Summer 8-1-2015

On the Selection of a Good Shape Parameter for RBF Approximation and Its Application for Solving PDEs

Lei-Hsin Kuo
University of Southern Mississippi

The University of Southern Mississippi

ON THE SELECTION OF A GOOD SHAPE PARAMETER FOR RBF APPROXIMATION AND ITS APPLICATIONS FOR SOLVING PDES

by

LEI-HSIN KUO

Abstract of a Dissertation
Submitted to the Graduate School
of The University of Southern Mississippi
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy

August 2015

ABSTRACT

ON THE SELECTION OF A GOOD SHAPE PARAMETER FOR RBF APPROXIMATION AND ITS APPLICATIONS FOR SOLVING PDES

by LEI-HSIN KUO

August 2015

Meshless methods utilizing Radial Basis Functions (RBFs) are a numerical method that require no mesh connections within the computational domain. They are useful for solving numerous real-world engineering problems. Over the past decades, after the 1970s, several RBFs have been developed and successfully applied to recover unknown functions and to solve Partial Differential Equations (PDEs).

However, some RBFs, such as Multiquadratic (MQ), Gaussian (GA), and Matérn functions, contain a free variable, the shape parameter, c. Because c exerts a strong influence on the accuracy of numerical solutions, much effort has been devoted to developing methods for determining shape parameters which provide accurate results. Most past strategies, which have utilized a trail-and-error approach or focused on mathematically proven values for c, remain cumbersome and impractical for real-world implementations.

This dissertation presents a new method, Residue-Error Cross Validation (RECV), which can be used to select good shape parameters for RBFs in both interpolation and PDE problems. The RECV method maps the original optimization problem of defining a shape parameter into a *root-finding* problem, thus avoiding the local optimum issue associated with RBF interpolation matrices, which are inherently ill-conditioned.

With minimal computational time, the RECV method provides shape parameter values which yield highly accurate interpolations. Additionally, when considering smaller data sets, accuracy and stability can be further increased by using the shape parameter provided by the RECV method as the upper bound of the c interval considered by the LOOCV method. The RECV method can also be combined with an adaptive method, knot insertion, to achieve accuracy up to two orders of magnitude higher than that achieved using Halton points.

The University of Southern Mississippi

ON THE SELECTION OF A GOOD SHAPE PARAMETER FOR RBF APPROXIMATION AND ITS APPLICATION FOR SOLVING PDES

by

Leihsin Kuo

A Dissertation
Submitted to the Graduate School
of The University of Southern Mississippi
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy

Approved:

_____
Dr. Ching-Shyang Chen, Committee Chair
Professor, Mathematics

_____
Dr. James V. Lambers, Committee Member
Associate Professor, Mathematics

_____
Dr. Haiyan Tian, Committee Member
Associate Professor, Mathematics

_____
Dr. Huiqing Zhu, Committee Member
Assistant Professor, Mathematics

_____
Dr. Karen S. Coats
Dean of the Graduate School

August 2015

ACKNOWLEDGMENTS

I would like to gratefully thank my advisor, Dr. C.S. Chen for his support, and instructive guidance during my graduate studies at the University of Southern Mississippi. He encouraged me to explore the research which I am interested in, and advised me to create my own original idea for my dissertation. His advise was paramount in providing a well rounded experience along my long-term career goals. Additionally, I greatly appreciated my committee members for spending time on editing this dissertation.

I take this opportunity to record our sincere thanks to all the faculty members of the Department of Mathematics for their encouragement and help.

Many thanks go to Dr. Joseph Kolibal at the University of New Haven, who taught me to think as a mathematician and led me to understand the difference between mathematics and engineering.

I am grateful for being invited as a visiting scholar at National Taiwan University (NTU) by Dr. D. L. Young in May and June 2012. Because of Dr. D. L. Young's financial support, I had the opportunity to study the Houbolt time marching scheme for the Wave equations.

Furthermore, I would like to acknowledge the financial support given by Dr. Y. C. Hon at City University of Hong Kong (CityU), from June through August 2012. This support provided me the opportunity to explore the research of shape parameter for RBFs.

I also want to thank Dr. C. H. Tsai, Dr. C. M. Fan, Dr. G. M. Yao, Dr. E. Cenek, Dr. T. H. Hsu, Dr. F. F. Dou, Corwin A. Stanford, Ryanne McNeese, Alex Cibotarica, Thir Raj Dangal, Balaram Khatri-Ghimire, Anup Lamichhane, Megan Richardson, Jaeyoun Oh Roberts, Tulsi Upadhyay, Suanrong Chen,Miao Yu, Vika Veunvi, Koshal Dahal, Daniel Lanterman, Hiba Naccache, Wei Zhao, and the members of the lab in CityU and NTU. Your true friendship is a rare treasure of my life.

Most importantly, I would like to thank my family. None of this would have been possible without their unconditional love and support.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Figure

LIST OF TABLES

Table

LIST OF ABBREVIATIONS

IC	-	Initial Conditions
SP	-	Shape Parameter
GS	-	Gaussian
BC	-	Boundary Conditions
CM	-	Collocation Methods
MQ	-	Multiquadric
MM	-	Meshless Method or Meshfree Method
SPI	-	Shape Parameter Indicator
SVD	-	Singular Value Decomposition
PDE	-	Partial Differential Equation
CFD	-	Computational Fluid Dynamics
RBF	-	Radial Basis Function
KIA	-	Knot Insertion Algorithm
CFD	-	Computational Fluid Dynamics
MPS	-	Method of Particular Solutions
FEM	-	Finite Element Method
TSVD	-	Truncated Singular Value Decomposition
MAPS	-	Method of Approximate Particular Solutions
MKIA	-	Modified Knot Insertion
RBFCM	-	Radial Basis Function Collocation Method
KBCMM	-	Kernel Based Collocation Meshless Methods
SPDRBF	-	Strictly Positive Definite Radial Basis Function
CSPDRBF	-	Conditional Strictly Positive Definite Radial Basis Function

Chapter 1

INTRODUCTION

1.1 Introduction

A Partial differential equation (PDE) is a differential equation that involves an unknown multivariable function and at least one corresponding partial derivative of that function. PDEs can be used to model multidimensional dynamical systems and therefore to describe a wide variety of physical phenomena including: simple motion, waves, vibrations, fluid flow, sound, heat distribution and dissipation, and elasticity, as well as phenomena in electrodynamics, and quantum mechanics.

Many mesh-based numerical methods had been proposed for solving PDEs; for instance, the Finite Element Method (FEM) [3, 4, 37, 40, 56], the Finite Difference Method (FDM), the Boundary Element Method (BEM), and the Finite Volume Method (FVM), among others.

The FEM method, in particular, has achieved great success in industry and academia. Many examples of FEM-based commercial software, such as ANSYS and ProCast, have been developed and used in the fields of Scientific Computing and Engineering to approximate solutions for coupled physics problems with large computational domains.

However, because phenomena at the mesoscopic scale can differ significantly from macroscopic phenomena, developments in the fields of Nanotechnology and Micro-Electro-Mechanical Systems (MEMS) have created a demand for numerical PDE solution methods which do not rely on mesh schemes. Numerical mesh-based methods are simply unable to estimate effects accurately for nanotechnology projects. Fortunately, since the early 1990s several meshless methods have been proposed in the fields of Mathematics, Science, and Engineering, which may yet prove useful in addressing this problem.

In general, meshless methods can be classified into domain-type and boundary-type methods. Boundary-type meshless methods, such as the Method of Fundamental Solution (MFS) [20, 31], the Hyper-singular Meshless Method (HMM) [71], the Trefftz method [12, 15, 13], and the Boundary Knot Method (BKM) [14], can be utilized to obtain solutions for homogeneous PDEs. Domain-type meshless numerical methods, such as the Smoothed Particle Hydrodynamics (SPH) method [42] and the Kernel-Based Collo-

cation (KBC) method [43, 44], can be utilized to obtain solutions for non-homogeneous PDEs.

