 4.1 can be defined in many different ways depending upon the chosen robustness measure. As stated earlier, the condition number of the closed-loop system matrix $(A + BK)$ defined by (4.3) or (4.4) may be one reasonable criterion to evaluate the robustness of the system. This indicates the maximum effect of perturbations upon the system matrices, and has been shown to be effective within the context of pole placement by full-state feedback [121].

By a property of the geometric mean, $\kappa_P(X)$ in (4.4) is bounded above by

$$\hat{\kappa}(X) = \frac{1}{2} (||X||^2_2 + ||^{-1}X||^2_2).$$

(4.82)

Thus (4.82) may also be considered as an appropriate measure of robustness for the system under consideration. The pole assignment problem can now be reformulated as follows.

Problem 4.4:

$$\min \quad F(X, Y) = \text{tr}[XX^T] + \text{tr}[YY^T]$$

$$\text{s.t.} \quad G_1(X, Y, A) = Q_{11}^T(XAY - A) = 0$$
$$G_2(X, Y, A) = Q_{12}^T(XAY - A)V_1 = 0$$
$$G_3(X, Y) = XAY - I = 0$$
$$\lambda_{2j-1} = \lambda_{2j}, j = 1, ..., n_e$$
$$\alpha_j \leq \sigma_j \leq \beta_j, j = 1, ..., n_e$$
$$\lambda_1 \leq \lambda_i \leq \bar{\lambda}_i, i = 1, ..., n$$
The objective function (4.83) can be replaced by

\[ \min F(Y) = \text{tr}[YY^T] \quad (4.90) \]

introducing an additional constraint

\[ G_i \equiv \text{diag}[X^TX]_i = 1.0, \forall i \quad (4.91) \]

where \( \text{diag}[X^TX]_i \) is the \( i \)th diagonal element of the matrix \( X^TX \).

This is a nonlinear programming problem with a nonlinear objective function and linear and nonlinear constraints. However, it is not a typical nonlinear programming problem because of the structure of the constraints; the constraints are a set of matrix valued functions instead of vector valued functions. Solving this type of problem using some optimization methods, the typical nonlinear optimization techniques may not be efficient. For example, suppose \( A \in \mathbb{R}^{10\times 10}, B \in \mathbb{R}^{10\times 1}, C \in \mathbb{R}^{1\times 10} \), then the number of variables and the number of nonlinear constraints in the model suggested will be 210 and 188, respectively, to assign real poles to the desired locations. That is, the Jacobian matrix of the constraints will be \( 210 \times 188 \), and hence, to use any optimization software package, it will be required to calculate \( 210 \times 188 \) partial derivatives of the constraints, which will be time-consuming. Certainly, some efficient methods to solve this problem need to be developed.

### 4.5 Concluding Remarks

The robust pole assignment problem is to find the controller which guarantees the closed-loop poles stay within prescribed regions for all uncertainties in the underlying plant. A measure of robustness for pole assignment is defined in terms of the condition number of the closed-loop system matrix. It is demonstrated that the measure so defined may be a reasonable criterion to evaluate system robustness.
In fact, the condition number of the closed-loop system matrix measures the sensitivity of its eigenvalues with respect to parameter variations. A robust output feedback pole assignment problem has been formulated as an optimization problem in terms of a scalar valued objective function with matrix arguments and matrix valued constraints with matrix arguments. Necessary and sufficient conditions for existence of a feasible solution to the robust pole assignment problem have been derived, and it is shown that the robust full-state feedback pole assignment problem can be solved as a special case of output feedback pole assignment problem. Since no existing methods/algorithms seem to effectively solve the problem formulated in this chapter, the next chapter will present a new optimization method for control system design involving Optimization of Matrix Valued Functions which will solve such a robust pole assignment problem efficiently.
Chapter 5

Optimization with Matrix Valued Functions

5.1 Introduction

An optimization method is the way of finding an optimum. However, it should be understood that the optimum of a given problem is not necessarily the best for the problem under consideration but it is simply one of the most favourable solutions for the model formulated, provided the model is a true representation of the system. Therefore, when one talks about optimization methods, one should always think about the model for the problem under study. The concept of an optimal solution and the choice of methods for finding such a solution are entirely problem dependent. The pole assignment problem formulated in the previous chapter is one of the optimization problems, but because of its special structure, no ordinary nonlinear optimization method can be applied efficiently. The typical nonlinear programming solution methods need to be modified in order to solve the problem. In a typical
nonlinear programming formulation, the necessary conditions for an optimal solution are usually derived in terms of the Jacobian of the constraints. However, if the constraints are in the form of matrix valued functions, their derivatives or Jacobian must be defined, and appropriate formulae to derive them need to be developed. In this chapter, an efficient optimization method for solving problems with matrix valued constraints is discussed. The method will solve the robust pole assignment problem formulated in terms of a scalar valued objective function with matrix arguments and matrix valued constraints with matrix arguments. To develop the method, a concept of derivative of matrix valued function is proposed and some formulae to calculate the derivatives are derived. The Lagrangian multiplier rule in optimization theory is extended for the matrix valued optimization problem. Two algorithms for solving such problems are described. The first one will be applied for solving the robust pole assignment problem where the objective function and the constraints are everywhere twice-continuously differentiable and the second one is for solving the case of non-smooth objective function and/or constraints. Numerical examples are tested with the algorithms proposed and compared with other existing algorithms. The results show that the proposed algorithms assign the poles in the desired locations more accurately and give much better robustness than any other algorithm appearing to date in the literature.

5.2 Lagrangian Function in terms of Matrix Valued Function

Consider the following nonlinear programming problem with equality constraints.

**Problem 5.1**

\[
\begin{align*}
\min \ & f(x) \\
\text{s.t.} \ & g(x) = 0
\end{align*}
\]
where $f : \mathbb{R}^n \to \mathbb{R}$, $g : \mathbb{R}^m \to \mathbb{R}^m$, and both $f$ and $g$ are everywhere twice-continuously differentiable.

It is well known that, in general, an equality constrained nonlinear programming problem can be solved by applying the Lagrangian multiplier rule, provided certain regularity conditions hold (see, e.g., Luenberger [149]). Defining the Lagrangian function as

$$L(x, \mu) = f(x) + \mu^T g(x)$$

where $\mu \in \mathbb{R}^m, \mu^T = (\mu_1, ..., \mu_m)$, necessary conditions for an optimal solution are

$$\frac{\partial L}{\partial x} = \frac{\partial f(x)}{\partial x} + \frac{\partial g(x)}{\partial x} \mu = 0$$

$$\frac{\partial L}{\partial \mu} = g(x) = 0$$

Consider the application of the same process to Problem 4.4 but ignoring the linear constraints. Define a Lagrangian function for a matrix form as:

$$L(X, Y, \Lambda, U_1, U_2, U_3) = \text{tr}(XX^T) + \text{tr}(YY^T) + U_1 \text{vec}G_1 + U_2 \text{vec}G_2 + U_3 \text{vec}G_3$$

or equivalently,

$$L(X, Y, \Lambda, U_1, U_2, U_3) = \text{tr}(XX^T) + \text{tr}(YY^T) + \text{tr}[U_1^T G_1] + \text{tr}[U_2^T G_2] + \text{tr}[U_3^T G_3]$$

where $U_1, U_2, U_3, \bar{U}_1, \bar{U}_2, \bar{U}_3$ are appropriately dimensioned matrices and vectors, respectively, and $\text{vec}^T[.]$ denotes a string of row vectors of the matrix $[.]$ expanded row by row. The following may be expected to hold from generalization of (5.2) and (5.3).

$$\frac{\partial L}{\partial X} = 0, \quad \frac{\partial L}{\partial Y} = 0, \quad \frac{\partial L}{\partial \Lambda} = 0$$

and

$$\frac{\partial L}{\partial U_i} = 0, \quad \text{for } i = 1, 2, 3$$
Athans [18] justified that the two spaces, $\mathbb{R}^{n \times m}$ which is the set of all real $n \times m$ matrices and $\mathbb{R}^{(n,m)}$ which is the $(n,m)$-dimensional Euclidean vector space, are algebraically and topologically equivalent. The inner products $\langle X, Y \rangle$, $\forall X, Y \in \mathbb{R}^{n \times m}$ and $\langle x, y \rangle$, $\forall x, y \in \mathbb{R}^{(n,m)}$, are identical so that $\langle \psi(X), \psi(Y) \rangle = \langle X, Y \rangle$, where

$$X = \begin{bmatrix} x_{11} & \cdots & x_{1m} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nm} \end{bmatrix}, \quad x = \text{vec}X = \psi(X).$$

Similarly, a mapping $\varphi$ can be defined in terms of $\Lambda$ and $\lambda$. Through the pair of mappings $\psi$ and $\varphi$, $\mathcal{L}(X, Y, \Lambda, U_1, U_2, U_3)$ which is the Lagrangian function with the matrix arguments, $X$, $Y$ & $\Lambda$ in equation (5.4) becomes $L(x, y, \lambda, U_1, U_2, U_3)$, which has the vector arguments, $x$, $y$ & $\lambda$. It can thus be observed that the Lagrangian Multiplier Rule for vector valued constraints can be generalized to matrix valued constraints.

### 5.3 Derivative of Matrix Valued Functions

In order to substantiate the discussion in the last section, the concepts of matrix derivative and trace must be developed. The matrix derivative is not necessarily uniquely defined. Dwyer and MacPhail [58] define a *symbolic matrix derivative* which is convenient for regression and correlation analysis. Other definitions can be found in Brewer [30], Graham [98], Neudecker [156], Parring [187], Pollock [191], Turnbull [234], Vetter [236] etc. In this chapter, a definition more suitable to the optimization problems under consideration is considered as proposed in [176].
**Definition 5.1**: Matrix derivative

Let

\[ F(X) = \begin{bmatrix} F_{11} & \cdots & F_{1n} \\ \vdots & \ddots & \vdots \\ F_{m1} & \cdots & F_{mn} \end{bmatrix} \]  

where \( X \in \mathbb{R}^{m \times n} \) and \( F_{ij} \) are scalar-valued functions of matrix \( X \).

Then the derivative of the matrix valued function \( F(X) \) with respect to the matrix variable \( X \) is defined by

\[ \frac{dF}{dX} = \begin{bmatrix} \frac{\partial F_{11}}{\partial X} & \cdots & \frac{\partial F_{1n}}{\partial X} \\ \vdots & \ddots & \vdots \\ \frac{\partial F_{m1}}{\partial X} & \cdots & \frac{\partial F_{mn}}{\partial X} \end{bmatrix} \]  

(5.8)

where \( \frac{dF}{dX} \) denotes the matrix differential.

The derivative of the matrix valued function \( F(X) \) with respect to a scalar variable \( x_{ij} \) is defined by

\[ \frac{\partial F}{\partial x_{ij}} \equiv \begin{bmatrix} \frac{\partial E_{i1}}{\partial x_{ij}} & \cdots & \frac{\partial E_{in}}{\partial x_{ij}} \\ \vdots & \ddots & \vdots \\ \frac{\partial E_{m1}}{\partial x_{ij}} & \cdots & \frac{\partial E_{mn}}{\partial x_{ij}} \end{bmatrix} \]  

where \( \Theta \) denotes the matrix operator.

With this definition, the following lemma provides the formulae required to calculate the derivatives of matrix valued functions.
Theorem 5.1 [176]:
Suppose $X \in \mathbb{R}^{n \times n}$, $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{m \times k}$, and $x_{ij}$, $\forall i, j$ are independent of each other. Let $\text{vec}D \equiv (d_{11}, \cdots, d_{1m}; d_{21}, \cdots, d_{2m}; \cdots; d_{n1}, \cdots, d_{nm})^T$, for any $n \times m$ matrix $D$, and let $F(X)$ be a matrix valued function.

The matrix derivatives for the following matrix valued functions are obtained as follows.

i) If $F(X) = AX$, then $\frac{\partial F}{\partial X} = \text{vec}A \text{vec}^T I_p$

ii) If $F(X) = XB$, then $\frac{\partial F}{\partial X} = \text{vec}I_n \text{vec}^T B^T$

iii) If $F(X) = AXB$, then $\frac{\partial F}{\partial X} = \text{vec}A \text{vec}^T B^T$

iv) If $G(X) = U \text{vec}F(X)$, for any appropriately dimensioned vector $U$, then $\frac{\partial G}{\partial X} = U \text{vec}(\frac{\partial F}{\partial X})$

Proof:
1. Let

$$F(X) = AX = \begin{bmatrix} a_{11} \cdots a_{1n} \\ \vdots \\ a_{m1} \cdots a_{mn} \end{bmatrix} \begin{bmatrix} x_{11} \cdots x_{1p} \\ \vdots \\ x_{n1} \cdots x_{np} \end{bmatrix} = \begin{bmatrix} \sum_j a_{1j} x_{1j} \cdots \sum_j a_{1j} x_{jp} \\ \vdots \\ \sum_j a_{nj} x_{1j} \cdots \sum_j a_{nj} x_{jp} \end{bmatrix}$$

\begin{equation} \tag{5.13} \end{equation}
Thus by (5.23) and (5.24),

\[
\begin{align*}
\frac{\partial F}{\partial X} &= \begin{bmatrix}
\frac{\partial F_{11}}{\partial x_{11}} & \cdots & \frac{\partial F_{11}}{\partial x_{1p}} \\
\vdots & \ddots & \vdots \\
\frac{\partial F_{m1}}{\partial x_{11}} & \cdots & \frac{\partial F_{m1}}{\partial x_{1p}} \\
& & \\
\frac{\partial F_{m1}}{\partial x_{m1}} & \cdots & \frac{\partial F_{m1}}{\partial x_{mp}} \\
\end{bmatrix}
\begin{bmatrix}
\frac{\partial F_{11}}{\partial x_{p1}} & \cdots & \frac{\partial F_{11}}{\partial x_{n1}} \\
\vdots & \ddots & \vdots \\
\frac{\partial F_{m1}}{\partial x_{mp}} & \cdots & \frac{\partial F_{m1}}{\partial x_{np}} \\
\end{bmatrix}
\begin{bmatrix}
a_{11} 0 \cdots 0 a_{1n} 0 \cdots 0 a_{11} \\
\vdots \\
\vdots \\
a_{m1} 0 \cdots 0 a_{m1} 0 \cdots 0 a_{m1} \\
\vdots \\
a_{mn} 0 \cdots 0 a_{mn} 0 \cdots 0 a_{mn}
\end{bmatrix}
\end{align*}
\]

\[
= [a_{11} \cdots a_{1n}; a_{m1} \cdots a_{mn}]^T [1 0 \cdots 0; 0 1 \cdots 0; \cdots; 0 1]
\]

\[
= \text{vec} A \text{vec} F
\]
Since by (5.24),

\[
\frac{\partial F_{ij}}{\partial X} = \begin{bmatrix}
    a_{11}b_{ij} & \cdots & a_{1n}b_{ij} \\
    \vdots & \ddots & \vdots \\
    a_{in}b_{ij} & \cdots & a_{in}b_{ij}
\end{bmatrix},
\]

and in general,

\[
\frac{\partial F_{ij}}{\partial X} = \begin{bmatrix}
    a_{ij}b_{ij} & \cdots & a_{ij}b_{ij} \\
    \vdots & \ddots & \vdots \\
    a_{nj}b_{ij} & \cdots & a_{nj}b_{ij}
\end{bmatrix}, \quad \text{for } \forall i, j.
\]

Thus,

\[
\frac{\partial F}{\partial X} = \begin{bmatrix}
    \begin{pmatrix}
        a_{11}b_{11} & \cdots & a_{1n}b_{11} \\
        \vdots & \ddots & \vdots \\
        a_{in}b_{11} & \cdots & a_{in}b_{11}
    \end{pmatrix} & \begin{pmatrix}
        a_{11}b_{12} & \cdots & a_{1n}b_{12} \\
        \vdots & \ddots & \vdots \\
        a_{in}b_{12} & \cdots & a_{in}b_{12}
    \end{pmatrix} \\
    \vdots & \ddots & \vdots \\
    \begin{pmatrix}
        a_{m1}b_{11} & \cdots & a_{mn}b_{11} \\
        \vdots & \ddots & \vdots \\
        a_{mn}b_{11} & \cdots & a_{mn}b_{11}
    \end{pmatrix} & \begin{pmatrix}
        a_{m1}b_{12} & \cdots & a_{mn}b_{12} \\
        \vdots & \ddots & \vdots \\
        a_{mn}b_{12} & \cdots & a_{mn}b_{12}
    \end{pmatrix}
\end{bmatrix}
= \left[ a_{11} \cdots a_{1n}; \cdots ; a_{m1} \cdots a_{mn} \right]^T \left[ b_{11} \cdots b_{m1}; \cdots ; b_{1n} \cdots b_{mn} \right]
= \text{vec} A \text{vec}^T B
\]

4. Straightforward from the linearity of the derivative operator.

The following formulae for the derivatives of the trace of a matrix are found in Athens [18].
Derivatives of trace of matrix

\[
\frac{\partial}{\partial X} \text{tr}(X) = I \tag{5.18} \\
\frac{\partial}{\partial X} \text{tr}(AX) = A^T \tag{5.19} \\
\frac{\partial}{\partial X} \text{tr}(AXB) = A^T B^T \tag{5.20} \\
\frac{\partial}{\partial X} \text{tr}(XX^T) = 2X \tag{5.21}
\]

With the above formulae and Theorem 5.1, the following theorem can be derived.

**Theorem 5.2:**

Let \( G(A) = \text{tr} [ A^T F(X) ] \), where \( A \in \mathbb{R}^{m \times m} \), \( F : \mathbb{R}^{m \times m} \rightarrow \mathbb{R}^{m \times m} \). Then,

\[
\frac{\partial G}{\partial X} = \sum_{i,j} A_{ij} \frac{\partial F_{ij}}{\partial X}, \tag{5.22}
\]

where \( A_{ij} \) and \( F_{ij} \) are \( i,j \text{'th} \) element of the corresponding matrix and the matrix valued function, respectively.

**Proof:**

Let

\[
G(X) = \text{tr} \left[ \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} F_{11} & \cdots & F_{1m} \\ \vdots & \ddots & \vdots \\ F_{m1} & \cdots & F_{mm} \end{bmatrix} \right] = \text{tr} \left[ \begin{bmatrix} \sum_j a_{j1} F_{j1} & \cdots & \sum_j a_{jm} F_{jm} \\ \vdots & \ddots & \vdots \\ \sum_j a_{jm} F_{j1} & \cdots & \sum_j a_{jm} F_{jm} \end{bmatrix} \right]
\]

Thus,

\[
\frac{\partial G}{\partial X} = \text{tr} \left[ \begin{bmatrix} \sum_j a_{j1} \frac{\partial F_{j1}}{\partial X} & \cdots & \sum_j a_{jm} \frac{\partial F_{jm}}{\partial X} \\ \vdots & \ddots & \vdots \\ \sum_j a_{jm} \frac{\partial F_{j1}}{\partial X} & \cdots & \sum_j a_{jm} \frac{\partial F_{jm}}{\partial X} \end{bmatrix} \right] = \sum_j a_{j1} \frac{\partial F_{j1}}{\partial X} + \cdots + \sum_j a_{jm} \frac{\partial F_{jm}}{\partial X} = \sum_{i,j} A_{ij} \frac{\partial F_{ij}}{\partial X} \]
With equation (5.6) - (5.7) and Theorem 5.1 and Theorem 5.2, the necessary conditions for an optimal solution to Problem 4.4 can now be stated as follows provided the gradients of the constraints at the stationary points are linearly independent.

\[
\frac{\partial L}{\partial X} = 2X + \sum_{i,j} U_{1,i,j} \frac{\partial G_{1,i,j}}{\partial X} + \sum_{i,j} U_{2,i,j} \frac{\partial G_{2,i,j}}{\partial X} + \sum_{i,j} U_{3,i,j} \frac{\partial G_{3,i,j}}{\partial X} = 0 \quad (5.23)
\]

\[
\frac{\partial L}{\partial Y} = 2Y + \sum_{i,j} U_{1,i,j} \frac{\partial G_{1,i,j}}{\partial Y} + \sum_{i,j} U_{2,i,j} \frac{\partial G_{2,i,j}}{\partial Y} + \sum_{i,j} U_{3,i,j} \frac{\partial G_{3,i,j}}{\partial Y} = 0 \quad (5.24)
\]

\[
\frac{\partial L}{\partial \Lambda} = \sum_{i,j} U_{1,i,j} \frac{\partial G_{1,i,j}}{\partial \Lambda} + \sum_{i,j} U_{2,i,j} \frac{\partial G_{2,i,j}}{\partial \Lambda} + \sum_{i,j} U_{3,i,j} \frac{\partial G_{3,i,j}}{\partial \Lambda} = 0 \quad (5.25)
\]

In addition the equality constraints \( G_i = 0, \ i = 1,2,3 \) must also hold.

### 5.4 Necessary Conditions of Optimality

For Problem 4.4, the necessary condition for an optimal solution, which is analogous with the Lagrangian multiplier rule in optimization theory, can be stated as follows.

**Theorem 5.3** : Necessary conditions for the matrix valued optimization problem.

Suppose the gradient vectors of \( G_i, i = 1,2,3 \) are linearly independent, at \((X^*, Y^*, \Lambda^*)\) which is a local minimum to Problem 2, then there exist constant matrices \( U_1, U_2, U_3 \) such that equations (5.23),(5.24) and (5.25) are satisfied at \((X^*, Y^*, \Lambda^*)\), where

\[
\frac{\partial G_1}{\partial X} = \text{vec}(Q_1^TX^*\Lambda^*)\text{vec}^T(Y^*T\Lambda^T) \quad (5.26)
\]

\[
\frac{\partial G_1}{\partial Y} = \text{vec}(Q_1^TX^*)\text{vec}^TI \quad (5.27)
\]

\[
\frac{\partial G_1}{\partial \Lambda} = \text{vec}(Q_1^TX^*)\text{vec}^T(Y^*T) \quad (5.28)
\]

\[
\frac{\partial G_2}{\partial X} = \text{vec}(Q_2^TX^*\Lambda^*)\text{vec}^T(V_1^TY^*T\Lambda^T) \quad (5.29)
\]

\[
\frac{\partial G_2}{\partial Y} = \text{vec}(Q_2^TX^*)\text{vec}^TV_1^T \quad (5.30)
\]
5.5 Proposed Algorithms

There exist a number of algorithms to solve constrained non-linear programming problems. However the choice of algorithm must depend upon the properties of the objective functions and the constraints. If these functions are nonlinear and smooth, then the problem can be solved by the classical Lagrangian method as stated in (5.2) - (5.3).

