Adaptive Radial Basis Function Interpolation for Time-Dependent Partial Differential Equations

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

by

Syeda Laila Naqvi
Department of Mathematics,
University of Leicester,
England, United Kingdom.

June 2013.

Abstract

In this thesis we have proposed the meshless adaptive method by radial basis functions (RBFs) for the solution of the time-dependent partial differential equations (PDEs) where the approximate solution is obtained by the multiquadrics (MQ) and the local scattered data reconstruction has been done by polyharmonic splines. We choose MQ because of its exponential convergence for sufficiently smooth functions. The solution of partial differential equations arising in science and engineering, frequently have large variations occurring over small portion of the physical domain, the challenge then is to resolve the solution behaviour there. For the sake of efficiency we require a finer grid in those parts of the physical domain whereas a much coarser grid can be used otherwise.

During our journey, we come up with different ideas and have found many interesting results but the main motivation for the one-dimensional case was the Korteweg-de Vries (KdV) equation rather than the common test problems. The KdV equation is a nonlinear hyperbolic equation with smooth solutions at all times. Furthermore the methods available in the literature for solving this problem are rather fully implicit or limited literature can be found using explicit and semi-explicit methods. Our approach is to adaptively select the nodes, using the radial basis function interpolation.

We aimed in, the extension of our method in solving two-dimensional partial differential equations, however to get an insight of the method we developed the algorithms for one-dimensional PDEs and two-dimensional interpolation problem. The experiments show that the method is able to track the developing features of the profile of the solution. Furthermore this work is based on computations and not on proofs.

Dedicated to:
Allah, the Almighty,
And
the Merciful .

Acknowledgements

In the name of Allah, most gracious, most merciful, I would like to say a very big thanks to my Allah for answering my prayers and for guiding me throughout my life and to protect me. May Allah shower His countless blessing and peace upon all His prophets in particular, the last prophet Hazrat Muhammad (peace be upon him (PBUH), and His Ahlulbait who have always been a source of inspiration and guidance in all walks of life.

I would like to express the deepest appreciation to all the people who made this thesis a success. First and the foremost, my utmost gratitude to my supervisor, Prof. Jeremy Levesley, who has an attitude and substance of a genius. He continually and convincingly conveyed a spirit of adventure in research and an excitement in regard to teaching. I must say that Prof. Jeremy Levesley has been a true inspiration not just in my research but also in my life. He has been a source of light in the times when i was quite discouraged. I must say thanks to him for his support, patience, time and for his friendly and caring nature. I would like to extend my gratitude to my co-supervisor, Dr. Fazal-i-Haq (Associate Professor), and I appreciate his support, help, encouragement and advice throughout my time as his student.

I am grateful to all the staff members of the mathematics department and the administration of the university for their help and making so many things easy for me. I am heartily thankful to Dr. Emmanuil Georgoulis, for his valuable ideas, suggestions and discussions throughout my studies in Leicester. He has been a great source of learning for me and despite of his busy schedule he always managed time for my research.

I must acknowledge the support of my family and I salute and express my deep appreciation to my caring and loving father Syed Ali Anjum Naqvi for his training, support, encouragement and my mother Fakhra Ali for her prayers and dreams for my success during the whole of my life. My warmest thanks goes to my beloved husband Syed Mustafa Shah for his patience, support, his sense of understanding and help throughout my stay. The hard time he had during my stay in Leicester means a lot to me and he owes much more than a thanks.

I wish to thank all my honourable teachers and I would also like to extend my appreciation and thanks to all my relatives, friends and fellow research students, Jainxia, Ana, Juxi, Mat, Fazli, Misha, Masha, Peter and Sabika.

My profound gratitude goes to my sponsor, the Higher Education Commission (HEC) of Pakistan for funding my studies of this degree in the UK. Finally, I wish to thank the Vice Chancellor Prof. Syeda Farhana Jehnagir of Shaheed Benazir Bhutto University, Peshawar, Pakistan, for being supportive in all aspects and for granting me study leave to undertake this programme.

Syeda Laila Naqvi,
Leicester, England, UK.
June, 2013.

Contents

List of Figures

List of Tables

Notations and abbreviations

Notation

[$\kappa(A)$] The *condition number* of a matrix A with respect to a given matrix norm ... 22

[σ_{max}] Maximum singular value of a given matrix ... 23

[σ_{min}] Minimum singular value of a given matrix ... 23

[λ_{max}] Maximum eigenvalue of a given matrix ... 23

[λ_{min}] Minimum eigenvalue of a given matrix ... 23

[θ_{crs}] Threshold for coarsening ... 12

[θ_{crs}] Threshold for refining ... 12

[π_m^d] The space of d-variate polynomials of degree not exceeding m ... 20

[$h_{X,\Omega}$] The *fill distance* of the data corresponding to the data sites X in Ω ... 22

[q_X] The *separation distance* of the data sites X in a given domain ... 22

[$\lfloor r \rfloor$] The largest integer that does not exceed r, where $r \in \mathbb{R}$ ... 27

[$\lceil r \rceil$] The smallest integer not less than r, where $r \in \mathbb{R}$ ... 63

[$C^{2k}(\mathbb{R}^d)$] Space of smooth functions on $\mathbb{R}^d$ of smoothness order $2k$ ... 27

[$V_X(x)$] Voronoi tile for any $x \in X$ ... 86

Abbreviation

[RBF] Radial Basis Function ... 3

[MQ] Multiquadrics ... 7

[DOF] Degrees Of Freedom ... 3

[PDE] Partial Differential Equation 1

[PD] Positive Definite 21

[CPD] Conditionally Positive Definite 21

[A.E] Absolute Error 43

[RMS] Root Mean Squared 76

[ODEs] Ordinary Differential Equations 34

[NLPDEs] Non-linear Partial Differential Equations 34

[FEM] Finite Element Method 1

[FDM] Finite Difference Method 1

[FVM] Finite Volume Method 1

[KdV] Kortweg- de Vries 5

[RBF − MOL] Radial basis functions-Method of lines 16

[BC] Boundary Condition 4

Chapter 1

Introduction

1.1 Background

Partial differential equations (PDEs) with solutions that have highly localized properties appear in many areas of application. The solutions of such PDEs sometimes exhibit steep gradients, corners, and rapid topological changes (change of shape, developing discontinuity). Such examples include, shock hydrodynamics [20, 86], transport in turbulent flow fields, moving front problems [7, 23], combustion processes [1], reactive or non-reactive flows [65], and singularities in interface flows [8]. The exact/analytical solution for a large class of PDEs is generally difficult to derive except for the simplest PDE problems. We then require a numerical solution of such problems, which is simply the numerical evaluation of analytical solution. There are cases when even the numerical solution becomes hard to achieve, such as for PDEs solved in complex geometries with nonlinearities.

Numerical simulation is an ongoing research area in engineering and science problems. Most of the techniques for these simulations in complex geometries depends upon discretizing the domain using a grid or triangulation. The major methods include the finite difference (FDM), finite volume (FVM), finite element (FEM), and spectral methods. In finite difference methods the derivatives are approximated using differences of function values on the grid. They are popular for their simplicity but in complex geometries obtaining weights for the grid stencils requires an effort to achieve uniform accuracy.

In FEM, the creation of a mesh grid in higher dimensions is a significant task. The discretization in all of these traditional methods involves some sort of mesh generation or triangulation of the region of interest. In mesh-based methods to create, maintain, and update (i.e., re-meshing) complex meshes takes significant amount of time.

This leads to the question about the possibility of generating the grid points for problems with difficult geometries and irregular domains in higher dimensions. For this reason one might think of *meshfree/meshless* methods. Meshfree methods are meshless in a way that they does not require connectivity of grid/mesh points, however finding nearest neighbours is computationally less intensive than the mesh generation in mesh-based methods. In the next section we will give a brief introduction to the meshfree methods.

1.2 Motivation

Scattered data approximation is a recent fast growing research area and deals with the problem of reconstructing an unknown function from data which may have no particular structure. The applications of the scattered data approximation are prominent in applied mathematics, computer science, geology, biology, engineering to business studies. To name a few specific applications, finance modeling, surface reconstruction, point cloud modeling, medical applications, fluid-structure interaction and the solution of the partial differential equations.

A meshfree method is a numerical method used to establish a system of algebraic equations for the whole domain of the problem without using a predefined mesh for the domain (or boundary) discretization. We will make use of meshfree methods for our scattered data approximation because mesh generation is one of the most time consuming part of any mesh-based numerical simulation. The meshfree method gives an economical alternative to methods such as those using wavelets, multivariate splines, finite elements, finite difference and finite volume, where all require the connectivity of nodes.

Radial basis function (RBF) methods are not tied to a grid and in turn, belong to a category of the above mentioned meshfree methods. This is done by composing a univariate basic function with a norm usually Euclidean, which makes the problem insensitive to the dimension and makes it virtually one-dimensional.

Most of the physical models in science and engineering have variations, shocks or steep gradients where the solutions exhibit large variations over small portions of the domain of interest. The variations usually can be seen in the coefficients, forcing terms, or in the boundary conditions. The use of a uniform mesh for such problems can be computationally expensive especially in multidimensions, where the required degrees of freedom (DoF) i.e., number of nodes/meshes/points can be prohibitively large. An alternative to the uniform grid, called *mesh adaptivity*, is to flag more points in the region of high variation and few points in rest of the domain. Mesh adaptivity is preferred over the uniform grid for the sake of efficiency. The adaptive grid should reflect the profile of the solution. The literature for the adaptive methods using different approaches can be found in [43, 21, 54, 46, 23, 68, 80].

The problems listed in Section 1.1 motivate us to introduce some kind of meshfree adaptive discretization for the PDEs which have highly localized features in space or time. Our motivational model is the third-order nonlinear KdV equation, which shall be discussed briefly in the next section.

1.2.1 Hyperbolic partial differential equations

For numerical methods, hyperbolic PDEs are considered to be a challenging task. Several issues arise for hyperbolic PDEs which do not occur for parabolic or elliptic PDEs [2]. Korteweg- de Vries (KdV) equation belongs to the hyperbolic class of PDEs. We will first discuss about the general challenges for hyperbolic PDEs and numerical schemes for them and then we will give a brief overview of our motivational model i.e., KdV equation.

Hyperbolic PDEs

Hyperbolic PDEs are conservative i.e., non-dissipative and suffers delicate issues of how the boundary conditions (BC) are applied. A simple example of this can be given using linear advection equation. For instance, $u_t - u_x = 0,\ 0 \leq x \leq 1,\ t \geq 0$, let us have $u(0,t) = u_0(x)$, then the solution will be $u(x,t) = u_0(t+x)$. The solution depends on the initial data travelling leftwards with speed $\frac{dx}{dt} = -1$. In this situation we requires only one boundary condition to be specified at $x = 1$ as the characteristic curve will exit at $x = 0$. Similarly for $u_t + u_x = 0$ solution will consist of the initial data travelling rightwards with speed $\frac{dx}{dt} = 1$ which require the boundary condition to be specified only at $x = 0$. The propagation of the information traveling along the characteristic curve in more complex situations needs to be handled very carefully.

It is often encountered that numerical schemes can suffer with numerical dispersion whether or not the PDE is dispersive. For instance, let us again consider the linear advection equation using initial condition $u(x,0) = e^{-kx}$, (k is the wave number). The PDE is not dispersive in itself but the numerical scheme for this PDEs using finite difference semi-discretization in space, suffers from numerical dispersion. To handle this issue, LaxWendroff and leapfrog schemes are discussed in [2].

The eigenvalue stability is another big concern if using explicit method for integrating the ODEs in time where-as using implicit methods can be used but they are computationally expensive. The nonlinear hyperbolic PDEs may have discontinuities in the solutions or in its derivatives, even with smooth initial condition and the error propagates with time. For hyperbolic PDEs unlike the parabolic PDEs, if we have a discontinuity in the initial condition it will travel in time and may not diffuse. For instance, considering the advection equation with step function as the initial condition

$$u_0(x) = \begin{cases} 1, & \text{if } x < 0, \\ 0, & \text{if } x \geq 0, \end{cases}$$

the discontinuity will propagate in time. Figure 1.1 is showing this behaviour. The

detailed discussion for evolutionary PDEs and more issues related to hyperbolic PDEs can be found in [2].

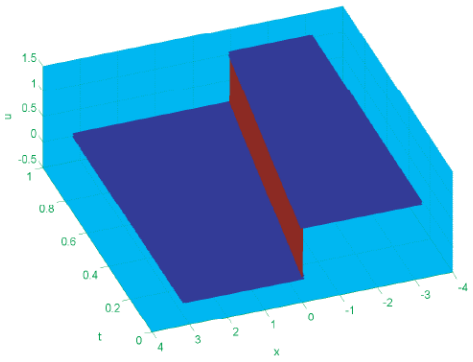


Figure 1.1: Discontinuity propagating in time for advection equation with step function as initial condition.

Korteweg- de Vries (KdV) equation

The nonlinear KdV equation

$$u_t + \epsilon u u_x + \mu u_{xxx} = 0,$$

is a hyperbolic model with smooth solutions at all times. The model describes the interaction between nonlinearity and dispersion. The PDE is of special interest in terms of numerical analysis, that is, an exactly solvable model whose solutions can be exactly specified. For an appropriate initial condition, Garden et al. [33] have shown the existence and uniqueness of the exact solution of the KdV equation.

There have been many methods proposed for the numerical solution of the KdV equation. However, many of these are fully implicit [36] which can be quite expensive. A limited literature concerning explicit schemes or semi-explicit schemes is available in literature [79, 87]. The numerical solution of PDEs by the traditional methods like finite difference, finite element can be expensive in terms of mesh generation for dimensions greater than three. Recently the KdV equation has been solved by RBFs coupled with the method of lines (RBF-MOL) in [74]. The RBF-MOL becomes our motivation to solve the problem with a new adaptive approach which is to solve by using as small number of DoF as possible, to achieve better results.

In terms of adaptivity, a question comes to mind about the possibility of an easy way of generating the grid points and to adaptively select the grid points. The structure of the grid must allow for easy addition and deletion of points, and to determine and process neighbors. For this reason one might think of a meshless method where RBFs can provide a positive answer to this question. We combined the meshfree RBFs with a new adaptive approach for the numerical solution of this problem. The adaption is based on the method of Iske [46]. We will introduce the RBFs in the next chapter in detail however we will give a brief summary of the RBF interpolation and the known adaptive methods based on RBFs.

1.3 Radial basis functions

Radial basis function interpolation, use linear combinations of the translates of one single basic univariate function. The RBFs have been praised in a variety of settings, where one of the chief reason for their success is their ease of implementation in multivariate scattered data approximation. In the last decade, RBF meshless method has become a viable choice for solving PDEs. Kansa introduced this method in [51, 50] for solving elliptic and parabolic PDEs. Franke and Schaback in [32] have used collocation using RBFs. Larsson and Fornberg have applied the idea for elliptic PDEs [53]. A number of papers have been published combining RBFs with the method of lines (MOL) [78, 39, 9, 38, 37, 40, 74] for time dependent PDEs.

A brief introduction of RBF interpolation is given here; the details are given in the next chapter. Let $\mathbf{x} = (x_1, \ldots, x_d) \in \mathbb{R}^d$ and $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\} \subseteq \mathbb{R}^d$ the given data set, RBF interpolant takes the form,

$$s(\mathbf{x}) = \sum_{j=1}^{N} \lambda_j \phi(\| \mathbf{x} - \mathbf{x}_j \|_2), \tag{1.1}$$

where $\| \cdot \|$ is the Euclidean distance, and $\phi : [0, \infty] \to \mathbb{R}$. The interpolation condition

$s(\mathbf{x}_i) = f(\mathbf{x}_i)$ gives the simple linear system

$$\mathbf{A}_{\phi,X}\lambda = f(\mathbf{x}_i), \tag{1.2}$$

to be solved for λ. The interpolation matrix $\mathbf{A}$ is guaranteed to be nonsingular for many choices of ϕ subject to some restrictions (see Chapter 2). We have a great deal of flexibility in choosing a suitable RBF. The exponential convergence depend on how we choose the RBF basis function.

The popular choices of the RBFs are:

- Infinitely smooth: multiquadrics $\phi(r) = (1 + c^2r^2)^{1/2}$ and Gaussian $e^{-(cr)^2}$; both contain a free parameter called the shape parameter.
- Thin plate splines $r^{2k}\log(r)$, where $k \in \{1, 2, ...\}$.
- Compactly supported Wendland's functions [83].

Our numerical approximations are based on *multiquadrics* (MQ). The method was introduced by Hardy in 1971 and the exponential convergence rate of the MQ interpolation for smooth functions was proved by Madych and Nelson in 1992 [55]. The RBF methods suffer with ill-conditioning while giving good accuracy. This relation is described as the uncertainty principal in RBF interpolation and was documented by Schaback in [71]. The infinitely smooth RBFs can gives exponential convergence as compared to the piecewise smooth RBFs where the convergence is of algebraic order. For instance, cubic splines in 1-D, as a special case of RBF interpolation becomes $\phi(r) = |r|^3$. The order of the approximation for such splines (when approximating smooth functions), is $\mathbf{O}(h^4)$ due to the jump in the third derivative at the origin [31]. The limited smoothness of piecewise smooth functions makes the infinitely smooth RBFs a favourable choice [13, 17]. The convergence of the RBFs containing the shape parameter, can be discussed in *stationary* and *non-stationary* settings. In stationary approximation, the number of

the centers N is fixed with varying shape parameter, whereas a non-stationary approximation keeps the shape parameter fixed and increase N. In the non-stationary setting with increasing N the error behaves as

$$|f(\mathbf{x}) - s(\mathbf{x})| \leq e^{-\frac{K(c)}{h}}, \tag{1.3}$$

where $K(c)$ is a constant depends on the value of the shape parameter and h is the fill distance (see Definition 2.9). However, the errors occurs due to rounding, it is often difficult to obtain highly accurate results. In the stationary setting, the estimate (1.3) cannot be used as K varies with c.

1.3.1 Differentiation matrices

The RBF differentiation matrices can be obtained by differentiating (1.1) at nodes $\mathbf{x}_1, ..., \mathbf{x}_N$. For one-dimensional case it will be of the form

$$\frac{d}{dx}(s(x)) = \sum_{i=1}^{N} \lambda_i \frac{d}{dx}\phi(|x - x_i|).$$

The differentiation matrices obtained by RBF collocation, often have eigenvalues with positive real parts. Sarra [69] investigated numerical accuracy and stability for one and two-dimensional hyperbolic problems. It was observed that bounding condition number, can give stable eigenvalue for symmetric RBF collocation. Whereas the eigenvalue stability can be achieved for asymmetric collocation for structured center locations i.e., Chebyshev pseudospectral grid. The structured grid works well where the symmetric matrix becomes singular. In a very recent study Chen et al. [16] proposed a fully discretized (both in space and time) RBF methods for hyperbolic PDEs. To analyze the stability condition, space discretization is kept uniform.

Platte and Driscoll [62] have proved theoretically that RBF differentiation matrices using the conditionally positive definite functions (CPD), for the time-dependent problems are stable for periodic domains. For non-periodic domains highly dissipative time

stepping can be used for stable discretization. They further discussed that for Gaussian RBF, special node distribution is crucial for stable one-dimensional collocation. Fornberg et al. [30] observed the behaviour of differennt RBFs near boundaries in an interval and its extension to 2-D. He observed that boundary clustering of nodes using Chebyshev interpolation is a good alternative than using high order polynomials which causes Runge phenomenon.

For nonlinear higher order hyperbolic PDEs, the eigenvalue stability of MOL-RBF is difficult both in theoretical and numerical context. For the KdV equation we observed that the CFL condition is influenced by shape parameter and the nonlinearity present in the problem. Adapting the solution for time dependent hyperbolic PDEs, where the solution becomes from smooth to steep in time, require scattered and unstructured mesh around the steep gradients or discontinuities (shocks). The investigation of eigenvalue stability for such problems is still in its early stages.

1.4 RBF based adaptive methods

Interpolation by radial basis functions is a viable choice in several adaptive methods in a number of time-independent and time-dependent settings. For instance, Driscoll and Heryudono [23] have proposed the resudial subsampling method for interpolation, boundary-value, and initial-boundary-value problems with high degrees of localization. Their adaptive method is based on RBF and it has been observed that adapting the shape parameter is crucial for better results. Their method is fitting the unknown function on equally spaced nodes, then computing the interpolation error at points halfway between the nodes, they added those points where the error exceeds a refining thresholds to become centers and deleted the points where the error is less than a coarsening threshold. A combination of B-spline techniques using the scaled multiquadric RBFs has been presented in Bozzini et al. [11]. In their method scaled multiquadrics have proven to provide smoother interpolant on the liner B-splines. Schaback et al. [42] have used a greedy adaptive algorithm where the method is of linear convergence. Behren et al. [7]

have combined a semi-Lagrangian method with local RBF approximation, specifically, thin plate splines for linear transport equations. An extension of the method can be in seen in [5] for the non-linear transport equations i.e., Burgers equation and the Buckley-Leverett equation which describes a two-phase fluid flow in a porous medium.

Davydov [21] has proposed an adaptive meshless discretization based on generalized finite difference stencils. Generalized stencil were obtained using five-point formula and the weights have been computed using RBFs. To generate the finite difference stencils they preferred RBFs due to their robustness on highly irregular data over the local polynomial least square method which is the best known approach considered in literature for the generation of the stencils. Iske et al.[46] combined the adaptive particle methods with the scattered data reconstruction through polyharmonic splines, which plays a key role in their method. The numerical stability and approximation behavior of the polyharmonic splines have been discussed. Their method performed well on real world problems, for instance, tracer transport problem in the arctic stratosphere and a popular test case scenario from hydrocarbon reservoir modelling, termed the five-spot problem. Saara [68] modified a simple moving grid algorithm from finite differences to RBF methods. The references [23, 46, 74] have been the main motivation of our adaptive method.

1.5 Main achievements

The adaptive RBF method is proposed for the one-dimensional time dependent KdV equation. The method has further extended to some more challenging problems. To get an insight into the extension of the method in 2-D, we present results for 2-D interpolation problem. This thesis contains computational results for RBF simulations of standard PDE problems. All the numerical experiments are run in MATLAB on Windows 7 system, running at 3.10 Ghz with 8 GB memory.

Our first adaptive experiment is for the one-dimensional non-linear KdV equation briefly introduced in Section 1.2.1. The KdV equation has been solved in [74] using a

uniform grid, but as the variation occur only in a small part of the domain we adapt the solution in those regions. Our adaptive method solved the problem with fewer number of grid points than the non-adaptive method with the same accuracy proposed by [74]. It can be seen that the adaptive grid reflects the profile of the solution. The approximation has been obtained from multiquadrics due to its spectral convergence for sufficiently smooth functions. To obtain the residuals we performed local scattered data reconstruction using polyharmonic splines. Further extension has been made for the interactions of two solitons for the same model. The adaptive method performs well on the interaction and adapt the solution when the taller wave catches up, interacts with, and then passes the shorter wave.

The adaptive method also performs well on some difficult PDEs such as Burgers and the Alleh-Cahn equation. The efficiency of the adaptive grid and the utility of RBFs, can be seen in higher dimensions. We extended the adaptive method for the two-dimensional interpolation problem on the Franke's function. In the future we aims in extension for the two-dimensional PDEs though not in this thesis.

1.5.1 A brief view of the adaptive method proposed

For an insight of the adaptive method we will briefly give an overview of the one-dimensional interpolation problem. We will interpolate the known function $f(x)$ over the interval [a,b] with the equally spaced initial discretization i.e., $x_i = a + (b-a)ih$ where $h = 1/(N+1)$ and $i = 0, 1, ..., N$.

The approximate solution i.e., $u(\mathbf{x})$, is obtained using global multiquadrics. On this approximation, we reconstruct the solution using polyharmonic splines which will become the natural spline in 1-D. This spline reconstruction will use approximate solutions obtained by MQ and will interpolate the value of x_i on a set of nearest neighbours of the point x_i i.e., $\mathcal{N}_{x_i}$. In our examples we reconstruct using 7 nearest neighbours, which will give us the local influence around every point.

The next step is to obtain the residual $\eta(x) = |u(x_i) - s_{\mathcal{N}_{x_i}}(x_i)|$ for every point in

$\mathbf{X}$, based on the thresholds θ_{ref} and θ_{crs} . If the difference $n(x) > \theta_{ref}$, the solution will lie in the region of high variation and we will add more points there. For the 1-D refinement if the residual at a check point exceeds θ_{ref}, we will add point to left and to the right of that check point. The small value of $n(x) < \theta_{crs}$, indicates that the solution is smooth enough, and hence we will remove such points. The coarsening is done by sorting the residuals and deleting a specific percentage of the sorted best error. This percentage depends on the problem we are dealing with.

In our numerical experiments we are using center dependent shape parameters to prevent the growth of the condition number $\kappa(A)$ which is due to the fact that, the RBF method put more nodes in the localized region which remarkably decreases the separation distance. Varying the shape parameters with the centers results in, improving accuracy and numerical conditioning [31]. We will adapt the shape parameters as per problem. The rule is that a formula for c_j could depend on spacing of centers x_j to its nearest neighbours.

The adaptive algorithm for one-dimensional (1-D) interpolation problem starts with the node set $\mathbf{X}$, refining θ_{ref} and coarsening θ_{crs} thresholds. The idea is to push all the residuals below θ_{ref}. The algorithm starts with equal distribution of nodes in the domain. In step 2.(b) of Algorithm 1, the adaptive selection of nodes relies on the local reconstruction using polyharmonic splines where the natural splines are used in 1-D. In step 3, residuals will point out the regions of high variation.