For the MFS, a boundary-type KBC method was first proposed to approximate the solutions of homogeneous elliptic-type partial differential equations [47, 55]. Coupled with the use of the Method of the Particular Solutions (MPS), the MFS has been extended to solving inhomogeneous partial differential equations [30, 31]. With the derivation of the particular solution of Helmholtz-type equations using RBFs [11], the MFS has further extended to solving various types of time-dependent problems [10]. In addition, the Eulerian-Lagrangian method (ELM) was combined with MFS to deal with the multi-dimensional Burgers and wave equation (called Eulerian-Lagrangian method of fundamental solutions, ELMFS) [34, 73]; the time-marching method of fundamental solution (TMMFS) has been successfully applied in the wave equations and the result is competitive with the finite element method (FEM) [33, 72].

Recently, the method of approximate particular solutions (MAPS) [8, 9] has been developed by simply applying the particular solution of the given differential operator as the kernel basis without using the MFS. The MAPS is inspired by the work of Sarra [59] which showed that the integrated RBFs are more stable and accurate than the regular RBFs.

However, the challenge of choosing the location of source points of the MFS and the trade-off principle of determining the shape parameter of Radial Basis Functions (RBFs) have posed difficulties for the TMMFS and MAPS.

In the early 1990s, Madych and Nelson [53] showed the convergence rate of kernels MQ, Inverse Multiquadric (IMQ), and Gaussian (GA) are exponential converge on reproducing kernel Hilbert space by taking either shape parameter, c, or fill distance, h. In particular, the errors of approximation tend to zero when $h \to 0$. In addition, the same results also proved by Wu and Schaback [68] use a different approach. After that, Yoon [70] showed the MQ RBF can also converges exponentially in a Sobolov space for PDEs.

However, in the numerical implementation using RBFs with large values of shape parameter or small values of fill distance, the interpolation matrix can be very ill-conditioned and this can seriously influence the numerical result as what it promised. For this phenomenon is named trade-off principle [61], and has been deeply discussed in many publications.

For the trade-off principle, Carlson and Foley [6] showed that the influence of location of collocation data points is less than shape parameter on the accuracy of the optimal solution. In other words, the choice of the shape parameter is playing central roles of finding the optimal solution for MQ RBF interpolation, and to select an optimal shape parameter, c, become an open issue to compromise the accuracy and stability of MQ RBF approximation before the interpolation matrix becomes overly ill-conditioned.

Meshless methods offer several advantages over mesh-based methods; they are simpler, they can be more accurate, and since they require no mesh scheme, they can be more efficient. This dissertation focuses on meshless methods utilizing Multiquadric Radial Basis Functions (MQ RBFs) to create mesh-free algorithms that are significantly simpler to employ than the FDM, FVM, FEM, and BEM approaches, and which may eliminate the need for meshes, thus eliminating the need to calculate connections and optimize mesh triangles.

For the shape parameter, c, Hardy proposes $c = 0.815\,d$, where $d = \frac{1}{N}\sum_{j=1}^{N} d_j$, and d_j is distance of the given data point $\mathbf{x}_j \in \mathcal{X}$ to its nearest neighbor for MQ [35]. Franke suggests $c = D/(0.8\sqrt{N})$, where D indicates the diameter of the smallest circle which including all interpolation points of the set $\mathcal{X}$ [28]. In the early 1990s researchers began to utilize *Leave-One-Out* Cross Validation (LOOCV) [58, 65] to successfully select an optimal shape parameter, c, within an appropriate interval for RBF interpolation.

For the shape parameter for MQ RBF interpolation, Fornberg and his co-workers [17, 26, 27, 48, 49] proposed several algorithms and demonstrated that the trade-off principle can be overcome but it takes a lot of computational time. For Gaussian RBF interpolation, Fasshauer and McCourt also proposed a new approach which can stably evaluate the RBF interpolants [22].

This dissertation presents a new approach: Residue-Error Cross Validation (RECV) to select a optimal shape parameters for RBFs, both in interpolation and PDEs. By itself, the RECV method provides similar accuracy to the LOOCV method, but with improved stability and reduced computational time. The RECV method can also be utilized with the LOOCV method for higher accuracy, and can combined with an adaptive method, Modified Knot Insertion (MKI), to achieve accuracy up to two orders of magnitude higher than that promised using Halton uniformly distributed points.

1.2 Outline

For interpolation or PDE problems using Radial Basis Functions (RBFs), numerical solutions often yield significantly different results, in terms of accuracy, depending on three factors: 1) value of the shape parameter (usually contained in RBFs), 2) number of collocation points, and 3) distribution method of the collocation points.

Chapter 1 of this dissertation contains an overview of the relevant literature regarding meshless methods utilizing RBFs for interpolation and PDE problems.

Chapter 2 (p. 5) begins with a study of global RBF interpolation problems [35, 36], including the collocation technique, the effect of the trade-off principle when using RBFs [61], as well as previous strategies to select optimal shape parameters for RBF interpolation.

In Chapter 3 (p. 22), the Residue-Error Cross Validation (RECV) method is proposed. The RECV method is a new approach to select an optimal shape parameter, c, for MQ RBF interpolation, based on examining error behavior. Chapter 3 also introduces the Modified Knot Insertion (MKI) adaptive method, an adaptive point method based on Franke's knot insertion [29] associated with optimal shape parameter which selected by RECV method.

In Chapter 4 (p. 42), eight test functions, including Franke's six classic benchmark analytical functions [28], are utilized to examine the properties and demonstrate the efficiency of the RECV method and the MKI adaptive method for RBF interpolation.

Chapter 5 (p. 70) explores the suitability and effectiveness of the RECV method and the MKI method for use with PDE problems. Chapter 5 begins by introducing two domain-type Kernel-Based Collocation Meshless Methods (KBCMM), Kansa's collocation method [43, 44], and the Method of Approximate Particular Solution (MAPS) [8, 9] as solvers for time independent PDEs. Chapter 5 then continues with numerical experiments in which the RECV method and the MKI adaptive method are adopted for use with three PDE problems, including Poisson's equation and convection-diffusion-reaction equations with mixed boundary conditions.

Chapter 6 (p. 93), extends the PDEs to time dependent problems. This chapter begins by introducing the Halbolt time discretisation scheme [39, 62, 66, 72], and continues with a demonstration of how to combine both Kansa's method and the MAPS with the Halbolt method to solve wave equations. Numerical experiments are performed to demonstrate the stability and accuracy achievable by combining the Houbolt method with KBC methods.

In Chapter 7 (p. 104), conclusions regarding the suitability and performance of the RECV method and the MKI method are discussed, particularly in regards to MQ RBF interpolation problems and PDE problems (using either Kansa's method or MAPS). Directions for future research are outlined and explained.

Chapter 2

LITERATURE REVIEW - A BACKGROUND OF RBF INTERPOLATION

Since the early 1970s, Radial Basis Functions (RBFs) began to receive attention in the fields of Scientific Computation and Engineering. Because RBFs can be utilized to approximate scattered data in high-dimensional interpolation problems, RBF quickly emerged as part of the cornerstone of the modern approximation theory.

This chapter briefly introduces RBF interpolation, including key concepts such as collocation technique, interpolation matrix invertibility, and the trade-off principle. The chapter continues with a discussion of several strategies commonly used in the selection of optimal shape parameters, as well as their respective advantages and disadvantages.

2.1 Defining Radial Basis Functions (RBFs)

RBFs were introduced by Rolland Hardy in the early 1970s, during his development of Multiquadratic (MQ) interpolation [35, 36]. Hardy's MQ RBF method focuses on the recovery of unknown functions from known data without triangulation, re-meshing, or other geometric programming efforts.

While Hardy introduced the first RBF, known as Hardy's MQ, following his work several other RBFs, such as Gaussian (GA), Inverse Multiquadric (IMQ), Thin Plate Spline (TPS), and Matérn, have been found and used with Hardy's collocation method to solve interpolation problems, to generate machine learning algorithms, and to find numerical solutions for differential equations.

Definition 2.1.1. ([64, 21]) Let $\mathbb{R}^d$ be d-dimensional Euclidean space. A function $\Phi : \mathbb{R}^d \to \mathbb{R}$ is called a Radial Basis Function (RBF) if

$$\Phi(\mathbf{x}) = \Phi(\mathbf{y}), \quad \text{whenever} \quad \|\mathbf{x}\| = \|\mathbf{y}\|, \quad \mathbf{x}, \mathbf{y} \in \mathbb{R}^d \tag{2.1}$$

where $\|\cdot\|$ is the Euclidean norm (*i.e.*, the l_2 norm) on $\mathbb{R}^d$.