However, Problem 4.4 is formulated in matrix form with extra linear constraints and upper and lower bounds on the variables. It may thus call for other methods which enable the matrix valued function and the extra constraints to be taken care of effectively. Also the fact that the objective function and the set of constraints are non-convex will produce other difficulties. The following two algorithms are suggested. The first one applies to the case where the objective function and the constraints are everywhere twice-continuously differentiable, any Problem 4.4 and the second one is for the case of non-smooth objective function and/or constraints.

- Algorithm 1:
  
  **Step 1.** Calculate the gradient of the objective function and the Jacobian of the constraints applying the Theorem 5.1.
  
  **Step 2.** Choose a feasible starting point if available, otherwise choose any
point as a starting point.

**Step 3.** If the poles are going to be assigned to the fixed points, go to step 5. If the poles are assigned in a specified region, set \( i = 1 \) and go to step 4.

**Step 4.** Set lower bounds = upper bounds for all poles except the \( i \)th pole.

**Step 5.** Select any suitable optimization software package available to generate Kuhn-Tucker points (e.g. NAG Manual [173]: E04UCF, MATLAB Optimization Tool Box [97], or other optimization packages may be used).

**Step 6.** Find a stationary point satisfying the necessary condition in Theorem 5.3. If the poles are going to be assigned in a specified region go to step 7, else go to step 8.

**Step 7.** Set \( i = i + 1 \), until \( i = n \). And repeat step 5.

**Step 8.** If algorithm converges to a desired accuracy stop, an optimal solution is obtained. Otherwise go to step 9.

**Step 9.** If algorithm fails to generate Kuhn-Tucker point, compute feedback gain \( F \) and eigenvalues assigned using current X or Y matrix. If poles are already in a feasible region, stop, an optimal solution is obtained, otherwise go to step 10

**Step 10.** Check Constraint Qualification (see, e.g., [23]) at the most current point. If it is satisfied, stop, there is no optimal solution in the neighbourhood of the current point. If it is not satisfied, adjust the upper and lower bounds of the poles to be assigned, and go to step 2.

**Algorithm 2:**

**Step 1.** Find a feasible solution for the fixed poles \( \lambda_i^j \), \( i = 1, \ldots, n \) using any available algorithm. Set \( i = 1 \).
Step 2. Choose an allowable interval of uncertainty (IOU) \( l_i > 0 \). Let \([a_i(k), b_i(k)]\) be the IOU at the beginning of \( k^{th} \) iteration with respect to the \( i^{th} \) pole to be assigned. Set \( k = 1 \), and let \( a_i(1) = \lambda_i, b_i(1) = \bar{\lambda}_i \), where \( \lambda_i \) and \( \bar{\lambda}_i \) are the upper and the lower bounds of poles to be assigned, respectively.

Let \( \alpha(k) = a_i(k) + (1 - \tau)[b_i(k) - a_i(k)] \), \( \beta(k) = a_i(k) + \tau[b_i(k)] \), where \( \tau \) is a search ratio (e.g., \( \tau = 0.618 \)).

Step 3.
If \( b_i(k) - a_i(k) < l_i \), then let \( \lambda_i^* = \frac{1}{2}[b_i(k) - a_i(k)] \), and \( i = i + 1 \) until \( i = n \), and go to step 2. Otherwise, if \( F(\lambda_1^*, \ldots, \lambda_{i-1}^*, \alpha(k), \lambda_{i+1}^0, \ldots, \lambda_n) > F(\lambda_1^*, \ldots, \lambda_{i-1}^0, \beta(k), \lambda_{i+1}^0, \ldots, \lambda_n) \) go to step 4, else goto step 5.

Step 4.
Let \( a_i(k + 1) = \alpha(k), \; b_i(k + 1) = b_i(k), \; \alpha(k + 1) = \beta(k), \; \beta(k + 1) = a_i(k + 1) + \tau[b_i(k + 1) - a_i(k + 1)] \). Go to step 6.

Step 5.
Let \( a_i(k + 1) = a_i(k), \; b_i(k + 1) = \beta(k), \; \beta(k + 1) = \alpha(k), \; \alpha(k + 1) = a_i(k + 1) + (1 - \tau)[b_i(k + 1) - a_i(k + 1)] \). Go to step 6.

Step 6. Let \( k = k + 1 \) and go to step 3.

5.6 Numerical Examples/Comparisons to Other Algorithms

The method described above is tested on some examples and compared to other algorithms appearing in the literature, including Chu, Nichols, & Knutsky [45], Miminis [167] and Roppenecker & O'Reilly [206].
Example 5.1:

The algorithm suggested by Chu et al [45] to obtain a robust solution seeks to minimize the condition numbers and the distance between assigned and desired poles simultaneously. This example employs the same system as described in example 3.3 whereby Chu et al obtain the poles at \((-0.973, -2.226, -2.918, -3.80)\) with a condition number 778.2 to assign the poles at \((-1, -2, -3, -4)\). Miminis algorithm which is based on a direct eigenstructure assignment approach using the deflation method obtains only three poles \((-1.0, -2.99999, -3.99999)\) at the desired locations and one pole at an arbitrary location of 0.0 with condition number 61.143. Two different cases are tested by the algorithm suggested in this chapter. When the poles are to be assigned precisely to the fixed positions of \((-1, -2, -3, -4)\), then \((-1.0, -2.0, -2.99997, -4.00001)\) are achieved:

\[
\begin{bmatrix}
-46.9999978492 & 13.4025422184 & -23.6266337024 \\
-19.8675177987 & 5.2241032607 & -11.0000010155
\end{bmatrix}
\]

The condition number has not been significantly improved, but the assigned poles are much closer to those desired using the proposed algorithm compared to the other algorithms. When the poles are to be assigned within a specified region of \(-20 \leq \lambda_i, i = 1, \ldots, n \leq -0.01\), then \((-0.01000, -0.4360161796, -2.46748142, -19.9999999997)\) are obtained with a much more improved condition number 18.063, and the output feedback gain matrix is:

\[
\begin{bmatrix}
-84.2883464668 & 14.0705659791 & 39.6308180642 \\
50.0539136422 & -8.484341063 & -23.9134976030
\end{bmatrix}
\]

Example 5.2:

In Example 3.4, for assigning poles to \((-0.20, -1.0, -5.0566, -8.6659)\), Slade [226] assigns them \((-0.1995, -1.0004, -5.0566, -8.6654)\) with a condition number
11.368, while Miminis assigns \((-0.20, -0.09999, -5.05660, -8.66589)\) with a condition number 51.1919. When the poles are to be assigned precisely to the fixed positions by the suggested algorithm \((-0.19999, -1.00000, -5.05659, -8.66589)\) are obtained with a condition number of 6.797, and the output feedback gain matrix

\[
\begin{bmatrix}
-0.2891179824 & -0.1049637251 & 0.1256089807 \\
-0.1251367374 & 0.3011657839 & 0.5250445615
\end{bmatrix}
\].

When the poles are to be assigned within a specified region of \(-20 \leq \lambda_i, i = 1, \ldots, n \leq -0.01\), then \((-0.0098669532, -1.2002272939, -8.3053988709, -19.9904211120)\) are obtained with a condition number 3.129, and the output feedback gain matrix

\[
\begin{bmatrix}
0.0496682656 & -2.7543017927 & 0.0437679212 \\
1.0507911628 & 0.0320730009 & 0.7139080444
\end{bmatrix}
\].

**Example 5.3 [206]:**

Roppenecker & O’Reilly’s algorithm is a type of parametric eigenstructure assignment approach. It assumes that the desired closed-loop eigenvalues are distinct and that no one of the closed-loop eigenvalues is the same as any open-loop eigenvalue. One of their test examples cited from Bruun & Kümmel [33] is compared with the results obtained by the algorithm proposed in this chapter.

Consider the following problem with dimension \(n = 5, m = 3, p = 3\).

\[
A = \begin{bmatrix}
0. & 0. & -0.0034 & 0. & 0. \\
0. & -0.041 & 0.0013 & 0. & 0. \\
0. & 0. & -1.1471 & 0. & 0. \\
0. & 0. & -0.0036 & 0. & 0. \\
0. & 0.094 & 0.0057 & 0. & -0.0510
\end{bmatrix}
\].
where the open-loop poles are \((0, 0, -0.051, -0.041, -1.1471)\) and the poles to be assigned are \((-1.1461, -0.0572, -0.5220, -0.2159, -0.1898)\), for which Roppe­necker et al obtain the output feedback gain matrix

\[
B = \begin{bmatrix}
-1 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0.948 \\
0.916 & -1 & 0 \\
-0.598 & 0 & 0
\end{bmatrix},
C = \begin{bmatrix}
0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0
\end{bmatrix}.
\]

To assign the poles to fixed positions by the proposed algorithm, poles of \((-1.14610, -0.05721, -0.52193, -0.21591, -0.18979)\) are obtained with a condition number 91.761, and the output feedback gain matrix

\[
\begin{bmatrix}
0.1353 & 0.1682 & -0.0113 \\
-3.9866 & -1.2456 & 0.6324 \\
-8.0450 & 22.5540 & -1.9750
\end{bmatrix}
\]

with a condition number 348.491.

If the poles are required to be assigned in the the region specified by \(-2 \leq \lambda_i, i = 1, ..., n \leq -0.01\), then poles of \((-2.0, -0.08032, -1.14836, -0.04116, -1.18387)\) with a condition number 10.158 and the output feedback gain

\[
\begin{bmatrix}
-0.0732904813 & 0.2843053080 & 0.0609631299 \\
-1.3262040194 & 1.3257705972 & 0.6992608274 \\
-11.5632876111 & 8.4731571558 & -3.058670364
\end{bmatrix}
\]

are obtained.

These results are summarized in the tables below.
### Table 1. (Example 1)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>CNK</th>
<th>Fletcher</th>
<th>Miminis</th>
<th>GOS(fix)</th>
<th>GOS(var)*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Poles</td>
<td>-0.997</td>
<td>-1.0</td>
<td>0.**</td>
<td>-1.0</td>
<td>-0.0100000000</td>
</tr>
<tr>
<td>Assigned</td>
<td>-2.002</td>
<td>-2.0</td>
<td>-1.0</td>
<td>-2.0</td>
<td>-0.4360161796</td>
</tr>
<tr>
<td></td>
<td>-3.000</td>
<td>-3.0</td>
<td>-2.99999</td>
<td>-2.9997</td>
<td>-2.4674814236</td>
</tr>
<tr>
<td></td>
<td>-3.998</td>
<td>-4.0</td>
<td>-3.99999</td>
<td>-4.00001</td>
<td>-19.9999999997</td>
</tr>
<tr>
<td>Condition No.</td>
<td>786.7</td>
<td>884.77</td>
<td>61.143**</td>
<td>499.18</td>
<td>18.063</td>
</tr>
</tbody>
</table>

* To assign within a specified region: $-20 \leq \lambda_i, i = 1, ..., n \leq -0.01$

** Not all poles are assignable.

### Table 2. (Example 2)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>CNK</th>
<th>Miminis</th>
<th>GOS(fix)</th>
<th>GOS(var)*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Poles</td>
<td>-0.1995</td>
<td>-0.2000</td>
<td>-0.19999</td>
<td>-0.0098669532</td>
</tr>
<tr>
<td>Assigned</td>
<td>-1.0004</td>
<td>-0.99999</td>
<td>-1.00000</td>
<td>-1.2002272939</td>
</tr>
<tr>
<td></td>
<td>-5.0566</td>
<td>-5.0566</td>
<td>-5.05659</td>
<td>-8.3053988709</td>
</tr>
<tr>
<td>Condition No.</td>
<td>11.368</td>
<td>80.621</td>
<td>6.796</td>
<td>3.129</td>
</tr>
</tbody>
</table>

* To assign within a specified region: $-20 \leq \lambda_i, i = 1, ..., n \leq -0.01$

### Table 3. (Example 3)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Rop&amp;O'R</th>
<th>Miminis</th>
<th>GOS(fix)</th>
<th>GOS(var)*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Poles</td>
<td>-0.0572</td>
<td>-0.0572</td>
<td>-0.05721</td>
<td>-0.04116</td>
</tr>
<tr>
<td>Assigned</td>
<td>-0.1898</td>
<td>-0.1898</td>
<td>-0.18979</td>
<td>-0.08032</td>
</tr>
<tr>
<td></td>
<td>-0.2159</td>
<td>-0.2159</td>
<td>-0.21591</td>
<td>-1.13837</td>
</tr>
<tr>
<td></td>
<td>-0.5220</td>
<td>-0.5219</td>
<td>-0.52193</td>
<td>-1.14536</td>
</tr>
<tr>
<td></td>
<td>-1.1461</td>
<td>-1.1461</td>
<td>-1.14610</td>
<td>-2.0</td>
</tr>
<tr>
<td>Condition No.</td>
<td>348.49</td>
<td>340.99</td>
<td>91.761</td>
<td>10.158</td>
</tr>
</tbody>
</table>

* To assign within a specified region: $-2 \leq \lambda_i, i = 1, ..., n \leq -0.01$
Results of more examples have also been compared with those produced by the algorithm suggested in this chapter, and it was found that, in general, the latter algorithm provides the better results in the sense that smaller condition numbers are achieved.

5.7 Concluding Remarks

An efficient method to calculate gradients and Jacobians for high dimensional matrix valued functions appearing in necessary conditions of optimality has been developed. Algorithms for solving optimization problems in matrix forms have been suggested. Examples show that the proposed algorithms provide much more robust results than the major algorithms in the literature. In particular, when closed-loop poles are assigned in a specified region rather than at exact locations, the algorithms give much better robustness which is of vital importance in practical control system design. Nevertheless, the method is seen to have some limitations. Although the objective function is smooth and convex, the constraint set is not necessarily a convex set. Moreover, there is no guarantee that constraint qualifications hold at a stationary point of interest, thus it does not exclude the possibility of a non-Kuhn-Tucker point being an optimal solution. The algorithm does not globally converge; it may be sensitive to the starting point for some ill-conditioned problems. The wider the upper and lower bounds of the poles to be assigned, the higher is the chance of failure to generate a set of feasible solutions, because of the non-convexity of the constraints set. In the next chapter, a homotopy approach which is a globally convergent method will be introduced to overcome these difficulties.
Chapter 6

A Global Method - The Homotopy Approach

6.1 Introduction

In the field of control system design, the assignment of closed-loop poles with prescribed time-response characteristics to robust locations may readily be formulated as an optimization problem as seen in the previous chapter. Such a formulation will generally involve a high degree of nonlinearity. The objective function and/or constraints may be \textit{badly behaved} even if they are smooth. In addition, the constraints set may not necessarily be convex. The solution of problems via optimization methods generally involves solving a set of Kuhn-Tucker type linear and/or nonlinear equations which produces stationary points for optimal solutions. The classical methods, including Newton type methods which have been applied for many years, are still of interest. However, most of these methods have a common drawback in that global convergence is not guaranteed. Such methods may fail to converge to
an optimal solution if the chosen starting points are not sufficiently close to the solution. In theory it may not be required that a starting point is a feasible solution. Indeed there are methods to reach a feasible solution starting from an infeasible one such as the *big M* method in linear programming or the *two phase* method or the penalty function method. However in practice, the feasibility of an initial starting point may significantly affect the algorithms’ ability to proceed to successive iterations without terminating at an infeasible solution. The selection of a suitable starting point may be difficult, ad hoc and time-consuming. It is well known that [10], [81], [88], [105], [143], [239], [243], [246] the homotopy methods provide an elegant theoretical framework with the potential to circumvent such difficulties.

The homotopy method for optimization, sometimes called a parametric optimization method, pathfollowing method or continuation method, is a branch of mathematical programming. The first study appears to be that of Manne [158] and the field developed rapidly in the late Seventies and throughout the Eighties, with the appearance of several hundred papers. Some survey works in this area may be found in the literature([9], [10], [19], [99], [152], [215], [239]), [250]. Textbook type references, include Garcia and Zangwill [85], Bank, Guddat, Klatte, Kummer and Tammer [21] and Guddat, Guerra Vasques and Jongen [101]. The solving of equations by the use of Homotopy methods can be found in the literature ([5], [6], [11], [20], [36], [43], [44], [84], [86], [84], [86], [124], [125], [198], [199], [240],[241], [246], [244]).

This method may be effectively applied to solve engineering problems in which the data in the model depend on parameters whose variations may be controlled by a system designer or exogenously. It has been used to analyze the change in the solution to problems for which there are small variations in the system parameters and also to find the solution to the original system by systematically varying an appropriately defined artificial parameter or parameters. This idea is applied to
solve constrained optimization problems by suitable manipulation of the active constraint set corresponding to the parameter variations. It has the advantage of global convergence under some mild assumptions and is efficient for solving ill-conditioned optimization problems. It has been stated that the robust output feedback pole assignment problem in [176] defines a nonlinear optimization problem for which the selection of suitable starting points has proved difficult for some ill-conditioned problems. The homotopy approach is seen to provide an elegant theoretical framework with the potential to circumvent this problem. The method is known to be numerically stable and suitable for solving highly nonlinear problems for which an initial feasible solution may be difficult to estimate. Although the methods appear in the literature of the last three or four decades, there is little work addressing their practical application to control systems design. Some pertinent studies appear in references (Harris et al [110], Richter and DeCarlo [201], Kabamba et al [116], Wayburn and Seader [251], Richter and Collins [200]). Recently, Richter and Hodel [202] suggested a method for solving an algebraic Riccati equation using a homotopy approach. Zigic, et al [258] proposed several algorithms to solve the Lyapunov type equations for the model reduction problem. In addition, Phatak and Keerthi [189] have considered a homotopy approach for stabilizing a single-input system with constraints on the gain vector and the stability region. As far as optimization for control engineering applications is concerned, the method seems to be still in the formative stage of development.

This chapter considers the application of the homotopy approach to robust pole placement. The basic concepts of homotopy, choice of homotopy functions, basic properties and main theorems to implement the homotopy methods are introduced. Some existing algorithms from the literature are reviewed and new algorithms based on the homotopy approach to solve optimization problems are proposed. Numerical examples for robust pole assignment by output feedback including the traces of homotopy paths are given.
6.2 Overview of Homotopy Methods

6.2.1 Concept of Homotopy

The following definition which can be found in any general topology textbook may be useful for understanding the basic concept of homotopy method since the method is based on this concept.

Definition: Homotopy

Let $X, Y$ be topological spaces and $f : X \rightarrow Y$, $g : X \rightarrow Y$. A map $H : X \times [0,1] \rightarrow Y$ is called a homotopy from $g$ to $f$ if $H(x, 0) = g(x)$ and $H(x, 1) = f(x)$ for all $x \in X$. $g$ is called homotopic to $f$, $H(x, t) = 0$ a homotopy system of equations (sometimes called a one-parameter imbedding function) and $t$ a homotopy variable, $t \in [0,1]$. 

A homotopy approach may be used to find points at which necessary conditions for the solution of an optimization problem are satisfied. This amounts in general to solving a system of equations of the form $f(x) = 0, f : \mathbb{R}^n \rightarrow \mathbb{R}^n, x \in \mathbb{R}^n$. The homotopy approach in particular is based on constructing and solving another system of equations, called the homotopy equations, with $g(x) = 0$, $g : \mathbb{R}^n \rightarrow \mathbb{R}^n, x \in \mathbb{R}^n$ whose solution is known in terms of an appropriately defined homotopy map. In the literature, several different types of homotopy maps have been introduced for this purpose including [85]

$$H_1(x,t) = tf(x) + (1-t)g(x), \quad 0 \leq t \leq 1, \quad g(x_0) = 0 \quad (6.1)$$

(linear homotopy)

$$H_2(x,t) = tf(x) + (1-t)(x - x_0), \quad 0 \leq t \leq 1, \text{ when } m = n \quad (6.2)$$

(fixed-point homotopy)
\[ H_0(x,t) = f(x) - (1-t)f(x_0) \] (Newton homotopy).

With an appropriately chosen homotopy function, the solution path of \( H_i(x,t) = 0 \), \( i = 1,2,3 \) is then traced starting from \( x(0) = x_0 \) at \( t = 0 \), moving on to \( x(1) \) at \( t = 1 \). In general, the homotopy path \( x: [0,1] \to \mathbb{R}^n \) defined by \( H(x,t) = 0 \) will describe a path from a known point \( x(0) = x_0 \) to an unknown solution \( x(1) \).

Equations (6.1) - (6.3) are examples of particular homotopy maps. Any appropriate function

\[ H(x,t): \mathbb{R}^{n+1} \to \mathbb{R}^n, \quad x \in X \subset \mathbb{R}^n \text{ and } t \in [0,1] \] (6.4)

may be acceptable. However, it should be noted that not every choice of homotopy map may necessarily create the homotopy path which leads to a solution to the system of equations of interest (see for example, Watson, [243]). Garcia and Zangwill [83] showed that under some assumptions, the set of all solutions to \( H(x,t) = 0 \) is a finite number of disjoint continuously differentiable paths, and any path is either a loop in \( clX \times [0,1] \) or starts from a boundary point of \( clX \times [0,1] \) and ends at another boundary point of \( clX \times [0,1] \). Watson, et al [248] concluded that 'The homotopy zero curves may not be smooth if the homotopy parameter represents a physical meaningful quantity, but they can always be obtained via certain generic constructions using an artificial (i.e. nonphysical) homotopy parameter'.

Chow et al [43] and Watson [243], [246], introduce a homotopy map

\[ H_a(x,t) = tf(x) + (1-t)g_a(x), 0 \leq x \leq 1 \] (6.5)

and show that for almost every \( a \in IntX, \ H_a^{-1}(0) \) contains a smooth curve which will lead from a zero of \( g_a \) to a zero of \( f \), provided the following conditions hold:

**Condition for Existence of Smooth Curve:**

i) \( H_a \) is transversal to zero, i.e., for almost all \( a \) the Jacobian matrix of \( H \) has full rank on \( H_a^{-1}(0) \).
(CHAPTER 6. HOMOTOPY APPROACH)

ii) $H_a(x, 0) = 0$ has a unique solution.

iii) $H_a(x, 1) = f(x)$

iv) $H_a^{-1}(0)$ is bounded.

The above property eliminates the difficulties including bifurcations, singular and ill-conditioned behaviour to solve fixed-point and find zero curves with homotopies. In practice, it guarantees that it is sufficient to solve the homotopy equations for a few important values of the homotopy parameter. The method is described more formally as below.

6.2.2 Imbedding Methods

Let $B$ be a closed unit ball in $\mathbb{R}^n$ and $f : B \rightarrow B$ be the function whose zero is sought. Let $s : B \rightarrow B$ be a simple function with a known zero. Construct a continuous map (the homotopy) $\phi : [0, 1] \times B \rightarrow B$ such that $\phi(x, 0) = s(x)$ and $\phi(x, 1) = f(x)$.