ALGORITHM 1. Adaptive RBF for 1-D interpolation.

INPUT: $\mathbf{X}$, θ_{ref} and θ_{crs}.

while {

1. Compute the MQ-interpolant u.
2. For each $\mathbf{x} \in \mathbf{X}$ {
   - **(a)** Determine a set $N_{\mathbf{x}} \subset \mathbf{X}$ of neighbours of $\mathbf{x}$.
   - **(b)** Compute s, the natural splines interpolant satisfying $s|_N = u|_N$ for each $\mathbf{x}$ but not at $\mathbf{x}$.
   - **(c)** Using s the local interpolant around $\mathbf{x}$, interpolate at $\mathbf{x}$.

   }
3. Compute the residuals $|u(\mathbf{x}_i) - s_{N_{\mathbf{x}_i}}(\mathbf{x}_i)|$ for each $\mathbf{x_i}$.
4. Adaptively refine when $|u(\mathbf{x}_i) - s_{N_{\mathbf{x}_i}}(\mathbf{x}_i)| > \theta_{ref}$.
5. Coarsen when $|u(\mathbf{x}_i) - s_{N_{\mathbf{x}_i}}(\mathbf{x}_i)| < \theta_{crs}$
6. Adapt the shape parameters.

UNTIL Every residual is less than θ_{ref}.

}

OUTPUT: A new set of adaptively selected set of centers.

To get a better understanding let us implement the algorithm on 1-D interpolation problems.

Example: Let us consider the so called Runge function $f(x) = \frac{1}{1+25x^2}$ over the interval $[-1, 1]$ with the equally spaced initial discretization $\mathbf{X} = \{x_1, ..., x_N\}$. We started with 20 equally spaced points and considered $\theta_{ref} = 10^{-4}$ and $\theta_{crs} = 10^{-6}$. The adaptive interpolation clusters nodes around the steep gradient. It is evident from Figure (1.2) that our method converges in four iterations with the maximum error $= 5.2 \times 10^{-7}$ with

adaptively adjusted points.

Another example can be given for $\tanh(60\mathbf{x} - 0.1)$ it is evident from Table (1.1) and Fig (1.3) that the adaptive method converges in 7 iterations with a total number of 66 points. It is evident from Fig 1.3 that the nodes are clustering around the steep gradient. We can see that the condition number given in Table (1.1) increases rapidly and then becomes stable.

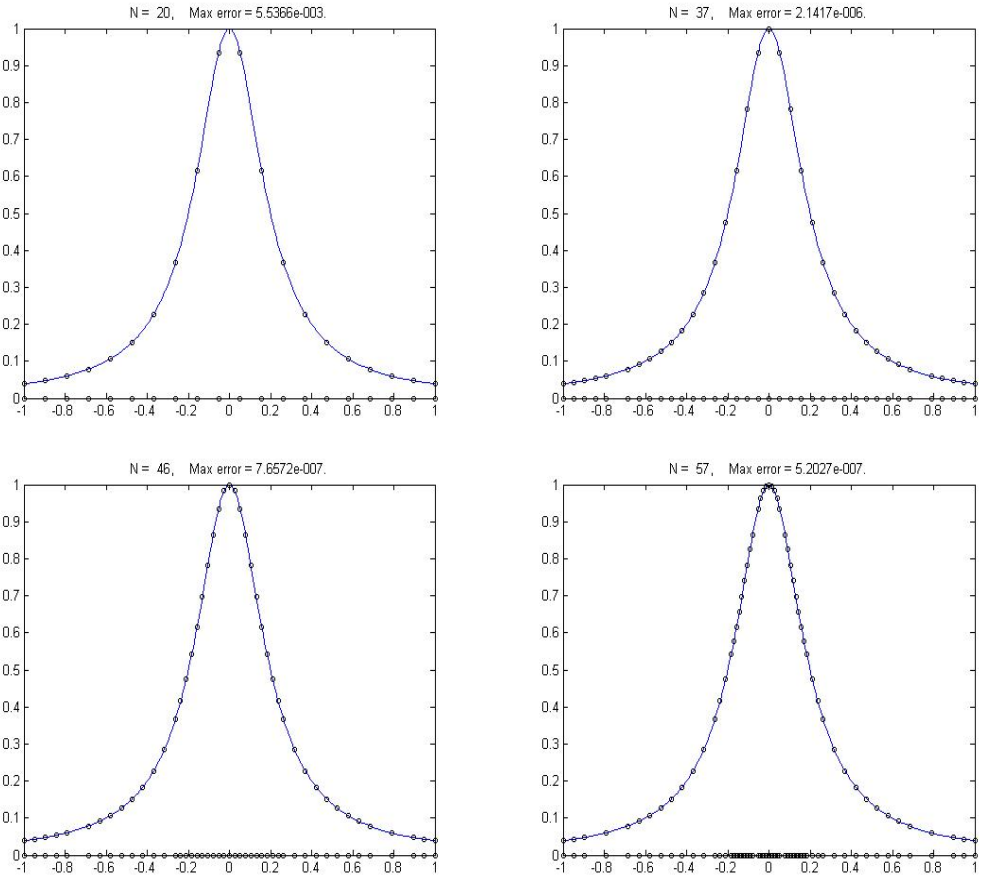


Figure 1.2: The adaptive interpolation of the Runge function $f(x) = \frac{1}{1+25x^2}$

Table 1.1: Adaptive interpolation for $\tanh(60x - 0.1)$.

Iteration	N	$\kappa(A)$	Max Error
1	20	3.7E+03	4.3E-01
2	22	3.1E+04	4.0E-01
3	25	1.0E+05	1.2E-01
4	31	3.7E+05	8.5E-03
5	36	1.1E+06	7.6E-04
6	54	3.2E+06	9.4E-05
7	66	5.4E+06	7.4E-05

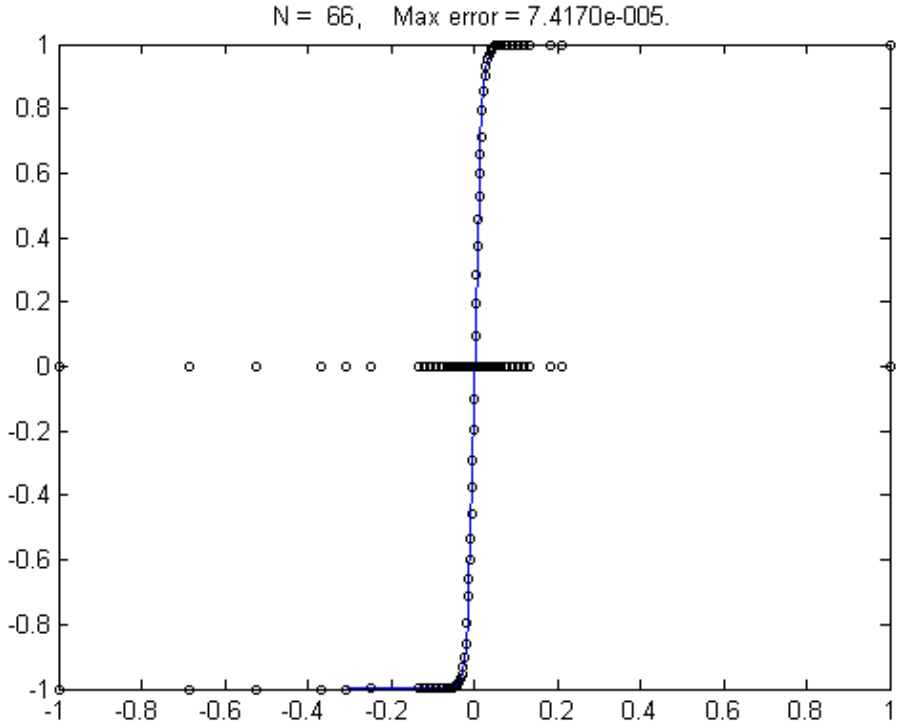


Figure 1.3: The adaptive interpolation for $\tanh(60x - 0.1)$.

1.6 Outlines

This thesis is organized as follows,

In **Chapter 2**, we will give an overview of the radial basis function interpolation. The MQ-RBF has been the primary tool for the interpolation in our experiments. In **Chapter 3**, we propose radial basis function in a method of lines fashion. The method performs well and is in a good comparison with other methods proposed for the gener-

alized Burgers-Huxley equation. The so called method of lines is also discussed both for 1-D generalized Burgers-Huxley equation and 2-D Allen-Cahn equation. In **Chapter 4** we discussed our motivational model the KdV equation. We will discuss about the PDE and challenges for numerical methods for this problem, in detail. In **Chapter 5**, we propose the adaptive RBF method for KdV equation. In our method, approximations are obtained by multiquadrics due to their exponential rate of convergence for sufficiently smooth functions. The residuals have been obtained by local reconstruction through polyharmonic splines which coincides with the natural splines in the one-dimensional case. The method essentially solve the KdV equation with lease number of grid points and insert more points in the region of high activity and coarsen otherwise. We observed that our method compared to the non-adaptive RBF-MOL [74] gives promising results for both, single soliton and interaction of two solitons solutions for the KdV equation. The numerical experiments are discuss for the 1-D time-dependent PDEs, the KdV equation, Allen-Cahn equation and the famous Burger's equation. The 2-D adaptive interpolation is presented for the Franke's function. We discussed the algorithms for the construction of the Voronoi diagram in the refinement context. In **Chapter 6**, We conclude the results obtained by our method and discuss the future extensions of the method.

Chapter 2

Radial Basis Functions

2.1 Scattered data interpolation problem

Multivariate scattered data interpolation requires the recovery of an unknown function f from a finite set of discrete data. The data consists of measurements (called the *data values*) obtained at certain pairwise distinct locations (called the *data sites*). Our interest is not just to approximate the data values at the data sites but to deduce the information at locations different from those at which the measurements are given. In other words, we wish to find a function (a good fit) "$\boldsymbol{s}$" which matches the given measurements at the corresponding locations i.e., the function "$\boldsymbol{s}$" passes through the data. This approach is known as interpolation. If the location on which the measurements are taken do not lie on a uniform or a regular grid then the process is called scattered data interpolation.

Definition 2.1 *(**Multivariate scattered data interpolation problem**) Let* $\mathbf{X} \subseteq \mathbb{R}^d$*, given the data* $(\mathbf{x}_i, f(\mathbf{x}_i))$*;* $i = 1, ..., N$ *where* $f : \mathbb{R}^d \to \mathbb{R}$*, the multivariate scattered data interpolation problem is to find a smooth function* $s : \mathbb{R}^d \to \mathbb{R}$ *such that* $s(\mathbf{x}_i) = f(\mathbf{x}_i)$*;* $i = 1, ..., N$ *where* s *is called the interpolant to the data.*

Here $\mathbf{x}_i$ are the measurements locations, and $f(\mathbf{x}_i)$ are the corresponding measurements. We will assume that these values are obtained by sampling a data function f at

data locations.

A convenient approach for solving the scattered data problem is to make the assumption that the interpolant s(.) is a linear combination of certain basis functions, $\phi_i(\mathbf{x}); i = 1, ..., N$, i.e.,

$$s(\mathbf{x}) = \sum_{i=1}^{N} \lambda_k \phi_k(\mathbf{x}) \quad \text{where } \mathbf{x} \in \mathbb{R}^d. \tag{2.1}$$

Solving the interpolation problem under this assumption

$$s(\mathbf{x}_i) = f(\mathbf{x}_i),\ i = 1, ..., N, \tag{2.2}$$

leads to the system of linear equations of the form

$$\mathbf{A}\lambda = f, \tag{2.3}$$

where the entries of the interpolation matrix $\mathbf{A}$ are given by

$$\begin{aligned} \mathbf{A}_{j,k} &= \phi_k(\mathbf{x}_j), \text{ where } j, k = 1, ..., N, \\ \text{and } \lambda &= [\lambda_1, ..., \lambda_N]^T, f = [f_1, ..., f_N]^T. \end{aligned}$$

The unique solution of the problem exists, whenever matrix $\mathbf{A}$ is non singular. Non-singularity of matrix $\mathbf{A}$ under some mild restrictions i.e., constant shape parameters and sometimes adding a low order polynomial is guaranteed in [59].

2.2 Introductory concepts

Definition 2.2 *(**Vector-norms**) Let $v \in \mathbb{R}^d$ the vector norms are defined as*

- *(ℓ_p-norm) The p-norm, $1 \leq p < \infty$ is defined as $\| v \|_p = (\sum_{i=1}^{n} |v_i|^p)^{1/p}$,*

- $(\ell_1 - norm)\ \| v \|_1 = \sum_{i=1}^{n} |v_i|$,
- $(\ell_2 - norm)\ \| v \|_2 = (\sum_{i=1}^{n} |v_i|^2)^{1/2}$,
- $(\ell_\infty - norm) = \| v \|_\infty \max_{1 \le i \le n} |v_i|$.

Definition 2.3 ***(Matrix-norms)*** *Let $A : \mathbb{R}^n \to \mathbb{R}^m$ and $v \in \mathbb{R}^n$. The norms of the matrix A induced by the vector norms are defined as*

- *(ℓ_p-norm)* $\| A \|_p = \max\limits_{v \in R^n \setminus 0} \frac{\|Av\|_p}{\|v\|_p}$,
- $(\ell_1 - norm)\ \| A \|_1 = \max_{1 \le j \le n} \sum_{i=1}^{n} |A_{ij}|$,
- $(\ell_2 - norm)\ \| A \|_2 = \sqrt{\rho(A^T A)}$ *where ρ is the spectral radius,*
- $(\ell_\infty - norm)\ \| A \|_\infty = \max_{1 \le i \le n} \sum_{j=1}^{n} |A_{ij}|$.

Definition 2.4 (Multi-index notation) *Let $\mathbb{N}_0$ denote the set of non-negative integers. A d-dimensional multi-index is a d-tuple*

$$\alpha = (\alpha_1, \ldots, \alpha_d) \in \mathbb{N}_0^d$$

For $\mathbf{x} = (x_1, \ldots, x_d) \in \mathbb{R}^d$, we define

$$|\alpha| = \alpha_1 + \cdots + \alpha_d,$$

$$\mathbf{x}^\alpha = x_1^{\alpha_1} \cdot x_2^{\alpha_2} \cdots x_d^{\alpha_d},$$

and

$$D^\alpha = \left(\frac{\partial}{\partial x_1}\right)^{\alpha_1} \cdots \left(\frac{\partial}{\partial x_d}\right)^{\alpha_d} = \frac{\partial^{|\alpha|}}{\partial x_1^{\alpha_1} \ldots \partial x_d^{\alpha_d}}.$$

It is effective to use the multi-index notation. For instance, the following example shows it is less laborious to use this notation.

Example: Let us suppose $d = 3$, and $\alpha = (\alpha_1, \alpha_2, \alpha_3)$, $\alpha_j \in \mathbb{N}$ for $j = 1, 2, 3$. For u be a function of three variables, x_1, x_2, x_3,

$$\sum_{|\alpha|=3} D^\alpha u = \frac{\partial^3 u}{\partial x_1^3} + \frac{\partial^3 u}{\partial x_1^2 \partial x_2} + \frac{\partial^3 u}{\partial x_1^2 \partial x_3} + \frac{\partial^3 u}{\partial x_1 \partial x_2^2} + \frac{\partial^3 u}{\partial x_1 \partial x_3^2} + \frac{\partial^3 u}{\partial x_2^3} + \frac{\partial^3 u}{\partial x_1 \partial x_2 \partial x_3} + \frac{\partial^3 u}{\partial x_2^2 \partial x_3} + \frac{\partial^3 u}{\partial x_2 \partial x_3^2} + \frac{\partial^3 u}{\partial x_3^3}.$$

Let us denote π_m^d , by space of d-variate polynomials of degree not exceeding than m.

Definition 2.5 (Condition of unisolvancy) *The data sites $\mathbf{X} \subseteq \mathbb{R}^d$ with $N \geq M = \dim \pi_m^d$ are called π_m^d -unisolvent if the zero polynomial is the only polynomial from the space π_m^d that vanishes on all of them.*

The definition comes from polynomial interpolation, in which case it guarantees a unique solution for interpolation to given data at a subset of points $\mathbf{X} \subset \mathbb{R}^d$ by a polynomial of degree m.

Example: Three collinear points in $\mathbb{R}^2$ are not 1-unisolvent since a linear interpolant, i.e., a plane trough three arbitrary heights at these 3 collinear points is not uniquely determined. On the other hand, if a set of points in $\mathbb{R}^2$ contains 3 non-collinear points, then it is 1-unisolvent.

Definition 2.6 *(**Positive definite matrix**) A real symmetric matrix A is called positive semi-definite if its associated quadratic form is non-negative, i.e.,*

$$\sum_{j=1}^N \sum_{k=1}^N \lambda_j \lambda_k A_{jk} \geq 0,$$

for $\lambda = [\lambda_1, ..., \lambda_N]^T$.

If the only vector λ that turns the above quadratic form into an equality is the zero vector, then A is called positive definite.

Definition 2.7 *(**Positive definite function**) A real valued continuous function $\Phi : \mathbb{R}^d \to \mathbb{R}$ is positive definite on $\mathbb{R}^d$ if and only if it is even and*

$$\sum_{j=1}^{N}\sum_{k=1}^{N}\lambda_j\lambda_k\phi(\mathbf{x}_j-\mathbf{x}_k)\geq 0,$$

for any N pairwise different points $\{\mathbf{x}_1,\ldots,\mathbf{x}_N\}\in\mathbf{R}^d$*, and* $\lambda=[\lambda_1,...,\lambda_N]^T$*.*

The function ϕ *is strictly positive definite on* $\mathbb{R}^d$ *if the only vector* λ *that turns the above into the equality is the zero vector.*

Positive definite functions have a vital role in approximation theory and statistics, the property is crucial for interpolation. A synonym for the positive definite function is function of positive type.

Definition 2.8 *(**Conditionally positive functions**) A continuous real function* $\Phi:\mathbb{R}^d\to\mathbb{R}$ *is said to be conditionally positive definite of order m on* $\mathbb{R}^d$ *if and only if*

$$\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j\phi(\mathbf{x}_i-\mathbf{x}_j)>0,$$

holds for all possible pairs (λ,X) of choices $\lambda=[\lambda_1,...,\lambda_N]$ and $X=\{\mathbf{x}_1,...,\mathbf{x}_N\}\subseteq\mathbb{R}^d$ satisfying the vanishing moment conditions

$$\sum_{j=1}^{N}\lambda_j P(\mathbf{x}_j)=0,$$

for all $P\in\pi_{m-1}^d$.

It is not possible to list all of the analogues and generalizations of positive definite functions but some of the properties of the PD and CPD functions are listed below (details can be found in [76]).

- If $\phi_1,...,\phi_N$ are positive-semi definite and $\lambda_j\geq 0$, then $\Phi:=\sum_{j=1}^{N}\lambda_j\phi_j$ is also positive-semi definite. For a positive definite ϕ_j if the corresponding λ_j is also positive then Φ is also positive definite.

- A conditionally positive definite function of order $m=0$ on $\mathbb{R}^d$ is positive definite on $\mathbb{R}^d$.

- If each ϕ_j is positive definite, then the limit $\Phi=\lim_{j\to\infty}\phi_j$.

- The product of two positive definite functions is positive definite.

- If $\phi_1, ..., \phi_d$ are positive definite and integrable functions on $\mathbb{R}$. Then the tensor product function

$$\Phi(\mathbf{x}) = \phi_1(x_1)...\phi(x_d),\ \mathbf{x} = (x_1, ..., x_d)^T \in \mathbb{R}^d$$

  is also positive definite.

- A positive definite function is always conditionally positive define of any order.

Definition 2.9 *(**Fill distance**) The fill distance corresponding to the data sites* $\mathbf{X}$ *in* Ω *is defined as:*

$$h_{\mathbf{X},\Omega} = \sup_{\mathbf{x}\in\Omega} \min_{\mathbf{x}_j\in\mathbf{X}} \| \mathbf{x} - \mathbf{x}_j \|_2.$$

This is sometimes called the covering radius. The geometrical interpretation of fill distance is the largest possible empty ball amongst the data sites. It is a measure of the data distribution and can indicate how well the domain Ω is filled with the data in the set $\mathbf{X}$.

Definition 2.10 *(**Separation distance**) The separation distance is defined as:*

$$q_{\mathbf{X}} = \tfrac{1}{2}\min_{i\neq j} \| \mathbf{x}_i - \mathbf{x}_j \|_2.$$

This is sometime called as packing radius. The geometrical interpretation of the separation distance is that no two balls of radius "$q_{\mathbf{X}}$" centered at each RBF centers will overlap.

Definition 2.11 *(**Condition number**) The condition number of a matrix* $\boldsymbol{A}$ *with respect to any matrix norm* $\| \cdot \|$ *is defined as*

$$\kappa(\boldsymbol{A}) = \| \boldsymbol{A} \| \| \boldsymbol{A}^{-1} \|.$$

The condition number of a matrix depends on the separation distance (see Definition 2.10), and provides information on the numerical stability of the interpolation process. To do so we have to investigate both the maximum and the minimum eigenvalues. A condition number is used to quantify the sensitivity to perturbations of a linear system, such as Equation (2.3), and to estimate the accuracy of a computed solution. Using the 2-norm, the matrix condition number is

$$\kappa(\mathbf{A}) = \| \mathbf{A} \|_2 \| \mathbf{A}^{-1} \|_2 = \frac{\sigma_{\max}}{\sigma_{\min}}.$$

where σ are the singular values of $\mathbf{A}$, in particular, when A is positive definite this ratio becomes $\frac{\lambda_{\max}}{\lambda_{\min}}$.

A well conditioned matrix will have a small condition number $\kappa(\mathbf{A}) \approx 1$, i.e., a system of equations is considered to be well-conditioned if a small change in the coefficient matrix or a small change in the right hand side results in a small change in the solution vector. While an ill-conditioned matrix will have a large condition number and a small change in the coefficient matrix or a small change in the right hand side results in a large change in the solution vector.

2.3 Radial basis functions

Definition 2.12 *A function $\Phi : \mathbb{R}^d \to \mathbb{R}$ is said to be radial if there exist a univariate function $\phi : [0, \infty] \to \mathbb{R}$ such that $\Phi(\mathbf{x} - \mathbf{x}_j) = \phi(r)$, where $r = \| \mathbf{x} - \mathbf{x}_j \|$ and $\| \cdot \|$ is some norm on $\mathbb{R}^d$ (usually the Euclidean norm).*

The definition can be explained as that for a finite set of distinct points $\mathbb{R}^d$ called the centers, Φ at any point $\mathbf{x} \in \mathbb{R}^d$ at certain distance from the centers is constant. This shows that Φ is radially symmetric about the centers. The function ϕ can be called as *basic* function whereas Φ can be called *basis* function. The reason is that one single basic function generates all of the basis function which we have used in the expansion

(2.1). The 2-norm invariance under all the Euclidean transformation has made the radial function interpolants quite desirable in applications. However what makes radial functions most useful for applications is the fact that the interpolation problem becomes insensitive to the dimension d of the space in which the data sites lie. Instead of having to deal with the a multivariate function Φ (whose complexity will increase with increasing space dimension d) we can work with the same univariate function ϕ for all choices of d.

RBF interpolation methods use linear combination of translates of one function $\phi(r)$ of a single real variable. Given a set of centers $\mathbf{X}$ in $\mathbb{R}^d$, the RBF interpolant takes the form

$$s(x) = \sum_{j=1}^{N} \lambda_j \phi(\| \mathbf{x} - \mathbf{x}_j \|_2). \tag{2.4}$$

The coefficients, λ, are chosen by enforcing the interpolation condition

$$s(\mathbf{x}_j) = f(\mathbf{x}_j),$$

at a set of nodes that typically coincides with the centers. Enforcing the interpolation condition at N centers results in a $N \times N$ linear system.

$$\mathbf{A}\,\lambda = f,$$

$$\begin{bmatrix} \phi(\| x_1 - x_1 \|_2) & \phi(\| x_1 - x_2 \|_2) & \dots & \phi(\| x_1 - x_N \|_2) \\ \phi(\| x_2 - x_1 \|_2) & \phi(\| x_2 - x_2 \|_2) & \dots & \phi(\| x_2 - x_N \|_2) \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ \phi(\| x_N - x_1 \|_2) & \phi(\| x_N - x_2 \|_2) & \dots & \phi(\| x_N - x_N \|_2) \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ . \\ . \\ . \\ \lambda_N \end{bmatrix} = \begin{bmatrix} f(x_1) \\ f(x_2) \\ . \\ . \\ . \\ f(x_N) \end{bmatrix}.$$

The above linear system has to be solved for the coefficients λ. The matrix $\mathbf{A}$ is called the interpolation matrix or the system matrix and consist of the functions serving as the basis of the approximation space. To evaluate the interpolant at points $\mathbf{y}_i$ for $i = 1, 2, ..., M$ using Equation (2.4), $M \times N$ evaluation matrix H is formed with entries

$$h_{ij} = \phi(\| \mathbf{y}_i - \mathbf{x}_j \|_2),\ i = 1, ...M \text{ and } j = 1, ...N.$$

Then the interpolant is evaluated at the M points by the matrix multiplication

$$H\,\lambda = f.$$

2.4 RBF interpolation with polynomial reproduction

Sometimes the assumption on the form Equation (2.1), for solution to the scattered data interpolation problem (see definition 2.1) is extended by adding certain polynomials to the expansion, i.e., $s(\mathbf{x})$ is now assumed to be of the form

$$s(\mathbf{x}) = \sum_{j=1}^{N} \lambda_i \phi(\parallel \mathbf{x} - \mathbf{x}_j \parallel) + p(\mathbf{x}), \quad \text{where } \mathbf{x} \in \mathbb{R}^d \text{ and } p \in \pi_{m-1}^d. \tag{2.5}$$

Equation (2.5) can be written as

$$s(\mathbf{x}) = \sum_{j=1}^{N} \lambda_j \phi(||\mathbf{x} - \mathbf{x}_j||) + \sum_{l=1}^{M} d_l p_l(\mathbf{x}), \tag{2.6}$$

where the polynomials $p_1, ..., p_M$ form a basis for the $M = \begin{pmatrix} d+m-1 \\ m-1 \end{pmatrix}$ -dimensional linear space π_{m-1}^d of polynomials of total degree less than or equal to $m-1$ in d variables.