It is important to note that the RBF, Φ, will be radially symmetric about its center, and the value of Φ is constant, representing the distance between known data points.

2.2 RBF interpolation

Interpolation involves taking given known data points and creating a virtual surface, an interpolant, which must pass through all the known data points. Therefore, in order to recover an unknown function, u, from a given dataset, $\mathcal{X}$, containing N distinct scattered data points in the domain, $\Omega \subseteq \mathbb{R}^d$, a necessary assumption for RBF interpolation is that the unknown function, u, can be expressed as linear combinations,

$$u(\mathbf{x}) \approx \tilde{u}(\mathbf{x}) := \sum_{j=1}^{N} \alpha_j \Phi\left(\|\mathbf{x}-\mathbf{x}_j\|_2\right), \ \mathbf{x} \in \Omega, \tag{2.2}$$

where $\|\cdot\|_2$ is the Euclidean norm, and Φ is a kernel which can contain any radial basis function, such as a *Multiquadric* (MQ) RBF,

$$\Phi(r,c) = \sqrt{r^2+c^2}, \tag{2.3}$$

or a *Gaussian* (GA) RBF,

$$\Phi(r,c) = \exp(-r^2/c^2), \tag{2.4}$$

where

$$r = \|\mathbf{x}\|_2 = \sqrt{x_1^2+\cdots+x_d^2}, \quad \mathbf{x} \in \mathbb{R}^d.$$

If the given dataset, $\mathcal{X}$, contains no noise, then the coefficients, $\{\alpha_j\}_{j=1}^N$, are chosen by forcing the exact interpolation condition,

$$\tilde{u}(\mathbf{x}_j) = u(\mathbf{x}_j), \quad j = 1,\cdots,N. \tag{2.5}$$

Combining (2.2) and (2.5) leads to a system of linear equations,

$$\mathbf{A}\boldsymbol{\alpha} = \mathbf{u}_{\mathcal{X}}, \tag{2.6}$$

where $\boldsymbol{\alpha} = [\alpha_1,\ldots,\alpha_N]^T$, and $\mathbf{u}_{\mathcal{X}} = [u(\mathbf{x}_1),\ldots,u(\mathbf{x}_N)]^T$, and the interpolation matrix, $\mathbf{A} = \mathbf{A}_{ij} = \Phi(r_{ij}) \in \mathbb{R}^{N\times N}$. A distance matrix, r_{ij}, contained within $\mathbf{A}$ can be is expressed as follows:

$$r_{ij} = \begin{bmatrix} \|\mathbf{x}_1-\mathbf{x}_1\|_2 & \|\mathbf{x}_1-\mathbf{x}_2\|_2 & \ldots & \|\mathbf{x}_1-\mathbf{x}_N\|_2 \\ \|\mathbf{x}_2-\mathbf{x}_1\|_2 & \|\mathbf{x}_2-\mathbf{x}_2\|_2 & \ldots & \|\mathbf{x}_2-\mathbf{x}_N\|_2 \\ \vdots & \vdots & \vdots & \vdots \\ \|\mathbf{x}_N-\mathbf{x}_1\|_2 & \|\mathbf{x}_n-\mathbf{x}_2\|_2 & \ldots & \|\mathbf{x}_N-\mathbf{x}_N\|_2 \end{bmatrix}. \tag{2.7}$$

Listing 2.1: DMatrix.m

```
function [DM] = DMatrix(data, ctrs)
[M,~] = size(data); [N,d] = size(ctrs);
DM = zeros(M,N);
for i=1:d
    DM = DM+(repmat(data(:,i),1,N)-repmat(ctrs(:,i)',M,1)).^2;
end
DM = sqrt(DM);
end
```

Listing 2.2: RBFInterpolation2D.m

```
function [] = RBFInterpolation2D()
% Define the Hardy's MQ
rbf = @(c,r) sqrt(r.^2+c^2); c=2.6;
% Define unknwon function which will be recovered
%by MQ interpolation.
testfunction = @(x,y)  sinc(2*x).*sinc(2*y);
% Generate center and test points
N=100; Nt=2500;
[X,Y] = meshgrid(linspace(-1,1,sqrt(N)));
ctrs=[X(:) Y(:)]; clear X Y;
[X,Y] = meshgrid(linspace(-1,1,sqrt(Nt)));
test=[X(:)Y(:)];
% Generate sample data in center and test points
z = testfunction(ctrs(:,1),ctrs(:,2));  % samples
zt = testfunction(test(:,1),test(:,2));
% Distance matrix
DM = DMatrix(ctrs,ctrs);
DMt = DMatrix(test, ctrs);
% Interpolation mareix
A = rbf(c,DM);
% solve for coefficients Alpha
Alpha = A\z;
% Evaluation
At = rbf(c,DMt);
Error = At*Alpha-zt;
% Recreate surface by MQ interpolant
figure(1)
mesh(X,Y,reshape(At*Alpha,sqrt(Nt),sqrt(Nt))); hold on
plot3(test(:,1),test(:,2),zt,'.');
% Plot error
figure(2)
mesh(X,Y,reshape(Error,sqrt(Nt),sqrt(Nt)));
end
```

Example 2.2.1. Consider a surface defined by the cardinal sine function (sinc function):

$$u(x,y) = \operatorname{sinc}(2x) \cdot \operatorname{sinc}(2y). \tag{2.8}$$

MQ RBF can be employed with shape parameter, $c = 2.6$, to interpolate (2.8), yielding 100 equally spaced points in the domain, $[-1,1]^2$ (see Figure 2.1 (a), p. 9). The profile of (2.8) is shown in Figure 2.1 (b).

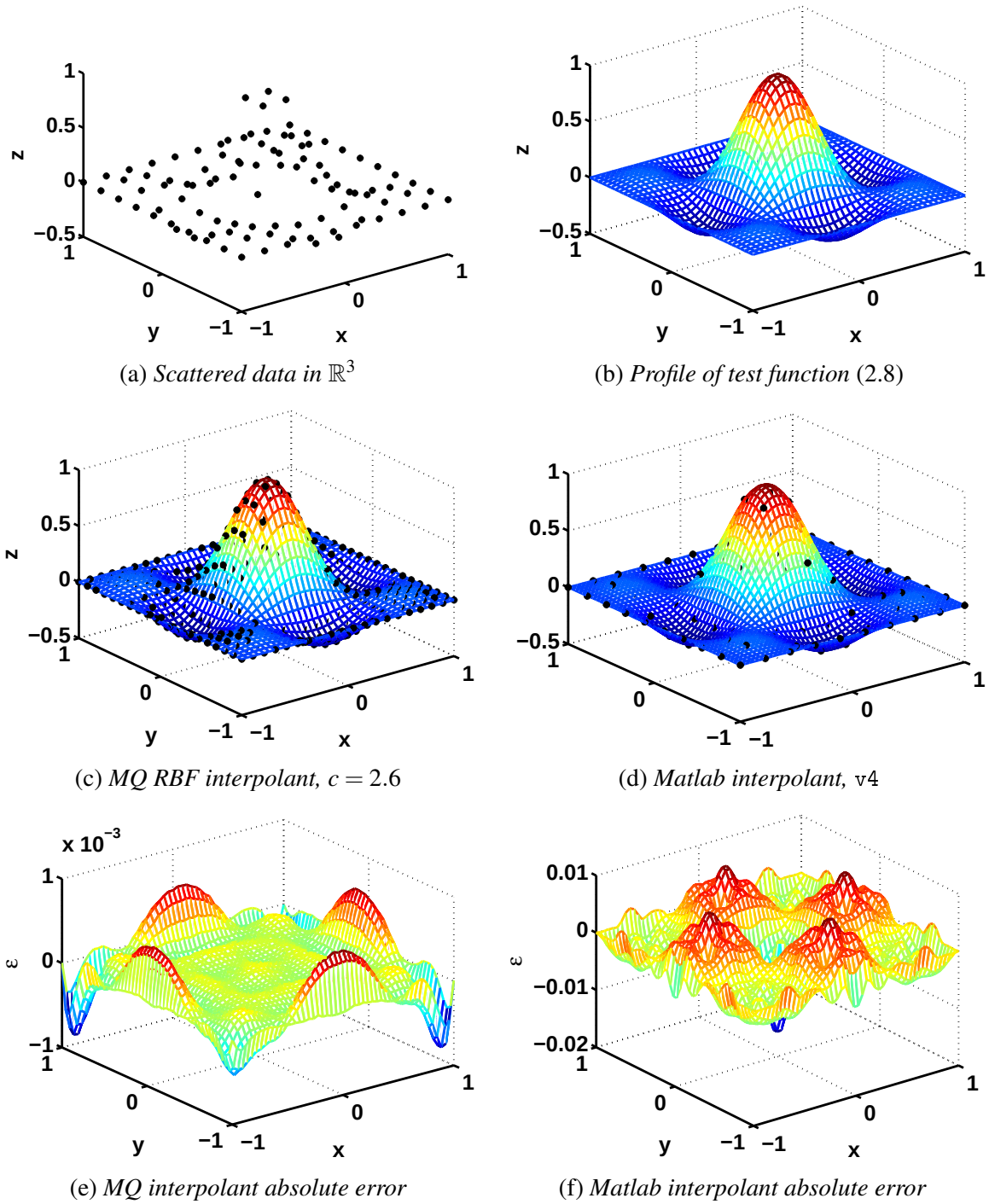
Figure 2.1 (c) shows the MQ RBF approximation for (2.8) generated with the script, Listing 2.2, which calls the `DMatrix.m`[1] subroutine, Listing 2.1. For comparison, Figure 2.1 (d) shows the interpolant surface for (2.8) generated by Matlab's standard interpolation command, `griddata`, utilizing the `v4` method (see Listing 2.3, line 15).