By solving the equation $\phi(x, t) = 0$ in $B \times [0, 1]$, one attempts to move from the known zero of $s(x)$, at $t = 0$ to the unknown zero of $f(x)$, at $t = 1$. In general, moving from a zero of $s(x)$ to a zero of $f(x)$ may or may not be possible. Sometimes $t$ is treated as an independent variable and sometimes it is a dependent variable with the arc length or some other parameter as the independent variable. Watson et al [243, 250] introduce the following lemma based on Sard’s Theorem [211] in Differential Topology.

**Lemma 6.1:**

Let $\rho : \mathbb{R}^n \times [0, 1] \times \mathbb{R}^n \rightarrow \mathbb{R}^n$ be a $C^2$ map such that the Jacobian matrix $J_\rho(a, t, x)$ has full rank on $\rho^{-1}(0) = \{(a, t, x)|0 \leq t \leq 1, x \in \mathbb{R}^n, J_\rho(a, t, x) = 0\}$. Such a $\rho$ is said to be transversal to zero. Then for almost all $a \in \mathbb{R}^n$, the map $\rho_a(t, x) = \rho(a, t, x)$ is also transversal to zero, i.e., the Jacobian matrix of $\rho_a(t, x)$ has full rank on $\rho_a^{-1}(0) = \{(t, x)|\rho_a(t, x) = 0\}$ with probability one.
Watson and Fenner [247] re-state the theorem introduced by Chow et al [43] which is fundamental to the generation of algorithms using homotopy methods as follows.

**Theorem 6.1:**

Define \( \rho : [0,1] \times B \rightarrow \mathbb{R}^n \) by \( \rho_a(t, x) = t(x - f(x)) + (1 - t)(x - a) \), where \( f : B \rightarrow B \) is a \( C^2 \) map such that the Jacobian matrix of \( x - f(x) \) is nonsingular at every fixed point of \( f \). Then for almost all \( a \) in the interior of \( B \), the set \( \{(t, x) | 0 < t < 1, x \in B, \rho_a(t, x) = 0\} \) of zeros of \( \rho_a \) consists of

i) a finite number of closed loops having finite length in \( (0,1) \times B \),

ii) a finite number of arcs having finite length in \( (0,1) \times B \) with end points in \( \{1\} \times B \),

iii) a curve of finite length starting at \( (0, a) \) and ending at \( (1, \bar{a}) \), where \( \bar{a} \in B \) is a fixed point of \( f \).

The curves in (i), (ii) and (iii) are disjoint and continuously differentiable, i.e., with probability one, there exists a smooth and finite length zero curve of \( \rho_a(t, x) \) emanating from \( (0, a) \) which reaches a fixed point \( \bar{x} \) of \( f \) at \( t = 1 \). An algorithm to compute a fixed point of \( f \) is designed to follow a smooth zero curve of \( \rho_a(t, x) \) from \( (0, a) \) to a fixed point \( \bar{x} \) of \( f \). Generation of the zero curve is the main process. It has been shown that this algorithm is globally convergent with probability one. That is, for all starting points \( a \), except possibly those in a set of Lebesgue measure zero, the algorithm converges globally. The method of solving an optimization problem using this idea involves starting with a system whose solution is known and solving a sequence of systems generated by varying the system parameter systematically until a desired solution is achieved. This approach is called the parametric optimization method.
As an example, consider the problem ([85], pp6) of solving the following nonlinear systems of equations.

\[
\begin{align*}
    x_1^3 - 3x_1^2 + 8x_1 + 3x_2 - 36 &= 0 \\
    x_1^2 + x_2 + 4 &= 0
\end{align*}
\]  

(6.6)

Define a homotopy function by

\[
H(x,t) = tf(x) + (1-t)g(x), \quad 0 < t < 1
\]

(6.7)

where \( f(x) = [f_1, f_2]^T \) is the function in (6.6) and \( g(x) \) is defined by

\[
\begin{align*}
    g_1(x) &= x_1^3 + 8x_1 + 3x_2 \\
    g_2(x) &= x_2
\end{align*}
\]  

(6.8)

Note that a solution to (6.8) giving \( g_1 = 0, g_2 = 0 \) is \( (x_1^0, x_2^0) = (0, 0) \).

It follows that the homotopy function defined in (6.7) will be

\[
\begin{align*}
    x_1^3 + 8x_1 + 3x_2 - t(3x_1^2 + 36) &= 0 \\
    tx_1^2 + x_2 + 4t &= 0
\end{align*}
\]  

(6.9)

The embedded parametric problem may be stated as follows. Starting from \( (x_1(0), x_2(0)) = (0,0) \) which is a known solution to \( H(x,0) = 0 \), obtain the solution \( (x_1(1), x_2(1)) \) for \( H(x,1) \).

Solving for \( x_1 \) and \( x_2 \) in terms of parameter \( t \), it can be seen that

\[
\begin{align*}
    x_1(t) &= 6t \\
    x_2(t) &= -36t^3 - 4t
\end{align*}
\]  

(6.10)

describes a path from a starting point \( (x_1(0), x_2(0)) = (0,0) \) at \( t = 0 \) to a solution \( (x_1(1), x_2(1)) = (6, -40) \) as \( t \) increases to 1.

Now consider the following constrained optimization problem.

\[
\begin{align*}
    \min f(x) \quad \text{s.t} \quad g_1(x) &= 0, \quad g_2(x) \leq 0, \quad \text{where} \quad f : \mathbb{R}^n \to \mathbb{R}, \\
    &\quad g_1 : \mathbb{R}^n \to \mathbb{R}, \quad g_2 : \mathbb{R}^n \to \mathbb{R}.
\end{align*}
\]  

(6.11)
The above typical optimization problem can be embedded into the following parametric optimization problem.

\[
\begin{aligned}
\min f(x, t), & \quad t \in T \subseteq R \\
\text{s.t.} & \quad x \in M(t) = \{x \in R^n | h_i(x, t) = 0, i \in I, h_j \leq 0, j \in J\}
\end{aligned}
\]

where \( h : R^n \rightarrow R^{n+p}, \quad I = \{1, \ldots, m\}, \quad J = \{p+1, \ldots, m+p\} \) (6.12)

Define the Lagrangian function

\[
L(x, t, \lambda, \mu) = f(x, t) + \sum_{i \in I} \lambda_i h_i(x, t) + \sum_{j \in J} \mu_j h_j(x, t)
\]  

(6.13)

The Kuhn-Tucker necessary conditions for this parametric optimization problem can be written as

\[
\frac{\partial L}{\partial x} = \nabla_x f(x, t) + \sum_{i \in I} \lambda_i \nabla_x h_i(x, t) + \sum_{j \in J} \mu_j \nabla_x h_j(x, t) = 0
\]  

(6.14)

\[
\frac{\partial L}{\partial \lambda_i} = h_i(x, t) = 0, \quad i \in I
\]  

(6.15)

\[
\frac{\partial L}{\partial \mu_j} = h_j(x, t) \leq 0, \quad j \in J
\]  

(6.16)

\[
\mu_j h_j(x, t) = 0, j \in J
\]  

(6.17)

\[
\mu_j \geq 0, j \in J
\]  

(6.18)

Applying the ideas of Kojima et al [131] and Gfrerer et al [88], the above conditions can be reformulated in the following theorem without the complementary slackness condition (6.17).

**Theorem 6.2:**

Let

\[
y_j = \begin{cases} 
\mu_j \geq 0 & \text{if } h_j(x, t) = 0 \\
h_j & \text{if } h_j(x, t) < 0
\end{cases}
\]  

(6.19)

\[
y_j^+ = \max\{0, y_j\}, \quad y_j^- = \min\{0, y_j\}
\]  

(6.20)

\[
\lambda_i = y_i, \quad i \in I, \quad \text{and} \quad \mu_j = y_j^+, \quad j \in J
\]  

(6.21)

Then, \((x, \lambda, \mu)\) satisfies Kuhn-Tucker condition iff \((x, y)\) satisfies
[CHAPTER 6. HOMOTOPY APPROACH] 109

\[ \nabla_x f(x, t) + \sum_{i \in I} y_i \nabla_x h_i(x, t) + \sum_{j \in J} y_j^+ \nabla_x h_j(x, t) = 0 \]  \hspace{1cm} (6.22)

\text{and} \hspace{1cm} y_j^+ - h_j(x, t) = 0, \hspace{0.5cm} j \in J \hspace{1cm} (6.23)

Proof:

i) Suppose \((x, \lambda, \mu)\) satisfies the Kuhn-Tucker equation. Then by the complementary slackness condition (6.17), \(\forall j \in J' \cup J, \hspace{0.5cm} \text{if} \hspace{0.5cm} h_j < 0, \hspace{0.5cm} \text{then} \hspace{0.5cm} \mu_j = 0. \) Thus by definition \(y_j = h_j < 0, \hspace{0.5cm} \text{and} \hspace{0.5cm} y_j^+ = 0, \hspace{0.5cm} j \in J' \text{ and } y_j^- = h_j(x, t). \) On the other hand, \(\forall j \in J' = J \setminus J', \hspace{0.5cm} \text{if} \hspace{0.5cm} h_j = 0, \hspace{0.5cm} \text{then} \hspace{0.5cm} y_j = \mu_j \geq 0, \hspace{0.5cm} \text{and hence,} \hspace{0.5cm} y_j^+ = \mu_j \text{ and } y_j^- = 0, \hspace{0.5cm} j \in J. \) Thus, \(\sum_{j \in J} \mu_j \nabla h_j(x, t) = \sum_{j \in J} y_j^+ \nabla h_j(x, t) \)

\[ y_j^- - h_j(x, t) = 0, \hspace{0.5cm} j \in J \]

ii) Now suppose \((x, y)\) satisfies (6.22) and (6.23). Then for \(j \in J', \hspace{0.5cm} h_j < 0, \hspace{0.5cm} y_j = h_j \) and \(y_j^- = y_j, \hspace{0.5cm} y_j^+ = 0. \) Thus \(\sum_{j \in J'} y_j^+ \nabla h_j = 0. \) Also, \(\forall h_j = 0, \hspace{0.5cm} j \in J', \hspace{0.5cm} y_j = \mu_j, \hspace{0.5cm} y_j^- = 0, \hspace{0.5cm} y_j^+ \geq 0, \hspace{0.5cm} \mu_j = y_j = y_j^+ \geq 0. \) Thus, \(\sum_{j \in J} y_j^+ \nabla h_j = \sum_{j \in J} \mu_j \nabla h_j. \) Since \(y_j^- = h_j(x, t), \hspace{0.5cm} \forall j \in J', \hspace{0.5cm} 0 = y_j^+ y_j^- = \mu_j h_j(x, t), \hspace{0.5cm} \forall j \in J'. \) Hence, \(\mu_j h_j = 0. \) ■

Finding stationary points to the parametric problem is equivalent to finding the kernel of the systems of equations (6.22) and (6.23).

6.3 Homotopy Algorithms

6.3.1 Existing Algorithms

Like other methods, the homotopy approach for solving constrained optimization problems involves finding zeros of the set of equations described in terms of Kuhn-Tucker conditions. To find a root of \(F(x) = 0, \) consider an appropriately constructed
Suppose the equation (6.24) has a solution \( x = x(t) \), which is continuous with respect to \( t, \forall t \in [0,1] \), or suppose there exists a continuous \( x : [0,1] \to X \) such that \( H(x(t),t) = 0, \forall t \in [0,1] \). Note that \( x \) describes a trajectory starting at a given point \( x_0 \) and ending at \( x_N = x(1) \) which is a solution of \( H(x,1) = 0 \). This observation can be stated formally as follows in terms of a homotopy approach.

To solve the system of equations \( F(x) = 0 \), define a homotopy mapping

\[
H : X \times [0,1] \subset \mathbb{R}^{n+1} \to \mathbb{R}^n
\]

for which a continuous \( x : [0,1] \to X \) will satisfy

\[
H(x(t),t) = 0, \forall t \in [0,1]
\]

such that \( H(x,1) = F(x) \) and \( H(x_0,0) = 0 \) with some known \( x_0 \).

To obtain \( x(t) \) on \([0,1]\) for (6.26), the interval \([0,1]\) is partitioned by

\[
0 = t_0 < t_1 < \ldots < t_N = 1
\]

and

\[
H(x,t_i) = 0, i = 1,\ldots,N
\]

will then be solved for each \( t_i, i = 1,\ldots,N \) iteratively. The corresponding \( x_i \) which is a solution to \( H(x,t_i) = 0 \) will be used as a starting point for finding \( x_{i+1} \), the solution to \( H(x,t_{i+1}) = 0 \). For sufficiently small \( t_{i+1} - t_i \), algorithms which converge to an optimal solution can be constructed. Though many methods and algorithms have been introduced there are basically two different approaches to solve the system of equations by homotopy methods. The first is to solve the system of equations directly using classical methods for solving linear or nonlinear simultaneous equations. The second is by solving initial value problems. A brief introduction is given as follows.
6.3.1.1 Direct Approaches

Any classical methods can be applied to solve (6.28). For example, Ortega et al. ([180], pp232), and García and Zangwill ([85], pp300) consider Newton type methods for which the $k^{th}$ step of Newton iteration proceeds as follows.

\[
x_{i,h+1} = x_{i,h} - \nabla_x H(x_{i,h}, t_i)^{-1} H(x_{i,h}, t_i), \quad k = 0, 1, ..., n_i - 1
\]

\[
x_{1,0} = x_0, x_{i+1,0} = x_{i,ni}, i = 1, ..., N - 1
\]

where $n_i$ is the number of steps to be taken for $t_i$. The algorithm should include a process to find a partition (6.27) and integers $n_1, ..., n_N$ such that (6.29) is well defined and generates the sequence

\[
x_{N,h+1} = x_{N,h} - \nabla_x H(x_{N,h}, 1)^{-1} H(x_{N,h}, 1), \quad k = 0, ..., n_N - 1
\]

which will converge to $x(1)$. Gfrerer et al. [88] and Guddat et al. [101], [105] suggest the following algorithm using an active-constraints set method.

Consider, instead of the original problem (6.12), the following auxiliary parametric optimization problem with active constraints only.

\[
P^k(t) : \quad \min f(x, t), \quad t \in T \subseteq R^r
\]

\[
s.t. \quad h_i(x, t) = 0, i \in I, \quad h_j(x, t) \leq 0, \quad j \in J_k,
\]

\[
t \in [t_k, t_{k+1}]
\]

where $I = \{1, ..., p\}$, $J_k = \{l, ..., s\} \subseteq J$, $t_k < t_{k+1}$.

\[
k = 0, 1, ..., N, \quad t_0 = 0, \quad t_N = 1.
\]

Note that $J_k$ is an index set of active inequality constraints.

**Algorithm:**

Let $0 = t_0 < t_1 < ... < t_N = 1$ and $I_0 = I(t_0), \quad I_k = I(t_k)$. Select an index set of active inequality constraints, $J_k$.  

Let $I(t)$ be the index set of active constraints, ie.

$$I(t) \equiv \{ j \in \{1,...,m\} : h_j(x(t),t) = 0 \}$$

and let

$$I^+(t) \equiv \{ j \in I(t) : \mu_j > 0 \}, \quad I^-(t) \equiv \{ j \in I(t) : \mu_j = 0 \}.$$  

### step 0.
Let $k = 0, x^k = x(k), I_k = I(k), u^k = (u_j(k))_{j \in I_k}$.

### step 1.
If, $k = N$, stop.

### step 2.
$k = k + 1$

### step 3.
Find the solution $(x^k, \lambda^k, \mu^k)$ of $P^{k-1}(t^k)$ in (6.31) and (6.32) by an appropriate method.

### step 4.
If $h_j(x^{k-1}, \lambda^{k-1}, \mu^{k-1}) \leq 0, j \notin I_k$ and $\mu_j^k \geq 0, j \in I_{k-1}$, then $I_k = I_{k-1}$ and go to step 1.

### step 5.
Find a Kuhn-Tucker point $\bar{x} \in (t^{k-1}, t^k)$ by solving $h_j(x^{k-1}(\bar{t}), \bar{t}) = 0$ or some $j \notin I - k - 1$ or $\mu^{k-1}(t) = 0$ for some $j \in I_{k-1}, \bar{x} = x^{k-1}(\bar{t}), \bar{\mu}_j = \mu^k_j k - 1(\bar{t})$ for $j \in I_{k-1}, \bar{\mu} = 0$ or $j \in I_{k-1}$.

### step 6.
Determine the index set $I(t), t \in (\bar{t}, \bar{t} + \epsilon)$, for sufficiently small $\epsilon > 0$. Let

$I_{k-1} = I(t), x^{k-1} = \bar{x}, u^{k-1} = (\bar{\mu}_j), j \in I(t)$. Go to step 3.

To see how the above algorithm works, consider the following example [105].

$$\min \quad f(x) = x_1^2 + 2x_2 + \frac{1}{2}x_3^2$$

$$\text{s.t.} \quad g_1(x) = x_1^2 + x_2^2 - 1 \leq 0$$

$$g_2(x) = x_1 - x_2 - \frac{5}{4} \leq 0$$

By solving the Kuhn-Tucker equation for the above problem, it is not difficult to find the Kuhn-Tucker point $x = (0, -1)^T$, $\lambda = (\frac{1}{2}, 0)^T$. In order to apply the homotopy
method to generate a stationary point, define homotopy maps for (6.36) – (6.38) as

\[
\begin{align*}
    h_0(x, t) &= x_1^2 + (4t - 2)x_2 + \frac{1}{2}x_2^2 \\
    h_1(x, t) &= x_1^2 + x_2^2 - 1 \\
    h_2(x, t) &= x_1 - x_2 - t - \frac{1}{4}
\end{align*}
\]

(6.39) (6.40) (6.41)

Then the equations (6.22) and (6.23) in theorem 2 will yield

\[
\begin{align*}
    \frac{\partial H}{\partial x} &= \begin{pmatrix} 2x_1 + 2y_1^+ x_1 - y_1^- \\
    4t - 2 + x_2 + 2y_2^+ x_2 + y_2^+ 
\end{pmatrix} = 0 \\
    y_1^- - g_1(x) &= y_1^- - (x_1^2 + (4t - 2)x_2 + \frac{1}{2}x_2^2) = 0 \\
    y_2^- - g_2(x) &= y_2^- - (x_1 - x_2 - t - \frac{1}{4}) = 0
\end{align*}
\]

(6.42) (6.43) (6.44)

where \( y_1^+ \) and \( y_2^+ \) are defined in (6.20).

i) Let \( t = 0 \). Then \( x = (0,1)^T, \lambda = (\frac{1}{2}, 0) \) is a Kuhn-Tucker point. Thus only constraint \( h_1 \) is binding. For \( t \geq 0 + \varepsilon \), for small \( \varepsilon \geq 0 \), and fixed \( x = (0, 1), y^+ = (\frac{1}{2} - 2t, 0)^T \) will be obtained from equation (6.42).

ii) Let \( t = \frac{1}{4} \). Then \( x = (0, 1)^T, y^+ = (0, 0)^T \) is found to be a Kuhn-Tucker point. Thus both of constraints \( h_1 \) and \( h_2 \) may be free. For \( t \geq \frac{1}{4} + \varepsilon \), for small \( \varepsilon \geq 0 \), any fixed \( y^+ = (0,0) \), \( x \) will have the value of \((0,2 - 4t)\).

iii) Let \( t = \frac{1}{4} \). Then, \( x = (0,0), y^+ = (0, 0)^T \) is a Kuhn-Tucker point. And \( x \) will have the same value as the case of \( t = \frac{1}{4} \) above.

iv) Let \( t = \frac{3}{4} \). Then, \( x = (0,-1)^T, y^+ = (0, 0) \) is a Kuhn-Tucker point. For \( t \geq \frac{3}{4} + \varepsilon \), for small \( \varepsilon \geq 0 \), and fixed \( y^+ = (0,0)^T \), \( x \) will have the value of \((0,2 - 4t)^T\), while for \( t \geq \frac{3}{4} + \varepsilon \), for small \( \varepsilon \geq 0 \), and \( x = (0,-1)^T, y^+ = (0,2 - 4t)^T \).

v) Let \( t = 1 \). Then, Kuhn-Tucker point is \( x = (0,-1)^T, y^+ = (2t - \frac{3}{2}, 0)^T \) from which \( y^+ = (\frac{1}{2}, 0)^T \) will be obtained.
Other algorithms may be found in references [88], [89], [104], [105], [144], [241].

6.3.1.2 Initial Value Problem Methods

An alternative approach is to replace the solution of the Kuhn-Tucker equation with the solution of an initial value problem. Consider a general form of the set of equations of which zeros are to be found,

\[ 0 = \phi(t) = H(x(t), t), \forall t \in [0, 1] \]  

(6.45)

where \( H \) has continuous partial derivatives with respect to \( x \) and \( t \). By the chain rule,

\[ \frac{d\phi(t)}{dt} = \frac{\partial H}{\partial x} \frac{dx}{dt} + \frac{\partial H}{\partial t} \cdot t \in [0, 1] \]  

(6.46)

Equation (6.46) with initial condition \( x(0) = x_0 \) is sometimes called the Davidenko differential equation. For a solution \( x = x(t) \) to (6.45), it will be true that \( \frac{dH(x, t)}{dt} = 0 \), \( \forall t \), and hence \( x(t) \) satisfies the differential equation

\[ \frac{\partial H}{\partial x} \frac{dx}{dt} = -\frac{\partial H}{\partial t}, \quad t \in [0, 1] \]  

(6.47)

and vice versa. If \( \frac{\partial H}{\partial x} \) is nonsingular for all \( x \) and \( t \), (6.47) can be written in the form

\[ \frac{dx}{dt} = -(\frac{\partial H}{\partial x})^{-1} \frac{\partial H}{\partial t}, \quad t \in [0, 1], \quad H(x(0), 0) = 0 \]  

(6.48)

Also it can be shown (see Ortega and Rheinbold [180], pp 231) that if \( \frac{\partial H}{\partial x} \) is nonsingular, where \( H \) is defined in (6.3) and

\[ \| (\frac{\partial H}{\partial x})^{-1} \| \leq \beta, \quad \forall x \in \mathbb{R}^n, \quad \beta > 0. \]  

(6.49)

then (6.48) has a unique solution \( x : [0, 1] \rightarrow \mathbb{R}^n \) for any fixed initial point.
6.3.1.3 Nonlinear Complementarity Problems

One of the frequently used methods to solve nonlinear programming problems is the so-called complementarity problem. In optimization theory, it is well known that solving a complementarity problem is sometimes more efficient than solving the original problem. A nonlinear complementarity problem is constructed as follows.