Enforcing the interpolation conditions $s(\mathbf{x}_i) = f(\mathbf{x}_i), i = 1, ..., N$, leads to a system of N linear equations in $N+M$ unknowns λ_j and d_l, one usually adds the M additional conditions to ensure a unique solution. Let us impose the interpolation conditions on our interpolant s to get

$$\sum_{j=1}^{N} \lambda_j \phi(||\mathbf{x}_k - \mathbf{x}_j||) + \sum_{l=1}^{M} d_l p_l(\mathbf{x}_k) = f(\mathbf{x}_k), \quad \text{for } 1 \le k \le N. \tag{2.7}$$

Equation (2.7) is a linear system of N equations in $N+M$ unknown variables in coefficient vector $\lambda = [\lambda_1, ..., \lambda_N]^T$ of the major part and $\mathbf{d} = [d_1, ..., d_M]^T$ of the polynomial

part of the interpolant. So we have M additional unknowns and the linear system is under determined. To eliminate these additional degrees of freedom, we add the following M constraint conditions:

$$\sum_{k=1}^{N} \lambda_k p_l(\mathbf{x}_k) = \mathbf{0}, \quad for\ 1 \leq l \leq M, \tag{2.8}$$

where $p \in \pi_{m-1}^d$ and m is the order of the basis function ϕ. This can be explained in an other way, that the rank of p was the dimension of the appropriate polynomial space. This requires that the underlying data set have a $\pi_{m-1}(\mathbb{R})$ unisolvent subset. If the data set fails to have such a subset then some nonzero polynomial of degree $m-1$ or less will vanish on the data which will make the polynomial reproduction impossible i.e., rank(p) will fall below the dimension of $\pi_{m-1}(\mathbb{R}^d)$.

Here the radial basis functions $\phi(||x - \mathbf{x}_j||)$ are translates of a conditionally positive definite radial function $\phi : [0, \infty) \to \mathbb{R}$ of order m.

We have assumed the interpolant "s" to be a linear combination of the translates $\phi(||\mathbf{x} - \mathbf{x}_j||)$ of a conditionally positive definite radial function ϕ at centers $\mathbf{x}_j$ plus a polynomial procession term from the space π_{m-1}^d. The use of the additional polynomial term is somewhat arbitrary in the sense that other set of M linearly independent functions could be used. However, it is clear that addition of a polynomial of total degree at most $m-1$, guarantees the well posedness of system given in Equation (2.5). In other words if the data come from a polynomial of total degree at the most $m-1$, then they are fitted exactly by our interpolant.

The classical choice for radial basis function ϕ along with their order m are shown in Table(2.1).

2.5 Compactly supported radial basis functions

The compactly supported radial basis functions (CSRBFs) were recently developed in [83, 85, 82]. The order of the CSRBFs can be taken as $m = 0$ in this case we do not need

Table 2.1: Radial Basis Functions.

RadialBasisFunction	$\varphi(r) =$	*Parameters*	*Order m*
Polyharmonic Splines	r^{ν}	$\nu > 0, \nu \notin 2\mathbb{N}$	$m \geq [\frac{\nu}{2}]$
Thin Plate Splines (TPS)	$r^{2k}\log(r)$	$k \in \mathbb{N}$	$m \geq k+1$
Gaussians	$e^{-(cr)^2}$	$c > 0$	$m \geq 0$
Multiquadrics(MQs)	$(1+c^2r^2)^{\frac{\nu}{2}}$	$\nu > 0,\ \nu \notin 2\mathbb{N},\ c > 0$	$m \geq [\frac{\nu}{2}]$
Inverse Multiquadrics(IMQs)	$(1+c^2r^2)^{\frac{\nu}{2}}$	$v < 0,\ c > 0$	$m \geq 0$

to append the polynomial part as in Equation 2.5. The compactly supported RBFs can be used for a fixed dimension d where as the RBFs listed in Table 2.1 can be used in arbitrary d. The Wendland's compactly supported RBFs are of the form

$$\phi_{d,k}(r) = \begin{cases} \mathbf{P}_{d,k} & r \in [0,1] \\ 0 & r > 1 \end{cases}$$

$\phi_{d,k}(r) : [0,\infty] \to R$ are the positive definite radial functions with the compact support normalized to the unit interval [0,1]. Where $\mathbf{P}_{d,k}$ is the univariate cut-off polynomial $(.)_+ : R \to [0,\infty)$ defined as

$$(x)_+ = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases}$$

The degree of $\mathbf{P}_{d,k}$ is $\lfloor d/2 \rfloor + 3k + 1$ where the symbol $\lfloor \rfloor$ denotes the largest integer less than or equal to x.

The use of locally supported basis function is of general importance in numerical analysis. Compactly supported basis can be efficient for fast evaluation of the interpolant and lead to banded interpolation matrices. The compactly supported radial basis functions are listed in Table 2.2.

Table 2.2: Wendland's Compactly Supported RBFs

Dimension d	Radial Basis Function	C^{2k}
$d = 1$	$\phi_{1,0} = (1-r)_+$	C^0
	$\phi_{1,1} = (1-r)_+^3(3r+1)$	C^2
	$\phi_{1,2} = (1-r)_+^5(8r^2+5r+1)$	C^4
$d \leq 3$	$\phi_{3,0} = (1-r)_+^2$	C^0
	$\phi_{3,1} = (1-r)_+^4(4r+1)$	C^2
	$\phi_{3,2} = (1-r)_+^6(35r^2+18r+3)$	C^4
	$\phi_{3,3} = (1-r)_+^8(32r^3+25r^2+8r+1)$	C^6
$d \leq 5$	$\phi_{5,0} = (1-r)_+^3$	C^0
	$\phi_{5,1} = (1-r)_+^5(5r+1)$	C^2
	$\phi_{5,2} = (1-r)_+^7(16r^2+7r+1)$	C^4

2.6 Well-posedness of the radial basis interpolation problem

Let us discuss the existence of a unique solution to the above interpolation problem. We discuss it separately for $m = 0$ and $m > 0$.

Suppose $m = 0$, i.e., the basic function ϕ is of order 0. In other words ϕ is positive

definite; e.g. Gaussian and some of the multiquadrics in Table 2.1. In this case we do not need to add the polynomial term, and the interpolant s as given in Equation (2.4) can be used:

$$s(\mathbf{x}) = \sum_{j=1}^{N} \lambda_j \phi(\| \mathbf{x} - \mathbf{x}_j \|). \tag{2.9}$$

In this case we do not need the additional conditions Equation (2.8). The interpolation conditions $s(\mathbf{x}_i) = f(\mathbf{x}_i)$ gives the following simpler system,

$$A_{\phi,X}\lambda = f(\mathbf{x}_i), \tag{2.10}$$

where $\mathbf{A}_{\phi,X}$, λ and $f(x_i)$ have the same meaning. Since ϕ is positive definite, matrix $\mathbf{A}_{\phi,X}$ is guaranteed to be positive definite. In this case Equation (2.10) has a unique solution. So the RBFs interpolation problem is well-posed for the case $m = 0$.

Now we turn to discuss the problem for $m > 0$. In other words ϕ is conditionally positive definite of order $m \geq 1$. In this case our interpolant s given by Equation (2.6) must contain a nontrivial polynomial term, and we must add the M additional conditions using Equation (2.8) to eliminate the M additional degrees of freedom. So we have to search for the solution of the augmented problem given by Equation (2.7) using (2.8).

Before discussing the solution of Equation (2.7), we consider its corresponding homogenous system

$$\begin{pmatrix} \mathbf{A}_{\phi,\mathbf{X}} & p_{\mathbf{X}} \\ p_{\mathbf{X}}^T & \mathbf{O} \end{pmatrix} \begin{pmatrix} \lambda \\ \mathbf{d} \end{pmatrix} = \mathbf{0},$$

where

$$\begin{aligned} \mathbf{A}_{\phi,\mathbf{X}} &= [\phi(\| \mathbf{x}_k - \mathbf{x}_j \|)]_{1\leq j,k\leq N} \in \mathbb{R}^{N\times N} \\ p_{\mathbf{X}} &= [p_l(\mathbf{x})]_{1\leq k\leq N, 1\leq l\leq M} \in \mathbb{R}^{N\times M} \\ \mathbf{O} &= \text{Null matrix} \in \mathbb{R}^{M\times M}. \end{aligned}$$

which is equivalent to the following equations

$$\mathbf{A}_{\phi,\mathbf{X}} \cdot \lambda + p_{\mathbf{X}} \cdot \mathbf{d} = \mathbf{0}, \tag{2.11}$$

$$p_{\mathbf{X}}^T \cdot \lambda = \mathbf{0}. \tag{2.12}$$

Multiplying Equation (2.11) with λ^T, we get

$$\lambda^T \mathbf{A}_{\phi,\mathbf{X}} \lambda + (p_{\mathbf{X}}^T \cdot \lambda)^T \cdot d = \mathbf{0},$$

using Equation (2.12) gives

$$\lambda^T \mathbf{A}_{\phi,\mathbf{X}} \lambda = \mathbf{0}. \tag{2.13}$$

But we know that ϕ is conditionally positive definite of order $m \geq 1$ and it is clear from Definition 2.8 that Equation (2.13) together with condition given by Equation (2.12) implies $\lambda \equiv \mathbf{0}$ and so Equation (2.11) becomes

$$p_{\mathbf{X}}^T \cdot \mathbf{d} = \mathbf{0},$$

which implies

$$\mathbf{d} = \mathbf{0}, \text{ provided that } P_{\mathbf{X}} \text{ is injective.}$$

So the corresponding homogenous system has unique solution $[\lambda, \mathbf{d}]^T = \mathbf{0}$.
This guarantees the non singularity of the matrix

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{\phi,\mathbf{X}} & p_{\mathbf{X}} \\ p_{\mathbf{X}}^T & \mathbf{O} \end{pmatrix},$$

which implies the well-posedness of system given by Equation (2.6). p_X will be injective if and only if the zero polynomial is the only polynomial from the space π_{m-1}^d that vanishes on all of the data sites $X = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$. Such a point set is called π_{m-1}^d-unisolvent (see Definition 2.5). Hence injectivity of the matrix $p_{\mathbf{X}}$ is guaranteed by imposing the mild

condition of unisolvency of the data points which proves the non-singularity of the matrix **A**.

Chapter 3

Meshless method of lines using radial basis function method

A ***mesh*** can be described as a net created by connecting nodes in a predefined manner. The synonyms can be grid, cells or elements. A ***meshfree*** method does not require a predefined mesh and no mesh generation is required throughout the process of solving the problem. The traditional methods such as finite element, finite volume, or finite difference requires the generation of a mesh for the underlying problem i.e., the triangulation of the region of interest. For problems involving steep gradients, sharp corners, or moving boundaries requires more flexibility in some parts of the domain of interest. Meshfree methods can be preferred over grid based method for such problems. They can be taken as a response to the limitations of the mesh-based methods.

Meshfree methods are the topic of recent research in many areas of computational science and approximation theory. The applications can be found in artificial intelligence, computer graphics, image processing, optimization and the numerical solution of (partial) differential equations. The original motivation for two of the most common basic meshfree approximation methods (radial basis functions and moving least squares) came from applications in geodesy, geophysics, mapping and metrology Later, the applications were found in many areas such as numerical solution of PDEs, arti-

ficial intelligence, learning theory, neural networks, signal processing, sampling theory, statistics, finance and optimization.

Radial basis function methods have become the primary tool for interpolating multi-dimensional scattered data. The method has shown promising results for solving PDEs in complex and irregular domains. RBF methods are not tied to a grid and in turn belong to a category of methods called ***meshless methods***. RBF methods succeed in very general settings by composing a univariate function with the Euclidean norm which turns a multi dimensional problem into one that is virtually one dimensional.

RBF methods enjoys numerous advantages for example, simple implementation,does not require connectivity of nodes, insensitive of the dimension and spectral convergence can be achieved. The spectral convergence depends on how we choose the basis function. But in addition the the advantages mentioned above, RBF methods have some drawbacks for example, finding optimal value of the shape parameter, ill-conditioning, solving a full matrix (except for compactly supported RBFs) and choosing nodes.

3.1 Meshless method of lines using radial basis function

RBF methods are praised for their simplicity and ease of implementation in multivariate scattered data approximation [12], and they are becoming a popular choice for the numerical solution of the PDEs. In the last decade RBF meshless method has been considered as the novel and prospective numerical method for solving PDEs [32, 74, 23, 51, 50]. The method was first introduced by Kansa [51, 50] for the numerical solution of elliptic and parabolic PDEs. The radial basis function approach is used to solve various problems in application including transport schemes on a sphere [29], Allen-Cahn equation [23], and the MOL-RBF scheme for time dependent KdV equation [74].

3.1.1 Elements of the method of lines (MOL)

The idea of the MOL is to replace the spatial(space boundary-value) derivatives in the PDE with algebraic approximations [72]. Which in our case is radial basis functions.

After this discretization the spatial derivatives are no longer stated explicitly in terms of the spatial independent variables. Thus, only the initial-value variable, typically time in time dependent problems, remain. With only one independent variable we have a system of ODEs that approximates the original PDE.

We will then integrate this system of ODEs in time by applying any integration algorithm for the initial-value ODEs to compute an approximate numerical solution to the PDE. The prominent feature of the MOL is the use of an existing, and generally well established, numerical method for ODEs. Many of the numerical routines for ODEs can be found in [72] and reference therein. The name method of lines is motivated by the fact that graph of the approximated solution $\upsilon(x,t)$ is a set of vertical parallel lines in $x-\upsilon(x,t)$ plane where the height of each line is $\upsilon(x,t)$.

3.1.2 Meshless method of lines using radial basis functions for the generalized Burgers-Huxley equation

A large class of problems arising in physical science and engineering, are modeled by nonlinear partial differential equations (NLPDEs). The solution of most of the NLPDEs is still very difficult to obtain analytically and numerically. The search of more efficient methods (analytical/numerical) for solving such problems is an active area of research. Burgers-Huxely equation belongs to one of the most famous NLPDEs. The equation is a relevant model to describe the interaction between diffusion, convection and reaction. The generalized Burgers-Huxley equation was investigated by Satsuma [70] in 1987 is of the form

$$u_t + \alpha u^{\delta} u_x - u_{xx} = \beta u(1-u^{\delta})(u^{\delta}-\gamma), \quad a \leq x \leq b, \quad t \geq 0, \tag{3.1}$$

with the initial condition

$$u^0(x) = u(x,0) = \left(\frac{\gamma}{2} + \frac{\gamma}{2}\tanh(\omega_1 x)\right)^{\frac{1}{\delta}}, \tag{3.2}$$

and boundary conditions

$$u(x,t) = \left(\frac{\gamma}{2} + \frac{\gamma}{2}\tanh(\omega_1(x-\omega_2 t))\right)^{\frac{1}{\delta}}, \quad x \in \partial\Omega, \quad t > 0. \tag{3.3}$$

The exact solution of (3.1)-(3.3) is given by [81], i.e.,

$$u(x,t) = \left(\frac{\gamma}{2} + \frac{\gamma}{2}\tanh(\omega_1(x-\omega_2 t))\right)^{\frac{1}{\delta}}, \quad a \le x \le b, \quad t \ge 0, \tag{3.4}$$

where

$$\omega_1 = \frac{-\alpha\delta + \delta\sqrt{\alpha^2 + 4\beta(1+\delta)}}{4(1+\delta)}\gamma,$$

$$\omega_2 = \frac{\alpha\gamma}{1+\delta} - \frac{(1+\delta-\gamma)(-\alpha + \sqrt{\alpha^2 + 4\beta(1+\delta)})}{2(1+\delta)},$$

and $\alpha, \beta, \gamma, \delta$ are constants such that $\alpha \ge 0$, $\beta \ge 0, \delta > 0$ and $\gamma \in (0,1)$.

In addition to the other nonlinear terms, equation (3.1) contains a convection term uu_x and a dissipation term u_{xx}.

The generalized Burgers-Fisher equation is given by

$$u_t + \alpha u^\delta u_x - u_{xx} = \beta u(1-u^\delta), \quad a \le x \le b, \quad t \ge 0, \tag{3.5}$$

with the initial condition,

$$u_0(x) = u(x,0) = \left(\frac{1}{2} + \frac{1}{2}\tanh(a_1 x)\right)^{\frac{1}{\delta}}. \tag{3.6}$$

The boundary conditions are extracted from the exact solution of Equation (3.5) given by

$$u(x,t) = \left(\frac{1}{2} + \frac{1}{2}\tanh(a_1(x - a_2 t))\right)^{\frac{1}{\delta}}, \tag{3.7}$$

where ,

$$a_1 = \frac{-\alpha\delta}{2(1+\delta)},$$

and

$$a_2 = \frac{\alpha}{1+\delta}\frac{\beta(1+\delta)}{\alpha}.$$

Many researchers have established numerical methods for solving the generalized Burgers-Huxley equation. When $\alpha = 0$ and $\delta = 1$ equation (3.1) becomes the Huxley equation which describes nerve pulse propagation in nerve fibres and wall motion in liquid crystals. For $\beta = 0$ equation (3.1) reduces to the famous Burgers equation which is a relevant model in shock wave formulation for high Reynolds number. This behaviour is due to the nonlinear advection term uu_x and the dissipation term u_{xx}. Equation (3.5) has applications in various fields of science and engineering such as chemical kinetics, population dynamics and branching Brownian motion processes.

Most recently Jan [48] solved the generlized Burgers Huxley equation (3.1) by RBF collocation using the multiquadrics in space, and finite difference approximation in time. Batiha et al. [4] presented the variational iteration method (VIM), Adomain decomposition method (ADM) for Burger's-Huxley equation by Hashim et al.[41] and numerical solution of generlized Burgers-Huxley equation using pseudospectral method and Darvishi's preconditioning by Javidi [49]. We present the solutions of Equation (3.1) and Equation (3.5) using method of lines and radial basis function. The method, to our knowledge, is new for solving the Burgers-Huxley equation. In our examples we solve the problem for higher values of δ, i.e., $\delta = 2, 4, \text{and } 6$.

3.1.3 RBF interpolation

Given the data at N nodes $x_i \in \mathbb{R}$, where $i = 1, 2, ..., N$, the basic form of the RBF approximation for the function $u = u(x)$ is denoted by

$$u(x) \;=\; \sum_{i=1}^{N} \lambda_i \, \phi_i \;=\; \Phi^T(x)\lambda, \tag{3.8}$$

where $\Phi(x) \;=\; [\phi_1(x), ..., \phi_N(x)]^T$ and $\lambda \;=\; [\lambda_1, ..., \lambda_N]^T$.

We will denote u_N as the approximate solution to u. Let $u_N(\mathbf{x}_i) \;=\; u(x_i) \;=\; u_i$ at N collocation points. Then

$$\mathbf{A}\,\lambda \;=\; \mathbf{u}, \tag{3.9}$$

where $\mathbf{u} \;=\; [u_1, ..., u_N]^T$ and $\mathbf{A}$ is the interpolation matrix described in Section 2.3. By RBF interpolation given in equation 3.8 and using equation 3.9 we get,

$$u(x,t) \simeq \; u_N(x,t) \;=\; \sum_{i=1}^{N} \; \lambda_i \, \phi_i \;=\; \Phi^T(x) \; \mathbf{A}^{-1} \; \mathbf{u}, \tag{3.10}$$

where

$$\mathbf{u} \;=\; [u_1(t), ..., u_N(t)]^T.$$

To replace the derivative terms in equation 3.1 i.e., u_x and u_{xx}, we can use equation 3.8 to obtain derivative of u_x at centers $\mathbf{x}_1, \cdots, \mathbf{x}_N$ by differentiating the basis function

$$\frac{d}{dx}(u(x)) = \sum_{i=1}^{N} \lambda_i \frac{d}{dx}\phi(|\mathbf{x} - \mathbf{x_i}|) = \mathbf{A}_x\lambda, \tag{3.11}$$

where $(\mathbf{A}_x)_{ij} = \frac{d}{dx}[\phi(|x - x_j|)]_{x=x_i}$. For instance, derivative term u_x in equation 3.10 can be replaced with the discretized derivative matrix

$$\begin{bmatrix} \frac{d}{dx}u(x_1) \\ . \\ . \\ . \\ \frac{d}{dx}u(x_N) \end{bmatrix} = \begin{bmatrix} \\ \mathbf{A}_x \\ \\ \end{bmatrix} \begin{bmatrix} \lambda_1 \\ . \\ . \\ . \\ \lambda_N \end{bmatrix} = \begin{bmatrix} \\ \mathbf{A}_x \\ \\ \end{bmatrix} \begin{bmatrix} \\ \mathbf{A}^{-1} \\ \\ \end{bmatrix} \begin{bmatrix} u_1 \\ . \\ . \\ . \\ u_N \end{bmatrix}.$$

Let us denote

$$\mathbf{D}_x = \mathbf{A}_x \mathbf{A}^{-1}. \tag{3.12}$$

Higher order derivatives can obtained in a similar way by differentiating equation 3.11.

We will impose the boundary conditions i.e., at u(a,t) and u(b,t) can be extracted from (3.7). One way of doing this to make use of $(N-1)\times(N-1)$ matrix $\mathbf{D}_x$, obtained by taking away the first and last rows and columns i.e.,

$$\begin{bmatrix} \frac{d}{dx}u(x_0) \\ \frac{d}{dx}u(x_1) \\ . \\ . \\ . \\ \frac{d}{dx}u(x_{N-1}) \\ \frac{d}{dx}u(x_N) \end{bmatrix} = \begin{bmatrix} \\ \\ \\ \mathbf{D}_x \\ \\ \\ \\ \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ . \\ . \\ . \\ u_N \\ u_{N+1} \end{bmatrix}.$$

By applying equation 3.10 on 3.1, and then collocating on the nodes $a = x_1 < x_2 <$ $, ..., < x_{N-1} < x_N = b$ in [a,b] we obtain,

$$\frac{du_i}{dt} + \alpha u_i^{\delta}(\mathbf{D}_x(x_i)\mathbf{u}) - (\mathbf{D}_{xx}(x_i)\mathbf{u} = \beta u_i(1-u_i^{\delta})(u_i^{\delta}-\gamma), \quad \text{for} \quad i = 1, 2, ..., N, \tag{3.13}$$

here u_i stands for $u_i(t)$.

The MOL-RBF discretized form of (3.1) can be written in terms of the column vectors $\mathbf{U} = [u_1, u_2, ..., u_N]^T$, and discretized differentiation matrices D_x and D_{xx} as,

$$\frac{dU}{dt} + \alpha \mathbf{U}^{\delta} \divideontimes (D_x \mathbf{U}) - (D_{xx}\mathbf{U}) = \beta \mathbf{U} \divideontimes (1 - \mathbf{U}^{\delta}) \divideontimes (\mathbf{U}^{\delta} - \gamma). \tag{3.14}$$

The notation $\divideontimes$ is used for the component by component multiplication of the two vectors which is the famous Hadamard product. Equation (3.14) can be written as

$$\frac{dU}{dt} = Q(\mathbf{U}), \tag{3.15}$$

where,

$$Q(\mathbf{U}) = -\alpha \mathbf{U}^{\delta} \divideontimes (D_x \mathbf{U}) + D_{xx}\mathbf{U}) + \beta \mathbf{U} \divideontimes (1 - \mathbf{U}^{\delta}) \divideontimes (\mathbf{U}^{\delta} - \gamma), \tag{3.16}$$

subject to the corresponding initial condition given in equation 3.6 as

$$\mathbf{U}(t_0) = [u_1^0(x_1), u_2^0(x_2), ..., u_N^0(x_N)]^T, \tag{3.17}$$

and the boundary conditions extracted from equation 3.7 as $u(a,t) = u_1(t)$ and $u(b,t) = u_N(t)$.

We have the following convergence theorems for the infinitely smooth RBFs [83].

Theorem 3.1 *Let Φ be one of the Gaussian or (inverse) multiquadrics. Suppose that Φ is conditionally positive of order m. Suppose further that $\Omega \subseteq \mathbb{R}^d$ is bounded and satisfies an interior cone condition. Denote the radial basis function interpolant to $f \in \mathcal{N}_\Phi(\Omega)$ based on Φ and $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$ by $s_{f,\mathbf{X}}$. Fix $\alpha \in \mathbb{N}_0^d$. For every $l \in \mathbb{N}$ with $l \geq \max\{|\alpha|, m-1\}$ there exists constants $h_0(l)$, $C_l > 0$ such that*

$$| \mathbf{D}^{\alpha} f(x) - \mathbf{D}^{\alpha} s_{f,\mathbf{x}} | \leq C_l h_{\mathbf{X},\Omega}^{l-|\alpha|} |f|_{\mathcal{N}_\Phi(\Omega)}$$

for all $\mathbf{x} \in \Omega$, provided that $h_{\mathbf{X},\Omega} \leq h_0(l)$.