Both appear to be good approximations, with the generated interpolants intercepting all of the data points. However, considering absolute error, the accuracy of approximation generated by the MQ RBF method (see Figure 2.1 (e)) is one order of magnitude higher than that achieved via Matlab's built-in algorithm (see Figure 2.1 (f)).

Listing 2.3: MatlabInterpolation2D.m

```
1  function [] = MatlabInterpolation2D()
2  % Define unknwon function which will be recover by
3  %built-in Matlab algorithm
4  testfunction = @(x,y)  sinc(2*x).*sinc(2*y);
5  % Generate center and test points
6  N=100; Nt=2500;
7  [X,Y] = meshgrid(linspace(-1,1,sqrt(N)));
8  ctrs(:,1)=X(:); ctrs(:,2)=Y(:); clear X Y;
9  [X,Y] = meshgrid(linspace(-1,1,sqrt(Nt)));
10 test(:,1)=X(:); test(:,2)=Y(:);
11 % Generate sample data in center and test points
12 z = testfunction(ctrs(:,1),ctrs(:,2));  % samples
13 zt = testfunction(test(:,1),test(:,2));
14 % Solve the interpolation problem
15 Z = griddata(ctrs(:,1),ctrs(:,2),z,X,Y,'v4')
16 % Evaluation
17 Error = Z(:)-zt;
18 % Recreate surface by MQ interpolant
19 figure(1)
20 mesh(X,Y,Z); hold on
21 plot3(test(:,1),test(:,2),Z(:),'.');
22 % Plot error
23 figure(2)
24 mesh(X,Y,reshape(Error,sqrt(Nt),sqrt(Nt)));
25 end
```

[1] Subroutine Listing 2.1 is a Matlab script designs to compute a distance matrix as described in (2.7) using two sets of data points within the domain, Ω; the script is modified by `DistanceMatrix.m` which introduced by G. Fasshauer [21], in order to cut down the among of memory usage while generating a distance matrix.



(a) *Scattered data in* $\mathbb{R}^3$

(b) *Profile of test function* (2.8)

(c) *MQ RBF interpolant,* $c = 2.6$

(d) *Matlab interpolant,* `v4`

(e) *MQ interpolant absolute error*

(f) *Matlab interpolant absolute error*

Figure 2.1: *Comparison of the accuracy of MQ RBF approximation and Matlab's built-in* `v4` *method. (see Example 2.2.1). Figure (a) shows the given sample points in the domain* $[-1,1]^2$*, while Figure (b) shows the profile of test function* (2.8)*. Figures (c) and (d) show the interpolants via MQ RBF and Matlab built-in* `v4`*, respectively, while Figures (e) and (f) show the error distribution for MQ RBF interpolant and Matlab built-in interpolant, respectively.*

2.3 RBF Interpolation Matrix Invertibility

In Section 2.2, Example 2.2.1 demonstrated how a highly accurate approximation utilizing MQ RBF can be achieved for any smooth function. However, solving the linear system, such as (2.6), imply there exists a unique matrix, $\mathbf{A}^{-1}$. Therefore, the coefficient $\boldsymbol{\alpha}$ can be obtained by,

$$\boldsymbol{\alpha} = \mathbf{A}^{-1}\mathbf{u}_{\mathcal{X}}, \tag{2.9}$$

and the invertibility is used to guarantee the matrix, such as $\mathbf{A}$, is invertible and nonsingular.

This section, introduces the central properties of RBF interpolation. Usually the invertibility of the RBF interpolation matrix, $\mathbf{A}$, in (2.6), could cause serious problems. In order to assure the matrix is always invertible, this section introduces the central properties of RBF interpolation as follows,

Definition 2.3.1. (positive definite) A real symmetric $N \times N$ matrix $\mathbf{A}$ is called positive semidefinite, if and only if its associated quadratic form

$$\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j \mathbf{A}_{ij} \geq 0 \tag{2.10}$$

for $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_N]^T$. The matrix $\mathbf{A}$ is called positive definite if the quadratic form (2.10) is zero only for $\boldsymbol{\alpha} = 0$.

Definition 2.3.2. A radial basis function $\Phi : \mathbb{R}^d \to \mathbb{R}$ on $[0, \infty)$ is strictly positive definite on $\mathbb{R}^d$, if for every set of distinct data points $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \subset \mathbb{R}^d$, the matrix $\mathbf{A}$ in (2.6) is positive definite.

The definition of strictly positive definite (Definition 2.3.2) guarantee that the system of equations in (2.6) using RBF, $\Phi(r)$, listed in Table 2.1 is invertible. But there are some powerful RBFs (see Table 2.2, p. 12) that fail to be strictly positive definite but satisfied the definition as follows:

Definition 2.3.3. A radial basis function $\Phi : \mathbb{R}^d \to \mathbb{R}$ on $[0, \infty)$ is conditional strictly positive definite of order m on $\mathbb{R}^d$, if for every set of distinct data points $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \subset \mathbb{R}^d$

$$\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j \Phi(\|\mathbf{x}_i - \mathbf{x}_j\|) > 0, \tag{2.11}$$

for all the non-zero $\{\alpha_1, \ldots, \alpha_N\}$ satisfied,

$$\sum_{i=1}^{N} \alpha_i p(\mathbf{x}_i) = 0 \tag{2.12}$$

for all polynomials p degree less than m.

In such case, in order to guarantee the interpolation matrix, $\mathbf{A}$ is positive definite when utilizing the Conditional Strictly Positive Definite Radial Basis Functions (CSPDRBFs) listed in Table 2.2, the trial function (2.2) should be rewritten as,

$$u(\mathbf{x}) \approx \tilde{u}(\mathbf{x}) := \sum_{j=1}^{N} \alpha_j \Phi\left(\|\mathbf{x}-\mathbf{x}_j\|_2\right) + p(\mathbf{x}), \tag{2.13}$$

with some polynomials, where the $p \in \prod_{m-1}^{d}$ denotes the space spanned by all d-variate polynomials of degree up to $m-1$ in $\mathbb{R}^d$, and pick a basis $\{p_1, \ldots, p_{m-1}\}$ of this space. Solving (2.13) leads to the matrix form,

$$\begin{bmatrix} \mathbf{A} & P \\ P^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} = \begin{bmatrix} u \\ 0 \end{bmatrix}, \tag{2.14}$$

where $\mathbf{A}_{ij} = \Phi(r_{ij})$, $P_{ij} = p_j(\mathbf{x}_i)$ is polynomials of degree up to $m-1$, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are unknown coefficients.

However, in Franke's conjecture [28], the interpolation matrix constructed by Hardy's MQ is non-singular and invertible without added polynomials within (2.2). Franke's conjecture also will be extended to domain type kernel based collocation meshless methods (see chapter 5) for PDE problems.

Tables 2.1 and 2.2 list some commonly used RBFs utilized within (2.2) and (2.13), respectively. The symbol r in the tables denotes the Euclidean distance, and K_ν is the modified Bessel function of order ν.