For a constrained optimization problem:

\[
\min f(x) \quad \text{s.t.} \quad g(x) \leq 0, \quad x \geq 0, \quad \text{where} \quad f : R^n \rightarrow R, \quad g : R^n \rightarrow R^n
\]  

(6.50)

the Kuhn-Tucker necessary conditions are

\[
\nabla_x L = \nabla f + \lambda^T \nabla g \geq 0, \quad \lambda \geq 0 \quad \text{(optimality condition)}
\]

(6.51)

\[
g(x) \leq 0, \quad x \geq 0 \quad \text{(feasibility condition)}
\]

(6.52)

\[
x^T (\nabla f + \lambda^T \nabla g(x)) = 0, \quad \lambda^T g(x) = 0
\]

(6.53)

(complementary slackness condition)

Let \((x^T, \lambda^T) \equiv x^T \text{ and } (\nabla_x L^T, -g^T(x)) \equiv F^T(x).\)

(6.54)

Then \(x \geq 0 \iff (x^T, \lambda^T)^T \geq 0 \iff x \geq 0, \quad \lambda \geq 0\)

(6.55)

\[
F(x) \geq 0 \iff (\nabla_x L^T, -g^T(x))^T \geq 0 \iff \nabla_x L, \quad -g(x) \geq 0
\]

(6.56)

\[
x^T F(x) = (x^T, \lambda^T)(\nabla_x L^T, -g^T(x))^T
\]

(6.57)

\[
= x^T \nabla_x L - \lambda^T g(x) = 0 \iff x^T \nabla_x L = 0, \quad \lambda^T g(x) = 0
\]

(6.58)

Thus (6.51) - (6.53) can be written as

\[
x^T F(x) = 0, \quad F(x) \geq 0, \quad x \geq 0
\]

(6.59)

which is called the nonlinear complementarity problem. It follows that the nonlinear complementarity programming method can be applied to generate Kuhn-Tucker points. Mangasarian [157] introduces the following property to solve the complementarity problem.
Theorem 6.3 [157]:
Let $\theta : \mathbb{R} \rightarrow \mathbb{R}$ be any strictly increasing function for which $\theta(0) = 0$. Then finding a $z \in \mathbb{R}^n$ satisfying the nonlinear complementarity problem (6.59):

$$z F(z) = 0, \quad F(z) \geq 0, \quad z \geq 0$$

is equivalent to solving the system of $n$ nonlinear equations

$$\theta(|F_i(z) - z_i|) - \theta(F_i(z)) - \theta(z_i) = 0, \quad i = 1, \ldots, n \quad (6.60)$$

Further, Watson [242], [244] suggests the following algorithm to solve the nonlinear complementarity problem in (6.59) by the homotopy approach. A strictly increasing function from $\mathbb{R}$ into $\mathbb{R}$ and $\theta(0) = 0$, $\theta(t) = t^3$, is considered so that the equation (6.60) yields

$$G_i(z) = |F_i(z) - z_i|^3 - (F_i(z))^3 + z_i^3 \quad (6.61)$$

and an artificial parameter homotopy map is defined by

$$\rho_\alpha(t, z) = t G(z) + (1 - t)(z - a) \quad (6.62)$$

The idea is to parameterize the zero curve of $\rho_\alpha(t, z)$ as the arc length and solve an initial value problem. To do this, define the parameterization $t = t(s), z = z(s)$ of the zero curve $\rho_\alpha(t, z)$ such that $\rho_\alpha(t(s), z(s)) = 0, t(0) = 0, z(0) = a$, where $s$ is arc length of the curve in the space $(t, z)$. The following initial value problem must now be solved.

$$\frac{d}{ds}\{\rho_\alpha(t(s), z(s))\} = \frac{\partial \rho}{\partial t} \frac{dt}{ds} + \frac{\partial \rho}{\partial z} \frac{dz}{ds} \quad (6.63)$$

$$= (G(z) - z + a, 1 - t + t \frac{\partial G}{\partial z}) \left( \frac{dt}{ds}, \frac{dz}{ds} \right) = 0.$$

with initial condition $t(0) = 0, z(0) = a$ and an additional equation

$$\|\left( \frac{dt}{ds}, \frac{dz}{ds} \right)\| = 1 \quad (6.64)$$
which determines the parameter $s$ as the arc length of the curve in the space $(t,z)$. Watson [242] used the homotopy map $-G(z)$ instead of $G(z)$ to ensure $\frac{ds}{dt}$ is non zero; this means that the zero curve will not turn back for any $a$. The modified homotopy map is

$$\rho_a(t,z) = tH(z) + (1-t)(z-a), \quad \text{where } H(z) = -G(z). \quad (6.65)$$

Watson [242], [244] introduces the following two theorems which prove the existence of a solution to the complementarity problem stated above and an algorithm for computing the solution.

**Theorem 6.4 [242]:**
Suppose the Jacobian of $H(z)$ is nonsingular at every zero of $H(z)$, and suppose there exists $r > 0$ such that $F_k(z) > 0$ if $z > 0$ and $z_k = \|z\|_\infty \geq r$. Then for almost all $a > 0$ there exists a zero curve $\gamma$ of $\rho_a(a,z)$ having finite arc length and connecting $(0,a)$ to $(1,z)$, where $z$ is a zero of $H(z)$.

**Theorem 6.5 [244]:**
Suppose every zero of $H(z)$ lies in the ball $\|z\| < r$, where $r$ is such that $z_k > 0$ and $F_k(z) \geq 0$ for some $j$ if $z \geq 0$ and $\|z\| \geq r$. Then $\exists \delta > 0$ such that for almost all $a \geq 0$ with $\|a\| < \delta$ there is a zero curve $\gamma$ of $\rho_a(t,z)$, along which $\nabla \rho_a(t,z)$ has full rank, connecting $(0,a)$ to $(1,z)$, where $z$ is a zero of $H(z)$.

Applying Mangasarian's theorem [157] for $\theta(t) = t^3$, it was shown that $z$ is a solution to the nonlinear complementarity problem if and only if it solves

$$H_i(z) = -G_i(z) = \theta(z_i) + \theta(F_i(z)) - \theta(|F_i(z) - z_i|)$$

$$= z_i^3 + (F_i(z))^3 - |F_i(z) - z_i|^3 = 0, \quad i = 1, \ldots, n. \quad (6.66)$$

The above can be used to solve constrained optimization problems. To see how homotopy methods solve constrained problems, consider the following one-parametric
optimization problem.

\[
\begin{align*}
\min_{x, a} & \quad f(x, a) : \mathbb{R}^{n+1} \rightarrow \mathbb{R}^k \\
g(x, a) & \leq 0, \; g : \mathbb{R}^{n+1} \rightarrow \mathbb{R}^k \\
h(x, a) & = 0, \; h : \mathbb{R}^{n+1} \rightarrow \mathbb{R}^m \\
\alpha & \in [0, 1]
\end{align*}
\]

Suppose \( f, h \) and \( g \) are as smooth as desired.

Define a Lagrangian function \( \mathcal{L}(x, \lambda, \mu, t) = f(x) + \lambda^T g(x, t) + \mu^T h(x, t) \)

The Kuhn-Tucker condition can be represented by

\[
\nabla \mathcal{L} = \nabla f - \lambda^T \nabla g - \mu^T \nabla h = 0
\]

\[
h(x, \alpha) = 0,
\]

together with the corresponding complementarity problem.

\[
H_i = -G_i = -\theta(|g_i(x, t) - \mu_i|) + \theta(g_i(x, t) + \theta(x, t)) = 0, \; i = 1, \ldots, m.
\]

This is a one-parametric optimization problem. One simple procedure to solve this problem is, to discretize the parameter space \([0, 1]\) as \(0 = t_0 \leq t_1 \ldots \leq t_n = 1\), and to solve each subproblem separately. If the separable problem is a convex problem, then it may not be difficult to develop some globally convergent algorithm. If it is not the case, there can be difficulties in finding a solution. However, a homotopy method which is a continuation method, whereby information about \(x(t_i)\) is used to compute \(x(t_{i+1})\) sequentially, can be applied to globalize a locally convergent optimization algorithm.

Vasudevan et al [235] suggest the following modified set of nonlinear equations to generate a stationary point using the homotopy method.

Define the homotopy map

\[
\rho_a(t, \mu_i) = tH + (1 - t)(\mu - a).
\]
and consider zeros of $\rho_a$ with

$$\rho_a(t, \mu_i) = tH + (1-t)(\mu_i - a_i) = 0.$$  \hfill (6.74)

where $a \in \mathbb{R}^n$ is chosen such that (6.70) and (6.71) are satisfied at $t = 0$. The Jacobian matrix of (6.72) with respect to $\{x, \lambda, \mu, t\}$ becomes singular if $\mu_{gi}(x, t) = 0$ for some $i$ for which the constraint enters active set. However the singularity problem can be avoided using (6.74).

**Watson Algorithm:**

Watson et al [248] suggests an algorithm which can be used to generate Kuhn-Tucker points from the necessary conditions, i.e., to solve the initial value problem given by (6.63) and (6.64):

step 1. Set $s = 0, y = (0, a), ypold = yp = (1, 0, ..., 0)$

step 2. if $\|y_1 < 0$ exit with error.

step 3. Compute a new vector $a = [\bar{\lambda}F(\bar{x}) + (1 - \bar{\lambda})/(1 - \bar{\lambda})]$.

If $\|a - new a - old a\| > 1 + constant \times \|old a\|$, error return

step 4. $ypold = yp$

step 5. Find a vector $z$ such that $D\rho_a(y) = 0$. If $z^T ypold < 0$, let $z = -z$

$yp = z/\|z\|$

step 6. If $y_1 < 0.99$ go to step 2

step 7. If $y_1 \geq 1$ set $(x, y)$ back to the previous point where $y_1 < 1$. Go to step 3

step 8. If $y_1 < 1$. Go to step 2

step 9. Obtain 0 at $y_1 = 1$ using an ordinary differential equation algorithm.
6.3.2 Proposed Algorithms for Solving Typical Optimization Problems

6.3.2.1 Difficulties in Implementing the Algorithms

The optimization algorithms based on the homotopy methods can be characterized as the iterative process of solving the sets of Kuhn-Tucker equations describing the necessary conditions of optimal solutions imbedded on appropriately defined homotopy functions. Several difficulties must be overcome before such algorithms can be successfully implemented. One of the difficulties is that most of the algorithms may not be efficient for large dimensional problems where the processing of the differential equation solver in Watson type algorithms or identifying the active constraints for a sparse system by Guddat type algorithms may be inefficient. Like all successive iteration methods, it is also important to choose appropriate stepsizes and the criteria for further partitioning of the subintervals. These must be sufficiently small so that the iterations converge but must also be as large as possible to minimize the total number of iterations for obtaining optimal solutions. Schwetlick [216], [217], suggests the following rule to choose the stepsize for the \textit{kth} iteration:

\begin{align}
& \text{\textit{kth stepsize}} \quad \Delta t_k = c_k \\
& \max\{1, \alpha, \alpha^2, \ldots\} \text{where} \quad c_k = \min\{1 - t_k, \Delta t_{k-1}/\alpha\} 
\end{align}

Georg [86], Wacker [239], and Wacker et al [240], suggest some rules to choose the stepsize for the successive iteration. However, there does not seem to be any general rule to minimize the computational effort; the total number of iterations to achieve an optimal solution will depend on the problem itself. In other words, an algorithm which solves a particular problem efficiently may not necessarily solve the other problems with the same degree of efficiency. Moreover, no existing algorithm based on homotopy methods seems to effectively solve some specially structured problems such as the matrix valued optimization problems modelled in Chapter 4.
[CHAPTER 6. HOMOTOPY APPROACH] 121

Further algorithmic development is required to solve these types of problems. Two algorithms for solving the homotopy equations in the forms of $H : \mathbb{R}^n \to \mathbb{R}^1$ which corresponds to the typical type of optimization problems are suggested. Another two algorithms for solving the homotopy equations in the forms of $H : \mathbb{R}^{n \times n} \to \mathbb{R}^1$ which corresponds to the matrix optimization will be proposed in the next section.

6.3.2.2 Algorithm I

An algorithm is proposed to solve (6.48) directly provided $(\frac{\partial H}{\partial \omega})^{-1}$ exists.

step 0. Select step size $\Delta t = t_{k+1} - t_k$ and a starting point $x_0$. Set $t = 0, x_k = x_0$.

step 1. Update $(\frac{\partial H}{\partial \omega})^{-1}$ and $\frac{\partial H}{\partial t}$.

step 2. Proceed an iteration $x_{k+1} = x_k - (t_{k+1} - t_k)(\frac{\partial H(x_\omega, t_\omega)}{\partial \omega})^{-1} \frac{\partial H(x_\omega, t_\omega)}{\partial t}$.

step 3. If $t < 1$, go to step 1. Else stop.

Example 6.1:

The example of (6.6) is solved by the above Algorithm I to obtain the following result.

Starting point: $(0.0, 0.0)$, $\Delta t = 0.0001$, Computation time: 1 sec on SGI 4D/480S.
Table 6.1. Homotopy path generated

<table>
<thead>
<tr>
<th>Iteration</th>
<th>t Value</th>
<th>x Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>0.1999000000000018 (0.6000000000, -0.4359460180)</td>
<td></td>
</tr>
<tr>
<td>2000</td>
<td>0.1998999999999943 (1.2000000000, -1.0877840360)</td>
<td></td>
</tr>
<tr>
<td>3000</td>
<td>0.29989999999999833 (1.8000000000, -2.1715140640)</td>
<td></td>
</tr>
<tr>
<td>4000</td>
<td>0.39989999999999723 (2.4000000000, -3.9031360720)</td>
<td></td>
</tr>
<tr>
<td>5000</td>
<td>0.49989999999999612 (3.0000000000, -6.4986500900)</td>
<td></td>
</tr>
<tr>
<td>6000</td>
<td>0.59989999999999502 (3.6000000000, -10.1740561080)</td>
<td></td>
</tr>
<tr>
<td>7000</td>
<td>0.69989999999999392 (4.2000000000, -15.1453541260)</td>
<td></td>
</tr>
<tr>
<td>8000</td>
<td>0.79989999999999282 (4.8000000000, -21.6285441440)</td>
<td></td>
</tr>
<tr>
<td>9000</td>
<td>0.89989999999999173 (5.4000000000, -29.8396261620)</td>
<td></td>
</tr>
<tr>
<td>10000</td>
<td>0.99989999999999063 (6.0000000000, -39.9946001800)</td>
<td></td>
</tr>
</tbody>
</table>

6.3.2.3 Algorithm II

Based on the sequential quadratic programming algorithm in the optimization theory, the following algorithms are proposed to solve the POP (6.12).

step 0. Partition $t \in [t_0 = 0, t_1, ..., t_N = 1]$. Set $s = 0, t_0 = 0$. Select $x_0, \epsilon$.

step 1. Let $t = t_s$. If $t_s = t_N$ go to step 4.

step 2. Approximate the original problem by a Quadratic Problem (QP) whereby

$$
\begin{align*}
  f(x, t_s) &= f(x_k, t_s) + \nabla f(x_k, t_s)^T (x - x_k) + \frac{1}{2} (x - x_k)^T H_f(x_k, t_s)(x - x_k) \\
  h_i(x, t_s) &= h_i(x_k, t_s) + \nabla h_i(x_k, t_s)i \in I \cup J \nonumber
\end{align*}
$$

step 3. Select any suitable software package available to solve the QP updated at step 2. If the QP solver fails to generate a solution at $t_{j+1}$, partition $[t_{j-1}, t_N = 1]$ more finely, and repeat step 3. If $||x - x_k||/||x_k|| < \epsilon$, let $s = s + 1$, go to step 1. Else, let $k = k + 1$, go to step 2.
step 4. Trace $x(t), t \in [0, 1]$ and identify an optimal solution $x(1)$.

Step 3 will be accomplished using a Quasi-Newton method, Ordinary differential equation solver, Nonlinear simultaneous equation solver, Linear and nonlinear complementarity programming, Sequential Quadratic Programming (SQP) routines, and/or Large scale programming routines.

Example 6.2:
Example in (6.36) - (6.38) is solved by the above Algorithm II. Step 3 was performed using NAG library routine [173], the result is as follows.

Starging point: (1.00000000, 1.00000000)

<table>
<thead>
<tr>
<th>Table 6.2. Homotopy path generated</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Iteration</strong></td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>46</td>
</tr>
<tr>
<td>47</td>
</tr>
<tr>
<td>48</td>
</tr>
<tr>
<td>49</td>
</tr>
<tr>
<td>50</td>
</tr>
</tbody>
</table>
6.3.3 The Application of the Proposed Algorithms to Robust Pole Assignment

6.3.3.1 Matrix Homotopy Function

No existing algorithm based on homotopy methods seems to effectively solve matrix valued optimization problems such as those in (4.83) - (4.89), or indeed more general forms of matrix valued function. To solve the robust pole assignment problem described in equations (4.83) - (4.89) by the homotopy method, define a Lagrangian function embedded with homotopy parameter $t$ as follows:

$$L(X, Y, A, U_1, U_2, U_3, t) = tr(XX^T) + tr(YY^T) + tr[U_1^T \hat{G}_1] + tr[U_2^T \hat{G}_2] + tr[U_3^T \hat{G}_3]$$

(6.77)

where $U_1 \in \mathbb{R}^{(n-m) \times n}, U_2 \in \mathbb{R}^{m \times (n-p)}$ and $U_3 \in \mathbb{R}^{n \times n}$ and $\hat{G}_1 : \mathbb{R}^{n \times n} \rightarrow \mathbb{R}^{n \times n}, \hat{G}_2 : \mathbb{R}^{m \times n+1} \rightarrow \mathbb{R}^{n \times (n-p)}$ and $\hat{G}_3 : \mathbb{R}^{n \times n+1} \rightarrow \mathbb{R}^{n \times n}$ are the homotopy functions appropriately defined with respect to $G_1, G_2$ and $G_3$, respectively.

Applying the analogous Lagrangian multiplier rule, it can be shown that the stationary points can be generated by solving the following system of equations.

$$\frac{\partial L}{\partial X} = 2X + \sum U_{1,ij} \frac{\partial \hat{G}_1}{\partial X} + \sum U_{2,ij} \frac{\partial \hat{G}_2}{\partial X} + \sum U_{3,ij} \frac{\partial \hat{G}_3}{\partial X} = 0$$

(6.78)

$$\frac{\partial L}{\partial Y} = 2Y + \sum U_{1,ij} \frac{\partial \hat{G}_1}{\partial Y} + \sum U_{2,ij} \frac{\partial \hat{G}_2}{\partial Y} + \sum U_{3,ij} \frac{\partial \hat{G}_3}{\partial Y} = 0$$

(6.79)

$$\hat{G}_i = 0, \quad i = 1, 2, 3$$

(6.80)

where the symbolic derivatives of the matrix valued functions $\hat{G}_i, i = 1, 2, 3$ with respect to the matrix variables $X$, defined as Definition 5-1, can be written as follows.

$$\frac{\partial \hat{G}_i}{\partial X} = \begin{bmatrix}
\frac{\partial \hat{G}_{i,11}}{\partial X_{11}} & \cdots & \frac{\partial \hat{G}_{i,1n}}{\partial X_{1n}} & \cdots & \frac{\partial \hat{G}_{i,11}}{\partial X_{1n}} & \cdots & \frac{\partial \hat{G}_{i,1n}}{\partial X_{1n}} \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
\frac{\partial \hat{G}_{i,11}}{\partial X_{1n}} & \cdots & \frac{\partial \hat{G}_{i,1n}}{\partial X_{1n}} & \cdots & \frac{\partial \hat{G}_{i,n-n}}{\partial X_{1n}} & \cdots & \frac{\partial \hat{G}_{i,n-n}}{\partial X_{1n}}
\end{bmatrix}$$

(6.81)
The are defined as in (6.81) - (6.82) replacing $X$ by $Y$. The derivation of the matrix valued functions in equations (6.78) - (6.79) can be easily calculated applying the Theorem 5.1.

For simplicity, suppose the poles are going to be assigned to fixed positions instead of to a specified region. For this particular problem, a linear homotopy mapping as defined in (6.1) may be considered to construct the imbedded problem as follows:

$$
\text{min } F(X, Y) = \text{tr}[XX^T] + \text{tr}[YY^T] \\
G_i(X, Y, t) = G_i(X, Y) + (1 - t)Q_i^T(Y - X\Lambda Y) \\
\text{s.t. } G_a(X, Y, t) = G_a(X, Y) + (1 - t)Q_a^T(Y - X\Lambda Y)V_i = 0 \\
G_a(X, Y, t) = XY - I = 0
$$

where $G_i(X, Y), i = 1, 2, 3$ are defined as in equation (4.84) - (4.86).

Note that for $t = 0$, an initial feasible solution to this imbedded problem can easily be identified as $Y = A, X = A^{-1}$. The following Newton homotopy function defined as (6.3) may also be considered, which will be valid in the case where $A$ is singular,

$$
\begin{align*}
\hat{G}_1(X, Y, t) &= G_1(X, Y) - (1 - t)Q_1^T(\Lambda - A) \\
\hat{G}_2(X, Y, t) &= G_2(X, Y) - (1 - t)Q_2^T(\Lambda - A)V_i \\
\hat{G}_a(X, Y, t) &= XY - I = 0
\end{align*}
$$
such that
\[ G_1(x_0, y_0) = Q_A^2(A - A) \]  
\[ G_2(x_0, y_0) = Q_A^2(A - A)\eta \]  
where \( x_0 = I, \quad y_0 = I \)

Denote \((x, y, u)\) by \(\bar{x}\) and let Kuhn-Tucker type equations be
\[ \tilde{h}(x, y, u, t) = 0 \]
\[ \tilde{c}(x, y, a, t) = 0 \]
\[ G_i(x, y, u, t) = 0, \quad i = 1, 2, 3 \]

As discussed in the previous sections, a homotopy method for solving sets of equations or optimization problems produces a zero curve which starts from a known initial solution \( x(0) = x_0 \) at \( t=0 \) and ends to an optimal solution \( x(1) = x^* \) at \( t=1 \). However, existence of such a curve depends on a choice of homotopy map. In nonlinear optimization theory, it is well known that the Kuhn-Tucker necessary conditions yield stationary solution provided an appropriate constraint qualifications hold. One such qualification which is most frequently used because of its ease of verification is the LICQ-CQ (Linearly Independent Constraint Gradient Constraint Qualification). Thus, it will be assumed that a constraint qualification holds for the set of constraints (6.85) at the stationary points. Under such an assumption and with other mild ones, it can be verified that the existence conditions for a smooth curve discussed in Lemma 6.1 will be satisfied. Kojima and Hirabayashi [131] discuss some conditions which determine the type of stationary solutions based on a Morse index [165] which is defined as the number of negative eigenvalues of the Hessian matrix of associated Lagrangian function.