The convergence theorem for the polyharmonic splines is given as

Theorem 3.2 *Suppose that $\Omega \subseteq \mathbb{R}^d$ is bounded and satisfies an interior cone condition. Let $\Phi(\mathbf{x}) = (-1)^{k+1} \parallel x \parallel_2^{2k} \log(\parallel x \parallel_2)$. Denote the interpolant of a function $f \in \mathcal{N}_\Phi(\Omega)$ based on the basis function and the set of centers $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\} \subseteq \Omega$ by $s_{f,\mathbf{X}}$. Then there exits constants h_0, $C > 0$ such that*

$$| \mathbf{D}^\alpha f(x) - \mathbf{D}^\alpha s_{f,\mathbf{x}} | \leq C_l h_{\mathbf{X},\Omega}^{l-|\alpha|} |f|_{\mathcal{N}_\Phi(\Omega)}$$

, for all $\mathbf{x} \in \Omega$ and all α with $|\alpha| \leq k-1$, provided that $h_{\mathbf{X},\Omega} \leq h_0$.

3.2 Classical fourth-order Runge-Kutta method

Space discretization of (3.1) using MQ gives system of equations given by (3.16), next stage is to integrate the system of ODEs in time, using the classical fourth order explicit Runge-Kutta (RK4) scheme. The is done in a method of lines fashion. The popular $RK4$ method to advance the approximate solution in time is

$$\begin{aligned} U^{n+1} &= U^n + \frac{\triangle t(K_1 + 2K_2 + 2K_3 + K_4)}{6} \quad \text{with} \\ K_1 &= Q(U^n), \\ K_2 &= Q\left(U^n + \frac{\triangle t}{2} K_1\right), \\ K_3 &= Q\left(U^n + \frac{\triangle t}{2} K_2\right), \\ K_4 &= Q(U^n + \triangle t K_3). \end{aligned} \tag{3.18}$$

Numerical schemes using explicit methods for time integration of the ODEs requires to hold the CFL condition. Courant, Friedrichs and Lewy in 1928 [19] formulated a necessary condition (*CFL condition*) for solving partial differential equations using difference methods. They formulated the CFL condition for the convergence of the difference scheme in terms of the domain of dependence. Runge-Kutta methods, belong

to an important family of implicit and explicit iterative methods for integration of ODEs in time and we can hope for stable result if the CFL condition is satisfied. The CFL restriction in 1-D is defined as

$$\frac{\Delta t}{(\Delta x)^{\alpha}} \leq C,$$

where Δt is time step and Δx is spatial spacing.

On other way to interpret this is, it is the maximum acceptable number, a time integrator can use. The Courant number is a measure of how much information pass through Δx for each Δt. A numerical scheme violating the CFL restriction, means that the time integrator is not interpreting what is physically happening and information propagating through more than one grid cell each time step. This will make the solution unstable.

The investigation of eigenvalues stability for the MOL-RBF is in its early stages, and CFL restriction for MOL-RBF is an open question. However the choice of Δt is due to rule of thumb for stability.

Rule of Thumb

The method of lines is stable if the eigenvalues of the (linearized) spatial discretization operator, scaled by Δt, lie in the stability region of the numerical ODE method [77].

Stability region of the ODE method is the subset of the complex plane consisting those eigenvalues (λ) in $\mathbb{C}$, for which the numerical approximation posses bounded solution when applied to the scalar linear model problem $u_t = \lambda u$, multiplied with Δt where Δt is time stepping. It can be seen in Figure 3.1 that the stability region of RK4 is the largest amongst RK4 of order 1,2,and 3 respectively.

There are other ODE integrators i.e., implicit trapezoidal rule which is second-order accurate. Details on ODE integrators can be found in references on numerical methods for ODEs [15]. We choose explicit four stage $RK4$ for integration of ODEs due to its larger stability region. For most of the hyperbolic PDEs the eigenvalues lies on the imaginary axis. The absolute stability region of RK4 intersects with the imaginary axis (See Figure 3.1) and this property makes it a favourable choice for solving time-

dependent hyperbolic PDEs.

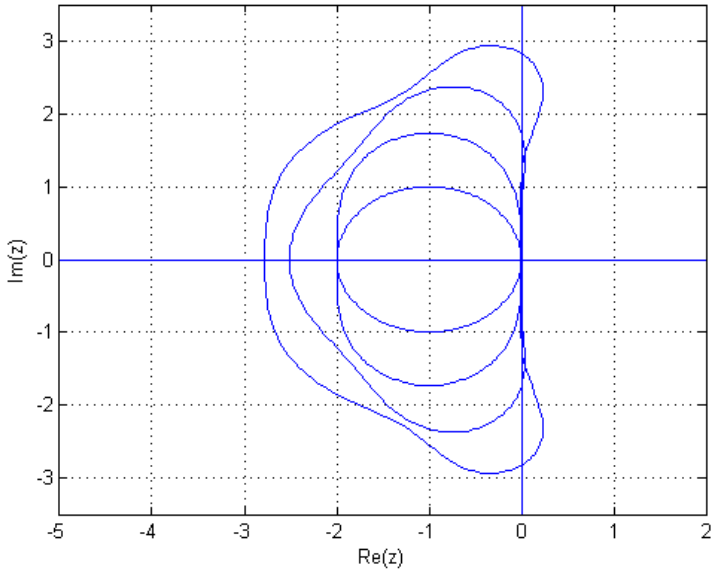


Figure 3.1: Stability region for Runge-Kutta method of order $1, 2, 3, 4$, the larger order is the larger the stability region. Stability region of RK4 has an intersection with imaginary axis.

For the stable time discretization, $RK4$ scheme requires the discretized PDE to have non-positive real parts of all its eigenvalues. The stable spectrum will fit in the stability region of $RK4$ scheme, if scaled by an appropriate time step Δt. A numerical study of accuracy and stability of RBF collocation methods for hyperbolic PDEs is discussed in [69].

In literature, for Runge-Kutta methods satisfying the special stability requirements such as total variation bounded (TVB), total variation diminishing (TVD), and strong stability preserving schemes (SSP), are discussed in a great detail. For solving nonlinear time-dependent PDEs where the solution exhibits discontinuities, much attention has been paid to the numerical schemes having the property of being total variation bounded. A much in-depth details can be found in [35, 27, 70, 75] and the reference therein.

For the total variation (TV) defined as

$$TV(u) = \sum_j |u_{j+1} - u_j|,$$

the TVB schemes satisfies

$$TV(u^n) \leq B,$$

for some fixed $B > 0$, depending only on TV(u^0) $n \geq 0$, and for Δt such that $n\Delta t \leq T$ will be known as TVB in $0 \leq t \leq T$. In our examples we are using explicit RK4 method for integrating the ODEs in time which possess the TVB property [44].

3.3 Numerical experiments

In this section the meshless RBF-MOL based approximate results are given for (3.1) and (3.5). A comparison of absolute errors of the RBF-MOL with variational iteration method (VIM) [4], meshless RBF collocation method [48], and the adomain decomposition method (ADM) [41] is presented.

All the computational work is performed with the nodal distance $dx = 0.1$ and time step $dt = 0.001$ otherwise mentioned explicitly. The computational domain is considered as [0,1] for the sake of comparison with ADM, VIM and RBF collocation method. To measure the difference between the exact solution and numerical solution the absolute error (A.E) is calculated. In our numerical experiments we have used multiquadrics as the radial basis function. Excellent agreement is found between the exact solution and the the results obtained by the MOL-RBF method.

Example 1. We consider the generalized Burgers-Huxley Equation (3.1) approximated with MOL-RBF and in Table (3.1) a comparison is made with Variational iteration method (VIM) [4] using the appropriate initial and boundary conditions. The parameters, $\gamma = 0.01, \alpha = \beta = 1, \delta = 2$ and $\delta = 4$, are taken to be the same as [4] for the sake of comparison. For $\gamma = 0.01, \alpha = \beta = 1, \delta = 6$ we compared the computations with Adomain decomposotion method (ADM) [41]. The kink-type solution obtained by MQ-RBF in comparison with the exact solution of Equation (3.5) is presented in Figure (3.1) where the parameters $\alpha = \beta = 1$, $\gamma = 2$, $\delta = 1$, $c = 2$ and $dt = 0.001$ at time

$t = 10$.

Table 3.1: MQ $\alpha = \beta = 1, \gamma = 0.01$, c=1.4

x	t				
	0.1	0.2	0.3	0.4	0.5
$\delta = 2$ RBF-MOL					
0.1	1.47E-06	1.25E-05	1.30E-05	1.32E-05	1.32E-05
0.3	3.11E-06	9.93E-06	1.15E-05	1.20E-05	1.22E-05
0.5	3.37E-06	1.140E-05	1.20E-05	1.21E-05	1.22E-05
$\delta = 2$ VIM [4]					
0.1	5.51E-05	1.10E-04	1.65E-04	2.20E-04	2.75E-04
0.3	5.51E-05	1.10E-04	1.65E-04	2.20E-04	2.75E-04
0.5	5.51E-05	1.10E-04	1.65E-04	2.20E-04	2.75E-04
$\delta = 4$ RBF-MOL					
0.1	4.52E-05	5.10E-05	5.30E-05	5.37E-05	5.39E-05
0.3	4.19E-05	5.70E-05	6.25E-05	6.44E-05	6.49E-05
0.5	2.92E-05	4.79E-05	5.47E-05	5.70E-05	5.77E-05
$\delta = 4$ VIM [4]					
0.1	2.17E-04	4.36E-04	6.53E-04	8.71E-04	1.09E-04
0.3	2.17E-04	4.35E-04	6.53E-04	8.70E-04	1.08E-04
0.5	2.17E-04	4.34E-04	6.52E-04	8.69E-04	1.08E-04
$\delta = 6$ RBF-MOL					
0.1	3.39-05	4.78E-05	5.09E-05	5.18E-05	8.75E-05
0.3	4.80E-05	7.05E-05	7.86E-05	8.12E-05	1.09E-04
0.5	5.22E-05	8.05E-05	9.06E-05	9.38E-05	9.97E-05
$\delta = 6$ ADM [41]					
0.1	3.48E-04	6.96E-04	1.04E-03	1.39E-03	1.74E-03
0.3	3.47E-04	6.95E-04	1.04E-03	1.39E-03	1.74E-3
0.5	3.47E-04	6.94E-04	1.04E-03	1.38E-03	1.73E-03

Example 2. The example illustrate the approximate solution of the generalized

Burger's-Fisher Equation (3.5). The results are given in Table (3.2) and (3.3) for $\alpha = 1, \beta = 0$, and for $\delta = 1, 2$ and 3, respectively. The comparison is made with the method based on collocation by RBFs [48].

Table 3.2: MQ $\alpha = 1$, $\beta = 0$, $c = 1.02$

x	t				
	0.5	1	2	5	50
$\delta = 1$ RBF-MOL					
0.1	6.79E-10	2.22E-08	7.08E-08	2.24E-07	5.67E-07
0.5	1.13E-08	3.92E-09	1.12E-08	5.71E-08	1.23E-07
0.9	5.42E-07	5.70E-07	6.23E-07	7.41E-07	7.120E-07
$\delta = 1$ RBF [48]					
0.1	1.0E-06	1.0E-06	1.0E-06	4.0E-06	6.0E-06
0.5	5.0E-06	3.0E-06	3.0E-06	9.0E-06	2.0E-06
0.9	3.0E-06	3.0E-06	3.0E-06	3.0E-06	3.0E-06
$\delta = 2$ RBF-MOL					
0.1	1.64E-07	1.83E-07	2.20E-07	3.21E-07	5.67E-07
0.5	5.05E-08	5.73E-08	6.82E-08	9.55E-08	1.23E-07
0.9	7.01E-07	7.17E-07	7.46E-07	8.04E-07	7.12E-07
$\delta = 2$ [RBF [48]]					
0.1	1.0E-06	0.0	2.0E-06	4.0E-06	3.0E-06
0.5	2.0E-06	1.0E-06	5.0E-06	1.1E-06	1.4E-06
0.9	0.0	2.0E-06	1.0E-06	3.0E-06	5.0E-06

Table 3.3: MQ $\alpha = 1$,$\beta = 0$, $\delta = 3$, $c = 1.02$, $dt = 0.0001$

x	t	A.E, RBF[48]	A.E, MOL-RBF
0.1	0.0005	6.0E-06	3.84E-08
	0.0010	1.9E-05	7.82E-08
0.5	0.0005	5.0E-06	2.85E-09
	0.0010	1.6E-05	5.69E-09
0.9	0.0005	4.0E-06	9.77E-08
	0.0010	1.5E-05	1.94E-07

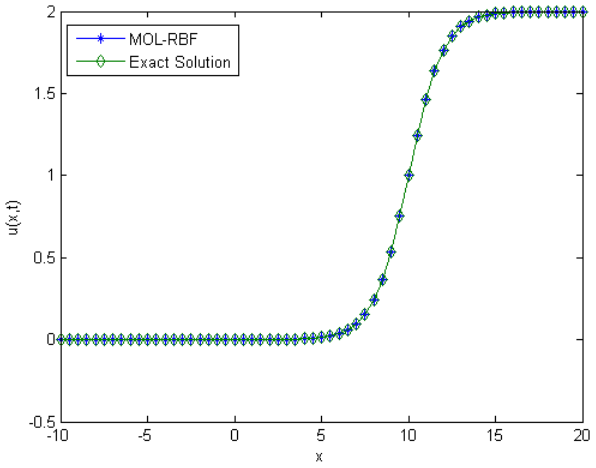


Figure 3.2: Solitary wave solution for Example 1, Parameters: $\alpha = \beta = 1, \gamma = 2, \delta = 1$

3.4 Allen-Cahn equation in two-dimensions

The Allen-Cahn equation in 2-D is with the Dirichlet boundary conditions is defined as:

$$\begin{aligned} u_t - \Delta u + \frac{1}{\epsilon^2} u(u^2 - 1) &= 0 \text{ in } \Omega_T := \Omega \times (0, T), \\ u &= 0 \text{ in } \partial\Omega_T := \partial\Omega \times (0, T), \\ u &= u_0 \text{ in } \Omega \times \{0\} \end{aligned} \tag{3.19}$$

where $\Omega \subset \mathbf{R}^N$ (N=2,3), $T > 0$ is a fixed constant, Δ is the Laplacian operator, ϵ is the interaction length.

Radial basis function methods makes the higher dimensional problems virtually one dimensional hence the MOL-RBF interpolation described in Subsection 3.1.3 can be

extended for $\Omega \subset \mathbf{R}^2$. The differentiation matrices for the Laplacian operator can be obtained in the simliar way as

$$(\mathbf{A_{xx}})_{ij} = \frac{d^2}{dx^2}[\phi(\| \mathbf{x} - \mathbf{x}_j \|)]_{\mathbf{x}=\mathbf{x}_i}.$$

$$(\mathbf{A_{yy}})_{ij} = \frac{d^2}{dy^2}[\phi(\| \mathbf{y} - \mathbf{y}_j \|)]_{\mathbf{y}=\mathbf{y}_i}.$$

Analogous to equation 3.12 D_{xx} and D_{yy} will be of the form

$$\mathbf{D}_{xx} = \mathbf{A}_{xx}\mathbf{A}^{-1},$$

and

$$\mathbf{D}_{yy} = \mathbf{A}_{yy}\mathbf{A}^{-1},$$

respectively. The MOL-RBF discretized form of equation 3.19 can be written in the form

$$\frac{du_i}{dt} = \mathbf{D}_{xx}\mathbf{u} + \mathbf{D}_{yy}\mathbf{u} - \frac{1}{\epsilon^2}u_i(u_i^2 - 1), \quad \text{for} \quad i = 1, ..., N, \tag{3.20}$$

here u_i stands for $u_i(t)$. The above system of ODEs will be integrated in time using explicit Runge-Kutta scheme of order four.

Example: For 3.19 we will use $\Omega = [-1, 1]^2 \subset \mathbf{R}^2$ as the computational domain with the initial condition

$$u_0(x, y) = \begin{cases} \tanh(\frac{3}{\epsilon}((x - 0.5)^2 + y^2 - (0.39)^2)) & \text{if } x > 0.14, \\ \tanh(\frac{3}{\epsilon}(y^2 - (0.15)^2)) & \text{if } -0.3 \leq x \leq 0.14, \\ \tanh(\frac{3}{\epsilon}((x + 0.5)^2 + y^2 - (0.25)^2)) & \text{if } x < -0.3. \end{cases}$$

where $\tanh = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ and $\epsilon = 0.05$. We will use $N = 41$ i.e., a total number of 1681 points. The time stepping is $\Delta t = 0.0001$ with a final time $T_f = 0.05$. MQ-RBF will be used for the global approximations with shape parameter $c = 10$.

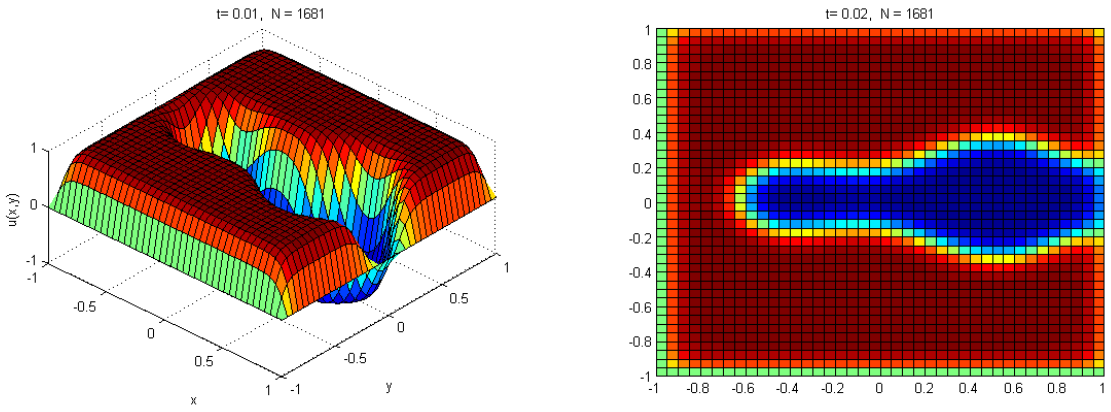


Figure 3.3: RBF-MOL solution for the Allen-Cahn equation using uniform distribution of nodes at $t = 0.01$.

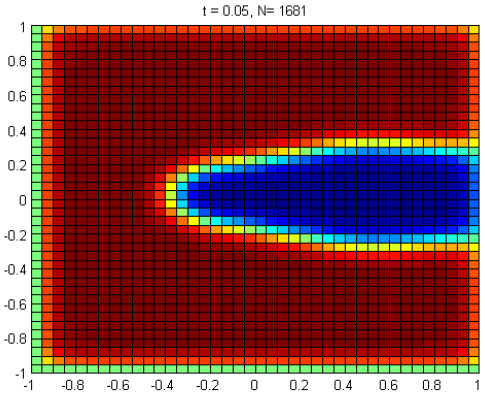


Figure 3.4: Solution at $t = 0.05$.

3.4.1 Conclusions

A radial basis function method, combining method of lines has been proposed to solve the generalized Burgers Huxley equation. We can see from Table (3.1) that our method is in good comparison with VIM [4] and ADM [41]. For the problem considered in Example 2, Table (3.2) and (3.3) are comparing the results obtained by RBF-MOL and RBF collocation method [48]. The results give us confidence to extend the method for more difficult 1-D problems, with an adaptive selection of nodes in space. In Section (3.4) we propose the RBL-MOL for the Allen-Cahn equation in 2-D. We aim in the extension of the method with adaptive selection of nodes in 2-D. Radial basis function belongs to the family of meshless methods so the method for 2-D Allen-Cahn equation

will remain essentially the same as for 1-D (See Subsection 3.1.2).

In the next chapter we will discuss the KdV equation and the difficulty for numerical methods to produce accurate results for this challenging PDE.

Chapter 4

Korteweg de Vries Equation

The Korteweg-de Vries equation belong to hyperbolic class of PDEs i.e., the linearization of this equation is a hyperbolic PDE. In general hyperbolic PDEs are considered to be a challenging task for numerical methods. In this chapter we will discuss this equation in detail and the challenges a numerical method may face. In next chapter we will use this PDE as a model problem for our adaptive algorithm.

4.1 Korteweg-de Vries equation and theory of solitons

The discovery of Korteweg-de Vries equation (1895) dated back to the discovery of soiltary waves (1834). Solitary waves were observed by a Scottish engineer John Scott Russell in 1834 in Russell's own words [66],

"I was observing the motion of a boat which was rapidly drawn along a narrow channel by a pair of horses, when the boat suddenly stopped not so the mass of water in the channel which it had put in motion; it accumulated round the prow of the vessel in a state of violent agitation, then suddenly leaving it behind, rolled forward with great velocity, assuming the form of a large solitary elevation, a rounded, smooth and well-defined heap of water, which continued its course along the channel apparently without change of form or diminution of speed. I followed it on horseback, and overtook it still rolling on at a

rate of some eight or nine miles an hour, preserving its original figure some thirty feet long and a foot to a foot and a half in height. Its height gradually diminished, and after a chase of one or two miles I lost it in the windings of the channel. Such, in the month of August 1834, was my first chance interview with that singular and beautiful phenomenon which I have called the Wave of Translation".

Despite the fact that Scott Russell's experimental observation was of fundamental importance it was not accepted by the science community, until Boussinesq in 1871 found a PDE with a solitary wave solution. In 1895 Diederik Korteweg and his doctoral student Gustav de Vries derived the equation, named as Korteweg-de Vries equation (KdV) [52]. The KdV equation was initially derived for the unidirectional propagation of small amplitude a, long wavelength λ in water of relatively shallow depth i.e., $h \ll \lambda$ Figure 4.1 explains the length scales used in the derivation of shallow water waves where λ is the wave length which is the measure of the spatial period of the wave, a is the amplitude which measure of the height of the wave i.e., the distance between the uninterrupted water to the peak of the wave and h is the depth of the water measured from the flat bottom of the water up to a quite free surface.

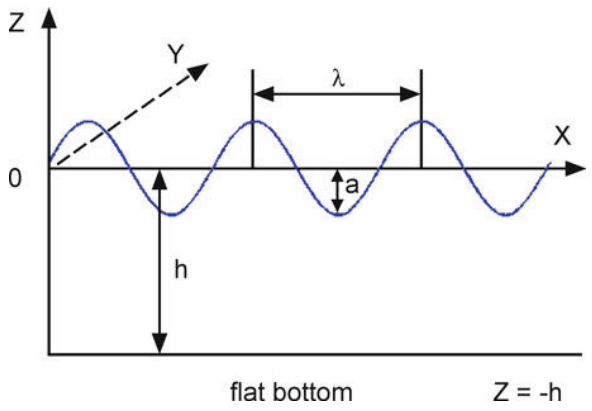


Figure 4.1: Periodic wave on the surface of water.

Korteweg and de Vries showed that this phenomenon could be described approximately by a nonlinear partial differential equation.

The conventional, non-dimensional version of the third-order nonlinear KdV equation

is

$$u_t + \epsilon u u_x + \mu u_{xxx} = 0, \tag{4.1}$$

where u represents height from the average water surface and x is the coordinate moving with the velocity of a linearized wave with $u \to 0$ as $|x| \to \infty$, x is a boundary-value (spatial) independent variable, t is an initial-value independent variable, and ϵ, μ are real constants.

The term $\epsilon u u_x$ is the nonlinear term which makes wave steep and the term μu_{xxx} is the dispersion term describes the spreading of the wave. The KdV equation is a nonlinear and dispersive wave equation i.e., waves of different wave number propagates at different velocities which is the characteristic of a *dispersive wave*. The nonlinear steepening can be balanced by dispersion effect. In the absence of dispersion term the equation produces the unique solution only for a finite time. The term u_t is the time evolution of propagation of unidirectional wave.

We can observe the linear dispersive relation of the equation by considering the simplest dispersive wave equation

$$u_t + u_x + u_{xxx} = 0. \tag{4.2}$$

The dispersion relation is

$$\omega = \lambda(1 - \lambda^2), \tag{4.3}$$

which can be obtained by substituting

$$u(x,t) = \ e^{i(\lambda x - \omega t)}, \tag{4.4}$$

into (4.2), where λ is the wave number, ω is the frequency, ω is a function of λ which means that the phase velocity depends on the wave number. From (4.3) we can see that

$$\lambda x - \omega t = \lambda\{x - (1 - \lambda^2)t\},$$

Hence the solution (4.4), with condition (4.3) describes a wave which travel at the velocity

$$m = \frac{\omega}{\lambda} = 1 - \lambda^2, \tag{4.5}$$

where m in (4.5)is called the phase velocity which by 4.5 describes that different components are traveling at different velocities which will spread or disperse the profile of the solution.

The balance of dispersion and nonlinearity makes a stationary waveform called a solitary wave which is a wave with infinite support [22]. The solitary wave propagates with uniform velocity without changing its shape and is consist of a single symmetrical hump. Solitary wave which is the phenomenon discovered by Scott Russell, first arose in connection with water waves it was later realized that it is a model equation which balance the nonlinearity and dispersion. The equation can be used to study nonlinear waves with an effect of dispersion in any medium.