Figure 2.2 shows some of commonly used RBFs, which contain a shape parameter, c, such as MQ, IMQ, GA, and Matérn, can force the solution of RBF interpolation problems exponentially approach to the function attempted to recover.

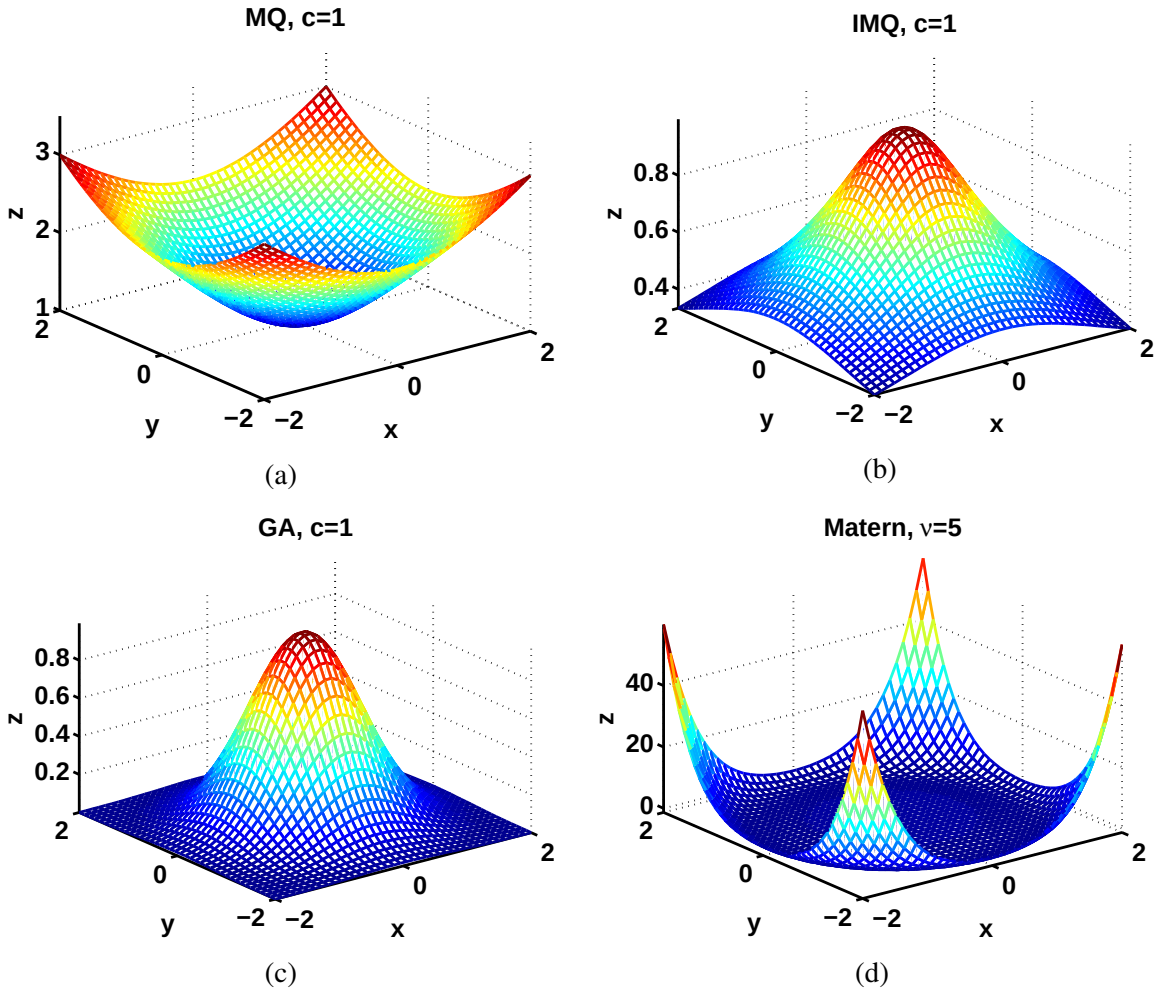
Figure 2.3 shows the graphs of GA with $c = 1$ and $c = 5$. Apparently, a larger value of shape parameter, c, causes Gaussian basis function become flatter, and a smaller value of c resulted in a more peaked basis function.

Table 2.1: *Strictly positive definite radial basis functions (SPD-RBFs).*

Type	RBF $\Phi(r)$
Gaussian	$\exp(-r^2/c^2)$
Inverse Multiquadric	$1/\sqrt{r^2+c^2}$
Matérn (Sobolev)	$r^{\nu}K_{\nu}(r)$
Matérn Spline	$e^{-cr}K_{\nu}(cr)$

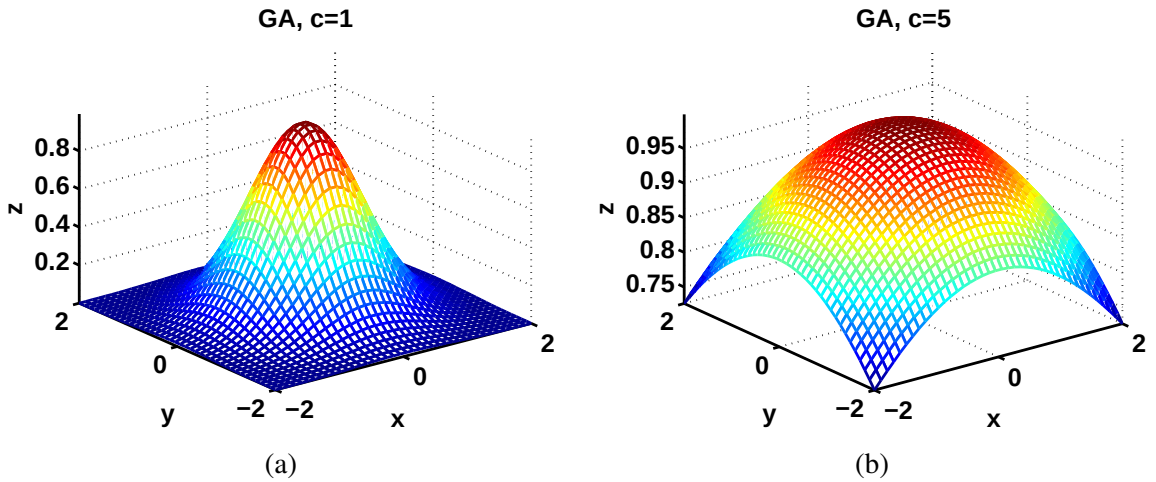
Table 2.2: *Conditional strictly positive definite radial basis functions (CSPD-RBFs).*

Type	RBF $\Phi(r)$	CSPD
Hardy's multiquadric	$\sqrt{r^2+c^2}$	1
Polyharmonic	$(-1)^{\beta/2}r^{\beta}$, $\beta>0$, $\beta\notin 2\mathbb{N}$	$\beta/2$
Thin-Plate Spline	$r^2\log(r)$	2



(a) (b) (c) (d)

Figure 2.2: *Some commonly used radial basis functions,* $\phi(r)$ *in* $\mathbb{R}^2$.



(a) (b)

Figure 2.3: *GA RBF with different shape parameter values.*

2.4 The Trade-Off Principle

This section briefly introduces the Trade-Off Principle (or the Uncertainty) named by R. Schaback [60, 61]. The Principle describes any interpolation utilizing RBFs which contain a shape parameter, c, (e.g., MQ, IMQ, GA, or Matérn RBFs). Essentially, when deciding on a shape parameter to optimize numerical results, a compromise must be made between achieving accuracy and achieving stability.

As shown in Example 2.2.1, RBF interpolation can be performed in Matlab with minimal coding to achieve highly accurate numerical results. The literature [5, 21, 64] indicates that, for a given scattered points set, $\mathcal{X}$, in the domain, Ω, errors occurring during MQ RBF interpolation can be decreased by modifying two variables: either shape parameter, c, or fill distance, h.

The fill distance, h, represents the maximum distance between any two data points in $\mathcal{X}$, and is defined as follows:

$$h := h(\mathcal{X}, \Omega) := \sup_{\mathbf{y}\in\Omega} \min_{\mathbf{x}\in\mathcal{X}} \| \mathbf{y} - \mathbf{x} \|_2 \, . \tag{2.15}$$

In the early 1990s, Madych [52, 53] provided theoretical proof that the absolute error of interpolation utilizing RBFs tends to machine precision when fill distance tends to zero.