The original robust pole assignment problem is then reduced to finding zeros of the following system of equations where for notational convenience \( \bar{x} \) is replaced by \( x \).
\[ H(x(t), t) = 0, \quad t \in [0, 1] \]  
\[ (6.90) \]
Suppose, without loss of generality, that the equation defined in (6.90) is such that

\[ H : \mathbb{R}^{n \times n+1} \times [0, 1] \rightarrow \mathbb{R}^{n \times n} \]  

(6.91)

Assume that the mapping \( X : [0, 1] \rightarrow \mathbb{R}^{n \times n} \) satisfying (6.90) is continuously differentiable on \([0, 1]\). Also it will be assumed that \( H \) is continuously differentiable with respect to both \( X \) and \( t \).

Define

\[ \phi(t) = H(X(t), t), \quad \forall t \in [0, 1] \]  

(6.92)

The Davidenko type differential equation which is defined for a scalar valued function in (6.46) can be extended for a matrix valued function as

\[ \frac{d\phi(t)}{dt} = \frac{\partial H}{\partial X} \frac{dX}{dt} + \frac{\partial H}{\partial t}, \quad t \in [0, 1] \]  

(6.93)

Thus \( X(t) \) must satisfy the following differential equation.

\[ \frac{\partial H}{\partial X} \frac{dX}{dt} = -\frac{\partial H}{\partial t}, \quad t \in [0, 1] \]  

(6.94)

And hence if \( \frac{\partial H}{\partial X} \) is nonsingular for all \( X \) and \( t \), (6.94) can be expressed by

\[ \frac{dX}{dt} = -\left( \frac{\partial H}{\partial X} \right)^{-1} \frac{\partial H}{\partial t}, \quad t \in [0, 1], \quad H(X(0), 0) = 0 \]  

(6.95)

The following theorem will be directly applied to evaluate the derivative of equation (6.90) at each iteration of the algorithm I which will be detailed later in this section.

**Theorem 6.6:**

Suppose \( A \in \mathbb{R}^{m \times n}, X \in \mathbb{R}^{m \times r}, Y \in \mathbb{R}^{r \times s} \). Let \( F(X, Y) = AXY \).

Then \( \frac{\partial}{\partial Y} F(X, Y) = A \otimes [\epsilon_{11}, \epsilon_{21}, \cdots, \epsilon_{n1}, \cdots, \epsilon_{ns}] \)  

(6.96)

where \( \epsilon_{ij} \) is the matrix whose \( ij^{th} \) element is 1, and all others are 0. The operator \( \otimes \) is the tensor product (right Kronecker product) defined as:

\[ A \otimes B \equiv \begin{pmatrix} a_{11}B & a_{12}B & \cdots & a_{1n}B \\ \vdots & \vdots & & \vdots \\ a_{m1}B & a_{m2}B & \cdots & a_{mn}B \end{pmatrix} \]  

(6.97)
CHAPTER 6. HOMOTOPY APPROACH

Proof

Note that

\[
F(X,Y) = AXY = \begin{bmatrix}
    a_{11} \cdots a_{1n} & x_{11} \cdots x_{1p} & y_{11} \cdots y_{1q} \\
    \vdots & \ddots & \vdots \\
    a_{m1} \cdots a_{mn} & x_{m1} \cdots x_{mp} & y_{m1} \cdots y_{mq}
\end{bmatrix}
\]

Thus

\[
\frac{\partial F}{\partial X} = \begin{bmatrix}
    (a_{11}y_{11} \cdots a_{11}y_{1p}) & (a_{11}y_{1q} \cdots a_{11}y_{1q}) \\
    \vdots & \ddots & \vdots \\
    (a_{m1}y_{m1} \cdots a_{m1}y_{mp}) & (a_{m1}y_{mq} \cdots a_{m1}y_{mq})
\end{bmatrix}
\]
Furthermore, 

\[
\frac{\partial F}{\partial Y \partial X} = \begin{pmatrix}
    a_{11} \cdots 0 & 0 \cdots 0 & \cdots & 0 \\
    0 & a_{11} \cdots 0 & \cdots & 0 \\
    0 & 0 & \ddots & 0 \\
    0 & 0 & \cdots & a_{11} \cdots 0 \\
    0 & 0 & \cdots & 0 \\
    \vdots & \ddots & \ddots & \ddots \\
    a_{11} \cdots 0 & \cdots & 0 \\
    0 & a_{11} \cdots 0 & \cdots & 0 \\
    0 & 0 & \ddots & 0 \\
    0 & 0 & \cdots & a_{11} \cdots 0 \\
    0 & 0 & \cdots & 0
\end{pmatrix}
\]

\[
= A \otimes [c_{11} c_{21} \cdots c_{pq}]
\]

6.3.3.2 General Matrix valued Optimization Problem

Now consider a more general form of optimization for a matrix valued function.

\[
\min f(X), \ f : R^{m \times n} \rightarrow R^1
\]

s.t \( G(X) \leq 0, \ G : R^{m \times n} \rightarrow R^{p \times q} \)
where \( f(X) \) and \( G(X) \) are continuously differentiable functions with respect to all values of \( X \).

One of the algorithms to be suggested in this section will solve an optimization problem with a scalar valued objective function of a matrix argument subject to the constraints which are expressed in terms of matrix valued functions. The basic idea is to solve a quadratic programming problem whereby the objective function of the original problem is approximated by a quadratic function while the constraints are linearized in a neighbourhood of the stationary points. It is thus necessary to consider how to approximate the nonlinear objective function with matrix argument by a quadratic function, and how to linearize constraints which are in the form of matrix valued functions. Turnbull [234] introduced the Taylor series expansion for a scalar function of a matrix argument. Based on the similar idea, the following theorem and corollaries may easily be derived. Applying these results, the nonlinear objective function in (6.99) can be approximated as a quadratic function, and the constraints in (6.100) can be linearized so that a general nonlinear programming problem can be approximated as a quadratic programming problem.

**Theorem 6.7: Linearization of a matrix valued function.**

Let \( F(X, Y) \) be a matrix valued function, where \( X, Y \) are matrices with appropriate dimensions. Then \( F(X, Y) \) can be approximated by the following first order terms.

Then \( \text{vec} F(X, Y) = \text{vec} F(X_0, Y_0) + F_X(X_0, Y_0)\Delta X + F_Y(X_0, Y_0)\Delta Y + o(\Delta X, \Delta Y) \)

where \( \text{vec} F \equiv (F_{11}, F_{12}, \cdots, F_{mn})^T \), \( F_X, F_Y \) are the partial derivatives of \( F \) with respective to \( X \) and \( Y \), respectively, as defined by Definition 5.1, and \( \Delta X \equiv \text{vec}(X - X_0), \Delta Y \equiv \text{vec}(Y - Y_0) \).

The following corollaries are also straightforward from those of scalar valued functions.
Corollary 6-1:
Let $F(X, Y) = XY$. Then $\text{vec}(F(X, Y)) = \text{vec}(X) \otimes Y + I \otimes Y X + (X \otimes I) Y + o(.)$

Corollary 6-2:
Let $F(X, Y) = AXYB$. Then $\text{vec}(F(X, Y)) = \text{vec}(AX) \otimes YB + A \otimes B Y \otimes AX + (A \otimes B Y) \otimes Y + o(.)$

Using theorem 8, corollaries 1 and 2, the problem in (6.99) - (6.100) can be approximated by a matrix valued quadratic programming problem where the quadratic objective function is minimized subject to the linear constraints.

6.3.3.3 Algorithms Proposed

Consider a Newton type iteration procedure such as that given in equation (6.29). If a partition (6.27) of $[0,1]$ and integer $n_i = 1, ... , N$ can be found such that the sequence of (6.29) is well defined and $x_N$ lies in the domain of convergence of the algorithm for solving (6.28), then the iteration process will be called numerically feasible [20], [144], closed [149] or numerically stable. It is desired to construct a numerically stable algorithm which will produce an optimal solution. Suppose an appropriate homotopy map is defined to generate corresponding continuous homotopy paths so that the proposed algorithm mapping is closed. The following two algorithms are suggested provided such a homotopy function is defined. Algorithm I solves (6.94) directly provided $(\frac{\partial H}{\partial X})^{-1}$ exists. Algorithm II extends the typical Sequential Quadratic Programming algorithm in optimization to solve more general matrix valued optimization problems.

6.3.3.1 Algorithm III

step 0. Select step size $\Delta t = t_{k+1} - t_k$ and a starting point $X_0$. Set $t = 0, X_k = X_0$.

step 1. Update $(\frac{\partial H}{\partial X})^{-1}$ and $\frac{\partial H}{\partial t}$. 

step 2. Proceed an iteration \( X_{k+1} = X_k - (t_{k+1} - t_k)(\frac{\partial H(X_k, t_k)}{\partial X})^{-1} \frac{\partial H(X_k, t_k)}{\partial t} \).

step 3. If \( t < 1 \), go to step 1. Else stop.

6.3.3.2 Algorithm IV

step 0. Partition \( t \in [0, t_0, \ldots, t_N = 1] \). Set \( j = 0, t_0 = 0 \). Select arbitrary starting point \( x_0 \), step-size \( \tau > 0 \) and stopping rule \( \varepsilon > 0 \). Let \( x_0^j = x_0 \).

step 1. Let \( k = 0, t = t_j, X_k = x_j^0 \). If \( t_j > t_N \) go to step 5.

step 2. Evaluate \( \nabla g_h \); the Jacobian of the matrix valued constraint \( g(x) \) at \( x_k \) for \( t = t_j \) by using formulae in [176], and evaluate \( H_k \); the Hessian of the Lagrangian function corresponding to (6.99)-(6.100).

step 3. For given \( x_k \) and \( t \), find \( d_k \) such that

\[
\min_d \nabla f(x_k^T) d + \frac{1}{2} H_k d \quad \text{s.t.} \quad \nabla g_h^T d = -g(x_k, t).
\]

Let \( x_{k+1} = x_k + d_k \) (6.103)

step 4. If step 3 fails to generate a solution at \( t_j \), partition \([t_{j-1}, t_N = 1]\) more finely, and repeat step 3. If \( ||x - x_k||/||x_k|| < \varepsilon \), let \( j = j + 1, x_j^0 = x_k \), go to step 1. Else, let \( k = k + 1 \), go to step 2.

step 5. Trace \( x(t), t \in [0, 1] \) and identify an optimal solution \( x(1) \).

At the initializing step, it is important to choose appropriate intervals of partitioning for the homotopy parameter \( t_{i+1} - t_i \), step-size \( \tau \) and the criteria for further partitioning of the subintervals. Though some rules have been suggested to choose the lengths of intervals for successive iterations (for example, Wacker et al., [240]),
they do not seem to guarantee to minimize the computational effort. Further algorithmic development is required. Note that for the particular robust pole assignment problem formulation in this study, since the objective function and the constraints are quadratic, the Hessian of the Lagrangean function associated is constant. And hence the Hessian updating in step 2 is not required. However, if it is not the case (for example, if poles to be assigned are in a specified region), the updated Hessian approximation may be carried out by appropriate methods such as Quasi-Newton update, rank-one update, rank-two update, etc (see, for example, Gill et al [91]). Step 3 involves solving the set of equation in the form of (6.90) by using a suitable quadratic programming package. The examples in the following section are solved using Nag Library Routine [173] in step 3. In step 4, it is possible to increase the level of partitioning so that the current solution will remain in the convergence region or to decrease it to accelerate the rate of convergence; this will clearly be problem dependent.

6.3.3.4 Stability of the Algorithm

Even under the assumption of the existence of a solution curve, a choice of suitable partitioning of homotopy parameter so that the iteration process will proceed without terminating at an infeasible solution or a discontinuity point is crucial. Avila [20] has discussed the relationship between partitioning of the interval, existence of a homotopy path and convergence properties. Guddat and Nowack [104] propose pathfollowing methods with jumps in the set of local minimizers. Other studies in the context of conditions whereby solutions exist and algorithms converge may be found in [84], [85], [89], [101], [115] and elsewhere.
CHAPTER 6. HOMOTOPY APPROACH

6.3.3.5 Numerical Example

Example 6.3:
Consider the same example in Example 3.4. Any appropriate homotopy mapping can be used to solve this problem. If the homotopy mapping defined in (6.85) is used to construct an imbedded problem, an initial solution can easily be found as 
\[ X = A^{-1}, Y = A. \]
The starting point thus may be chosen as
\[
\begin{bmatrix}
-0.54906721 & -6.17977527 & -1.50747614 & -4.72357399 \\
0.42511117 & 2.78410779 & 0.86916665 & 2.18079921 \\
2.15892411 & 13.0631438 & 3.9046376 & 9.31477473 \\
2.2288779 & 13.8537587 & 4.22669746 & 9.79116179 \\
1.3800000 & -0.20770000 & 6.71500000 & -5.67800000 \\
-0.58140000 & -4.29000000 & 0.00000000 & 0.67500000 \\
1.0670000 & 4.27300000 & -6.65400000 & 5.85300000 \\
0.04800000 & 4.27300000 & 1.34300000 & -2.10400000
\end{bmatrix}
\]
The algorithm IV generates the homotopy path shown in Table 6.3. As stated earlier, algorithms based on the homotopy approach achieve global converge; they are guaranteed to generate a sequence of points which converge to an optimal solution for almost all arbitrary chosen starting points, as seen in this example. From the results presented in the table, it is evident that though a feasible solution may not be generated during the iteration process, at the final stage a feasible and optimal solution will be obtained.

Example 6.4:
Suppose \((A, B, C)\) in Example 3.4 is transformed to \((\tilde{A}, \tilde{B}, \tilde{C})\) with
\[
\tilde{A} = TAT^T =
\begin{bmatrix}
-0.47560000 & 2.82530000 & 0.40030000 & -2.13310000 \\
-2.17920000 & -3.08630000 & 3.25180000 & -0.40030000 \\
1.18910000 & 5.23630000 & 0.12330000 & -2.15280000 \\
-6.53820000 & 4.21150000 & 4.66130000 & -8.22940000
\end{bmatrix}
\]
where
\[
T = \begin{bmatrix}
0.37796447 & 0.57735027 & 0.15430335 & 0.70710678 \\
0.37796447 & 0.57735027 & 0.15430335 & -0.70710678 \\
0.75592895 & -0.57735027 & 0.30860670 & 0.00000000 \\
0.37796447 & 0.00000000 & -0.92682010 & 0.00000000
\end{bmatrix}
\]

The poles to be assigned are the same as in Example 6.3. The homotopy mapping defined in (6.85) is used to construct an imbedded problem, where an initial solution is \( Y = A \) and \( X = A^{-1} \). The homotopy path obtained is shown in Table 6.4.

**Example 6.5 [206]:**

Another example where \( n = 5, m = 3, p = 3 \), is solved by the proposed algorithm where \( A, B \) and \( C \) are given as follows.

\[
A = \begin{bmatrix}
0 & 0 & -0.0034 & 0 & 0 \\
0 & -0.041 & 0.0013 & 0 & 0 \\
0 & 0 & -1.1471 & 0 & 0 \\
0 & 0 & -0.0036 & 0 & 0 \\
0 & 0.094 & 0.0057 & 0 & -0.0510
\end{bmatrix}
\]

\[
B = \begin{bmatrix}
-1 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0.948 \\
0.916 & -1 & 0 \\
-0.598 & 0 & 0
\end{bmatrix}
\]

\[
C = \begin{bmatrix}
0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0
\end{bmatrix}
\]
The open-loop poles are \((0, 0, -0.051, -0.041, -1.1471)\) and the poles to be assigned are \((-1.1461, -0.0572, -0.5220, -0.2159, -0.1898)\). Since the \(A\) matrix is singular, the homotopy mapping defined in (6.85) is inappropriate. The homotopy mapping given in equation (6.86) is therefore used. An initial values of \(X_0\) and \(Y_0\) can be chosen arbitrarily as long as \(X_0Y_0 = I\) holds. For this particular example, the following initial values are chosen.

\[
X_0 = \begin{bmatrix}
-1.14610700 & 1.00000000 & 1.00000000 & 1.00000000 & 1.00000000 \\
1.00000000 & -0.05721231 & 1.00000000 & 1.00000000 & 1.00000000 \\
1.00000000 & 1.00000000 & -0.52193050 & 1.00000000 & 1.00000000 \\
1.00000000 & 1.00000000 & 1.00000000 & -0.21591262 & 1.00000000 \\
1.00000000 & 1.00000000 & 1.00000000 & 1.00000000 & -0.18979777
\end{bmatrix}
\]

\[
Y_0 = \begin{bmatrix}
-0.38643204 & 0.16133769 & 0.11207358 & 0.14027998 & 0.14335898 \\
0.16133769 & -0.61837347 & 0.22750676 & 0.28476385 & 0.29101412 \\
0.11207358 & 0.22750676 & -0.49902286 & 0.19781182 & 0.20215359 \\
0.14027998 & 0.28476385 & 0.19781182 & -0.57483091 & 0.25303111 \\
0.14335898 & 0.29101412 & 0.20215359 & 0.25303111 & -0.58189409
\end{bmatrix}
\]

The algorithm generates the homotopy path shown in Table 6.5. The condition number achieved in this example, 91.76, is fairly large. However, it compares with the condition number of 348.49 obtained by Bruun et al. Experience shows that the algorithm based on the homotopy methods proposed in this paper produces results which are as robust as those achieved by the alternative approach used in [176]. The results also compare favourably with the results obtained using other output feedback algorithms [48], [167].

As discussed in Chapter 4, the condition number of the closed-loop system matrix measures the sensitivity of its eigenvalue changes with respect to its parameter variations. For example, suppose all elements of matrix \(A\) in Examples 6.3 and 6.6 are truncated two digits after the decimal points, respectively, while the other
elements, $B$, $C$ and the feedback gain $K$ remain unchanged. The eigenvalues corresponding to the new closed-loop system matrices become

**Example 6.3 ($\kappa_2 = 6.79757$):**

\[-5.05659997, -0.20000000, -8.66590000, -1.00000000]\n
$\rightarrow (-5.05059613, -0.19342667, -8.66088917, -1.00958801)$

**Example 6.5 ($\kappa_2 = 91.76129$):**

\[-1.14610700, -0.52193050, -0.18979777, -0.05721231, -0.21591262]\n
$\rightarrow (-1.14000000, -0.68160967, -0.04465299, -0.04000000, -0.21559754)$.

This demonstrates that Example 6.5 which has the larger condition number experiences the larger perturbation in its eigenvalues. However, minimizing this condition number has produced a robust closed-loop system in both cases. The examples presented here are of exact pole-placement problems. It has been shown in the research that to assign the poles of a closed-loop system in a specified region would produce better condition numbers; thus, better robustness.
**Table 6.3. Homotopy path generated (Example 6.3)**

<table>
<thead>
<tr>
<th>t value</th>
<th>Eigenvalues assigned</th>
<th>Cond No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-2.12724140 ± 4.30914794, -2.12724140, -4.30914794</td>
<td>9.91403524</td>
</tr>
<tr>
<td>0.2</td>
<td>-2.17327325 ± 12.13324691, -14.02704394, 2.09805509</td>
<td>34.72591973</td>
</tr>
<tr>
<td>0.4</td>
<td>-10.76508544, 1.21939642, -0.0051551, -2.21417982</td>
<td>8.94515731</td>
</tr>
<tr>
<td>0.6</td>
<td>0.40122028 ± 0.01865151, -9.08729611, -4.58387105</td>
<td>7.02782299</td>
</tr>
<tr>
<td>0.8</td>
<td>-8.62035083, 0.47079030, -0.0051410, -5.15022932</td>
<td>5.17972649</td>
</tr>
<tr>
<td>0.82</td>
<td>-8.6338471, 0.45373493, -0.71112675, -5.12473460</td>
<td>4.93416836</td>
</tr>
<tr>
<td>0.84</td>
<td>-8.63564704, 0.53732492, -0.92710235, -5.0291071</td>
<td>4.4913600</td>
</tr>
<tr>
<td>0.86</td>
<td>-8.60736806, 0.44933811, -0.91906473, -5.10203832</td>
<td>4.52219572</td>
</tr>
<tr>
<td>0.88</td>
<td>-8.5960953, 0.37833603, -0.92426307, -5.17205296</td>
<td>4.61405487</td>
</tr>
<tr>
<td>0.90</td>
<td>-8.54718906, 0.3087004, -5.2126592, -0.93628828</td>
<td>4.78156549</td>
</tr>
<tr>
<td>0.92</td>
<td>0.22550344, -5.21322128, -5.5109888, -0.94839349</td>
<td>5.06452704</td>
</tr>
<tr>
<td>0.94</td>
<td>-8.18747242, 0.1332356, -8.57377184, -0.96434197</td>
<td>5.43714790</td>
</tr>
<tr>
<td>0.96</td>
<td>-5.14779714, 0.0304351, -8.60420655, -0.98211034</td>
<td>5.86264528</td>
</tr>
<tr>
<td>0.98</td>
<td>-5.1025090, -0.0765337, -8.63603983, -0.99628862</td>
<td>6.32102846</td>
</tr>
<tr>
<td>1.00</td>
<td>-5.06509997, -0.2000000, -8.6659000, -1.0000000</td>
<td>6.79757500</td>
</tr>
</tbody>
</table>

The output feedback gain matrix with respect to the final values of poles assigned is

\[
\begin{bmatrix}
-0.2891180029 & -0.1049637291 & 0.1256089884 \\
-0.1251367545 & 0.3011657556 & 0.5250445580
\end{bmatrix}
\]

with the eigenvector matrix corresponding to the poles assigned;

\[
\begin{bmatrix}
-1.30157464 & 0.71368395 & 0.26867869 & 1.70425455 \\
-0.09424289 & 0.7323116 & -1.1186304 & 0.00778209 \\
-0.05399523 & -1.1424834 & 0.04888353 & -0.60529897 \\
-0.60619298 & -0.56912793 & 0.57429733 & -0.24197061
\end{bmatrix}
\]
### Table 6.4. Homotopy path generated (Example 6.4)

<table>
<thead>
<tr>
<th>t value</th>
<th>Eigenvalues assigned</th>
<th>Cond No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.63861386, ±3.99898500, -5.62175332, 0.81986855</td>
<td>9.91408</td>
</tr>
<tr>
<td>0.2</td>
<td>-9.66530729, 0.66002466, -0.87494474, 1.95185376</td>
<td>18.2375</td>
</tr>
<tr>
<td>0.4</td>
<td>-10.58402457, -0.02579496, -2.27836498, 2.15919297</td>
<td>8.89423</td>
</tr>
<tr>
<td>0.6</td>
<td>-9.14360234, -0.05662286, -4.10266402, 1.00301456</td>
<td>5.34935</td>
</tr>
<tr>
<td>0.8</td>
<td>-5.04969772, 0.48405770, -0.62603107, -8.53288953</td>
<td>4.69436</td>
</tr>
<tr>
<td>0.82</td>
<td>-4.99447639, -0.80113031, 0.54958617, -8.53288953</td>
<td>4.47889</td>
</tr>
<tr>
<td>0.84</td>
<td>-5.08453326, -0.79668058, 0.45759143, -8.48419335</td>
<td>4.51061</td>
</tr>
<tr>
<td>0.86</td>
<td>-5.14905821, -0.81783305, -8.44851472, 0.38865785</td>
<td>4.5931</td>
</tr>
<tr>
<td>0.88</td>
<td>-5.19112394, -0.84180487, -8.43108665, 0.32021882</td>
<td>4.78623</td>
</tr>
<tr>
<td>0.9</td>
<td>-5.20688099, -0.86921875, -8.43650701, 0.24730494</td>
<td>5.02633</td>
</tr>
<tr>
<td>0.92</td>
<td>-5.19016790, -0.90447774, 0.16959292, -8.46659217</td>
<td>5.35210</td>
</tr>
<tr>
<td>0.94</td>
<td>-5.16831030, 0.08753057, -8.51141043, -0.93549333</td>
<td>5.70980</td>
</tr>
<tr>
<td>0.96</td>
<td>-5.13638341, 0.00010020, -8.56255212, -0.96623751</td>
<td>6.07780</td>
</tr>
<tr>
<td>0.98</td>
<td>-5.09437044, -0.09470911, -8.61494648, -0.99921671</td>
<td>6.44314</td>
</tr>
<tr>
<td>1.00</td>
<td>-5.05660000, -0.20000003, -8.66590004, -0.99999997</td>
<td>6.79823</td>
</tr>
</tbody>
</table>

The output feedback gain matrix with respect to the final values of poles assigned is

\[
\begin{bmatrix}
-0.2890943138 & -0.1049723256 & 0.1256070524 \\
-0.1252246885 & 0.3011778047 & 0.5250393934
\end{bmatrix}
\]

The eigenvector matrix corresponding to the poles assigned is

\[
\begin{bmatrix}
0.98339067 & -0.11437147 & 0.13005204 & 0.38424464 \\
0.12607319 & -0.91923706 & 0.94288836 & 0.72636161 \\
0.04619975 & 0.23642553 & -0.86404505 & 1.09709559 \\
0.44199524 & -1.32732777 & -0.05613850 & 1.20456361
\end{bmatrix}
\]
Note that, as stated in corollary 1-1 and theorem 2, the above eigenvector matrix can be found from the result of Example 6.1 by post multiplying its corresponding eigenvector matrix by the chosen orthogonal transformation matrix \( T \). The proposed algorithm does not need any extra effort to assign complex poles. For example, suppose one pair of complex poles and two real poles are to be assigned at \((-3.0 \pm i1.2, -2.0, -1.0)\). The homotopy path, assigned poles and feedback gain matrix obtained are as shown in the following Table 6.5.