The heart of the observations about the KdV was the question that what happens when two "great waves of translation" runs into each other ?. This question was not answered by Russell and the observation was hidden in his report [66]. The phenomenon was first observed by Zabusky and Kruskal [87] while studying the continuum approximation to the nonlinear discrete mass string of Fermi-Pasta-Ulam. They observed that initially a sinusoidal profile would decompose into a series of interacting pulses which had the shape of solitary wave i.e., they considered

$$u_t + uu_x + \delta^2 u_{xxx} = 0 \tag{4.6}$$

with periodic boundary conditions

$$u(x, 0) = \cos \pi x, \;\; 0 \leq x \leq 2,$$

and u, u_x, u_{xxx} are periodic on [0,2] for all t, they choose $\delta = 0.0022$. The celebrated observation was that these nonlinear pulses can interact strongly and then continue

thereafter with almost no effect of interaction at all. One other way of saying this is,

- A localized wave propagates without change of its properties i.e., shape and velocity.
- Localized waves are stable against mutual collisions and retain their identities.

Zabusky and Kruskal [87] coined the name *soliton* for such waves. The usual way to refer the single-soliton solution is as a solitary wave, but when more than one of them appear in a solution they are called solitons. The KdV equation has become the streamline research since early 1960's [87, 26]. The discovery by Zubusky and Kruskal has led, in turn to a diverse area of research since many physical, mechanical and biological phenomena can be described using the theory of solitons. This makes the KdV equation and other equations that admits the solitary wave and soliton solution, an intense area of research [28, 64]. A detailed survey of results is presented by Miura [60].

4.1.1 Challenges and known numerical methods for solving KdV equation

The approximate solutions to the KdV equation have been obtained by well known numerical methods, like finite difference, finite elements and meshfree radial basis functions, as discussed. Miura [60] have presented a survey of results for this equation. The explicit schemes for the KdV equation suffers from stability issues and we have to keep the time step very small. For instance, the RBF-MOL scheme proposed in Chapter 3 for the KdV equation [74] is very sensitive with regard to the time stepping and becomes unstable with a considerably small increment in time stepping. Figure 4.2 is showing this observation

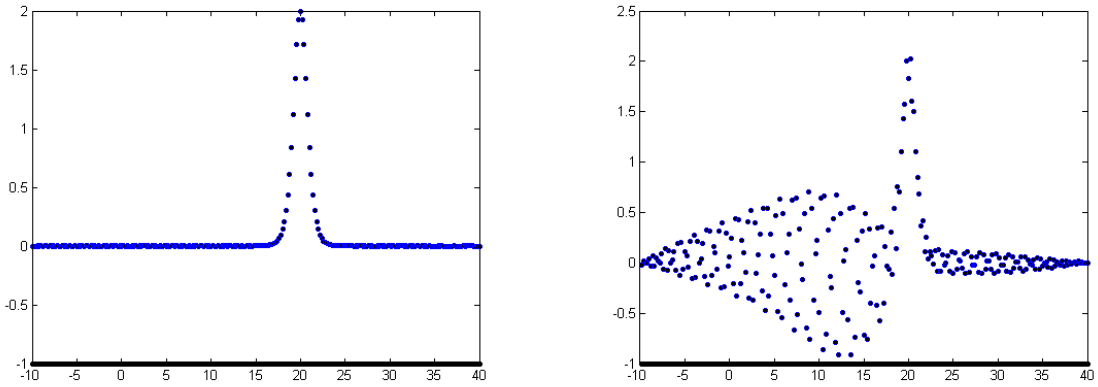


Figure 4.2: Solution of KdV equation using uniform distribution of nodes, becomes unstable with a very small increment in Δ Left: $\Delta t = 1.2953 \times 10^{-3}$, Right: $\Delta t = 1.2956 \times 10^{-3}$

Using radial basis function where the basis function contains a the shape parameter finding the CFL restriction is a difficult task and the reason is two fold, one is the non-linearity present in the equation, second varying the shape parameter. We numerically observed the CFL condition for MOL-RBF [74] although not discussed in the given article. The KdV equation is third order is space so for $\frac{\Delta t}{(\Delta)^{\alpha}} \leq C$ we expected $\alpha = 3$ but numerical experiments showed that $\alpha \approx 3.2$ which varies if we change the shape parameter. This issue requires a deep understanding while solving such problems using radial basis function interpolation.

On the basis of our experiments, we also observed convergence of the MOL-RBF for the KdV equation using equally spaced grid points. In this experiments we fixed shape parameter c and time step Δt for different Δx in space. It is evident from Figure 4.3 that the method is almost exponential in space.

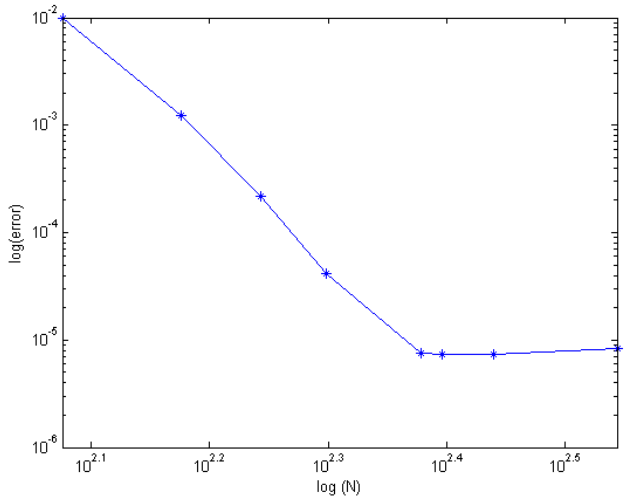


Figure 4.3: The plot of convergence for the MOL-RBF in spatial variable for KdV equation using $\Delta t = 0.01$, $c = 0.65$ at time $t = 0.5$

We observed that the increasing, the number of grid points, is making the system severely ill-conditioned i.e., 10^{10} and requires some preconditioning.

Implicit schemes can enjoy a greater numerical stability but they are expensive in terms of computations i.e., they require to solve the system of algebraic equations at each step. For the stiff ODEs implicit methods can be preferred over explicit methods as they do not have a restrictive absolute stability requirement. Another observation about numerical methods for this equation is that the numerical scheme has to be conservative in nature with long time stable results. For instance, leapfrog explicit scheme produce correct results for a while but at a later stage blows up. Numerical schemes for the KdV equation should be carefully designed such that they remains stable for long time integration and conservative in nature. An accurate numerical scheme should genuinely represent the amplitude of the true solution for long time integration. A numerical method with damping properties in long time integration will produce inaccurate solution.

Several methods have been proposed for numerical solution of the KdV equation. For instance,

- Zabusky and Kruskal [87] and Vliegenthart [79] have used the finite difference explicit scheme;

- Greig and Morris [36] have derived the Hopscotch algorithm based on implicit finite difference scheme;
- Winther [84] and Sanz-Serna and Christie [67] have used the finite element method;
- Shen [73] used spectral method and Cheng et al. [18] have used the discontinuous Galerkin method;
- Shen [74] has combined meshfree radial basis functions with method of lines;

Meshfree methods are considered to be a viable choice for solving PDEs. In the last decade radial basis function has been considered as a novel method for solving PDEs. Recently in [74] KdV equation has been solved by combining RBFs with the method of lines on a uniform distribution of nodes. However, solving the problem with adaptive radial basis function interpolation, to our knowledge, is new. Aadaptive radial basis function interpolation, has drawn attention of many researchers for solving the PDEs exhibiting high degrees of localization in space/time. Our method have used the adaptive distribution of nodes, where more nodes have been flagged in the regions of high activity and coarsened otherwise. Adaptive in comparison with non-adaptive method, solved the problem with as less as possible, numbers of nodes with the same accuracy. Our numerical scheme represents the amplitude of the true solution for long time integrations and is also conservative i.e., non-dissipate. We have also presented the numerical experiments for the interaction of solitons solution of the KdV equation.

4.1.2 Why choosing the KdV equation for adaptive algorithm?

The study of nonlinear Partial differential equations arising in physical applications is a growing research area. For many such equations questions regarding existence, uniqueness and stability of the solution are still unanswered. The KdV equation with the dispersion present in the equation is a challenging problem for numerical methods. We will now summarize our discussion for choosing this particular problem from hyperbolic class of nonlinear PDEs.

A very wide class of non-linear dispersive systems can be described by this equation or by one of its modified form. There are many positive reasons to make the present study, of interest, to summarize the discussion that it is a model nonlinear hyperbolic equation with smooth solutions for all times. The PDE is nondissipative and it is of dispersive nature as discussed in Section 4.1. The KdV equation is well know to satisfy a number of conservation laws in [61]. For instance, the conservation of mass $I_1(u) = \int_{-\infty}^{\infty}(u)dx =$ constant and conservation of energy $I_1(u) = \int_{-\infty}^{\infty}(u)^2dx =$ constant and are invariant with time.

The KdV equation for $\mu \ll 1$ is somehow close to burgers equation its dispersion limit. Burger's equation often develops solutions with shock discontinuities whereas the solution of KdV equation always remains smooth Many recent schemes appeared in the literature are fully implicit, on the other hand explicit and semi-explicit schemes and are too limited. The model equation is a nonlinear PDE with an exact solution that can be used to asses the accuracy of a numerical method. The PDE admits a soliton solution which is an intense area of research and have applications in many areas of physics, engineering, and biological sciences. Some nonlinear systems have solitary waves but not solitons, whereas the KdV is from one of the models having solitary waves which are solitons.

In the next chapter we will present the adaptive radial basis function method for the celebrated KdV equation.

Chapter 5

Adaptive Radial Basis Function Method

The essence of most of the adaptive methods for time-dependent PDEs is a cyclic procedure [23]. For instance, solve → error indicator → refine/coarse, and the cycle terminate when a stopping criterion is satisfied. In this thesis we have developed an adaptive RBF method which follows this outline.

The procedure *solve* in the time-dependent sense means that the solutions to the PDEs are obtained by marching in time using any explicit/implicit ODE solver. In our case this is done by combining MOL and RBFs. Once the approximate solutions are obtained the information is then passed to the *error indicator* which gathers the information about the local approximation quality of the the interpolation around $\mathbf{x} \in \mathbf{X}$. The local residual is the criteria for mesh refinement or coarsening. The decision is then passed to the *refine/coarse* stage. The refinement strategy for 1-D is double the number of points however 2-D refinement requires the construction of Voronoi points. We will now explain relevant feature of RBF interpolation, with the design of adaptive rules i.e., refining and coarsening strategy in the following sections.

The flow chart for the adaptive RBF interpolation is,

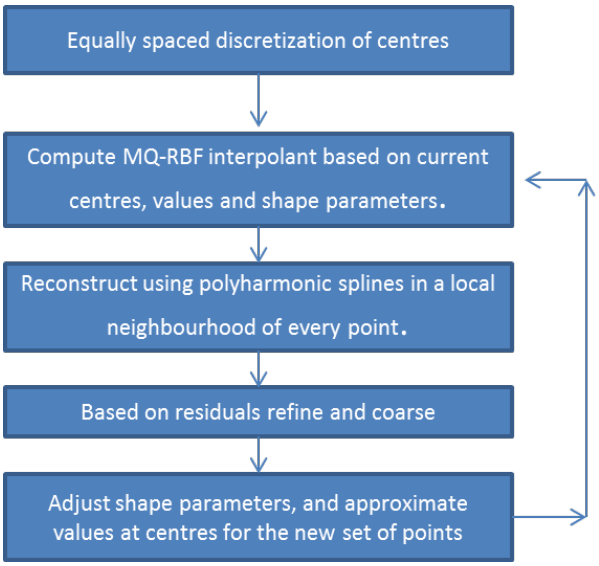


Figure 5.1: Adaptive flow chart, this cyclic process will be called at every time level for time-dependent PDEs.

The above is a cyclic process until we reach to a stopping criterion.

5.1 Adaptive interpolation

5.1.1 Solve

The adaptive interpolation is simple and straight forward to implement. Our adaptive method for time dependent PDEs, will be discretized in space by RBFs in a method of lines fashion. For all our experiments we have used MQ-RBF. The method of lines using RBFs is discussed in Chapter 3, where we explained MOL-RBF for solving the generalized Burgers-Huxely equation.

Solve; in the adaptive interpolation we advance the solution of the PDE on adaptively selected nodes using ODE integrators. By space discretization using MQ-RBF in a method of lines scheme, we get a system of ODEs to advance in time for the solution of underlying PDE. The choice of the ODE integrator is a flexible feature i.e., we can use the standard explicit/implicit ODE solvers to integrate our solution in time. In our numerical experiments for solving KdV equation and Allen-Cahn equation, we have used explicit fourth-order $RK4$ ODE integrator (see Section 3.3). For the Burgers equation,

we have used MATLAB built-in *ode*15*s* ODE solver. The solver is based on backward differentiation formula (BDF) which belongs to a family of implicit methods for solving ODEs.

The RBFs can be globally supported, infinitely differentiable and contains a free parameter. RBFs with such properties posses interpolation with spectral accuracy [13, 17] when the solution is smooth.

5.1.2 Adapting the shape parameter

It has been observed in [23] that adapting the shape parameter is crucial for adapting the profile of the solution. The shape parameter affects the accuracy and the conditioning of the interpolation matrix. Numerical experiments shows that adjustment of shape parameters with the number of centers can produce invertible well conditioned interpolation matrix in finite precision arithmetic. The adjustment of shape parameters means that every center will have its own shape parameter and the strategy is based on spacing to its nearest centers.

In the literature many researchers have attempted to find the optimal value of the shape parameter. The optimal value of the shape parameter means that it can produce the most accurate interpolant. In our numerical experiments we are adapting the shape parameter as a function of spacing to its nearest neighbours which has shown promising results in our numerical examples.

The following algorithm can be used to adapt the shape parameters as per center [23].

ALGORITHM. (Adapting the shape parameters)

INPUT: Set of centers.

1. Let dx be the differences between centers.
2. Obtain two arrays i.e., $[\text{Inf}, 1./dx]$ and $[1./dx, \text{Inf}]$.
3. Choose the minimum i.e., $c = \min([\text{Inf}, 1./dx], [1./dx, \text{Inf}])$.

OUTPUT: c the array of adaptively adjusted shape parameters for each center.

To prevent the growth of the *condition number* $\kappa(A)$ we need to raise the lower bound of the minimum eigenvalue $\lambda_{\min}$. Due to the Gershgorin's theorem $\lambda_{\max}$ is bounded for the quasi-uniformly spaced data. However $\lambda_{\min}$, as a function of data sites, or their separation distance, decays much faster. The separation distance is a natural choice compared to the fill distance, for lower bounds of the eigenvalues. The reason is that a point could have a big fill distance but for a badly conditioned interpolation process, only two of the points needs to be very close.
The lower bound for the generalized multiquadrics $\phi(x, c) = (1 + (\parallel cx \parallel)^2)^\beta$ given in [83]

$$\lambda_{\min} \geq C(d, \beta, c) q^{\beta - \frac{d}{2} + \frac{1}{2}} e^{\frac{-2M_d}{qc}},$$

where $\beta \in \mathbb{R} \setminus \mathbb{N}_0$, d is the dimension, $C(s, \beta, c)$ and M_d are some constants, q is the separation distance. The specialized case for the above using$\beta = 1/2, d = 1$ would take the form $\lambda_{min} \geq C(1, 1/2, c) q^{\frac{1}{2}} e^{\frac{-2M_1}{qc}}$.

The λ_{min} for the thin plate splines $\phi(x) = (-1)^{\beta+1} (\parallel x \parallel)^{2\beta} \log \parallel x \parallel$ is defined as,

$$\lambda_{\min} \geq C_d c_\beta (2M_d)^{-d-2\beta} q^{2\beta},$$

where C_d, c_β, are known constants, d is the dimension and $\beta \in \mathbb{N}$. The specialized case for the above using $\beta = 1$ and $d = 2$ would take the form $\lambda_{min} \geq C_2 c_1 (2M_2)^{-4} q^2$

The discussion on the lower bounds of other radial basis functions can be found in [82, 25]. To raise the lower bound of λ_{min} we are using center dependent shape

parameters and the strategy is quite straight forward, i.e., the shape parameter for each center is selected on its distance to its nearest neighbours. In case of using variable shape parameter the theoretical analysis of the RBF methods is quite complex. However, Bozzini et al., [10] have provided sufficient conditions for the non-singularity of the RBF interpolation matrix.

5.1.3 Error indicator

Development of an error estimator/indicator is an active area of research in finite elements [34]. Solving PDEs with adaptive approach relies on the error indicator which indicates the region of high variation [23, 45]. This information can be used to place finer grids in such regions. We have used a simple yet effective error indicator in our experiments i.e., the pointwise error $|u(\mathbf{x}) - s(\mathbf{x})|$ between a sufficiently smooth function u and polyharmonic spline interpolant satisfying $u|_{\mathcal{N}} = s|_{\mathcal{N}}$ in a local neighbourhood $\mathcal{N}$ around $\mathbf{x}$. In the next section we will discuss the reconstruction $s|\mathcal{N}$.

Reconstruction by polyharmonic splines

Polyharmonic splines were first established by Duchon [24] in 1977, where he extended the earlier work of Atteia [3] to higher dimensions. The polyharmonic splines were the first radial basic function interpolants to be actively researched in a theoretical context. In 2-D for the thin plate splines the problem was focussed on finding splines which minimised the bending energy of the infinite thin plate splines subject to the interpolation constraints. The polyharmonic splines are of the form,

$$\Phi_{d,k}(r) = \begin{cases} r^{2k-d}\log(r), & \text{if } d \text{ is even,} \\ r^{2k-d}, & \text{if } d \text{ is odd,} \end{cases} \tag{5.1}$$

where k is required to satisfy $2k > d$. We will take the order $m = k$ for $\Phi_{d,k}(r) \in CPD_d(m)$. This particular choice of m rather than the minimal choice $m = k - \lceil d/2 \rceil + 1$ means that, the Beppo Levi space $BL^k(\mathbb{R}^d)$ i.e.,

$$BL^k(\mathbb{R}^d) = \{f \in C(\mathbb{R}^d) D^\alpha u \in L^2(\mathbb{R}^d) \text{ for all } \mid \alpha \mid = k\} \subset C(\mathbb{R}^d) \tag{5.2}$$

is the optimal recovery space for the polyharmonic splines kernel $\Phi_{d,k}$. With this choice of $m = k$ the interpolant in equation (2.6) takes the form

$$s(\mathbf{x}) = \sum_{j=1}^{N} c_j \Phi_{d,k}(\|\mathbf{x} - \mathbf{x}_j\|) + \sum_{|\alpha|<k} d_\alpha \mathbf{x}^\alpha. \tag{5.3}$$

According to [24], the scattered data approximation by polyharmonic splines is optimal in its native reproducing kernel Hilbert space, as given by the Beppo Levi space. The seminal papers of Meinguet [56, 57, 58], shows that for a fixed point set $X \subset \mathbb{R}^d$ the interpolate s in (5.3) minimizes the energy

$$\mid f \mid^2_{BL^k} = \int_{\mathbb{R}^d} \sum_{|\alpha|=k} \binom{k}{\alpha} \left(D^\alpha f\right)^2 d(\mathbf{x}), \tag{5.4}$$

among all the functions f satisfying $s|_{\mathbf{X}} = f|_{\mathbf{X}}$ of the Beppo Levi space given by (5.2).

In other words, the reconstruction s minimizes the Beppo Levi energy functional $|.|_{(BL)^k}$ among all the recovery functions f in $BL^k(\mathbb{R}^d)$ i.e,

- $s(\mathbf{x}) = f(\mathbf{x})$ for all $\mathbf{x} \in \mathbf{X}$.
- $|s|_{(BL)^k} \leq |f|_{(BL)^k}$ for all $f \in BL^k(\mathbb{R}^d)$.

For instance, let us take m=d=2 the thin plate kernel will be $\phi_{2,2} = r^2 \log(r)$, with $\mathbf{x} = (x_1, x_2) \subseteq \mathbb{R}^2$ and for $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\} \subset \mathbb{R}^2$, the reconstruction will take the form

$$s(\mathbf{x}) = \sum_{j=1}^{N} \lambda_j \|\mathbf{x} - \mathbf{x}_j\|^2 \log(\|\mathbf{x} - \mathbf{x}_j\|) + d_0 + d_1 x_1 + d_2 x_2, \tag{5.5}$$

where the semi-norm is

$$|f|^2_{BL^2} = \int_{\mathbb{R}^2} \left(\frac{\partial^2 f}{\partial x_1^2} \right)^2 + 2 \left(\frac{\partial^2 f}{\partial x_1 x_2} \right)^2 + \left(\frac{\partial^2 f}{\partial x_2^2} \right)^2 dx_1 dx_2, \tag{5.6}$$

which reflects the bending energy for a thin plate of infinite extent. The motivation of the name thin plate comes from the fact that the resulting reconstruction minimized the bending energy $| \, . \, |^2_{BL}$ among all the interpolants in the Beppo-Levi space,

$$BL^2(\mathbb{R}^2) = \{ f : D^\alpha f \in L^2(\mathbb{R}^2) \text{ for all } | \, \alpha \, | = 2 \} \subset C(\mathbb{R}^2). \tag{5.7}$$

According to Iske [47] the advantages of the the polyharmonic splines can be viewed as,

- The polyharmonic splines reconstruction is well posed for arbitrary dimension and distribution of interpolation points.
- It is optimal in its corresponding Beppo Levi space $BL^m(\mathbb{R}^d)$ which is a relevant function space in nonlinear hyperbolic problems.
- The reconstruction by polyharmonic splines is meshfree which is flexible feature in terms of adaptivity. It is well suited for problems with shocks, steep gradients or singularities.
- The problems involves solving a square linear system which is small, if the number of nodes in $\mathbf{X}$ is small.
- The local approximation order of the polyharmonic splines kernel $\Phi_{d,m}$ is m, w.r.t C^m functions.

5.1.4 Natural splines as one-dimensional case of the polyharmonic splines

In one-dimensional case, polyharmonic splines becomes natural splines of order $2k$. ***Preposition***: Every natural cubic spline s has a representation of the form

$$s(x) = \sum_{i=1}^{N} \lambda_i \phi(|x - x_j|) + P(x),$$

where $x \in \mathbb{R}$, $\phi(r) = r^3, r \geq 0$, and $P \in \pi_1(\mathbb{R})$ satisfying the vanishing moments condition. More details of natural splines as RBF can be found in [83].

The properties of the natural splines can be seen as one-dimensional case for polyharmonic splines discussed in Section 5.1.3. We will now explain it briefly. For univariate case, i.e., $d = 1$, the polyharmonic spline takes the form

$$\Phi_{1,k} = r^{2k-1} \text{ where } k \geq 1,$$

which is natural splines of order $2k$. For the one-dimensional numerical experiments, the scattered data reconstruction has been done by $\Phi_{1,2} =\mid x - x_j \mid^3 + d_0 + d_1\mathbf{x}$ for all the common test problems. For the KdV equation, where the PDE has the third order spatial derivative we have used $\Phi_{1,3} =\mid x - x_j \mid^5 + d_0 + d_1 x + d_1 x^2$.

We can now give a precise definition of our error indicator.

Error indication

The error indicator can be considered as the function of the node set $\mathbf{X}$ i.e., $\eta : \mathbf{X} \to [0, \infty]$ which assigns a significance value to each node $\mathbf{x} \in \mathbf{X}$, $s_{\mathcal{N}_\mathbf{x}}$ is the polyharmonic spline reconstruction which matches the solution values at a node set $\mathcal{N}_\mathbf{x} \subset \mathbf{X}\backslash\mathbf{x}$ in a neighbourhood around $\mathbf{x}$ but not at $\mathbf{x}$ i.e.,

$$s_{\mathcal{N}_\mathbf{x}}(v) = u(v) \quad \forall v \in \mathcal{N}_\mathbf{x}. \tag{5.8}$$

The adaption relies on the error indicator defined as

$$\eta(\mathbf{x}) = |u(\mathbf{x}) - s_{\mathbb{N}_\mathbf{x}}(\mathbf{x})| \text{ for all } \mathbf{x} \in \mathbf{X}, \tag{5.9}$$

where u is the approximation to the true solution, obtained by the MQ-RBF.

The error indicator is the mean by which we check the variation of the true solution. It is done by calculating the pointwise deviation between approximate value of the solution $u(\mathbf{x})$ at $\mathbf{x}$ and the interpolated value at $\mathbf{x}$ using a local interpolant $s_{\mathbb{N}_\mathbf{x}}$ around $\mathbf{x}$. When $\mathbf{x}$ will lies in the region of high activity such as less regularity for u, around a discontinuity (shock) or around the steep gradients the error indicator $\eta(\mathbf{x})$ is expected to be big, whereas if it lies in the region where the solution is smooth $\eta(\mathbf{x})$ is expected to be small. Large values of $\eta(\mathbf{x})$ indicate that we need to flag more nodes in refinement context whereas smaller values are subject to coarsening.