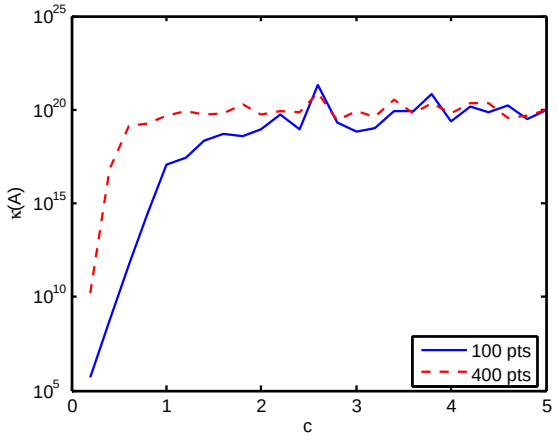
However, the interpolation matrix, $\mathbf{A}$ in (2.6), generated with RBFs, such as MQ, IMQ, or GA, is well known as a highly ill-conditioned matrix [35, 60, 61]. In this dissertation, the condition number of the interpolation matrix is denoted by $\kappa(\mathbf{A})$, which is defined:

$$\kappa(\mathbf{A}) = \|\mathbf{A}\| \, \|\mathbf{A}^{-1}\| = \frac{\sigma_{\max}}{\sigma_{\min}}, \tag{2.16}$$

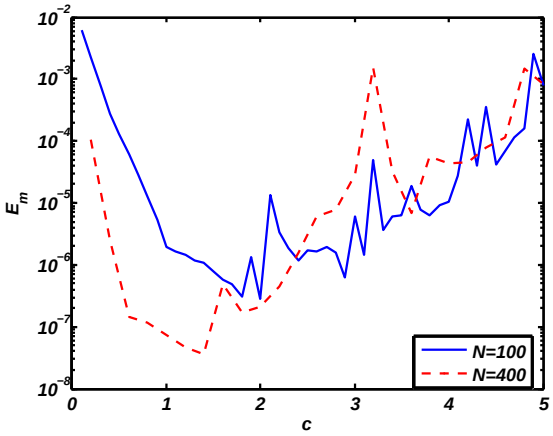
where $\sigma_{\max}$ and $\sigma_{\min}$ are the largest and smallest singular values of $\mathbf{A}$. The condition number can be used to estimate the degree to which a numerical solution will be affected by rounding errors [63]. Hardy [35] and others have demonstrated that, as h decreases or as c increases, $\kappa(\mathbf{A})$ increases, leading to greater instability and error. In other words, the numerical solution of RBF interpolations utilizing an inappropriate shape parameter value, c, or a small fill distance value, h, can be extremely unstable and inaccurate.

The desire to increase accuracy and stability in RBF interpolation has led to two different approaches: stationary interpolation, and non-stationary interpolation.

Stationary interpolation considers the total number of interpolation points, N, to be fixed; changing values of the shape parameter, c, modifies accuracy. In the second approach, non-stationary interpolation, the shape parameter, c, is fixed, while the number of interpolation points, N, is variable.

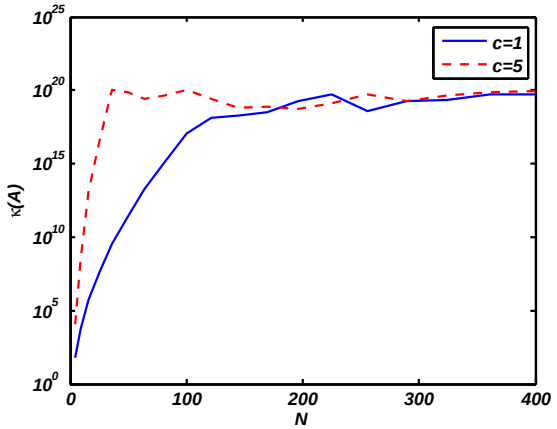


(a) *Condition number, $\kappa(A)$, versus shape parameter, c.*

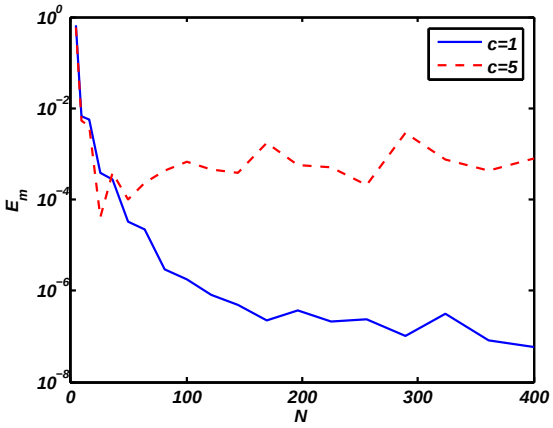


(b) *Maximum approximation error, E_m, versus shape parameter, c.*

Figure 2.4: *Stationary MQ RBF interpolation. The total number of interpolation points are fixed at $N = 100$ (solid line) and $N = 400$ (dashed line).*



(a) *Condition number, $\kappa(A)$, versus number of interpolation points, N.*



(b) *Maximum approximation errors, E_m, versus number of interpolate points, N.*

Figure 2.5: *Non-stationary MQ interpolation. Shape parameter are fixed at $c = 1$ (solid line) and $c = 5$ (dash line).*

In order to illustrate the Trade-off Principle and the instability issue that occurs when a matrix is ill-conditioned, Figure 2.4 (stationary interpolation) shows condition number, $\kappa(\mathbf{A})$, and maximum error, E_m, plotted with changing shape parameter, c; while Figure 2.5 (non-stationary interpolation) shows condition number, $\kappa(\mathbf{A})$, and maximum error, E_m, plotted with number of interpolation points, N, as a variable. In both cases, test function (2.8) is utilized (see Example 2.2.1, p.8).

Figure 2.4 (a) shows the condition number, $\kappa(\mathbf{A})$, for stationary MQ RBF interpolations constructed for $N = 100$, and $N = 400$ equally spaced interpolation points in the domain, $[-0.5, 0.5]^2$. For both data sets, $\kappa(\mathbf{A})$, shows exponential growth as c increases.

Figure 2.4 (b) shows maximum approximation error, E_m, versus shape parameter, c. For both data sets ($N = 100$, and $N = 400$) the accuracy of MQ approximation improves smoothly as shape parameter increases, until $c = 1.9$, and $c = 1.4$, respectively. Beyond these ideal points, as the shape parameter, c, becomes less appropriate, the condition number, $\kappa(\mathbf{A})$, grows extremely high and the matrix becomes increasingly ill-conditioned.

A similar error behavior can be seen in Figure 2.5, which shows results for non-stationary MQ RBF interpolations constructed with two different shape parameters, $c = 1$ and $c = 5$, for $N = 9$ (3^2) to $N = 400$ (20^2) equally spaced interpolation points in the domain, $[-0.5, 0.5]^2$.

Figure 2.5 (a) shows that, as interpolation points, N, increase, condition number, κ, grows rapidly. Figure 2.5 (b) shows that maximum approximation error increases rapidly, as well, as N increases.

Together, Figures 2.4 and 2.5 illustrate the Trade-Off Principle. The phenomenon is summarized: Increasing accuracy, by increasing c (shape parameter) or N (interpolation points), increases instability. The Trade-Off Principle applies to both stationary and non-stationary RBF interpolation, so long as the RBF includes a shape parameter.

Essentially, increasing either c (shape parameter) or N (interpolation points) can improve accuracy. However, when using either variable to improve accuracy, some method must be utilized to pick an optimal value for that variable in order to increase accuracy without increasing instability beyond an acceptable level. Most research in this area has focused on methods to determine ideal shape parameters. This is because c is a free variable, which can be changed easily. Changing values for N, interpolation points, however, can have a real-world cost, depending on the type of data being considered, as well as increasing computational costs, especially as data-sets become very large.

When discussing stationary MQ RBF interpolation, mathematically shape parameter, c, could be increased indefinitely to improve accuracy without limit. Madych [52, 53], for example, demonstrated that an accurate MQ RBF approximation could be made by a large c value and small fill distance, h.