<table>
<thead>
<tr>
<th>t value</th>
<th>Eigenvalues assigned</th>
<th></th>
<th>Cond No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-0.96794161 ± i2.19131852, -2.93269016, -2.19631672</td>
<td></td>
<td>9.91403</td>
</tr>
<tr>
<td>0.1</td>
<td>0.55441513 ± i2.19322557, -10.29346006, 0.43399611</td>
<td></td>
<td>34.3236</td>
</tr>
<tr>
<td>0.9</td>
<td>-3.25456650, -2.91568332, -1.70602925, -0.58025749</td>
<td></td>
<td>10.79570</td>
</tr>
<tr>
<td>0.91</td>
<td>-3.09989289 ± i0.40136660, -1.71584600, -0.61879144</td>
<td></td>
<td>10.70876</td>
</tr>
<tr>
<td>0.98</td>
<td>-3.03225179 ± i1.00418904, -1.92370293, -0.90797251</td>
<td></td>
<td>10.45061</td>
</tr>
<tr>
<td>0.981</td>
<td>-3.03171142 ± i0.99385511, -1.92747617, -0.91245576</td>
<td></td>
<td>10.45326</td>
</tr>
<tr>
<td>1.000</td>
<td>-3.00000000 ± i1.20000008, -1.99999992, -1.00000008</td>
<td></td>
<td>10.53854</td>
</tr>
</tbody>
</table>

The output feedback gain matrix with respect to the final values of poles assigned is

\[
\begin{bmatrix}
  0.203398296 & -0.0503092845 & -0.0270814881 \\
-0.0589773023 & 0.178687854 & 0.5883817299
\end{bmatrix}
\]
The eigenvector matrix corresponding to the poles assigned is

\[
\begin{bmatrix}
-0.34906134 + 0.44194730i & 0.91980726 & 0.69080573 \\
-0.15871475 + 0.17392491i & 0.05622453 & -0.05493451 \\
0.63221365 \pm i0.0 & -0.38222093 & 0.22590646 \\
0.39095388 + 0.27360461i & -0.06856063 & 0.68379738
\end{bmatrix}
\]

Table 6.6. Homotopy path generated (Example 6.5)

<table>
<thead>
<tr>
<th>t value</th>
<th>Eigenvalues assigned</th>
<th>Cond No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-0.69261152, -0.20508917, -1.14637905, -0.05301294, -0.04063631</td>
<td>3.88946</td>
</tr>
<tr>
<td>0.1</td>
<td>-0.44355449 \pm i, 0.15122322, -1.14011324, -0.06822017, -0.04160973</td>
<td>2.86809</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>0.9</td>
<td>-1.18709539, -0.78912739, 0.00979154, -0.11340589, -0.05238060</td>
<td>15.63106</td>
</tr>
<tr>
<td>0.91</td>
<td>-1.16573294, -0.58440310, -0.13873092, -0.21460419, -0.02809826</td>
<td>56.07427</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>0.98</td>
<td>-1.16356187, -0.46426906, -0.04385293, -0.23082273, -0.22258898</td>
<td>41.37004</td>
</tr>
<tr>
<td>0.981</td>
<td>-1.16299111, -0.46817943, -0.04406554, -0.23251671, -0.22342003</td>
<td>41.85066</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>0.999</td>
<td>-1.14717959, -0.51817744, -0.05690996, -0.19412618, -0.21579280</td>
<td>85.51794</td>
</tr>
<tr>
<td>1.000</td>
<td>-1.14610700, -0.52193050, -0.18070977, -0.05721231, -0.21591262</td>
<td>91.76129</td>
</tr>
</tbody>
</table>
The output feedback gain matrix with respect to the final values of poles assigned is

\[
\begin{bmatrix}
-0.0752904766 & 0.2843051695 & 0.0509630615 \\
-1.3262060314 & 1.3257710275 & 0.6992609004 \\
-11.5632836063 & 8.4731617847 & -3.0586645686 \\
\end{bmatrix}
\]

The eigenvector matrix corresponding to the poles assigned is obtained as

\[
\begin{bmatrix}
0.01450540 & -0.02521325 & -0.07122008 & -0.79590301 & 0.03261831 \\
-0.00288849 & -0.30613159 & -0.01583180 & -0.00085088 & -0.05795625 \\
2.45545847 & 3.81776944 & 5.85692013 & 0.11448447 & 6.63366249 \\
0.08963654 & -0.52318512 & -0.02997500 & 0.13265330 & -0.95168840 \\
-0.00801332 & -0.25913469 & -0.14021891 & -0.62802325 & -0.30367493
\end{bmatrix}
\]

Experience shows that the results obtained by the homotopy methods proposed in this chapter are as robust as those achieved by the alternative traditional approach used in Chapter 3 and also compare favourably with the results obtained using alternative output feedback algorithms [45], [167].

6.4 Concluding Remark

It is well known that the homotopy methods are mathematically elegant and guarantee global convergent properties. Nevertheless, they have not been widely applied to solve engineering problems. An approach based on the homotopy method has been proposed to solve the robust output feedback pole placement problem, which guarantees global convergence regardless of the feasibility of an initial starting point. The proposed algorithm has been seen to generate optimal solutions efficiently for both well-conditioned and ill-conditioned problems. More practical engineering application problems may be solved effectively by the proposed method.
Chapter 7

Case Study - Pole Assignment for an Aircraft Control System Design

7.1 Introduction

Aircraft flight control systems are interesting areas for the application of modern control theory. Since the aircraft has more degrees of freedom to be controlled than those of, say, surface vehicles, their operational requirements for flight conditions and their ability to change from one condition to the other will be more rigid. More complicated automatic flight control systems (AFCS), such as Vertical Short Take-Off and Landing (VSTOL), must have additional operational requirements to those for conventional aircraft, which may call for more sophisticated design considerations. Flight control is distinct from guidance of flight. McRuer et al ([162], pp13-15) define these two concepts as follows.

Guidance: The action of determining the course and speed, relative to some reference system, to be followed by a vehicle.

Control: The development and application to a vehicle of appropriate
forces and moments that (1) Establish some equilibrium state of vehicle motion (operating point control). (2) Restore a disturbed vehicle to its equilibrium (operating point) state and/or regulate, within desired limits, its departure from operating point conditions (stabilization). Flight control is concerned only with vehicle motion quantities measured in the aircraft, while the guidance involves axis system transformations that put the vehicle and target on comparable terms.

Problems related to flight control seem to have been raised even before aircraft were invented. McRuer and Graham [163] have traced back the history of flight control to the "Early Dawn Age". The Early Dawn Ages (from earliest times to 1901) are the period during which dynamical aircraft stability and flight control theory were studied by Maxwell, Routh and Maxim, etc., in 1868, 1877 and 1891, respectively. It is known that Hiram Maxim designed and built what they called the heavier-than-air machine in 1891, though unfortunately, it has never flown. Nevertheless, his idea of a flight control system which used an actuator to deflect the elevator and employed a gyroscope to provide a feedback signal, which was identical to a present-day pitch attitude control system remains unique ([161], pp10). In 1891, Lanchester was the first to investigate the dynamic stability of aircraft analytically. More detailed historical surveys may be found in [162] [163], [164]. In 1903, Bryan and Williams derived the conditions for the longitudinal stability of a machine heavier than air by introducing the linearized equation of motion which is the foundation of the dynamic stability of aircraft these days. Bryan introduced the theory of longitudinal and lateral motion in 1911. During the ninety years since the Wright Brothers invented the glider at the end of 1901, the aircraft and the flight control systems have developed rapidly. Ever since the first successful auto pilot aircraft, a U.S. Air Force Douglas C-54 crossed the Atlantic under automatic control, from Stephenville, Canada to Brize Norton, England, in 1947 (New York Times, Sept. 23, 1947), appearances of supersonic jet aircraft, automatic landing aircraft, fly-by-wire
control aircraft, manned space vehicles, Space Shuttle, V/STOL, and developing an all-electric airplane, etc. have increased the requirements for more sophisticated flight control systems development.

This chapter discusses the basic goals of flight control systems. Maintaining specified equilibrium states of aircraft motions and adjusting pilot control deficiencies are the main concern. Though aircraft motion involves highly nonlinear equations, they can be linearized for small deviations from stationary flight conditions such as constant altitude and velocity, and small angle of attack etc. A decoupled linearized model of the longitudinal and the lateral motions of an aircraft is known to be analytically tractable for control system design instead of employing a complicated full nonlinear model directly. By using a linearized model with fixed values of aerodynamic, equilibrium flight condition and other related parameters, pole locations of the closed-loop system matrix may characterize the quality of flight control. A lateral linearized model of seventh order and associated flight system matrices are used in the case study. The main purpose of this case study is to demonstrate how the robust pole assignment methods and algorithms proposed in this research may effectively be applicable for solving control system design problem rather than actual flight control system design. As discussed in the previous chapters, the main purposes of robust pole assignment is to locate the closed-loop poles to prescribed positions so as to make them minimally sensitive to perturbations of system parameters. With several different open-loop linearized lateral state matrix A's, three different controllers are designed by assigning closed-loop poles to prescribed positions using different pole assignment methods. Design 1 is obtained using a modal assignment method which is a direct pole assignment method. Designs 2 and 3 are obtained using robust pole assignment methods. It will be seen that the controller with the more robust poles assigned gives the more "robust flight control"; the controller obtained with smaller condition number maintains the closed-loop
system properties closer to the nominal ones as far as transient behaviour, closed-loop poles, command input variations, settling time and steady state errors are concerned, as the system parameters vary.

7.2 Purposes of Flight Control Systems

The basic goals of AFCS are to establish and maintain specified equilibrium states of the aircraft motion and stability, and to achieve aircraft handling quality. The stability requirement is to maintain both static and dynamic stability. If a disturbance to an aircraft causes the resulting forces and moments acting on the aircraft to tend initially to return the aircraft to the kind of flight path for which its controls are set, the aircraft is said to be statically stable. If, as a result of a disturbance, the aircraft tends to return eventually to its equilibrium flight path, and remains at that position for some time, the aircraft is said to be dynamically stable [161]. Handling quality requirements are for the aircraft to be safely controllable by the pilot without any exceptional piloting skill. More specifically, the flight control system is required ([162], pp625):

i) To establish and maintain the established flight path in the presence of disturbances such as gusts, winds, and wind shears.

ii) To reduce flight path errors to zero in a stable, well-damped, and rapidly responding manner.

iii) To establish an equilibrium flight condition.

iv) To limit the speed or angle of attack excursions from this established equilibrium flight condition.
These requirements will be achieved by employing both longitudinal control and lateral control which are concerned with longitudinal motion and lateral motion of aircraft. Longitudinal control includes the following operational functions ([162], pp603).

i) Pitch attitude control
ii) Vertical gust regulation
iii) Path (deviation from prescribed datum) control, frontside operation
iv) Path control, backside operation
v) Speed control
vi) Horizontal gust regulation

Lateral control includes ([162], pp613):

i) Roll attitude control
ii) Roll attitude regulation
iii) Rudder-induced rolling
iv) Turn coordination
v) Path control
vi) Inertial cross-coupling suppression

7.3 Mathematical Model

As stated above, the equations of aircraft motion may be decoupled into longitudinal motion and lateral motion. The former is related to forward velocity and pitching
attitude while the later is concerned with roll, yaw and sideslip velocity. Longitudinal motion and lateral motion of aircraft are controlled by three control surfaces; elevators, rudder and ailerons. Though these motions are originally described in full-nonlinear equations they are usually linearized in practical applications. The linearized flight control systems model can be expressed in the following standard form:

\[
\dot{x} = Ax + Bu \\
y = Cx
\]  

(7.1)  
(7.2)

where choices of the state variables and the control variables will depend upon the types of aircraft under study and design objectives.

In this case study, a lateral linearized model of a light fixed wing aircraft is considered, where the state vector, system's inputs and outputs are defined as follows [170]:

\[
x = \begin{bmatrix}
v \\ \rho \\ \gamma \\ \phi \\ \psi \\ \zeta \\ \xi \\ \zeta_o \\ \xi_o \\ \gamma_{wo}
\end{bmatrix}
\]

sideslip velocity(m/s)  
roll rate(rad/s)  
yaw rate(rad/s)  
roll angle(rad)  
washout filter state  
rudder angle(rad)  
aileron angle(rad)  
rudder angle command(rad)  
aileron angle command(rad)  
washed out yaw rate

\[
u = \begin{bmatrix}
\rho \\ v \\ \phi
\end{bmatrix}
\]

roll rate  
sideslip velocity  
bank angle

\[
y = \begin{bmatrix}
\rho \\ v \\ \phi
\end{bmatrix}
\]

(7.3)  
(7.4)  
(7.5)
and the $A$ and $B$ matrices are given by

$$A = \begin{bmatrix}
y_v & 0 & y_r & -U_0 & 0 & y_z & 0 \\
l_v & l_p & l_r & 0 & 0 & l_z & l_t \\
n_v & n_p & n_r & 0 & 0 & n_z & n_t \\
0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0.5 & 0 & -0.5 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \mu_1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \mu_2 \\
\end{bmatrix}, \quad B = \begin{bmatrix}
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & K_1 \\
0 & K_1 \\
\end{bmatrix} \tag{7.6}
$$

with appropriately defined $C$ matrix, where $y_v, l_v$, and $n_v$ are aerodynamic derivatives, $U_0$ is the nominal forward air speed and $\mu_i, K_i$ are the rudder and aileron actuator parameters, respectively.

### 7.4 Eigenstructure Assignment

In linear system theory, it is well known that the free response of the system $\dot{x} = Ax, x(0) = x_0$ is a composition of motions, along eigenvectors of the matrix $A$, which are called the modes of the system. A particular mode is excited by choosing the initial state to have a component along the corresponding eigenvector ([137], pp16). The response of a linear system depends upon (1) eigenvalues, (2) eigenvectors and (3) initial condition. As stated in earlier chapters, the eigenstructure defines the dynamical characteristics of the system. The eigenvalues determine the time-domain characteristic of a mode; they determine the rate of decay or rise-time of each mode, and the nature of any oscillatory behaviour. The eigenvector determines the shape of the mode. More specifically, the right eigenvectors determine the state variables participating in the response of each mode, while the left eigenvectors determine the modes excited by fluctuations in the state variables. An initial condition determines the degree to which each mode will be excited along the corresponding
eigenvector. Thus, for a control system design, the eigenvalue and eigenvector need to be chosen suitably. By using a decoupled linearized model of longitudinal and lateral dynamics, with reference to an equilibrium flight path and to fixed values of aerodynamic, propulsive, inertial and structural parameters, the handling qualities can be expressed in terms of pole location with an assigned tolerance in the complex plane [37]. The flight control requirements thus may be accomplished by assigning closed-loop poles to the specified region so as to make them least sensitive to perturbations of system parameters. A suitable choice of eigenvectors which characterize the shape of mode will also be a main concern. Sensitivity analysis for the controller designed with respect to variations of system parameters should not be excluded in the design process to complete the study.

Before discussing an aircraft flight control system design with a particular example, it is necessary to review an eigenstructure assignment method, the modal pole assignment method, which has been frequently used for flight control system design in application. A main merit of the method is eigenvectors as well as eigenvalues may be assignable under some mild assumptions. It is possible to assign a set of desired eigenvectors which are chosen so as to produce the required handling qualities. The aircraft lateral motion is characterized by roll, Dutch roll and spiral modes, and each of these modes are required to meet prescribed characteristics such as speed of mode (slow or fast), level of damping, etc. Suppose a set of suitable eigenvalues for the closed-loop has been chosen as

\[
\lambda_1, \lambda_2 : \text{(Dutch roll mode)} \\
\lambda_3 : \text{(spiral mode)} \\
\lambda_4 : \text{(roll mode)}
\]

(7.7)

For the modally assigned controller design, the desired eigenvectors corresponding to the poles to be assigned need to be appropriately defined. By inspecting the lateral linearized flight model (7.1) together with (7.6), it is clear that undesirable coupling between the rolling and yawing motions is inevitable without re-designing.
associated eigenvectors. The following choice of desired corresponding eigenvectors will decouple the rolling motions from the yawing motions to conform to handling quality criteria

\[
\begin{bmatrix}
1 & x \\
0 & 0 \\
x & 1 \\
0 & 0
\end{bmatrix} ;
\begin{bmatrix}
0 \\
x \\
x \\
x
\end{bmatrix} = 1 ;
\begin{bmatrix}
0 \\
x \\
x \\
x
\end{bmatrix} = x
\]

(7.8)

where \( x \) represents an unspecified component and 1 is a dominant one. The vectors \( v_1 - v_4 \) are chosen to decouple the rolling motions from the yawing motions so as to achieve favourable handling qualities. This may indeed prevent rolling motions from causing significant change in yawing motions and vice versa. In general, however, a desired eigenvector cannot be achieved by modal pole assignment methods. If an eigenvector \( v_i \) which lies precisely in the subspace spanned by the columns of \((\lambda_i I - A)^{-1}B\), it will be achieved exactly. But since, in practice, usually it is not the case, a best possible eigenvector which is the projection of a desired eigenvector onto the subspace spanned by the columns of \((\lambda_i I - A)^{-1}B\) is instead chosen. A detailed discussion of the modal pole assignment method may be found in [15].

### 7.5 Numerical Example

Consider now the design of an output feedback controller for the lateral dynamics of a light aircraft where the equations of motion are described by equations (7.1)-(7.2) and the state vectors, inputs and outputs are defined as (7.3)-(7.5). Nominal

<table>
<thead>
<tr>
<th></th>
<th>dutch roll</th>
<th>spiral</th>
<th>roll</th>
</tr>
</thead>
<tbody>
<tr>
<td>( v_1 )</td>
<td>[1 \ x ]</td>
<td>[0 ]</td>
<td>[0 ]</td>
</tr>
<tr>
<td>( v_2 )</td>
<td>[0 \ 0 ]</td>
<td>[x ]</td>
<td>[1 ]</td>
</tr>
<tr>
<td>( v_3 )</td>
<td>[x \ 1 ]</td>
<td>[x ]</td>
<td>[0 ]</td>
</tr>
<tr>
<td>( v_4 )</td>
<td>[0 \ x ]</td>
<td>[1 ]</td>
<td>[x ]</td>
</tr>
</tbody>
</table>
system matrices trimmed at 33 m/s airspeed are given as follows ([170], adapted).

\[
A = \begin{bmatrix}
-0.277 & 0.00 & -32.900 & 9.810 & 0.00 & -5.432 & 0.000 \\
-0.103 & -3.325 & 3.750 & 0.00 & 0.00 & 0.00 & -28.640 \\
0.365 & 0.00 & -0.639 & 0.00 & 0.00 & -9.490 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
\end{bmatrix}
\]

(7.9)

\[
B = \begin{bmatrix}
0.00 \\
0.00 \\
0.00 \\
0.00 \\
0.00 \\
0.00 \\
0.00 \\
0.00 \\
0.00 \\
0.00 \\
\end{bmatrix}, \quad C = \begin{bmatrix}
0.00 & 1.00 & 0.00 & -1.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
1.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 1.00 & 0.00 & 0.00 \\
\end{bmatrix}
\]

(7.10)

where the open loop poles are \((-0.5, -8.3591, 0.1217, -0.5018 \pm i3.5085, -10.0, -5.0)\). The design objective is to construct a suitable output feedback controller to provide both tracking capability and regulating capability. That is, (1) to ensure the ability of aircraft to change from one steady-state to another steady-state without excessive fluctuations in transient responses and (2) to minimize the possible transient fluctuations upon the aircraft. The open-loop transient responses of each state to an initial value of roll angle \(\phi = 0.1\) rad and sideslip velocity \(v = 1\) m/s, respectively, while all others are set to be zero are shown in Figs. 7.1-7.2. As one can see in the figures, most open-loop transient responses are diverging indefinitely as time elapses since the model is open-loop unstable. Thus, some poles need to be assigned to more appropriate locations. A set of suitable poles for the closed-loop
system are chosen as

\[
\begin{align*}
\lambda_1, \lambda_2 &= -1.5 \pm i1.5 \quad \text{(Dutch roll mode)} \\
\lambda_3 &= -0.05 + \alpha_2 \quad \text{(spiral mode)} \quad (7.11) \\
\lambda_4 &= -4 + \alpha_1 \quad \text{(roll mode)}
\end{align*}
\]

where \(\alpha_1, \alpha_2 \geq 0\) are taken to be appropriate increments to facilitate robust design.