Our reconstruction through the polyharmonic splines is local and the interpolant in (5.3) resulting from the interpolation conditions (5.8) under the vanishing moment conditions $\sum_{j=1}^{N} \lambda_j P(\mathbf{x}_j) = 0$ for all $P \in \pi_m^d$, will takes the form

$$\begin{pmatrix} A_{\phi,X} & P_\mathbf{X} \\ P_\mathbf{X}^T & \mathbf{O} \end{pmatrix} \begin{pmatrix} \lambda \\ \mathbf{d} \end{pmatrix} = \begin{pmatrix} u_\mathbf{X} \\ \mathbf{0} \end{pmatrix} \tag{5.10}$$

Our local reconstruction is obtained by interpolation through the seven nearest neighbours of the point of interest. We will obtain the set of nearest neighbours by using the RBF distance matrix

$$r = \begin{bmatrix} \| \mathbf{x}_1 - \mathbf{x}_1 \|_2 & \| \mathbf{x}_1 - \mathbf{x}_2 \|_2 & \dots & \| \mathbf{x}_1 - \mathbf{x}_N \|_2 \\ \| \mathbf{x}_2 - \mathbf{x}_1 \|_2 & \| \mathbf{x}_2 - \mathbf{x}_2 \|_2 & \dots & \| \mathbf{x}_2 - \mathbf{x}_N \|_2 \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ \| \mathbf{x}_N - \mathbf{x}_1 \|_2 & \| \mathbf{x}_N - \mathbf{x}_2 \|_2 & \dots & \| \mathbf{x}_N - \mathbf{x}_N \|_2 \end{bmatrix}$$

For instance, the first row of r contains distances of $\mathbf{x}_1$ with $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ we will first sort the row in ascending order and then pick sorted indexes $\mathcal{I} = \{2, ..., 8\}$. For the index $I = 1$ will represent the distance of the center with itself which will not be considered. From the center set $\mathbf{X}$ we will pick the centers on the index set $\mathcal{I}$. This set $\mathcal{N}_{\mathbf{x}_1}$ will then represent the set of nearest neighbours to $\mathbf{x}_1$. The same procedure can be repeated for every row which will give the set of nearest neighbours $\mathcal{N}_{\mathbf{x}_i}$ for $\mathbf{x}_i \in \mathbf{X}$

Since for local reconstruction problems, the number of N interpolation conditions is usually small also the dimension of the polynomial space π_m^d is small, the dimension of the resulting system (5.10) will be small. A comparison of the spectral condition number of polyharmonic spline with five different radial kernel functions can be found in [47].

5.1.5 Refine/ Coarse

The idea of refining and coarsening is as per the computational complexity of the model. We insert new nodes into the regions where the error indicator η in (5.9), gives us a bigger value than the predefined threshold i.e., we refine, where-as we remove nodes from $\mathbf{X}$ in regions where the value of η is smaller i.e., we coarse. This is to balance the approximation quality of the model against the required computational complexity. We will neither refine nor coarsen the end points. However for 2-D interpolation it is observed that largest errors occur at the boundary.

Refinement is 1-D context is, adding two nodes (left and right) around the node which lies in the region of high activity. Refinement in 2-D context is, adding the Voronoi points which shall be explained in Section 5.4. We did numerous experiments to maintain the balance between coarsening and refinement. We found that removing all the nodes which are less than coarsening threshold i.e., θ_{crs} is not an effective approach in our experiments. This will take away a whole bunch of information from the grid, and with less information adaptive method will become unstable. We therefore sort the residuals and remove a percentage i.e., residuals are ranked from worse to least worse and we only remove points from those nodes associated with a certain percentage at the

poor end of the scale. This percentage depends on the underlying problem. Coarsening a percentage is a crucial step in our adaptive method.

5.2 Adaptive RBF interpolation in the method of lines fashion

The adaptive RBF scheme for time-dependent PDEs is straight forward. We replace the spatial (boundary-value) derivatives in the PDE with algebraic approximations with radial basis functions with only one remaining independent variable, we have a system of ODEs that approximates the original PDE this approach is called the method of lines using RBFs. For instance, let us try to develop the adaptive scheme for the one-dimensional third order nonlinear KdV equation,

$$u_t + \epsilon u u_x + \mu u_{xxx} = 0. \tag{5.11}$$

The boundary and initial values can be derived from the exact solution given as

$$u(x,t) = 2\,\mathrm{sech}^2(x - 4t). \tag{5.12}$$

First we will perform the adaptive interpolation on the the initial condition

$$u(x,t_0) = 2\,\mathrm{sech}^2(x). \tag{5.13}$$

We will now explain the adaptive algorithm for time-dependent PDEs. To explain it, let us disuss our adaptive algorithm which will call Subroutine 1 for refining and coarsening in 1-D.

ALGORITHM 2. (Adaptive RBF for 1-D time-dependent PDEs)

INPUT: $\mathbf{X}$, θ_{ref} and θ_{crs}.

1. Calling Subroutine 1; gives adaptive nodes, values at nodes and shape parameters for given initial condition.

2. For $i = 1 : T_{final}$

   {

   (a) Compute; coefficient matrix, coefficients and differentiation matrices.

   (b) Advance the solution in time using ODE integrator and incorporate the boundary conditions.

   (c) Compute; the interpolant, using centers in step 1, computed coefficients in step(a) and updated solution in step (b).

   (d) Again calling Subroutine 1 compute adaptive; nodes, values at nodes and shape parameters, using the computed interpolant in step c.

   }

UNTIL The final time level achieved.

OUTPUT: Adaptive profile of the solution at final time level.

Step 2(c) will compute an interpolant for approximating the solution values at new added points. This step in the algorithm will create an evaluation matrix by using, step 1 for centers, step 2(a) for coefficients and step 2(b) for solution values. We can then use this evaluation matrix to find the approximate solutions at new added points in the domain. In our examples this interpolant is $MQ\text{-}RBF$.

Subroutine 1 which is called in step 1 and 2(d) is designed as

Subroutine 1. (Adaptive selection of nodes)

INPUT: Interpolant, θ_{ref}, θ_{crs}, N.

X= Distribution of nodes in domain into N equally spaced points.

While {

1. Compute the MQ-interpolant u.
2. For each $\mathbf{x} \in \mathbf{X}$ {
   - **(a)** Determine a set $\mathcal{N}_{\mathbf{x}} \subset \mathbf{X}$ of neighbours of $\mathbf{x}$.
   - **(b)** Compute s, the natural splines interpolant satisfying $s|_{\mathcal{N}} = u|_{\mathcal{N}}$ for each $\mathbf{x}$ but not at $\mathbf{x}$.
   - **(c)** Using s the local interpolant around $\mathbf{x}$, interpolate at $\mathbf{x}$.

   }
3. Compute the residuals $|u(\mathbf{x}_i) - s_{\mathcal{N}_{\mathbf{x}_i}}(\mathbf{x}_i)|$ for each $\mathbf{x_i}$.
4. Refine if $|u(\mathbf{x}_i) - s_{\mathcal{N}_{\mathbf{x}_i}}(\mathbf{x}_i)| > \theta_{ref}$.
5. Coarsen if $|u(\mathbf{x}_i) - s_{\mathcal{N}_{\mathbf{x}_i}}(\mathbf{x}_i)| < \theta_{crs}$
6. Adapt the shape parameters for the new set of adaptively selected point.
7. Interpolate at the new added points using interpolant in the input.

UNTIL Every residual is less than θ_{ref}.

}

OUTPUT: Adaptively selected centers $\mathbf{x} \subset \mathbf{X}$, approximate solution $u(\mathbf{x})$, adaptively adjusted shape parameters c.

Note: Subroutine 1; the input takes interpolant, which is the given initial condition at time $t_0 = 0$. At time $\tau = t_0 + m\triangle t$, where m is a positive integer. We will use the MQ interpolant to interpolate at the new added points. We will also input N, which is the number of nodes, for equally spaced distribution for adaptive algorithm to start

with. This distribution should not be too coarse or too fine. In step 3 the residuals will be computed as defined in Section 5.1.3. We will now discuss the method in detail.

The procedure starts with adaptive selection of nodes at initial stage. Calling Subroutine 1 by using initial condition as the "input interpolant". We will get a new set of nodes, values at nodes and adjusted shape parameters. We will now advance our solution in time. To march in time we will create the $(N+1) \times (N+1)$ differentiation matrices D_x and D_{xxx} for the first and the third derivatives in Equation (5.11). Let us say for the time dependent PDE method where the derivative may need to be evaluated thousands of times we will use the following way to evaluate the differentiation matrices,

$$D_n = \begin{bmatrix} \\ A_n \\ \\ \end{bmatrix} \begin{bmatrix} \\ A^{(-1)} \\ \\ \end{bmatrix},$$

with the entries $A_{ij} = \phi_{c_j} \parallel (\mathbf{x}_i - \mathbf{x}_j) \parallel\parallel$ and $A_n = \phi_{c_j} \frac{d^n}{dx^n}(\parallel \mathbf{x}_i - \mathbf{x}_j \parallel)$ where subscript n is referred to the order of the derivative. We can then solve the time stepping problem using any standard explicit/implicit ODE integrator. This will be done in step $2(b)$ of Algorithm 2. We have used the explicit Runge-Kutta scheme of order 4 in solving KdV equation and Allen-Cahn equation. In solving Burger's equation we will use stiff ODE integrator.

$$\begin{bmatrix} u_1 \\ u_2 \\ . \\ . \\ . \\ u_N \end{bmatrix}_t = \begin{bmatrix} \\ D_{(n)} \\ \\ \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ . \\ . \\ . \\ u_N \end{bmatrix}.$$

The boundary conditions will now be incorporated. Step $2(c)$ of Algorithm 2 will compute and update the "interpolant" to predict the approximate solution at the new added point. This interpolant is the input in Subroutine 1. This interpolant will be

computed using MQ-RBF for each time level. The errors will be compared at T_{final}.

The adaption depends on the developing features of the problem. For instance, for adapting the solution of the KdV equation first we did the adaption at every time level. Later we observed that computational cost can be saved by avoiding excessive cycles of adapting procedure, i.e., if the re-adaption occur after g time steps, i.e., at $\tau = t_0 + m\triangle t$, where m is positive integer. The selection of m depends upon the underlying problem. Small values will make the algorithm expensive and large values are not recommended either since, the solution may change its profile before the adaption occurs. At time τ we will use the same centers at $t = t_0$, but using the updated solution $[u_1^\tau, ..., u_N^\tau]^T$ and we will re-adapt by interpolating this updated solution. This will gives us a new set of RBF centers and shape parameters that capture the profile of the solution at $t = \tau$. This process is repeated with values at a new set of RBF centers for the next m time steps. The procedure will stop when we arrive at desired final time $t = T_{final}$.

5.3 Numerical experiments for time-dependent PDEs

We will now present numerical results for one-dimensional time-dependent PDEs with an extension of the method for two-dimensional interpolation problem. The adaptive method applied for the numerical experiments is described in Section 5.2. The thresholds used for refining and coarsening are denoted by θ_{ref} and θ_{crs} respectively. The choice of the thresholds is problem dependent. We will be using the classical Runge-Kutta (RK4) method for integrating in time unless otherwise stated. All the numerical experiments are run in MATLAB on Windows 7 system running at 3.10 Ghz with 8 GB memory.

5.3.1 Korteweg-de Vries (KdV) propagation of a single soliton

In this example, we study a single soliton solution of equation (5.11) with an exact solution given by (5.13). We choose $\epsilon = 6$ and $\mu = 1$ and the time step $\Delta t = 0.001$. The initial condition $u(x, 0)$ and boundary functions $f(t)$ and $g(t)$ can be obtained from the

exact solution in (5.13).

In order to compare the numerical results we will use the parameters used in [74]. The computational domain is $[-10, 40] \times [0, 5]$. The adaptive profiles of the single soliton for $t = 0, 1, 2, 3, 4, 5$ can be seen in Figures (5.2)-(5.4). The thresholds for refining $\theta_{ref} = 10^{-3}$ and for coarsening $\theta_{crs} = 10^{-6}$. The method is advanced in time using the classical RK4 method. Initially we started with 70 equally spaced points. For the one-dimensional problems we have used the polyharmonic splines which is the natural spline of order $2k$. For instance, for KdV equation the polyharmonic spline will be of the form

$$\phi_{1,k} = r^{2k-1} \tag{5.14}$$

with the order $m = k \geq 1$. In our examples we have used $k = 3$ which will be of the form

$$\phi_{1,3} = |x - x_j|^5 + d_0 + d_1 x + d_2 x^2. \tag{5.15}$$

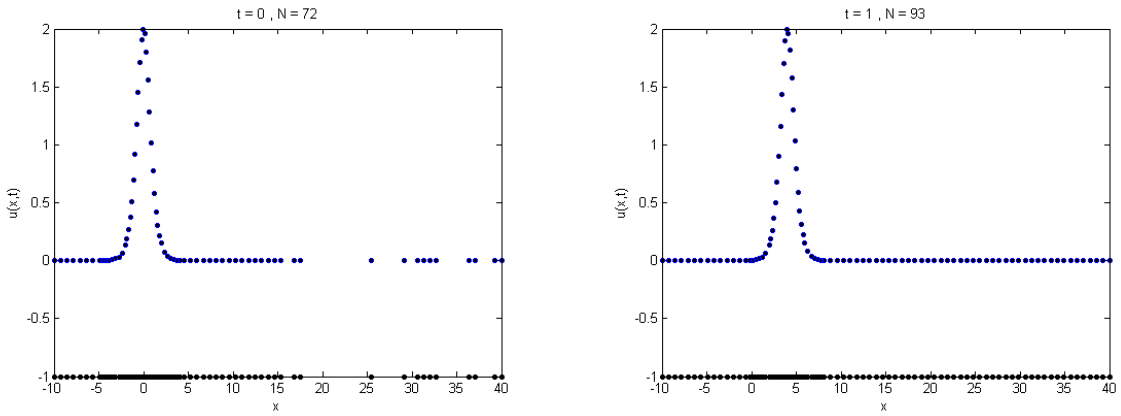


Figure 5.2: Left: Initial adaptive discretization at time $t = 0$, Right: Soliton moving from left to right, adaptive solution at time $t = 1$.

On the initial discretization of nodes, we adaptively select the nodes and the shape parameters $t_0 = 0$ (Figure 5.2). We then marched in time to obtain the solution at time $t = t_0 + \Delta t$. Once the solution is obtained we will again adapt the solution to obtain a new set of adaptively selected nodes and shape parameters. The procedure will continue after every 3 time steps and will stop when the final time arrives.

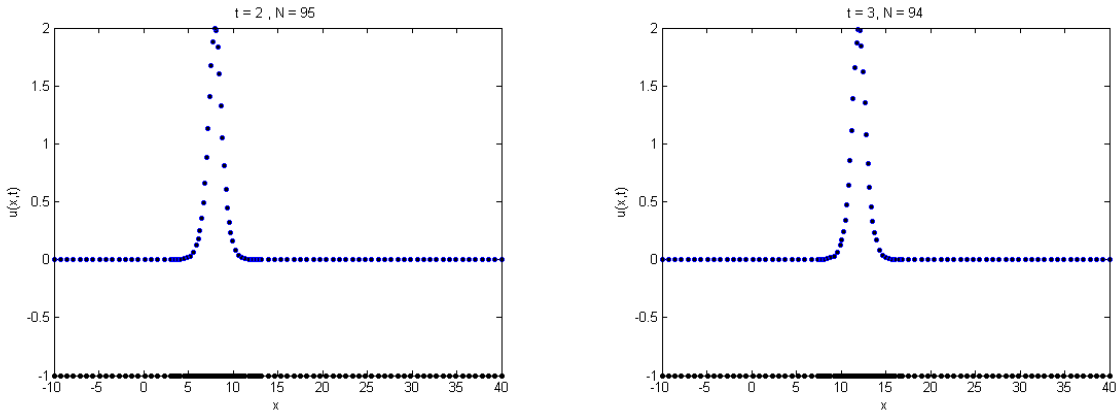


Figure 5.3: The profile of the adaptive solution at time $t = 2$ and $t = 3$

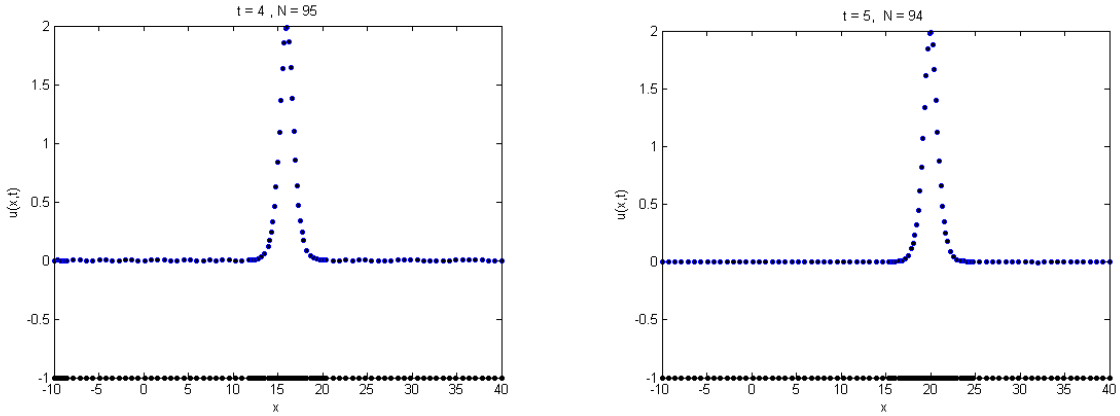


Figure 5.4: The profile of the adaptive solution at time $t = 4$ and $t = 5$

In 1-D experiments we adapted the shape parameters in adjustments with the centers. The MQ is flatter for smaller values of c and have a localized feature for bigger values. In the region of high activity we will use c be local which will makes the MQ peaked. We require center dependent shape parameters because the problems we are dealing with, are time dependent and the profile of the solution changes in time from smooth to steep. The balance between refining and coarsening is kept by removing 30%of the sorted best errors.

The adaptive method uses almost half the number of points than the non-adaptive method for the KdV (see Table 5.2). The performance of the method is compared using the maximum error, $\| U^N - u \|_\infty = \max_{1 \le i \le N} | U_i^N - u_i |$ and the root mean square

(RMS)= $\sqrt{\frac{\sum_{i=1}^{N}(U_i^N - u_i)^2}{N}}$. The adaptive method is shifting the points with increasing time, which is due to the soliton solution of the problem.

Adapting the solution at every time step can be expensive and hence avoided. We can expect inaccuracy or instability if we readapt, after the solution changes its profile. Hence we adapted the solution for KdV equation after every 3 time steps which saved the computational time and give accurate results Table (5.1). However adapting solution after 4 time steps gives maximum error $1.0E-2$ and RMS $3.5E-3$. This choice of adapting every 3 time steps is due to to our experiment, i.e., adapting the solution at every 5 time steps gives us unstable results. The unstable results are due to the reason that the solution changes its profile before the adaption occur.

Table 5.1: Comparison of Adaptive and Non-Adaptive methods for the single soliton at final time $t = 5$

Adaptive/Non-Adaptive RBF	No: of nodes	Max-Error	RMS errors
MQs	151	$5.9E-3$	$3.0E-3$
Gaussian	151	$1.2E-3$	$5.8E-4$
IMQs	251	$3.9E-3$	$1.2E-3$
Adaptive MQ	94	$3.1E-3$	$1.3E-3$

The goal of our adaptive method is to obtain a numerical solution in such a way that the residuals are less than a predefined threshold using smallest number of DoF. Table 5.1 is a comparison for KdV equation on uniform node distribution of nodes and adaptive distribution. We can see that our method solved the problem with smallest number of DoF with almost the same accuracy.

5.3.2 Interaction of two solitons

This example is studying the interaction of the two solitons solution of Equation (5.11) with $\epsilon = 6$ and $\mu = 1$. The initial condition is:

$$u(x, t_0) = 12\frac{3 + 4\cosh(2x - 8t_0) + \cosh(4x - 64t_0)}{(3\cosh(x - 28t_0) + \cosh(3x - 36t_0)^2}, \tag{5.16}$$

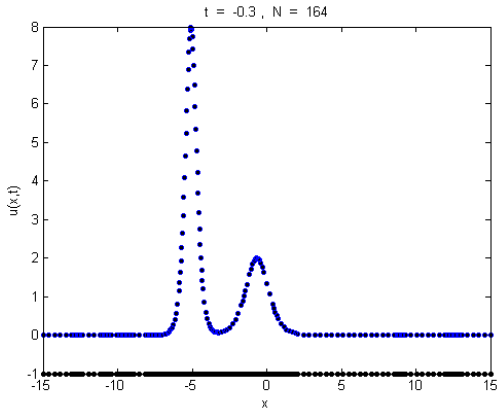
where t_0 is the initial time. The exact solution of (5.11) is given by

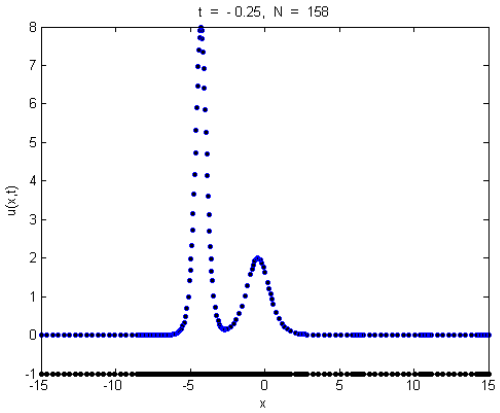
$$u(x,t) = 12\frac{3 + 4\cosh(2x - 8t) + \cosh(4x - 64t)}{(3\cosh(x - 28t) + \cosh(3x - 36t)^2}. \tag{5.17}$$

The boundary functions f(t) and g(t) can be obtained from the exact solution.

The computational domain is $[-15, 15] \times [-0.3, 0.3]$. The profile of interaction of the two solitons shows that the adaptive method is able to track the developing features of the profile of the solution. We have recorded the results for $t = -0.3$, -0.25, -0.2, -0.15, -0.1, -0.05, 0, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3 in Figures (5.5)-(5.7)

We started the adaptive interpolation with 80 equally spaced points where the refining threshold $\theta_{ref} = 10^-3$ whereas the coarsening $\theta_{crs} = 10^-7$. To maintain the balance in refining and deleting the number of points we coarsened 70% of the sorted errors. We solved this problem with MATLAB built-in ode15s ODE solver with $\Delta t = 0.0001$. The adaption occurred at every time level.





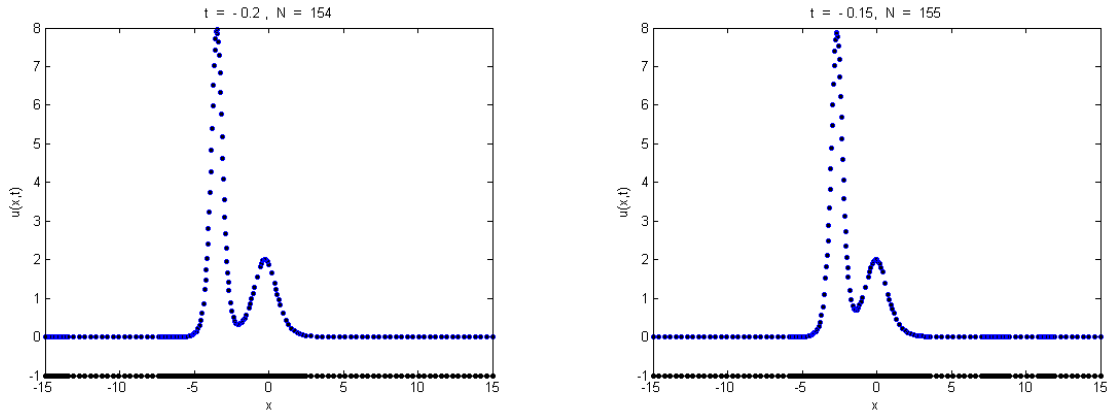
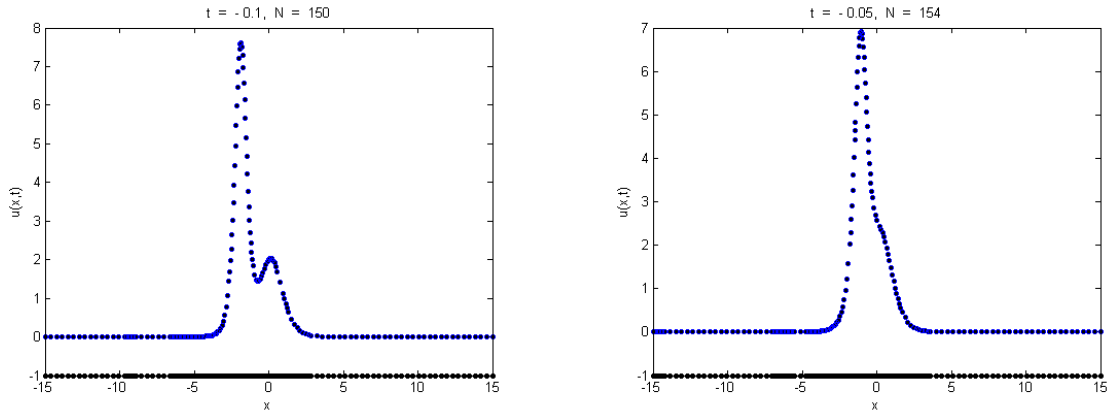


Figure 5.5: Interaction of two solitons moving from right to left, this profile is just before the interaction and recorded at time $t = -0.3, -0.25, -0.2$ and -0.15

Figure 5.5 is the profile of soltions moving from right to the left. We recorded this before the interaction actually happens. We can see that the adaptive algorithm is using a finer set of points where required.