Computationally, however, due the Trade-Off Principle, as c increases, so does κ. Therefore, efforts to determine an optimal shape parameter, c, for stationary MQ RBF interpolation need to strike a balance between accuracy and stability.

2.5 Leave-One-Out Cross Validation (LOOCV)

Leave-One-Out cross-validation (LOOCV) is a validation technique to evaluate the results in statistics analysis. In the past years, it has been successfully applied to select an optimal shape parameter for RBF but the technique is time consuming.

For a given interpolation points set $\mathcal{X} = \{\mathbf{x}_j\}_{j=1}^{N}$, the LOOCV takes out a single point to estimate the error of MQ approximation which is constructed by the rest $N-1$ data points. Therefore, the test points for searching for an optimal shape parameter can be omitted by repeating N times this procedure for whole interpolation points set.

To actually operate the LOOCV to select an optimal shape parameter, the given interpolation points set $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ needs to take out a single data point $\mathbf{x}_k$ with corresponding remaining $N-1$ data set:

$$\{\mathbf{x}_j^{[k]}\}_{\substack{j=1 \\ j\neq k}}^{N} = [\mathbf{x}_1, \ldots, \mathbf{x}_{k-1}, \mathbf{x}_{k+1}, \ldots, \mathbf{x}_N]^T,$$

can be used to construct the approximation $\tilde{u}^{[k]}(\mathbf{x})$:

$$\tilde{u}^{[k]}(\mathbf{x}) \;:=\; \sum_{j=1}^{N-1} \alpha_j^{[k]} \Phi\left(\|\mathbf{x} - \mathbf{x}_j^{[k]}\|_2\right), \tag{2.17}$$

and the error $E^{[k]}$ at the points $\mathbf{x}_k$ can be written:

$$E^{[k]} \;=\; |\tilde{u}^{[k]}(\mathbf{x}_k) - u(\mathbf{x}_k)|. \tag{2.18}$$

The accuracy of MQ approximation is determined by $E = [E^{[1]}, \ldots, E^{[N]}]^T$ which is corresponding to the entire interpolation points set $\mathcal{X}$.

The cost of CPU time in the procedure above is expensive and tedious. Rippa's paper[58], however, simplified this procedure by a single formula:

$$E^{[k]} = \frac{\alpha_k}{\mathbf{A}_{kk}^{-1}}, \tag{2.19}$$

where the α_k is the k^{th} coefficient of *full* data interpolation as (2.7), and $\mathbf{A}_{kk}^{-1}$ is the k^{th} diagonal element of the inverse matrix of corresponding interpolation. Hence, the errors of MQ interpolation can be completed in a single time by use (2.19). Then the Matlab built-in function `fminbnd` can be applied to find the minimum of cost function as Listing 2.4 [23] for optimal shape parameter, c.

2.6 LOOCV Versus Previous Methods: A Numerical Comparison

In 1971 Hardy [35] demonstrated that an optimal shape parameter for MQ RBF interpolation could be selected in $\mathbb{R}^2$ utilizing $c = 0.815\,d$, where $d = \frac{1}{N}\sum_{j=1}^{N} d_j$, and d_j is the distance of a given data point, $\mathbf{x}_j \in \mathcal{X}$, to its nearest neighbor.

Later, in 1982, Franke [28] proposed another method to define optimal shape parameters for MQ RBF interpolation, utilizing $c = D/(0.8\sqrt{N})$, where D is the diameter of the smallest circle which includes all interpolation points, $\{\mathbf{x}_j\}_{j=1}^{N}$.

Both of these methods provided relatively good shape parameters, especially when calculations are done on single precision (32 bits) arithmetic. In addition, Franke's shape parameter can be slightly modified and made more suitable for double precision (64 bits), as follows:

$$c = D/(0.8\sqrt[4]{N}), \tag{2.20}$$

The LOOCV method (see (2.19), p. 17) can be even more precise, and, unlike previous methods, offers additional precision when utilized with double precision arithmetic.

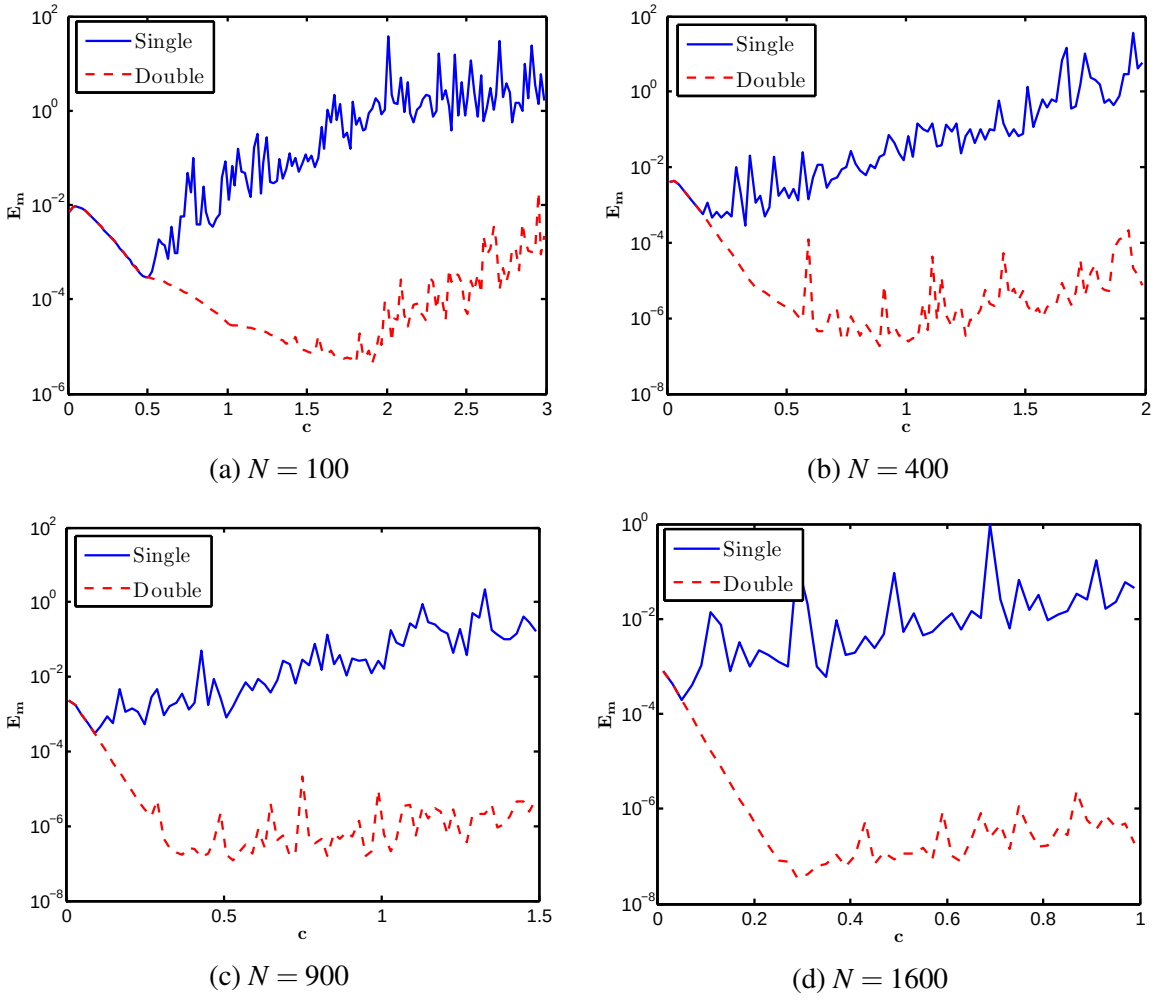
To illustrate the relative strengths and weaknesses of each approach, test function (2.8) in Example 2.2.1 was sampled for $N = 100$, $N = 400$, $N = 900$, and $N = 1600$ regular grid interpolation points in the computational domain $[-0.5, 0.5]^2$. Matlab scripts were then utilized to determine the shape parameter, c, values recommended by Hardy's Method (Listing B.1), Franke's Method (Listing B.2), the modified Franke method [2], and the LOOCV method (Listing 2.5, calling Listing 2.4).