The remaining poles are free in the left half plane. For this particular example, the following three different controller designs are considered to compare the transient responses and their robustness with respect to variations in the system parameters. Design I applies the modal pole assignment method, and Design II and Design III are based on the robust pole assignment method.

**Design I**
Since the dimension of the state is greater than the number of inputs plus the number of outputs, not all the poles may be assignable to desired positions. The number of assignable poles may not exceed \(\min\{n, m + p - 1\} = 5\). To apply the modal pole assignment method, the chosen poles to be assigned are

\[
\begin{align*}
\lambda_1, \lambda_2 &= -1.5 \pm i1.5 \\
\lambda_3 &= -0.05 \\
\lambda_4 &= -4
\end{align*}
\]

A set of desired eigenvectors is chosen as (7.8) so as to achieve the required handling qualities. The following results are obtained by the modal method.
The achievable eigenvectors are

\[
X = \begin{bmatrix}
0.99778 \pm 0.00000 & -0.00000 & 0.00673 \\
0.00000 \pm 0.00000 & -0.04548 & 0.95618 \\
0.03143 \mp 0.04401 & 0.27404 & -0.06066 \\
0.00000 \pm 0.00000 & 0.00636 & -0.23905 \\
-0.01493 \mp 0.00040 & 0.30449 & 0.00952 \\
0.03427 \mp 0.00806 & -0.01701 & -0.02335 \\
0.00053 \mp 0.00576 & 0.04902 & -0.15315
\end{bmatrix}
\]

with associated feedback gain matrix:

\[
K_1 = \begin{bmatrix}
0.0284451933 & -0.0073370497 & 0.0139478239 & -0.0087169467 \\
0.0565898178 & -0.0150820466 & 0.0005340570 & -0.006097815
\end{bmatrix}
\]

The closed-loop poles are at \((-1.50000001 \pm 1.49999998, -4.00000001, -0.05000000, -8.51078005 \pm 0.69943087)\) where the condition number of the closed-loop system matrix defined by \(\kappa(A + BK)\) as (4.17) is 922.366. Note that the eigenvectors obtained above are close to the desired modal matrix in (7.8).

**Design II**

A pair of complex poles \(\lambda_1, \lambda_2 = 1.5 \pm \iota 1.5\) corresponding to the Dutch roll mode are fixed and the \(\lambda_3\) is allowed to vary. The poles to be assigned are

\[
\lambda_1, \lambda_2 = -1.5 \pm \iota 1.5 \\
-1.0 \leq \lambda_3 \leq -0.05 \\
\lambda_4 = -4
\]

Using the proposed robust pole assignment algorithm, the feedback gain matrix is

\[
K_2 = \begin{bmatrix}
0.0231812254 & -0.0108873392 & 0.0146260958 & -0.0203104704 \\
0.0183167160 & 0.0172267139 & 0.0048128872 & 0.1263201633
\end{bmatrix}
\]

The assigned poles are \((-1.50000000 \pm \iota 1.50000000, -1.00000000, -4.00000000, -8.20310165 \pm \iota 1.12146684, -0.33479670)\) with condition number 728.193. In this...
case $\lambda_3$ has reached its lower bound.

**Design III**

Here the poles to be assigned are allowed more flexibility. One pair of complex poles are assigned to be a specified location and all the other poles are to be in a specified region instead of fixed positions. That is

$$\lambda_1, \lambda_2 = -1.5 \pm j1.5$$

$$-5 \leq \lambda_3, \lambda_4 \leq -3$$

and others are free in the left half plane. The robust pole assignment algorithm produced the following results.

Feedback gain matrix:

$$K_3 = \begin{bmatrix}
0.0381182931 & -0.0100151349 & 0.0132751184 & -0.0233988071 \\
0.0540216720 & 0.0524016936 & -0.0058095986 & 0.2540614614
\end{bmatrix},$$

where the poles have been assigned to $(-1.49999950 \pm j1.50000010, -4.21326704, -3.00000000, -7.02395993 \pm j0.00000000, -0.47981411)$ with a condition number of 593.591.

The closed-loop transfer functions $H_i(s) = \frac{\Delta i}{\Delta i}$, relating roll rate to rudder demand corresponding to each controller, $K_1, K_2$, and $K_3$, respectively, are as follows.

$$H_1 = \frac{\{(s + 1.5)^2 + 1.5^2\}(s + 0.5)}{(s + 1.5)^2 + 1.5^2} \frac{(s + 8.511)^2 + 0.999}{(s + 4.0)(s + 0.689)(s + 0.055)} \frac{(s + 2.384)^2 + 3.674^2}{(s + 0.499)}$$

$$H_2 = \frac{\{(s + 1.5)^2 + 1.5^2\}((s + 8.203)^2 + 1.121)(s + 4.0)(s + 0.689)(s + 0.335)}{(s + 4.233)(s - 1.788)(s + 0.303)}$$

$$H_3 = \frac{\{(s + 1.5)^2 + 1.5^2\}((s + 7.024)^2 + 1.0)(s + 4.213)(s + 4.709)(s + 3.0)}{(s + 1.6)^2 + 1.6^2} \frac{(s + 7.024)^2 + 1.0}{(s + 4.213)(s + 4.709)(s + 3.0)}$$

The modal pole assignment method is, as the name indicates, the one which attempts to decouple particular modes from the others while the robust pole assignment method seeks to assign the poles to maximally insensitive locations. Thus it is expected that the poles assigned by the modal assignment method will be modally decoupled so that command inputs to a specified state may not significantly affect
the required modes. However, robustly assigned controllers are expected to produce much more robust results though some couplings among particular modes may be inevitable. Note that, in $H_1$, the Dutch roll mode is cancelled by decoupling zeros, while the desired modal structure has not been maintained in $H_2$ and $H_3$. In other words, the rolling motion is exactly decoupled from the yawing motion when controller $K_1$ is applied to the nominal system as intended. However, controllers $K_2$ and $K_3$ do not define such a modal structure. More information about coupling and decoupling effects may be obtained from Fig. 7.3 which compares the modal shapes produced by the controllers obtained from both the modal pole assignment method and the robust pole assignment method, respectively, when an initial yaw rate $\gamma(0) = 0.02$ rad/sec is applied. As expected, it is observed that there are coupling effects on rolling motion into sideslip velocity or yaw rate in the robustly assigned controller designs. However, the modally decoupled controller has prevented rolling motions from causing significant yawing motions. The overall closed-loop transient responses of each design are compared in Figs. 7.4-7.6, followed by their detailed overview of Figs. 7.7-7.12. In Tables 7.8-7.9 and Table 7.10, times to reach steady state and the maximum overshoot of each of the trajectories are summarized. As seen in Fig. 7.4 or Figs. 7.7-7.8, Design I which is obtained using modal assignment method produces slow transient responses. In particular, settling times of yaw rate($x_3$), roll angle($x_4$), washout filter state($x_2$) and aileron angle($x_7$) for initial bank angle $\phi(0) = 0.1$ rad take more than 120 seconds (Fig. 7.7 and Table 7.8). Design II and Design III which are obtained by the robust pole assignment method give more rapid steady state transient responses in less than 10 seconds as seen in Figs. 7.5-7.6, 7.9-7.11. The command inputs applied at an initial sideslip velocity $v(0) = 1$ m/s are not very different in both modally decoupled and robustly assigned controllers as seen in Figs. 7.34,7.36, 7.38. At an initial roll angle $\phi_0 = 0.1$ rad, the initial aileron angle commands applied to the robustly assigned controllers are larger than the maximum aileron angle commands for the modally
decoupled design, but still do not exceed the physical limits (Figs. 7.33, 7.35, 7.37).

The transient response of each individual state for different designs, at $\phi(0) = 0.1$ rad and $v(0) = 1$ m/s, respectively, are compared more precisely in Figs. 7.13-7.26. Time history of sideslip velocity and roll rate for each design as in Figs. 7.13-7.16 exhibit to reach stable steady states in a first few seconds. However, roll angle response ($x_4$) and washout filter state ($x_5$) for Design I fail to reach a stable steady state in a reasonable time period (Figs. 7.19-7.22). The method of robust pole assignment in a specified region assigns the poles to the least sensitive location, but it does not attempt to assign the fastest ones. Nevertheless, for this particular example, the robust poles happen to be the fastest ones. The more robust controller has produced the shorter settling time. However, the mode of maximum overshoot does not seem to have any direct relationship between a more robust controller and a less robust controller.

As discussed in Chapter 2, no model can be perfect. That is, a model is not a full representation of the real system but simply a partial representation of it. In fact, unmodelled dynamics and/or simplified models may seriously misconstrue the results of studies. There may be other problems. Suppose, a model has been appropriately constructed and an optimal solution has been obtained. However, such an optimal solution will be meaningful only when the decision variables and all established relationships and the properties among the components of the system remain unchanged or at least have not changed significantly. Nevertheless, the internal parameter variations of the underlying system and external disturbances may be inevitable. Therefore, a good control system design must reduce sensitivity to such variations, disturbances and unmodeled dynamics. Robustness is one of the main concerns of system design as discussed in the previous chapters.

To see how the perturbations in the system parameters affect the lateral states for robust and non-robust controller designs, respectively, consider variations in
linearized lateral state matrix $A$ derived from a nonlinear simulation at 33 m/s airspeed for different yaw angles and roll angles ([16], adapted) given by Tables 7.2-7.3. The closed-loop matrices $M = [A + BKC]$ are calculated with corresponding nominal feedback gain matrix $K$'s obtained by Designs I, II and III, respectively. The closed-loop poles assigned for each controller and condition numbers $\kappa_2(M)$ obtained are shown in Tables 7.4 - 7.6. The damping factors and natural frequencies associated with the dominant poles for each perturbed system are shown in Table 7.7. Controller I, the modally decoupled controller, assigns the poles for the perturbed systems as shown in Table 7.4. Desired poles are assigned for the nominal model with condition number 902.566. As expected, when the system is perturbed, the assigned poles are sensitive and in some cases have moved to unstable locations. However, the robustly assigned controllers have produced a well conditioned closed-loop response for the perturbed systems(Table 7.5-7.6). Damping factors corresponding to the dominant poles for each perturbed system also seem to be more favourable in robust controllers than non-robust one(Table 7.7). Although natural frequencies do not exhibit any particular tendency, the Dutch roll mode maintains 0.7 damping ratio by all designed controllers. This damping ratio is the required level for flight handling qualities. Figs. 7.27-7.32 show how the transient response of each state induced by each design is sensitive with respect to the parameter variations in $A$ matrix. As expected, the more robust controller produces the more robust result. That is, all the transient responses of more robust controller with smaller condition number reach the steady state in shorter periods of time and are less sensitive to the parameter variations compared to those of less robust controllers which have larger condition numbers. The time history of control inputs for each design under nominal and perturbed systems are as shown in Figs. 7.33-7.38. It was observed that the command inputs for Design I have been continuously applied when the system is perturbed as seen in Figs. 7.39-7.40 which are re-plotted for extended time period up to 200 seconds for Figs. 7.33-7.34.
However, for Design III, the command inputs need to be applied for not more than 10 seconds (Figs. 7.37-7.38).

7.6 Concluding Remarks

In this case study, a linearized model of a light fixed wing aircraft is used to evaluate how much the controllers obtained by different methods are robust in the sense that how closely each controller maintains a given closed-loop system in a desired state regardless of the system parameters variations. The example in this case study has demonstrated that the robust controller with the robust closed-loop poles will maintain better closed-loop system properties. Design 3, the most robust controller obtained using robust pole assignment method by assigning the poles in a prescribed region instead of fixed locations, produces the fastest transient responses. Perturbations in $A$ result in smaller deviation to steady-state responses than one experienced with the other controllers. Input commands for the perturbed systems are not very different to those of the nominal system. This case study demonstrates that robust pole assignment may indeed guarantee least variations in the closed-loop system properties. Nevertheless, maximum overshoot of each transient response could not be reduced by manipulating solely the location of closed-loop poles. This may be because the condition number itself cannot influence anything directly concerning the behaviour of overshoots. Moreover, simply varying one of the closed-loop pole locations cannot adjust the overshoot of the corresponding trajectory for multi-input multi-output system. To solve this problem, it will be necessary to reformulate the pole assignment problem to incorporate an objective function which can measure a performance of overshoot.
roll angle = 0.0 rad

\[ A_1 = \begin{bmatrix}
-0.2540 & 0.0074 & -30.3390 & 9.8050 & 0 & -4.6200 & 0 \\
-0.0935 & -7.8620 & 3.8200 & 0 & 0 & 0 & -24.3590 \\
0.3366 & 0.0627 & -0.3075 & 0 & 0 & -8.0090 & 0 \\
0 & 1.0000 & -0.0325 & -0.0031 & 0 & 0 & 0 \\
0 & 0 & 0.5000 & 0 & -0.5000 & 0 & 0 \\
0.2700 & -0.1467 & 0.5689 & -0.1743 & -0.5689 & -10.0000 & 0 \\
-0.0102 & -0.0549 & 0.4452 & 0.2789 & -0.4452 & 0 & -5.0000 \\
\end{bmatrix} \]

roll angle = 0.15 rad

\[ A_2 = \begin{bmatrix}
-0.2562 & -0.0087 & -29.7880 & 9.7932 & 0 & -4.5400 & 0 \\
-0.1110 & -7.7900 & 2.9940 & 0 & 0 & 0 & -27.9500 \\
0.3330 & -0.0504 & -0.2799 & 0 & 0 & -7.8700 & 0 \\
0 & 1.0000 & -0.0584 & 0.0042 & 0 & 0 & 0 \\
0 & 0 & 0.5000 & 0 & -0.5000 & 0 & 0 \\
0.2700 & -0.1467 & 0.5689 & -0.1743 & -0.5689 & -10.0000 & 0 \\
-0.0102 & -0.0549 & 0.4452 & 0.2789 & -0.4452 & 0 & -5.0000 \\
\end{bmatrix} \]

roll angle = 0.25 rad

\[ A_3 = \begin{bmatrix}
-0.2667 & -0.0018 & -29.0800 & 9.8000 & 0 & -4.5000 & 0 \\
-0.1222 & -7.7650 & 3.9400 & 0 & 0 & 0 & -23.7599 \\
0.3320 & -0.2150 & -0.2880 & 0 & 0 & -7.8120 & 0 \\
0 & 1.0000 & -0.0250 & 0.0079 & 0 & 0 & 0 \\
0 & 0 & 0.5000 & 0 & -0.5000 & 0 & 0 \\
0.2700 & -0.1467 & 0.5689 & -0.1743 & -0.5689 & -10.0000 & 0 \\
-0.0102 & -0.0549 & 0.4452 & 0.2789 & -0.4452 & 0 & -5.0000 \\
\end{bmatrix} \]

Table 7.2: Lateral $A$ matrix variation with roll attitude: 33 m/s airspeed
[CHAPTER 7. CASE STUDY]

yaw angle = 0.0 rad

\[
A_4 = \begin{bmatrix}
-0.2564 & 0.0033 & -30.3900 & 0.8000 & 0 & -4.6340 & 0 \\
-0.0944 & -7.8735 & 3.3580 & 0 & 0 & 0 & -24.4300 \\
0.3371 & 0.1143 & -0.3000 & 0 & 0 & -8.0300 & 0 \\
0 & 1.0000 & -0.0402 & -0.0068 & 0 & 0 & 0 \\
0 & 0 & 0.5000 & 0 & -0.5000 & 0 & 0 \\
0.2790 & -0.1467 & 0.5689 & -0.1743 & -0.5689 & -10.0000 & 0 \\
-0.0102 & -0.0549 & 0.4452 & 0.2789 & -0.4452 & 0 & -5.0000
\end{bmatrix}
\]

yaw angle = 0.15 rad

\[
A_5 = \begin{bmatrix}
-0.2540 & -0.0083 & -30.0880 & 9.6855 & 0 & -4.5420 & 0 \\
-0.0964 & -7.7900 & 3.4500 & 0 & 0 & 0 & -23.9440 \\
0.3340 & -0.2090 & -0.2800 & 0 & 0 & -7.8740 & 0 \\
0 & 1.0000 & -0.0433 & 0.0133 & 0 & 0 & 0 \\
0 & 0 & 0.5000 & 0 & -0.5000 & 0 & 0 \\
0.2790 & -0.1467 & 0.5689 & -0.1743 & -0.5689 & -10.0000 & 0 \\
-0.0102 & -0.0549 & 0.4452 & 0.2789 & -0.4452 & 0 & -5.0000
\end{bmatrix}
\]

yaw angle = 0.25 rad

\[
A_6 = \begin{bmatrix}
-0.2527 & -0.0005 & -30.0030 & 9.4920 & 0 & -4.5130 & 0 \\
-0.1000 & -7.7700 & 3.9400 & 0 & 0 & 0 & -23.7940 \\
0.3330 & -0.1830 & -0.2900 & 0 & 0 & -7.8240 & 0 \\
0 & 1.0000 & -0.0114 & 0.0031 & 0 & 0 & 0 \\
0 & 0 & 0.5000 & 0 & -0.5000 & 0 & 0 \\
0.2790 & -0.1467 & 0.5689 & -0.1743 & -0.5689 & -10.0000 & 0 \\
-0.0102 & -0.0549 & 0.4452 & 0.2789 & -0.4452 & 0 & -5.0000
\end{bmatrix}
\]

Table 7.3: Lateral A matrix variation with sideslip angle attitude : 33 m/s airspeed
### Table 7.4: Closed loop poles for each feedback gain obtained by Design I

<table>
<thead>
<tr>
<th>System</th>
<th>Poles assigned</th>
<th>Cond. No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_0$</td>
<td>$-1.50000001 \pm i.49999998, -4.00000001, -0.00000001, -0.04999999, -8.20310165 \pm i.80997412, -8.88804370, -0.33239217$</td>
<td>728.19</td>
</tr>
<tr>
<td>$A_1$</td>
<td>$-1.04935013 \pm i.73060775, -4.00000001, -0.00000001, -0.04999999, -8.20310165 \pm i.80997412, -8.88804370, -0.33239217$</td>
<td>575.72</td>
</tr>
<tr>
<td>$A_2$</td>
<td>$-1.00344343 \pm i.84280339, -4.1123279, -0.96464977, -8.10153476 \pm i.71693033, -0.35003102$</td>
<td>575.72</td>
</tr>
<tr>
<td>$A_3$</td>
<td>$-1.0279322 \pm i.86539385, -4.5652982, -0.75993419, -8.09820988 \pm i.873204068, -0.29275156$</td>
<td>576.91</td>
</tr>
<tr>
<td>$A_4$</td>
<td>$-1.0448654 \pm i.7084666, -3.99863351, -0.93202738, -8.28678049 \pm i.88300798, -0.34347500$</td>
<td>570.47</td>
</tr>
<tr>
<td>$A_5$</td>
<td>$-1.0114299 \pm i.8827061, -4.0061195, -0.77521493, -8.09847279 \pm i.59292269, -0.31710153$</td>
<td>578.06</td>
</tr>
<tr>
<td>$A_6$</td>
<td>$-1.03534145 \pm i.86192389, -4.4586281, -0.76901586, -8.10758287 \pm i.840069302, -0.3183714$</td>
<td>577.66</td>
</tr>
</tbody>
</table>

Desired pole location: $(A_1, A_2, A_3, A_4) = (1.5 \pm i.5, -4.0, -1.0)$

### Table 7.5: Closed loop poles for each feedback gain obtained by Design II

<table>
<thead>
<tr>
<th>System</th>
<th>Poles assigned</th>
<th>Cond. No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_0$</td>
<td>$-1.50000001 \pm i.49999998, -4.00000001, -0.00000001, -0.04999999, -8.20310165 \pm i.80997412, -8.88804370, -0.33239217$</td>
<td>728.19</td>
</tr>
<tr>
<td>$A_1$</td>
<td>$-1.04935013 \pm i.73060775, -4.00000001, -0.00000001, -0.04999999, -8.20310165 \pm i.80997412, -8.88804370, -0.33239217$</td>
<td>575.72</td>
</tr>
<tr>
<td>$A_2$</td>
<td>$-1.00344343 \pm i.84280339, -4.1123279, -0.96464977, -8.10153476 \pm i.71693033, -0.35003102$</td>
<td>575.72</td>
</tr>
<tr>
<td>$A_3$</td>
<td>$-1.0279322 \pm i.86539385, -4.5652982, -0.75993419, -8.09820988 \pm i.873204068, -0.29275156$</td>
<td>576.91</td>
</tr>
<tr>
<td>$A_4$</td>
<td>$-1.0448654 \pm i.7084666, -3.99863351, -0.93202738, -8.28678049 \pm i.88300798, -0.34347500$</td>
<td>570.47</td>
</tr>
<tr>
<td>$A_5$</td>
<td>$-1.0114299 \pm i.8827061, -4.0061195, -0.77521493, -8.09847279 \pm i.59292269, -0.31710153$</td>
<td>578.06</td>
</tr>
<tr>
<td>$A_6$</td>
<td>$-1.03534145 \pm i.86192389, -4.4586281, -0.76901586, -8.10758287 \pm i.840069302, -0.3183714$</td>
<td>577.66</td>
</tr>
</tbody>
</table>

Desired pole location: $(A_1, A_2, A_3, A_4) = (1.5 \pm i.5, -4.0, -1.0)$

Table 7.4: Closed loop poles for each feedback gain obtained by Design I

Table 7.5: Closed loop poles for each feedback gain obtained by Design II
### Table 7.6: Close loop poles for each feedback gain obtained by Design III