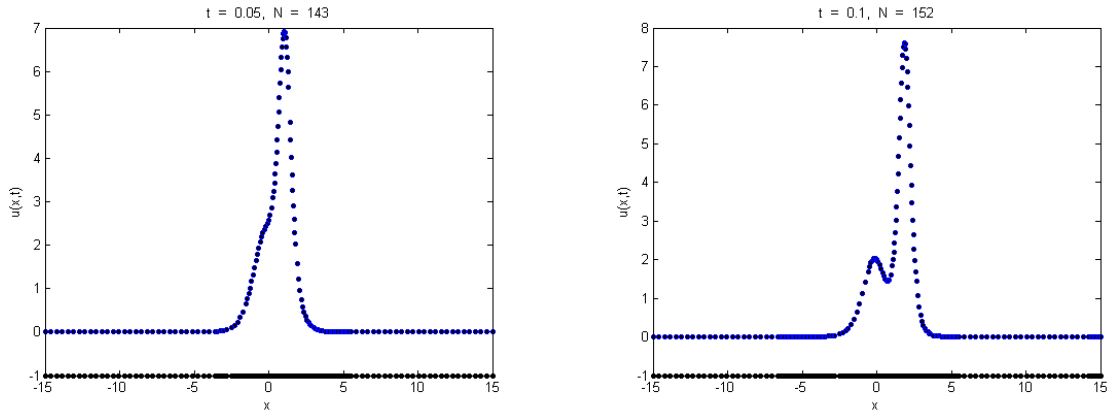
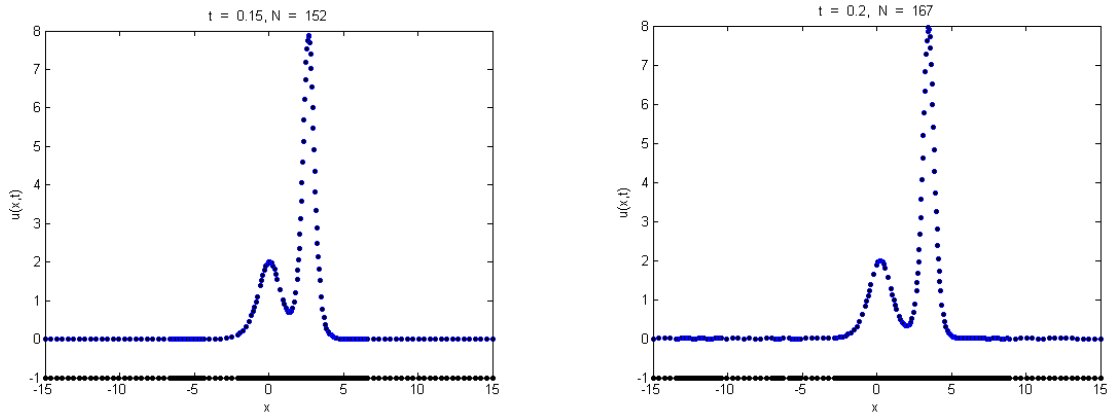


Figure 5.6: The smaller solitary wave is interacting with the larger wave at time $t = -0.1$, $t = -0.05$, $t = 0.05$ and $t = 0.1$.

Interaction of the two waves can be seen in Figure (5.6). We observed and recorded the results at the time levels when the interaction actually happened. We can see that at t=−0.05 the shorter wave passes through the bigger wave towards the left. At time t=0.1 it interacted with a total number of adaptively adjusted points $N = 152$.



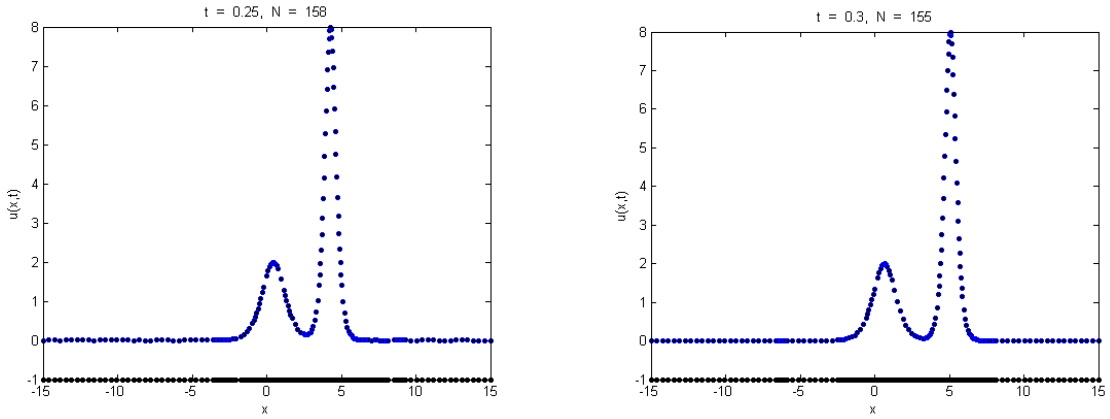


Figure 5.7: The waves retain to the original shape at $t = 0.25$, $t = 0.3$

Figure (5.7) shows that taller wave catches up, interacts with the shorter one and then passes towards the left. We can see that this is like almost no effect of interaction at all. Adaptive algorithm performed well to place a finer grid in a region of high variation. The process ends up with a total number of $N = 155$ adaptively distributed points.

Table 5.2: Comparison of Adaptive and Non-Adaptive methods for the of interaction of two solitons at final time $t = 0.3$

Adaptive/Non-Adaptive RBF	Degrees of Freedom	Max-Error	RMS errors
MQs	201	$9.5E-3$	$4.7E-3$
Gaussian	201	$1.4E-3$	$5.4E-4$
IMQs	201	$3.4E-3$	$5.7E-3$
Adaptive MQ	155	$4.9E-3$	$1.5E-3$

The adaptive method captured the developing features the solution at all times. In Table (5.2) we presented a comparison with the non-adaptive RBF method [74].

5.3.3 Burgers equation

Burgers equation is a nonlinear advection-diffusion problem [14]. It is defined as

$$\epsilon u_{\mathbf{xx}} - u u_{\mathbf{x}} = u_t, \tag{5.18}$$

where $\epsilon \geq 0$ is a given parameter and will be chosen such that(5.18) develops a shock. The solution then exhibits moving fronts that can be made arbitrarily sharp by decreas-

ing the kinematic viscosity ϵ as a coefficient of the second-derivative uu_{xx} term which is the dissipation term.

Burgers equation appears in various fields of applied mathematics. The applications can be seen in modelling of fluid dynamic, shock wave formation, traffic flow, turbulence, boundary layer behaviour. Initially the Burgers equation was introduced to describe turbulence in one space dimension.

We will now consider this moving front problem given by (5.18) which is a common test problem for adaptive methods [23] and references therein. The initial condition is

$$u(\mathbf{x}, 0) = \sin(2\pi\mathbf{x}) + \frac{1}{2}\sin(\pi\mathbf{x}), \tag{5.19}$$

with the boundary conditions

$$u(0, t) = u(1, t) = 0. \tag{5.20}$$

The boundary conditions are taken to be zero which diminish the amplitude of the wave with increasing time. Here $u(\mathbf{x}, t)$ is a wave that generates a steep front of width $O(\epsilon)$ and moves towards $\mathbf{x} = 1$. This is one of the principal reasons which makes Burger's equation a stringent test problem. The solution $u(\mathbf{x}, t)$ steepens with increasing time t and become difficult to resolve spatially.

The computational domain we are using is $(0, 1) \times [0, 1]$ with parameter $\epsilon = 10^{-}3$. Solution is advanced in time using the MATLAB built-in stiff ode-solver ode15s with $\Delta t = 0.01$. We have given the Jacobian matrix in *odetset* for the stiff solver *ode*15*s* to work faster. The thresholds $\theta_{ref} = 10^{-}5$ and $\theta_{crs} = 10^{-}8$. The method starts initially with 13 equally spaced nodes and the adaption occurs at every time level until the final time arrives. All the residuals becomes less than 10^{-5} with a total number of 155 adaptively selected nodes in the domain. The coarsening percentage for this problem is 20% of the sorted array of best errors.

The polyharmonic splines $\phi_{1,k} = r^{2k-1}$ reconstruction for the Burgers equation will

take the form

$$\phi_{1,2} = |x - x_j|^3 + d_0 + d_1 x. \tag{5.21}$$

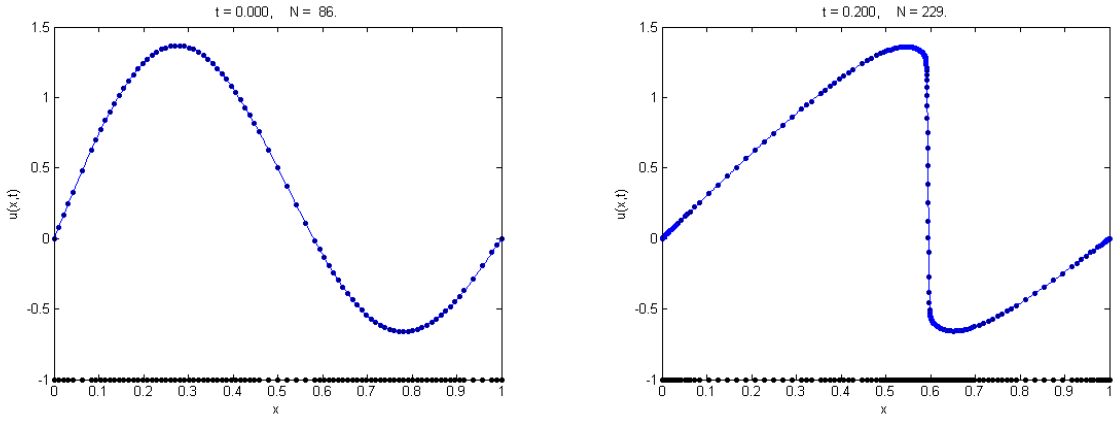


Figure 5.8: Adaptive RBF method for the Burgers equation at time $t = 0.4$ and $t = 0.6$

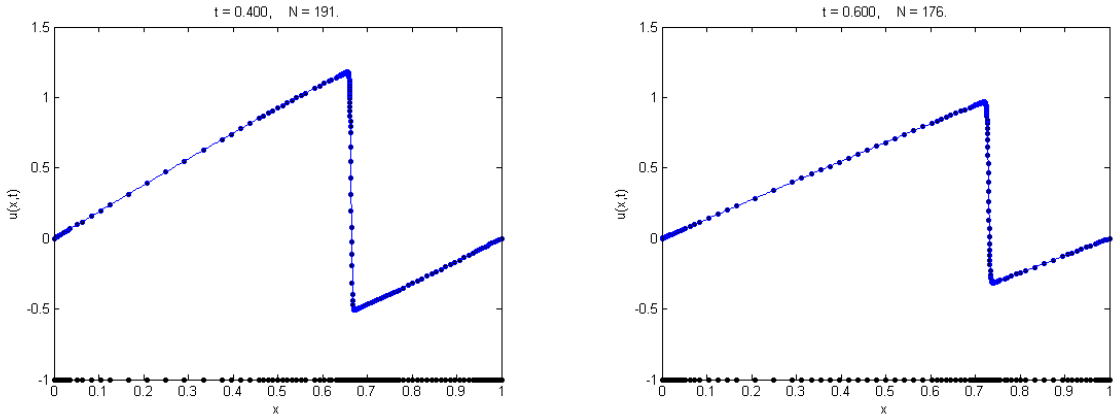


Figure 5.9: Adaptive RBF method for the Burgers equation at time $t = 0.4$ and $t = 0.6$

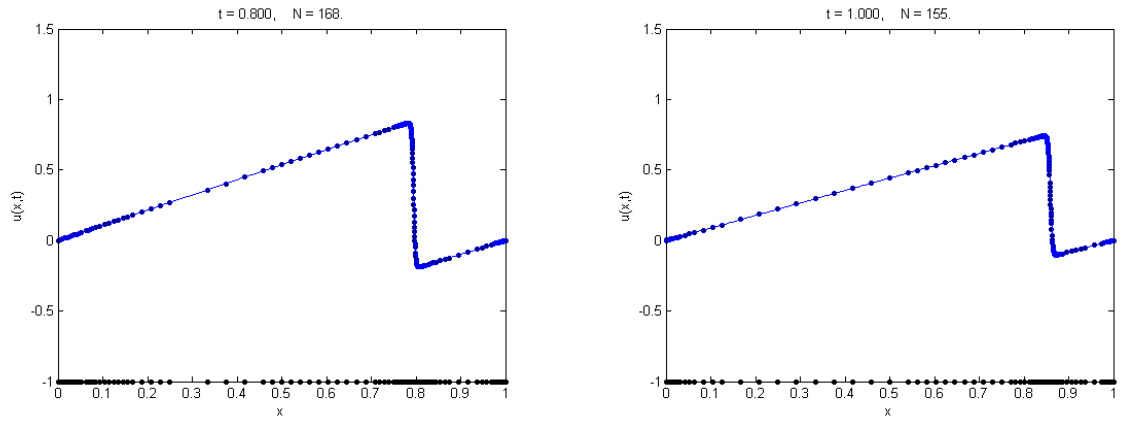


Figure 5.10: Adaptive RBF method for the Burgers equation at time $t = 0.6$ and $t = 0.8$

For the given smooth initial condition equation (5.19) the solution is moving towards 1, and generating a steep moving front. Due to the zero boundary conditions solution decays with time. In Figures (5.8)-(5.10) we can observe that our adaptive method is performing well.

5.3.4 Allen-Cahn Equation

The Allen-Cahn equation is defined as

$$u_t - u(1 - u^2) = \nu u_{xx}, \tag{5.22}$$

with the initial condition

$$u(x, 0) = 0.6x + 0.4 \sin[\frac{\pi}{2}(x^2 - 3x - 1)], \tag{5.23}$$

and the boundary conditions

$$u(-1, t) = -1, u(1, t) = 1. \tag{5.24}$$

In this experiment we have used parameter $\nu = 10^{-6}$ and the computational domain $[-1, 1] \times [0, 8.25]$. The adaption occurs every 15 time steps since we observed that the

solution is not changing its profile rapidly. We started the algorithm with 30 initially equally distributed points in the domain. The thresholds are $\theta_{crs} = 10^{-5}$ and $\theta_{ref} = 10^{-3}$. In this experiment we coarsened 30 percent of the residuals. Fourth order Runge-Kutta method is used for time stepping and the step size we used is 0.01.

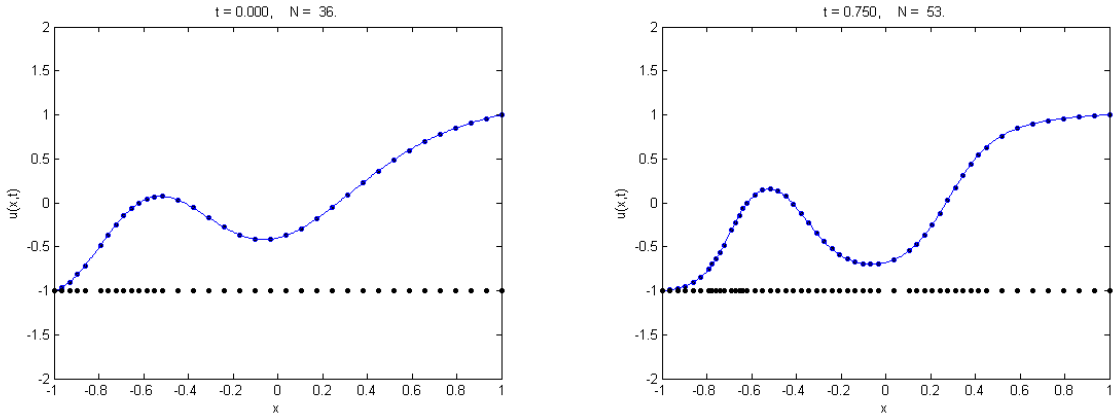


Figure 5.11: Adaptive solution for the Allen-Cahn equation at at time $t = 0$, and $t = 0.75$.

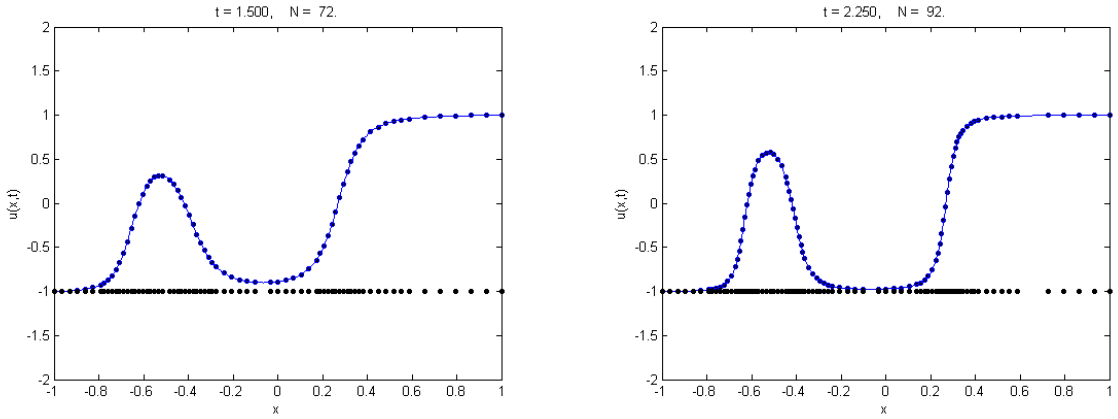


Figure 5.12: Adaptive solution for the Allen-Cahn at time $t = 1.5$ and $t = 2.25$.

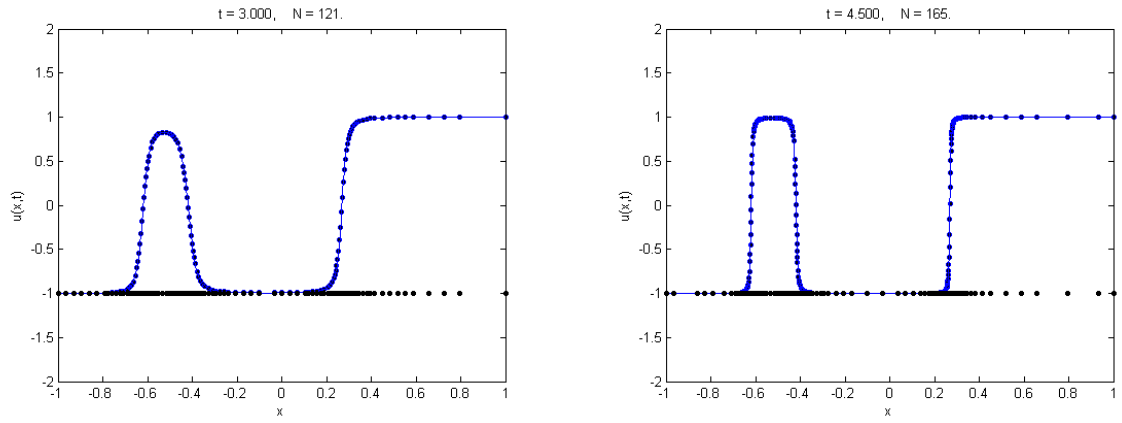


Figure 5.13: Adaptive solution for the Allen-Cahn at time $t = 3$ and $t = 4.5$.

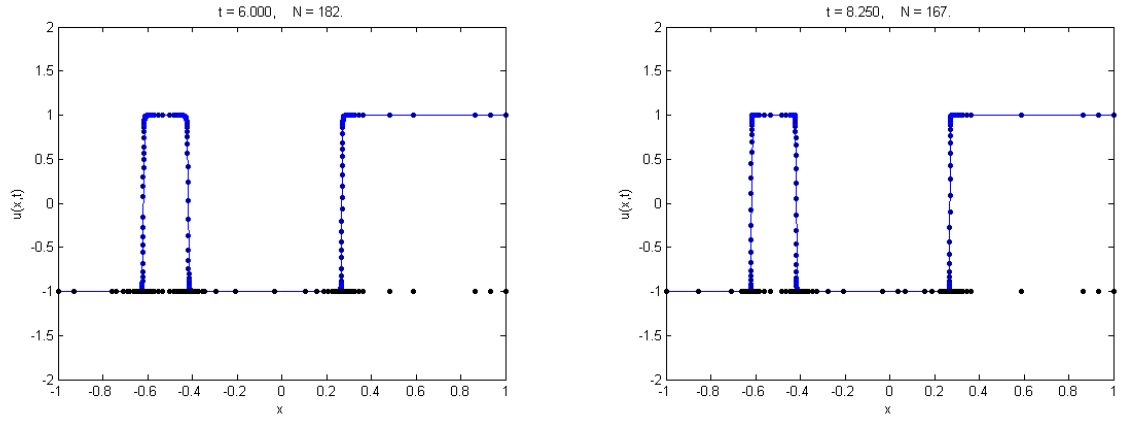


Figure 5.14: Adaptive solution for the Allen-Cahn at time $t = 6$ and at final time $t = 8.25$, it is evident that the nodes are clustering around steep gradients.

5.4 Interpolation in 2D

For the two-dimensional case we will consider the Franke-type function with the computational domain $[-1, 1]^2$. The function is defined by,

$$f(x, y) = e^{-0.1(x^2+y^2)} + e^{-5((x-0.5)+(y-0.5)^2)} + e^{-15((x+0.2)^2+(y+0.4)^2)} + e^{-9((x+0.8)^2+(y-0.8)^2)}. \tag{5.25}$$

The idea of adaptive method is straight forward in extension for 2-D interpolation problem. However, refining in 2-D require sophisticated algorithms specifically for points lying on the boundary.

5.4.1 Refinement in 2-D

Voronoi diagram is defined as

Definition 5.1 ***Voronoi Diagram*** *Suppose that* $\mathbf{X} \subset \mathbb{R}^d$ *is a finite point set in* $\mathbb{R}^d$. *Then, for any* $\mathbf{x} \in \mathbf{X}$ *, the point set*

$$V_{\mathbf{X}}(\mathbf{x}) = \{y \in \mathbb{R}^d) :\| y - \mathbf{x} \| = \min_{\overline{\mathbf{x}} \in X} \| y - \overline{\mathbf{x}} \|\} \subset \mathbb{R}^d)$$

is said to be the Voronoi tile of $\mathbf{x}$*, and the collection* $\{V_{\mathbf{X}}(\mathbf{x})\}_{\mathbf{x}\in\mathbf{X}}$ *of Voronoi tiles is called the Voronoi diagram of* $\mathbf{X}$.

The Voronoi tile $V_{\mathbf{X}}(\mathbf{x})$ is a non empty, closed and convex polyhedron, whose vertices are called *Voronoi points*. Construction of the Voronoi diagram can be found in [63].

Adding the Voronoi points i.e., $V_{\mathbf{X}}(\mathbf{x})$ in the refining context, for a point $\mathbf{x}$ is due to the local error estimates of the polyharmonic splines [45]. Bound of the pointwise error $|u(\mathbf{x}) - s(\mathbf{x})|$ where $u(\mathbf{x})$ is a sufficiently smooth function and $s(\mathbf{x})$ is the polyharmonic spline interplant satisfying $s|_{\mathcal{N}} = u|_{\mathcal{N}}$, is given by the estimate

$$|u(\mathbf{x}) - s(\mathbf{x})| \leq C \cdot h_{\mathcal{N},\rho}^{k-d/2}(\mathbf{x}), \tag{5.26}$$

where C is a constant depending on u, and $\rho > 0$ is radius and

$$h_{\mathcal{N},\rho}(\mathbf{x}) = \sup_{\|y-x\|<\rho} d_{\mathcal{N}}(y)$$

is the local fill distance of $\mathcal{N}$ around $\mathbf{x}$.

In order to refine around $\mathbf{x} \in \mathbf{X}$ the distance function $d_{\mathcal{N}} = \min_{\mathbf{x}\in\mathcal{N}} \| \cdot - \mathbf{x} \|$ shall be reduced for a reduction in estimate (5.26). The distance function $d_{\mathcal{N}}$ is convex on $V_{\mathbf{X}}(\mathbf{x})$ for any $\mathbf{x} \in \mathcal{N}$ another observation is that $d_{\mathcal{N}}$ has local maxima at the Voronoi points in the tile $V_{\mathbf{x}}$. This give us an understanding to add Voronoi points $V_{\mathbf{X}}(\mathbf{x})$, for a point $\mathbf{x}$ which, lies in the region of high variation. Adding $V_{\mathbf{X}}(\mathbf{x})$ will reduce the local error (5.26) around $\mathbf{x}$.

Refinement: A node $\mathbf{x} \in \mathbf{X}$ is refined by the insertion of the vertices of its Voronoi tile, i.e., the Voronoi points $V_{\mathbf{X}}(\mathbf{x})$ into the node set $\mathbf{X}$.

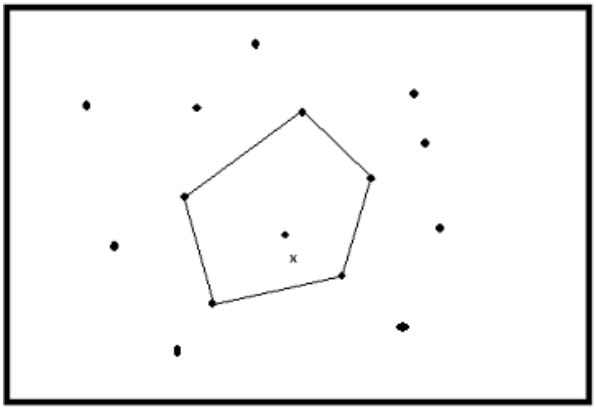


Figure 5.15: Voronoi tile for the point $\mathbf{x}$

Coarsening: A node $\mathbf{x} \in \mathbf{X}$ is *coarsened* by its removal from the current node set $\mathbf{X}$, i.e., $\mathbf{X}$ is modified by replacing the $\mathbf{X}$ by $\mathbf{X}/\mathbf{x}$.

We will now present the adaptive algorithm for 2-D interpolation problem.

5.4.2 Adaptive algorithm

We will now discuss the algorithm for interpolation in 2-D.

ALGORITHM 3. Adaptive RBF for 2-D interpolation.

INPUT: $\mathbf{X}$, θ_{ref} and θ_{crs}.

Repeat {

1. Compute the MQ-interpolant u.
2. For each $\mathbf{x} \in \mathbf{X}$ {
   - **(a)** Determine a set $\mathcal{N}_{\mathbf{x}} \subset \mathbf{X}$ of neighbours of $\mathbf{x}$.
   - **(b)** Compute s, the polyharmonic splines interpolant satisfying $s|_{\mathcal{N}} = u|_{\mathcal{N}}$ for each $\mathbf{x}$ but not at $\mathbf{x}$.
   - **(c)** Using s the local interpolant around $\mathbf{x}$, interpolate at $\mathbf{x}$.

   }
3. Compute the residuals $|u(\mathbf{x}_i) - s_{\mathcal{N}_{\mathbf{x}_i}}(x_i)|$ for each $\mathbf{x_i}$.
4. Calling Subroutine 2 for refinement.

UNTIL Every residual is less than θ_{ref}.

}

OUTPUT: A new set of adaptively selected set of centers.

Algorithm 3 will call Subroutine 2 and will do the adaptive interpolation for Frank's function given in 5.25. We will now give the Subroutine 2 for the refinement in 2-D.

Subroutine 2. (Voronoi points of interior and boundary nodes)

INPUT: **X**, residuals.

1. Find the Voronoi points.
2. Find index i.e., I=find(residuals$>\theta_{ref}$).
3. If I$\in \Omega = \mathbf{X} \subset \mathbf{R}^2$,
   - **(a)** Select the Voronoi points on such indices.
4. If I$\in \partial\Omega \subset \mathbf{X}$,,
   - **(a)** Select boundary points and the Voronoi points on such indexes.
   - **(b)** Remove the infinite ray from the Voronoi points
   - **(c)** Compute the inersection of the Vornoi diagram with the bounded computational domain.
5. Combine the interior and boundary points with their corresponding Voronoi points.

}

OUTPUT: Adaptively selected centers.

The polyharmonic spline reconstruction for two-dimensional problem, the equation (5.1) will takes the form $\phi_{2,k} = r^{2k-2}log(r)$ of order k. In our experiment we have used $k = 2$ which reduces the above as

$$\phi_{2,2} = \| \mathbf{x} - \mathbf{x}_j \|^2 \log(\mathbf{x} - \mathbf{x}_j) + d_1 + d_2 x_1 + d_3 x_2,$$

where x_1 and x_2 are the coordinates of the of $\mathbf{x} = (x_1, x_2) \in \mathbb{R}^2$. The thresholds for refining and coarsening are 10^{-2} and 10^{-5} respectively. The refining will be adding the Voronoi points of a point $\mathbf{x}$ which lies in the region of high variation and coarsening will be the removal of that point from the domain.

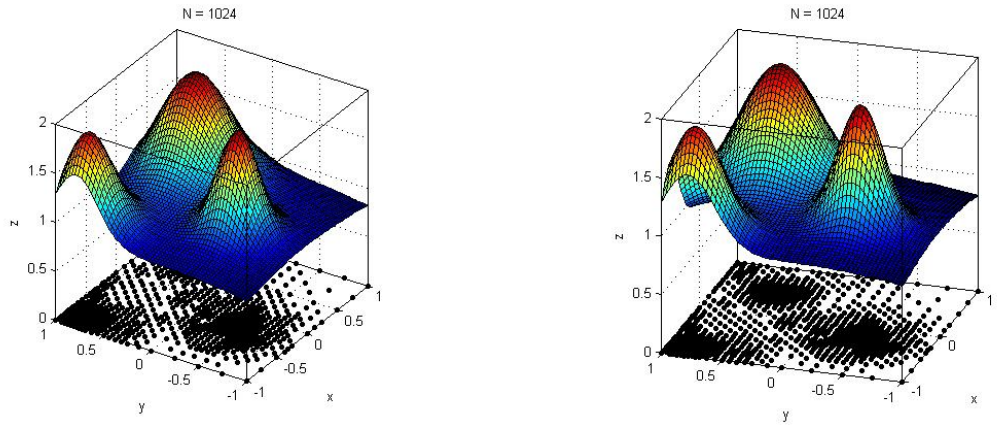


Figure 5.16: Adaptive interpolation for 2-D interpolation problem

Figure 5.16 shows that for the Frank's function the method is adapting the steep gradients efficiently and the interpolation process ends with 1024 adaptively selected nodes. The RMS is 1.2E-3 and the maximum error is recorded as 7.6E-3. We believe that the algorithm gives us some confidence for its improvement and extension to two-dimensional PDEs.

Chapter 6

Conclusions and future work

We developed a new adaptive method for the one-dimensional KdV equation, the goal of the method was to solve the problem with smallest number of degrees of freedom, i.e., with the smallest number of grid points. The extension of the method for 2-D interpolation provided some confidence to extend the method for two dimensional PDEs. In this chapter some findings and achievements of the adaptive method for time dependent PDEs using radial basis functions have been discussed where the RBFs are suitable for reconstruction on both the structured and unstructured grids. The chapter ends with some suggestions and recommendations, upon the completion of the thesis.

6.1 Conclusions

The adaptive spectral methods are preferred for problems with steep gradients, sharp corners, moving fronts and with singularities. We developed the adaptive scheme for single soliton and interaction of two solitons solution of the third order nonlinear KdV equation. A large number of nonlinear dispersive systems can be described by the KdV equation or one of its modified form. Some interesting numerical results are found during this journey, but keeping in mind that the RBF is relatively new subject many of its behaviours still require rigourous justifications, in this context the theoretical analysis is far from complete. Our method uses MQ-RBFs to approximate the solution, which

is due to its spectral convergence. For the indication of the error, we reconstruct the solutions in a local neighbourhood around every node using the polyharmonic splines which is a popular choice for adaptive schemes [6]. For instance, $\eta(\mathbf{x}) = |u(\mathbf{x}) - s_{\mathcal{N}_\mathbf{x}}(\mathbf{x})|$ for all $\mathbf{x} \in \mathbf{X}$, where $u(\mathbf{x})$ is the MQ approximation at the point and $s_{\mathcal{N}_\mathbf{x}}(\mathbf{x})$, is the local reconstruction around $\mathbf{x}$ but not at $\mathbf{x}$. The polyharmonic splines for the 1-D case becomes the natural splines, for the KdV equation we have used $\phi_{1,3} = |x - x_i|^5 + d_0 + d_1 x + d_2 x^2$.

Our goal was to obtain a numerical solution with less number of DoF as possible, by using an adaptive approach such that the errors are below a prescribed threshold. Solving the problem with less number of DoF can be efficient in saving the computer storage and adaption occur at a certain time level can be computationally less expensive.

The implementation of the schemes results in, solving the problem with almost half the number of the grid points than the non-adaptive scheme. The comparison can be seen in Table (5.1) for a single soliton and Table (5.2) for interaction of two solitons. The results are showing that our method is able to track the developing features in the solution profile. The method have shown promising results for the second order nonlinear difficult problems, i.e., the Burgers equation with the moving front solution and Allen-Cahn equation. The method is able to add more points in the region of high activity. The RBF collocation for the time-dependent PDEs in particular, the hyperbolic PDEs, are challenging for their time stability. We observed the CFL condition for KdV equation using the non-stationary RBFs, depend on the non-linear part and the shape parameter. Our 2-D interpolation problem for the Frank's function, we have used $\phi_{2,2} = \| \mathbf{x} - \mathbf{x}_i \|^2 \log(\| \mathbf{x} - \mathbf{x}_i \|) + d_1 + d_2\mathbf{x} + d_2\mathbf{x}$ which has been praised for several reasons. For instance, the scattered data approximation by polyharmonic splines is optimal in its native reproducing kernel Hilbert space, as given by the Beppo Levi space. Constructing the algorithm for refinement in 2-D is a motivation for the extension of the method for PDEs.

6.2 Outlook and future work

Our goal in this effort is to deliver the base of the method which may be useful for future research and applications. The utility of the RBFs is not quite evident for the 1-D problems, even though our results are satisfactory. We developed the algorithm for the 2-D interpolation problem and in future we will extend the idea, to solve 2-D time-dependent problems. The convergence analysis of the RBF with scattered data in a finite domain is a big concern as the RBF collocation methods use interpolation to approximate the solutions of PDEs. The use of center dependent shape parameters breaks the symmetry of the interpolation matrix $\mathbf{A}$ and the proof of its nonsingularity. This issue require a better understanding. In addition we require the CFL restriction for time dependent problems for both, the uniform and non-uniform distribution of grid points. This is an urgent question to be answered while solving time-dependent problems i.e., stability domain in terms of distributions of the nodes and shape parameters. Eigenvalue stability of the RBF differentiation matrices is another big concern. However this question needs to be answered theoretically for both, uniform and adaptive node distribution.

Bibliography

[1] M. J. H. Anthonissen. *Local defect correction techniques: analysis and application to combustion*. Eindhoven University of Technology, Eindhoven, 2001. Dissertation, Technische Universiteit Eindhoven, Eindhoven, 2001.

[2] U. M. Ascher. *Numerical methods for evolutionary differential equations*, volume 5 of *Computational Science & Engineering*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2008.

[3] M. Atteia. Fonctions "spline" et noyaux reproduisants d'Aronszajn-Bergman. *Rev. Française Informat. Recherche Opérationnelle*, 4(R-3):31–43, 1970.

[4] B. Batiha, M. S. M. Noorani, and I. Hashim. Application of variational iteration method to the generalized burger's-huxley equation. *CHAOS SOLITON FRACT*, 39:660–663, 2008.

[5] J. Behrens and A. Iske. Grid-free adaptive semi-Lagrangian advection using radial basis functions. *Comput. Math. Appl.*, 43(3-5):319–327, 2002. Radial basis functions and partial differential equations.

[6] J. Behrens, A. Iske, and M. Käser. Adaptive meshfree method of backward characteristics for nonlinear transport equations. In *Meshfree methods for partial differential equations (Bonn, 2001)*, volume 26 of *Lect. Notes Comput. Sci. Eng.*, pages 21–36. Springer, Berlin, 2003.

[7] J. Behrens, A. Iske, and Käserm M. Adaptive meshfree method of backward characteristics for nonlinear transport equations. In *Meshfree methods for partial differ-*

ential equations (Bonn, 2001), volume 26 of *Lect. Notes Comput. Sci. Eng.*, pages 21–36. Springer, Berlin, 2003.

[8] A. L. Bertozzi, M. P. Brenner, T. F. Dupont, and L. P. Kadanoff. Singularities and similarities in interface flows. In *Trends and perspectives in applied mathematics*, volume 100 of *Appl. Math. Sci.*, pages 155–208. Springer, New York, 1994.

[9] N. Bibi, S. I. A. Tirmizi, and S. Haq. Meshless method of lines for numerical solution of Kawahara type equations. *Appl. Math. (Irvine)*, 2(5):608–618, 2011.

[10] M. Bozzini, L. Lenarduzzi, M. Rossini, and R. Schaback. Interpolation by basis functions of different scales and shapes. *Calcolo*, 41(2):77–87, 2004.

[11] M. Bozzini, L. Lenarduzzi, and R. Schaback. Adaptive interpolation by scaled multiquadrics.

[12] M. D. Buhmann. *Radial basis functions: theory and implementations*, volume 12 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, Cambridge, 2003.

[13] M. D. Buhmann and N. Dyn. Spectral convergence of multiquadric interpolation. *Proc. Edinburgh Math. Soc. (2)*, 36(2):319–333, 1993.

[14] J. M. Burgers. A mathematical model illustrating the theory of turbulence. In *Advances in Applied Mechanics*, pages 171–199. Academic Press Inc., New York, N. Y., 1948. edited by Richard von Mises and Theodore von Kármán,.

[15] J. C. Butcher. *Numerical methods for ordinary differential equations*. John Wiley & Sons Ltd., Chichester, 2003.

[16] X. Chen and J. Jung. Matrix stability of multiquadric radial basis function methods for hyperbolic equations with uniform centers. *J. Sci. Comput.*, 51(3):683–702, 2012.

[17] A. H. D. Cheng, M. A. Golberg, E. J. Kansa, and G. Zammito. Exponential convergence and *h-c* multiquadric collocation method for partial differential equations. *Numer. Methods Partial Differential Equations*, 19(5):571–594, 2003.

[18] Y. Cheng and C. W. Shu. A discontinuous Galerkin finite element method for time dependent partial differential equations with higher order derivatives. *Math. Comp.*, 77(262):699–730, 2008.

[19] R. Courant, K. Friedrichs, and H. Lewy. ber die partiellen Differenzengleichungen der mathematischen Physik. *Mathematische Annalen*, 100:32–74, 1928.

[20] P. Cruza, M. Alvesb, F. D. Magalhesa, and A. Mendes. Solution of hyperbolic pdes using a stable adaptive multiresolution method. *Chemical Enginerring Science*, 58:1777–1792, 2003.

[21] O. Davydov and D. T. Oanh. Adaptive meshless centres and RBF stencils for Poisson equation. *J. Comput. Phys.*, 230(2):287–304, 2011.

[22] P. G. Drazin and R. S. Johnson. *Solitons: an introduction*. Cambridge Texts in Applied Mathematics. Cambridge University Press, Cambridge, 1989.

[23] T. A. Driscoll and A. R. H. Heryudono. Adaptive residual subsampling methods for radial basis function interpolation and collocation problems. *Comput. Math. Appl.*, 53(6):927–939, 2007.

[24] J. Duchon. Splines minimizing rotation-invariant semi-norms in Sobolev spaces. In *Constructive theory of functions of several variables (Proc. Conf., Math. Res. Inst., Oberwolfach, 1976)*, pages 85–100. Lecture Notes in Math., Vol. 571. Springer, Berlin, 1977.

[25] G. E. Fasshauer. *Meshfree approximation methods with MATLAB*, volume 6 of *Interdisciplinary Mathematical Sciences*. World Scientific Publishing Co. Pte. Ltd., Hackensack, NJ, 2007. With 1 CD-ROM (Windows, Macintosh and UNIX).

[26] E. Fermi, J. Pasta, and S. Ulam. Studies in nonlinear problems, i. *Los Alamos report LA 1940*, 1955.

[27] L. Ferracina and M. N. Spijker. Stepsize restrictions for total-variation-boundedness in general Runge-Kutta procedures. *Appl. Numer. Math.*, 53(2-4):265–279, 2005.

[28] A. T. Filippov. *The versatile soliton*. Birkhäuser Boston Inc., Boston, MA, 2000.

[29] N. Flyer and G.B. Wright. Transport schemes on a sphere using radial basis functions. *J. Comput. Phys.*, 226(1):1059–1084, 2007.

[30] B. Fornberg, T. A. Driscoll, G. Wright, and R. Charles. Observations on the behavior of radial basis function approximations near boundaries. *Comput. Math. Appl.*, 43(3-5):473–490, 2002. Radial basis functions and partial differential equations.

[31] B. Fornberg and J. Zuev. The Runge phenomenon and spatially variable shape parameters in RBF interpolation. *Comput. Math. Appl.*, 54(3):379–398, 2007.

[32] C. Franke and R. Schaback. Solving partial differential equations by collocation using radial basis functions. *Appl. Math. Comput.*, 93(1):73–82, 1998.

[33] C. S. Gardner, J. M. Greene, M. D. Kruskal, and R. M. Miura. Method for solving the korteweg-devries equation. *Phys. Rev. Lett.*, 19:1095–1097, Nov 1967.

[34] E. H. Georgoulis, P. Houston, and J. Virtanen. An *a posteriori* error indicator for discontinuous Galerkin approximations of fourth-order elliptic problems. *IMA J. Numer. Anal.*, 31(1):281–298, 2011.

[35] S. Gottlieb, D. Ketcheson, and C.W. Shu. *Strong stability preserving Runge-Kutta and multistep time discretizations*. World Scientific Publishing Co. Pte. Ltd., Hackensack, NJ, 2011.

[36] I. S. Greig and J. Ll. Morris. A hopscotch method for the Korteweg-de-Vries equation. *J. Computational Phys.*, 20(1):64–80, 1976.

[37] S. Haq, N. Bibi, S. I. A. Tirmizi, and M. Usman. Meshless method of lines for the numerical solution of generalized Kuramoto-Sivashinsky equation. *Appl. Math. Comput.*, 217(6):2404–2413, 2010.

[38] S. Haq, A. Hussain, and M. Uddin. RBFs meshless method of lines for the numerical solution of time-dependent nonlinear coupled partial differential equations. *Appl. Math. (Irvine)*, 2(4):414–423, 2011.

[39] S. Haq, A. Hussain, and M. Uddin. On the numerical solution of nonlinear Burgers'-type equations using meshless method of lines. *Appl. Math. Comput.*, 218(11):6280–6290, 2012.

[40] S. Haq, U. L Siraj, and M. Uddin. A mesh-free method for the numerical solution of the KdV-Burgers equation. *Appl. Math. Model.*, 33(8):3442–3449, 2009.

[41] I. Hashim, M. S. M. Noorani, and M. R. Said Al-Hadidi. Solving the generalized Burgers-Huxley equation using the Adomian decomposition method. *Math. Comput. Modelling*, 43(11-12):1404–1411, 2006.

[42] Y. C. Hon, R. Schaback, and X. Zhou. An adaptive greedy algorithm for solving large RBF collocation problems. *Numer. Algorithms*, 32(1):13–25, 2003.

[43] C. Hong, K. Li, and L. Wen-Jun. Numerical solution of pdes via integrated radial basis function networks with adaptive training algorithm. *Applied Soft Computing.*, 11:855–860, 2010.

[44] W. Hundsdorfer and M. N. Spijker. Boundedness and strong stability of Runge-Kutta methods. *Math. Comp.*, 80(274):863–886, 2011.

[45] A. Iske. *Multiresolution methods in scattered data modelling*, volume 37 of *Lecture Notes in Computational Science and Engineering*. Springer-Verlag, Berlin, 2004.

[46] A. Iske. Particle flow simulation by using polyharmonic splines. In *Algorithms for approximation*, pages 83–102. Springer, Berlin, 2007.

[47] A. Iske. On the construction of kernel-based adaptive particle methods in numerical flow simulation. *Springer Berlin Heidelberg*, 120:197–221, 2013.

[48] A. Jan. A computational meshless method for the generalized Burger's-Huxley equation. *Appl. Math. Model.*, 33(9):3718–3729, 2009.

[49] M. Javidi. A numerical solution of the generalized Burger's-Huxley equation by pseudospectral method and Darvishi's preconditioning. *Appl. Math. Comput.*, 175(2):1619–1628, 2006.

[50] E. J. Kansa. Multiquadrics—a scattered data approximation scheme with applications to computational fluid-dynamics. I. Surface approximations and partial derivative estimates. *Comput. Math. Appl.*, 19(8-9):127–145, 1990.

[51] E. J. Kansa. Multiquadrics—a scattered data approximation scheme with applications to computational fluid-dynamics. II. Solutions to parabolic, hyperbolic and elliptic partial differential equations. *Comput. Math. Appl.*, 19(8-9):147–161, 1990.

[52] D.J. Korteweg and G. de Vries. On the change of form of long waves advancing in a rectangular canal, and on a new type of long stationary waves. *Philos Mag*, Ser 5(39):422–443, 1895.

[53] E. Larsson and B. Fornberg. A numerical study of some radial basis function based solution methods for elliptic PDEs. *Comput. Math. Appl.*, 46(5-6):891–902, 2003.

[54] N. A. Libre, A. Emdadi, Shekarchi M. Kansa, E. J., and M. Rahimian. A fast adaptive wavelet scheme in RBF collocation for nearly singular potential PDEs. *CMES Comput. Model. Eng. Sci.*, 38(3):263–284, 2008.

[55] W. R. Madych and S. A. Nelson. Bounds on multivariate polynomials and exponential error estimates for multiquadric interpolation. *J. Approx. Theory*, 70(1):94–114, 1992.

[56] J. Meinguet. Basic mathematical aspects of surface spline interpolation. In *Numerische Integration (Tagung, Math. Forschungsinst., Oberwolfach, 1978)*, volume 45 of *Internat. Ser. Numer. Math.*, pages 211–220. Birkhäuser, Basel, 1979.

[57] J. Meinguet. An intrinsic approach to multivariate spline interpolation at arbitrary points. In *Polynomial and spline approximation (Proc. NATO Adv. Study Inst., Univ. Calgary, Calgary, Alta., 1978)*, volume 49 of *NATO Adv. Study Inst. Ser. C: Math. Phys. Sci.*, pages 163–190. Reidel, Dordrecht, 1979.

[58] J. Meinguet. Multivariate interpolation at arbitrary points made simple. *Z. Angew. Math. Phys.*, 30(2):292–304, 1979.

[59] C. A. Micchelli. Interpolation of scattered data: distance matrices and conditionally positive definite functions. *Constr. Approx.*, 2(1):11–22, 1986.

[60] R.M. Miura. The Korteweg-de Vries equation: a survey of results. *SIAM Rev.*, 18(3):412–459, 1976.

[61] R.M. Miura, C.S. Gardner, and M. D. Kruskal. Korteweg-de Vries equation and generalizations. II. Existence of conservation laws and constants of motion. *J. Mathematical Phys.*, 9:1204–1209, 1968.

[62] R. B. Platte and T. A. Driscoll. Eigenvalue stability of radial basis function discretizations for time-dependent problems. *Comput. Math. Appl.*, 51(8):1251–1268, 2006.

[63] F. P. Preparata and M. I. Shamos. *Computational geometry*. Texts and Monographs in Computer Science. Springer-Verlag, New York, 1985. An introduction.

[64] M. Remoissenet. *Waves called solitons*. Springer-Verlag, Berlin, third edition, 1999. Concepts and experiments.

[65] O. Roussel, K. Schneider, A. Tsigulin, and H. Bockhorn. A conservative fully adaptive multiresolution algorithm for parabolic PDEs. *J. Comput. Phys.*, 188(2):493–523, 2003.

[66] J. S. Russell. Reports on waves, 14th meeting of the british asscociation for the advancement of science. *John Murray, London.*

[67] J. M. Sanz-Serna and I. Christie. Petrov-Galerkin methods for nonlinear dispersive waves. *J. Comput. Phys.*, 39(1):94–102, 1981.

[68] S. A. Sarra. Adaptive radial basis function methods for time dependent partial differential equations. *Appl. Numer. Math.*, 54(1):79–94, 2005.

[69] S. A. Sarra. A numerical study of the accuracy and stability of symmetric and asymmetric RBF collocation methods for hyperbolic PDEs. *Numer. Methods Partial Differential Equations*, 24(2):670–686, 2008.

[70] J. Satsuma, M. Ablowitz, B. Fuchssteiner, and M. (Eds) Kruskal. Topics in soliton theory and exactly solvable nonlinear equations. *World scientific Singapore*, 1987.

[71] R. Schaback. Error estimates and condition numbers for radial basis function interpolation. *Adv. Comput. Math.*, 3(3):251–264, 1995.

[72] W. E. Schiesser and G. W. Griffiths. *A compendium of partial differential equation models.* Cambridge University Press, Cambridge, 2009. Method of lines analysis with Matlab.

[73] J. Shen. A new dual-Petrov-Galerkin method for third and higher odd-order differential equations: application to the KdV equation. *SIAM J. Numer. Anal.*, 41(5):1595–1619 (electronic), 2003.

[74] Q. Shen. A meshless method of lines for the numerical solution of KdV equation using radial basis functions. *Eng. Anal. Bound. Elem.*, 33(10):1171–1180, 2009.

[75] C.W Shu and S. Osher. Efficient implementation of essentially nonoscillatory shock-capturing schemes. *J. Comput. Phys.*, 77(2):439–471, 1988.

[76] J. Stewart. Positive definite functions and generalizations, an historical survey. *Rocky Mountain J. Math.*, 6(3):409–434, 1976.

[77] L.N. Trefethen. *Spectral methods in MATLAB*, volume 10 of *Software, Environments, and Tools*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2000.

[78] M. Uddin, S. Haq, and G. Qasim. A meshfree approach for the numerical solution of nonlinear sine-Gordon equation. *Int. Math. Forum*, 7(21-24):1179–1186, 2012.

[79] A. C. Vliegenthart. On finite-difference methods for the Korteweg-de Vries equation. *J. Engrg. Math.*, 5:137–155, 1971.

[80] L. Vrankar, E. J. Kansa, L. Ling, F. Runovc, and G. Turk. Moving-boundary problems solved by adaptive radial basis functions. *Comput. & Fluids*, 39(9):1480–1490, 2010.

[81] W. Y. Wang, Z. S. Zhu, and Y. K. Lu. Solitary wave sloutions of the generalized burger's-huxley equation. *J.Phys. A:Math. Gen*, 23:271–274, 1990.

[82] H. Wendland. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Adv. Comput. Math.*, 4(4):389–396, 1995.

[83] H. Wendland. *Scattered data approximation*, volume 17 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, Cambridge, 2005.

[84] R. Winther. A conservative finite element method for the Korteweg-de Vries equation. *Math. Comp.*, 34(149):23–43, 1980.

[85] Z. M. Wu. Compactly supported positive definite radial functions. *Adv. Comput. Math.*, 4(3):283–292, 1995.

[86] J. Xue and G. Liao. Least-squares finite element method on adaptive grid for PDEs with shocks. *Numer. Methods Partial Differential Equations*, 22(1):114–127, 2006.

[87] N. J. Zabusky and M. D. Kruskal. Interaction of "solitons" in a collisionless plasma and the recurrence of initial states. *Phys. Rev. Lett.*, 15:240–243, Aug 1965.