<table>
<thead>
<tr>
<th>System</th>
<th>Poles assigned</th>
<th>Cond. No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_0$</td>
<td>$-1.49909950 \pm 1.50000010, -4.21326703, -3.00000000, -7.02395993 \pm 1.00000000, -0.47981411$</td>
<td>593.59</td>
</tr>
<tr>
<td>$A_1$</td>
<td>$-1.06792012 \pm 1.768831611, -4.25551961, -2.48519465, -7.62157263, -6.95576286, -0.47239029$</td>
<td>468.56</td>
</tr>
<tr>
<td>$A_2$</td>
<td>$-0.97940300 \pm 1.74642455, -3.69722019, -2.48519465, -8.08009181, -5.91423043, -0.47430138$</td>
<td>457.60</td>
</tr>
<tr>
<td>$A_3$</td>
<td>$-1.0071863 \pm 1.69429775, -4.88615115, -2.86208690, -8.18728255, -5.40934017, -0.46850194$</td>
<td>452.54</td>
</tr>
<tr>
<td>$A_4$</td>
<td>$-1.07132464 \pm 1.79758236, -4.10968800, -2.46919307, -7.48770121, -7.19460721, -0.47271921$</td>
<td>468.44</td>
</tr>
<tr>
<td>$A_5$</td>
<td>$-0.95620292 \pm 1.65822362, -5.42335185, -2.86321180, -8.1540913, -4.92520902, -0.47248258$</td>
<td>461.84</td>
</tr>
<tr>
<td>$A_6$</td>
<td>$-1.0085663 \pm 1.69643342, -4.68703660, -2.86322456, -8.12811338, -5.65265977, -0.47187264$</td>
<td>458.71</td>
</tr>
</tbody>
</table>

Desired pole location: $(-1.5 \pm 1.5, \text{all others are free in stable region})$,
### Design I

<table>
<thead>
<tr>
<th>System</th>
<th>Dominant Poles assigned</th>
<th>Damping</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_0$</td>
<td>$-1.50000000 \pm i1.49999999$</td>
<td>0.7071</td>
<td>2.1213</td>
</tr>
<tr>
<td>$A_1$</td>
<td>$-1.04181280 \pm i1.76171587$</td>
<td>0.5090</td>
<td>2.0467</td>
</tr>
<tr>
<td>$A_2$</td>
<td>$-0.98296004 \pm i1.76070735$</td>
<td>0.4875</td>
<td>2.0165</td>
</tr>
<tr>
<td>$A_3$</td>
<td>$-1.0318626 \pm i1.76358626$</td>
<td>0.5059</td>
<td>2.0445</td>
</tr>
<tr>
<td>$A_4$</td>
<td>$-1.03092811 \pm i1.76102369$</td>
<td>0.5052</td>
<td>2.0406</td>
</tr>
<tr>
<td>$A_5$</td>
<td>$-1.0265338 \pm i1.78310906$</td>
<td>0.4862</td>
<td>2.0565</td>
</tr>
<tr>
<td>$A_6$</td>
<td>$-1.04415032 \pm i1.78440228$</td>
<td>0.5030</td>
<td>2.0874</td>
</tr>
</tbody>
</table>

### Design II

<table>
<thead>
<tr>
<th>System</th>
<th>Dominant Poles assigned</th>
<th>Damping</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_0$</td>
<td>$-1.50000000 \pm i1.49999999$</td>
<td>0.7071</td>
<td>2.1213</td>
</tr>
<tr>
<td>$A_1$</td>
<td>$-1.04935013 \pm i1.73069778$</td>
<td>0.5185</td>
<td>2.0239</td>
</tr>
<tr>
<td>$A_2$</td>
<td>$-1.00344343 \pm i1.84280398$</td>
<td>0.4782</td>
<td>2.0083</td>
</tr>
<tr>
<td>$A_3$</td>
<td>$-1.02703222 \pm i1.86504388$</td>
<td>0.4826</td>
<td>2.1299</td>
</tr>
<tr>
<td>$A_4$</td>
<td>$-1.04448654 \pm i1.70848666$</td>
<td>0.5216</td>
<td>2.0025</td>
</tr>
<tr>
<td>$A_5$</td>
<td>$-1.01141299 \pm i1.88827061$</td>
<td>0.4722</td>
<td>2.1421</td>
</tr>
<tr>
<td>$A_6$</td>
<td>$-1.03534145 \pm i1.86192529$</td>
<td>0.4360</td>
<td>2.1304</td>
</tr>
</tbody>
</table>

### Design III

<table>
<thead>
<tr>
<th>System</th>
<th>Dominant Poles assigned</th>
<th>Damping</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_0$</td>
<td>$-1.49999995 \pm i1.50000010$</td>
<td>0.7071</td>
<td>2.1213</td>
</tr>
<tr>
<td>$A_1$</td>
<td>$-1.06792012 \pm i1.75831611$</td>
<td>0.5191</td>
<td>2.0572</td>
</tr>
<tr>
<td>$A_2$</td>
<td>$-0.97940300 \pm i1.74642455$</td>
<td>0.4891</td>
<td>2.0023</td>
</tr>
<tr>
<td>$A_3$</td>
<td>$-1.00071863 \pm i1.59429778$</td>
<td>0.5316</td>
<td>1.8823</td>
</tr>
<tr>
<td>$A_4$</td>
<td>$-1.07132464 \pm i1.79758236$</td>
<td>0.5120</td>
<td>2.0926</td>
</tr>
<tr>
<td>$A_5$</td>
<td>$-0.98620292 \pm i1.65822562$</td>
<td>0.5112</td>
<td>1.9293</td>
</tr>
<tr>
<td>$A_6$</td>
<td>$-1.00085633 \pm i1.63643242$</td>
<td>0.5218</td>
<td>1.9182</td>
</tr>
</tbody>
</table>

Table 7.7: Damping factors and Natural Frequencies for Dominant Poles
### Table 7.8: Time to reach steady state for initial bank angle $\phi(0) = 0.1$ rad

<table>
<thead>
<tr>
<th>Design</th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_4$</th>
<th>$x_5$</th>
<th>$x_6$</th>
<th>$x_7$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design 1</td>
<td>10</td>
<td>80</td>
<td>120</td>
<td>120</td>
<td>70</td>
<td>130</td>
<td></td>
</tr>
<tr>
<td>Design 2</td>
<td>15</td>
<td>8</td>
<td>15</td>
<td>15</td>
<td>20</td>
<td>14</td>
<td>12</td>
</tr>
<tr>
<td>Design 3</td>
<td>10</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>18</td>
<td>10</td>
<td>3</td>
</tr>
</tbody>
</table>

### Table 7.9: Time to reach steady state for initial sideslip velocity $v = 0.1$ m/s

<table>
<thead>
<tr>
<th>Design</th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_4$</th>
<th>$x_5$</th>
<th>$x_6$</th>
<th>$x_7$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design 1</td>
<td>40</td>
<td>60</td>
<td>60</td>
<td>120</td>
<td>100</td>
<td>8</td>
<td>80</td>
</tr>
<tr>
<td>Design 2</td>
<td>7</td>
<td>8</td>
<td>10</td>
<td>16</td>
<td>17</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>Design 3</td>
<td>7</td>
<td>4</td>
<td>3</td>
<td>8</td>
<td>13</td>
<td>7</td>
<td>7</td>
</tr>
</tbody>
</table>

### Table 7.10: Maximum overshoot for $\phi(0) = 0.1$ rad, $v(0) = 1$ m/s

<table>
<thead>
<tr>
<th>Design</th>
<th>$\phi_1$</th>
<th>$\phi_2$</th>
<th>$\phi_3$</th>
<th>$\phi_4$</th>
<th>$\phi_5$</th>
<th>$\phi_6$</th>
<th>$\phi_7$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design 1</td>
<td>+0.32</td>
<td>–</td>
<td>+0.03</td>
<td>+0.1</td>
<td>+0.018</td>
<td>+0.0055</td>
<td>+0.0055</td>
</tr>
<tr>
<td></td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Design 2</td>
<td>+0.35</td>
<td>−0.1</td>
<td>−0.005</td>
<td>−0.1</td>
<td>+0.011</td>
<td>+0.0065</td>
<td>+0.022</td>
</tr>
<tr>
<td></td>
<td>−0.24</td>
<td>−0.07</td>
<td>−0.007</td>
<td>+0.1</td>
<td>+0.0078</td>
<td>+0.0085</td>
<td>+0.0032</td>
</tr>
<tr>
<td>Design 3</td>
<td>+1.0</td>
<td>0.0068</td>
<td>+0.037</td>
<td>+0.008</td>
<td>+0.031</td>
<td>+0.022</td>
<td>+0.0014</td>
</tr>
<tr>
<td></td>
<td>−0.3</td>
<td>−0.0092</td>
<td>−0.002</td>
<td>−0.002</td>
<td>−0.004</td>
<td>−0.0092</td>
<td>−0.0004</td>
</tr>
<tr>
<td></td>
<td>−0.1</td>
<td>−0.012</td>
<td>−0.001</td>
<td>−0.007</td>
<td>−0.001</td>
<td>−0.004</td>
<td>−0.003</td>
</tr>
<tr>
<td></td>
<td>+1.0</td>
<td>+0.016</td>
<td>+0.008</td>
<td>+0.003</td>
<td>+0.009</td>
<td>+0.024</td>
<td>+0.081</td>
</tr>
<tr>
<td></td>
<td>−0.3</td>
<td>−0.007</td>
<td>−0.002</td>
<td>−0.007</td>
<td>−0.011</td>
<td>+0.023</td>
<td>+0.0041</td>
</tr>
<tr>
<td></td>
<td>−0.3</td>
<td>−0.007</td>
<td>−0.002</td>
<td>−0.007</td>
<td>−0.011</td>
<td>+0.012</td>
<td>+0.004</td>
</tr>
</tbody>
</table>

$(\text{rad, rad/s})$
Figure 7.1: Open loop transient responses for $\phi(0) = 0.1$ rad

Figure 7.2: Open loop transient responses for $v(0) = 1 \text{ m/s}$
For initial yaw rate=0.2 rad/s

** solid line: Design 1, dot line: Design 2, dash dot line: Design 3.

Figure 7.3: Level of Decoupling of each design at $\gamma(0) = 0.2 \text{ rad/s}$
Figure 7.4: Overall responses for Design 1 at $\phi(0) = 0.1$ rad, $v(0) = 1$ m/s

Figure 7.5: Overall responses for Design 2 at $\phi(0) = 0.1$ rad, $v(0) = 1$ m/s

Figure 7.6: Overall responses for Design 3 at $\phi(0) = 0.1$ rad, $v(0) = 1$ m/s
Figure 7.7: Detailed view of transient responses for Design 1, $\phi(0)=0.1$ rad

Figure 7.8: Detailed view of transient responses for Design 1, $v(0)=1$ m/s
Figure 7.9: Detailed view of transient responses for Design 2, $\phi(0) = 0.1$ rad

Figure 7.10: Detailed view of transient responses for Design 2, $v(0) = 1$ m/s
Figure 7.11: Detailed view of transient responses for Design 3, $\phi(0)=0.1$ rad

Figure 7.12: Detailed view of transient responses for Design 3, $v(0)=1$ m/s
[CHAPTER 7. CASE STUDY]

Figure 7.13: Sideslip velocity ($X_1$) responses to initial bank angle, $\phi(0)=0.1$ rad

Figure 7.14: Sideslip velocity ($X_1$) responses to initial sideslip velocity, $v = 1$ m/s
Figure 7.15: Roll rate ($X_2$) responses to initial bank angle, $\phi(0)=0.1$ rad

Figure 7.16: Roll rate ($X_2$) responses to initial sideslip velocity, $v = 1$ m/s
Figure 7.17: Yaw rate ($X_3$) responses to initial bank angle, $\phi(0)=0.1$ rad

Figure 7.18: Yaw rate ($X_3$) responses to initial sideslip velocity, $v = 1$ m/s
Figure 7.19: Roll angle ($X_4$) responses to initial bank angle, $\phi(0)=0.1$ rad

Figure 7.20: Roll angle ($X_4$) responses to initial sideslip velocity, $v = 1$ m/s
Figure 7.21: Washout filter state \( X_5 \) to initial bank angle, \( \phi(0) = 0.1 \) rad

Figure 7.22: Washout filter state \( X_5 \) to initial sideslip velocity, \( v = 1 \) m/s
Figure 7.23: Rudder angle ($X_6$) responses to initial bank angle, $\phi(0)=0.1$ rad

Figure 7.24: Rudder angle ($X_6$) responses to initial sideslip velocity, $v = 1$ m/s
Figure 7.25: Aileron angle ($X_7$) responses to initial bank angle, $\phi(0)=0.1$ rad

Figure 7.26: Aileron angle ($X_7$) responses to initial sideslip velocity, $v = 1$ m/s
Figure 7.27: Responses of perturbed systems for Design 1, $\phi(0)=0.1$ rad
Figure 7.28: Responses of perturbed systems for Design 2, $\phi(0)=0.1$ rad
Figure 7.29: Responses of perturbed systems for Design 3, $\phi(0)=0.1$ rad
Figure 7.30: Responses of perturbed systems for Design 1, \( v(0) = 1 \) m/s
Figure 7.30: Responses of perturbed systems for Design 1, $v(0) = 1 \text{ m/s}$
Figure 7.32: Responses of perturbed systems for Design 3, $v(0) = 1 \text{ m/s}$
Figure 7.32: Responses of perturbed systems for Design 3, $v(0) = 1\text{ m/s}$
Figure 7.33: u-y Plot for nominal and perturbed systems with Design 1 at \( \phi(0) = 0.1 \)

Figure 7.34: u-y Plot for nominal and perturbed systems with Design 1 at \( v(0) = 1 \) m/s
[CHAPTER 7. CASE STUDY]

Figure 7.35: u-y Plot for nominal and perturbed systems with Design 2 at $\phi(0) = 0.1$

Figure 7.36: u-y Plot for nominal and perturbed systems with Design 2 at $v(0) = 1$ m/s
Figure 7.37: $u$-$y$ Plot for nominal and perturbed systems with Design 3 at $\phi(0) = 0.1$

Figure 7.38: $u$-$y$ Plot for nominal and perturbed systems with Design 3 at $u(0) = 1$ m/s
Figure 7.39: u-y Plot for nominal and perturbed systems with Design 1 at $\phi(0)=0.1$

Figure 7.40: u-y Plot for nominal and perturbed systems with Design 1 at $v(0)=1$ m/s
Chapter 8

Conclusion and Further Research

As in other fields of science, in control engineering, historically there was a gap between theory and its applications. For example, the basic theory of dynamical stability of aircraft was studied even before aircraft were invented, and still is being applied for designing flight control systems. In general, a theorist does not care much about any possible gap between his theory and the practical application of the theory, but a practical engineer tries to make every effort to compensate for the gaps by applying a known theory directly to industrial problems. The requirements of military weapon systems including radar and fire control systems during World War II, as well as those of guided missiles and the aerospace developments after the War have contributed to a rapid growth in control system technology. Moreover, undoubtedly the advanced modern computer technology, together with the need to develop highly accurate industrial robots, industrial process control, etc., have also stimulated control system design and analysis techniques. These days, in general, most theories seem to be being applied relatively quickly. However, still there seems to remain some gaps between practical applications in the field where the theory has originated and applications in other fields. Sometimes a practical method which has been successfully applied in a particular field of science may not necessarily have
been fully applied in other fields. For example, a theory in pure mathematics, the Catastrophe theory, has raised much interest in the field of applied mathematics. However, it does not seem to be as successful as expected in other fields of applied sciences though the potential applicability may be unlimited.

Optimization theory is a very well-used, fundamental technique and sufficiently refined in application as well as in theory. It has been broadly applied in the areas of applied mathematics, engineering, physics, economics and management, etc., for solving practical problems in related fields during the last few decades. As far as engineering applications are concerned, there are various types of problem which can be solved by their formulation in terms of optimization and the use of appropriate optimization techniques. Pole assignment problem is one such problem which has been extensively studied by many authors for the last twenty years. Some of them are formulated in the context of a typical optimization framework. Nevertheless, strangely enough, much work has not applied such a typical optimization technique by directly attempting to optimize an objective function subject to a given set of constraints. Although they claim that the underlying objective function will be optimized, most algorithms are confined to finding a solution which satisfies the constraints and some properties of the objective function chosen rather than optimal one. Their main concern is to find a feasible solution instead of an optimal solution. There seems to exist a gap between optimization theory and its application in control engineering design problems.

It has been shown that the robust pole assignment problem can be formulated as an optimization problem whose objective function is a scalar valued function with matrix arguments and the constraints are matrix valued functions with matrix arguments. This formulation is unique in the sense that the robust pole assignment problem attempts to directly optimize the given objective function subject to a set of matrix valued constraints. Though there are some pros and cons for a condition
number of the closed-loop system matrix being used as a performance criterion to measure the robustness of the underlying system, it has been demonstrated that minimizing the objective function defined in terms of condition number produces a favourable robust controller design, and hence choice of such an objective function for the robust pole assignment problem has been seen to be reasonable and tractable. However, it should be noticed that the objective function in the formulation is not the condition number itself but is the upper bound of it. Since the condition number is not everywhere differentiable, it has been replaced by a smooth objective function to be more tractable. Nevertheless, this will not reduce the validity of the mathematical model. Trying to minimize the condition number may be too ambitious or may even call for an unnecessary extra effort. The robust pole assignment problem is more concerned with finding a controller which guarantees that the closed-loop poles stay within prescribed regions for all uncertainties in the underlying plant. In fact, it may not necessarily be required to find the controller which will assign maximally insensitive poles. Instead, it may be satisfactory as long as the controller chosen provides the prescribed characteristic for every perturbed plant. Thus, minimizing the upper bound of condition number will be sufficient for the robust control system design purpose.

Unlike other robust pole assignment problem formulations in the literature, a characteristic of this example is that it can generate an optimal solution by directly applying appropriate nonlinear optimization algorithms. However, because of its high nonlinearity as well as its high dimensionality in the formulation, a difficulty of solving this type of problem lies in deriving the Kuhn-Tucker type necessary conditions and solving them in practice. Efficient optimization algorithms which are based on a different approach from the usual optimization methods have thus been studied. A concept of symbolic derivative of matrix valued function which is analogous to the ordinary derivative has been proposed together with the numerical algorithms for solving such problems efficiently. These facilitate the Jacobian
evaluations of matrix valued functions with matrix arguments and the solving of sets of nonlinear equations. Numerical examples have shown that the proposed algorithms provide much more robust results, as far as in terms of condition number is concerned, than other algorithms in the literature.

Most solution processes via optimization methods involve solving sets of linear and/or nonlinear equations which will produce stationary points for optimal solutions. In general, those processes may encounter some difficulties in that global convergence is not guaranteed if the chosen starting points are not sufficiently close to the solution sought. The homotopy method is seen to provide an elegant theoretical framework with the potential to overcome this disadvantage by widening the domain of convergence or generating starting points which are sufficiently close to optimal solutions for the application of other algorithms. The method has been extended to an efficient global convergent algorithm for solving the robust pole assignment problem. It has been demonstrated that the algorithm converge globally to an optimal solution effectively for almost all choice of initial starting points even for ill-conditioned problems.

Finally, it has been seen that a case study of robust pole assignment for an aircraft control system design problem using the methods proposed in this research has produced successful results. A main goal of flight control system design is to maintain stability of aircraft motions over varying flight conditions. Since pole locations of the closed-loop system of linearized model of the longitudinal and the lateral aircraft motions characterize the quality of flight control, assigning poles to their most robust locations by the proposed algorithm has produced the fastest and least sensitive steady-state responses for all perturbed systems. The robust output feedback pole assignment problem formulation as an optimization problem with a special structure of matrix valued functions, and its proposed solution methods have provided the better results in the sense that smaller condition numbers are achieved.
and are maximally insensitive to perturbations of the system matrices compared to other methods or algorithms.

Though the robust pole assignment problem has been formulated as a well structured optimization problem, and algorithms to solve the problem have been proposed, there are some limitations which further research work needs to resolve. One involves defining some other reasonable measures of system performance and compromising the effect of conflicting objectives. The other involves tackling non-smooth optimization problems in the control system design framework. The mathematical formulation of output feedback pole assignment problem as a particular form of nonlinear optimization problem in this study is a part of the whole problem as usual in most optimization problem formulations. A choice of objective function in terms of condition number of the closed-loop system matrix may provide information about the maximum effect of the perturbations in the system matrices or the overall sensitivity of the eigenvalues of a given matrix. Thus, by minimizing the condition number, the least sensitive pole locations may be sought. However, it may not meet other performance measures of robustness such as $LQG$, $H^\infty$ and $\mu$ objectives, etc., which have appeared in other robust control problem formulations. This study has not focused on a realistic control system design but has concentrated on developing solution methods based on optimization theory. To solve more practical engineering problems incorporated with the optimization theory, further research may be required to link optimization theory and control system design. This may include alternative modelling with different measures of robustness such as LQG and $H^\infty$-norm type measure in collaboration with the techniques of Multi-objective optimization and Non-smooth optimization method. Since the condition number may be just one possible measure of system robustness, other robustness indices, appearing in other formulations need to be considered. The linear-quadratic-Gaussian (LQG) problem involves achieving a design objective which is to keep the actual plant state as near to its ideal, desired value as possible for all time horizons.
under consideration. Thus the objective function will include a measure to minimize tracking errors which is usually expressed in quadratic cost functional in the form of

\[ J(x, u) = \int_0^\infty (x^T Q x + u^T R u) dt \]

where \( Q \) and \( R \) are appropriately defined weighting matrices. This ensures that the transient behaviour of the closed-loop system is close to the desired one so that the trajectory sensitivity may be minimized.

\( H^\infty \) design considers the objective function in the form of

\[ J = \sup_{\omega} \sigma_{\text{max}}(E(j\omega)) \]

which provides more flexibility to achieve design objectives in frequency domain when model uncertainty is the main concern. \( H^\infty \) objectives may include requirements such as the bandwidth and the magnitude of resonance peaks of a transfer function. By optimizing the objective function, it may be possible to achieve (1) robust internal stability, (2) disturbance attenuation, and (3) other nominal performance requirements. These different design objectives may sometimes be conflicting, they may even be incommensurable, that is, performances cannot be measured in a common unit and no weighting factors may be appropriately assigned. Multiobjective optimization will thus be one of the recommended approaches for control system design. Another interesting topic is non-smooth optimization. A non-smooth optimization method will provide broader applications to solve more realistic practical problems in control system design when not all of the objective functions and/or the constraints are sufficiently smooth, which is the usual case.
References


[REFERENCES] 196


[REFERENCES] 197


[References] 198


[REFERENCES]


[REFERENCES]


[57] Dupačová, J., 'On some connections between parametric and stochastic programming', in Guddat, J., H. Th. Jongen, B. Kummer and F. Nozicka (eds),
[REFERENCES]


[REFERENCES] 202


[REFERENCES]


[REFERENCES]


[REFERENCES]


[REFERENCES]


[REFERENCES]


REFERENCES


[REFERENCES]


[REFERENCES]


[REFERENCES]


[REFERENCES] 213


[185] Pappalardo, M., 'Stability studies in parametric optimization via the imagespace approach', in Guddat, J., H.Th. Jongen, B. Kummer and F. Nozicka(eds),


[REFERENCES]


[REFERENCES]


[REFERENCES]


[REFERENCES]


[REFERENCES]