Figure 2.6 shows maximum approximation error, E_m, for the range of possible shape parameter values, c, for $N = 100$, $N = 400$, $N = 900$, and $N = 1600$, when interpolated on single and double precision arithmetic (see Example 2.2.1, p. 8). Figures 2.6 (a)–(d) show that maximum approximation error, E_m, is the same for single and double precision arithmetic when c is small; however, double precision are able to utilize significantly larger shape parameters and achieve lower maximum error. This example demonstrates that the values for optimal shape parameters are dependent on the precision of the utilized arithmetic.

Table 2.3 shows maximum approximation error, E_m, of interpolants generated by the c values suggested by the Hardy method, the Franke method, and the Modified Franke method.

Comparing the numerical results in Table 2.3 with the different optimal shape parameter, c, values suggested by Figures 2.6 (a)-(d) reveals the relative strengths and weaknesses of each method. The shape parameter, c, values suggested by the Hardy and Franke methods

[2] To interpolate using the Modified Franke's shape parameter, the command in line 25 of the Matlab script (Listing B.2) should be replaced with the command, "`c = sqrt(2/0.64/sqrt(N(i)));`".



(a) $N = 100$ (b) $N = 400$

(c) $N = 900$ (d) $N = 1600$

Figure 2.6: *Maximum approximation error, E_m, versus shape parameter, c, for different numbers of points, N, for the MQ approximation described in Example 2.2.1. Comparing the c values predicted by the Hardy, Franke, and Modified Franke methods (see Table 2.3) to these results illustrates the relative strengths and weaknesses of the approaches.*

are more suitable for use on single precision arithmetic. On the other hand, the shape parameters suggested by the Modified Franke method are more suitable for use on double precision arithmetic, but might suggest overly high c values when used on a single precision.

To compare the relative accuracy of the LOOCV method, Table 2.4 shows maximum approximation error, E_m, of interpolants generated using the shape parameters suggested by the LOOCV method (from the ranges $c \in [0, 1.5]$, and $c \in [0, 20]$), for $N = 100$, $N = 400$, $N = 900$, and $N = 1600$, when interpolated on single and double precision arithmetic.

Examining the results in Table 2.4 shows that, for small ranges of c, i.e., $c \in [0, 1.5]$, the LOOCV method suggests optimal c values when run on a double precision arithmetic, but

Table 2.3: *Maximum approximation error, E_m, of interpolants generated with the Hardy, Franke, and Modified Franke shape parameter, c, values (for various numbers of points, N, sampled from* (2.8)*, from Example 2.2.1).*

		Hardy		Franke		Modified Franke	
Precision	N	c	E_m	c	E_m	c	E_m
Double	100	0.09	8.23E-03	0.18	4.53E-03	0.56	2.64E-04
	400	0.04	3.85E-03	0.09	1.81E-03	0.40	5.46E-06
	900	0.03	1.77E-03	0.06	7.77E-04	0.32	3.12E-07
	1600	0.02	5.79E-04	0.04	2.43E-04	0.28	3.59E-08
Single	100	0.09	8.23E-03	0.18	4.53E-03	0.56	5.91E-04
	400	0.04	3.85E-03	0.09	1.81E-03	0.40	3.67E-03
	900	0.03	1.77E-03	0.06	7.82E-04	0.32	5.09E-04
	1600	0.02	5.79E-04	0.04	2.40E-04	0.28	6.14E-03

suggests overly high c values when utilized on a single precision arithmetic. Additionally, the LOOCV method suggests overly high c values when considering a larger c range ($c \in [0, 20]$) on both single and double precision arithmetic. This pattern of overly high recommended c values can be explained as a local optimum issue [58]. Overcoming this issue has been a goal within the field for several years.

Table 2.4: *Maximum approximation error, E_m, of interpolants generated with LOOCV method shape parameter, c, values, for different ranges of c (for various numbers of points, N, sampled from* (2.8)*, from Example 2.2.1).*

		$c \in [0, 20]$		$c \in [0, 1.5]$	
Precision	N	c	E_m	c	E_m
Double	100	10.92	4.84E+01	1.01	2.90E-05
	400	5.65	8.69E-01	0.58	1.00E-06
	900	7.64	4.38E-01	0.30	5.79E-07
	1600	9.44	1.30E+00	0.29	2.97E-08
Single	100	7.64	3.62E-01	0.45	3.84E-04
	400	8.33	6.11E+00	0.32	7.46E-04
	900	12.52	4.57E+00	0.36	3.22E-03
	1600	15.28	9.14E+00	0.92	8.25E-02

Listing 2.4: CostLoocv.m

```
function [ceps] = CostLoocv(c,DM,rbf,rhs)
  A = rbf(c,DM);
  invA = inv(A);
  EF = (invA*rhs)./diag(invA);
  ceps = norm(EF,inf);
  end
```

Listing 2.5: RBFInterpolation2DLOOCV.m

```
function []=RBFInterpolation2DLOOCV()
tic
% Define Hardy's MQ.
rbf = @(c,r) sqrt(r.^2+c^2);
% Define unknown function which will be recovered.
%by MQ interpolation.
testfunction = @(x,y) sinc(2*x).*sinc(2*y);
% Define number of center points.
N = [100 400 900 1600];
% Define upper and lower bounded of shape parameter for optimization.
minc =0; maxc = 2;
% Generate test points.
[X,Y] = meshgrid(linspace(-0.5,0.5,sqrt(2500)));
test=[X(:) Y(:)]; clear X Y;
% Sample data in test poins.
zt = testfunction(test(:,1),test(:,2));
for i=1:length(N)
    % Generate center points.
    [X,Y] = meshgrid(linspace(-0.5,0.5,sqrt(N(i))));
    ctrs=[X(:) Y(:)]; clear X Y;
    % Create right hand side vector.
    z = testfunction(ctrs(:,1),ctrs(:,2));
    % Compute distance matrix.
    DM = DMatrix(ctrs,ctrs);
    DMt = DMatrix(test,ctrs);
    % LOOCV for optimal c.
    [c(i),~] = fminbnd(@(c) CostLoocv(c,DM,rbf,z),minc,maxc,...
    optimset('TolX',1e-3));
    % Solved the system of equations.
    A = rbf(c(i),DM);
    Alpha = A\z;
    % Evaluate the error at given test points.
    At = rbf(c(i),DMt);
    Error = At*Alpha-zt;
    Maxerr(i) = norm(Error, inf);
    fprintf('Number of interpolant: %i\n', N(i))
    fprintf('Maximum error: %e\n', Maxerr(i))
    fprintf('c = %f\n', c(i))
    fprintf('CPU time = %f\n', toc)
end
end
```

Chapter 3

DEVELOPING THE RECV METHOD FOR SHAPE PARAMETER IN RBF INTERPOLATION

To date, the problem of minimizing costs when selecting a shape parameter has been a classic problem within the fields of computational science, mathematics, and engineering. Numerous studies examine the issue of determining an ideal shape parameter when interpolating with Multiquadric (MQ), Gaussian (GA), and Matérn RBFs.

So far, efforts which have focused on mathematically proven shape parameter values have remained difficult to apply in real-world implementations. Other methods, which tend to utilize a trail-and-error approach, yield results which are not optimally accurate or which ignore the effect of the precision of the solver. Further efforts have utilized global optimization methods such as genetic algorithms [1, 19] or simulated annealing [2] with the LOOCV method. While these approaches tend to yield good shape parameter values, they can be computationally expensive and time consuming.

This dissertation presents a new method, Residue-Error Cross Validation (RECV), which can be used to select an optimal shape parameters for RBFs in both interpolation and PDE problems. The RECV method maps the original optimization problem of defining a shape parameter into a *root-finding* problem, thus avoiding the local optimum issue associated with RBF interpolation matrices, which are inherently ill-conditioned.

With minimal computational time, the RECV method provides shape parameter values which yield highly accurate interpolations. Additionally, when considering smaller data sets, accuracy and stability can be further increased by using the shape parameter provided by the RECV method as the upper bound of the c interval considered by the LOOCV method.

The RECV method can also be combined with an adaptive method, knot insertion, to achieve accuracy up to two orders of magnitude higher than that achieved using Halton uniformly distributed points.

3.1 Residue Error vs. Optimal Shape Parameter

Residue Error

Consider a given dataset, $\mathcal{X} = \{\mathbf{x}_j\}_{j=1}^{N}$, containing N distinct interpolation points, and an unknown function, u, sampled at the given points in domain, Ω. As has been amply demonstrated [5, 21, 64], any RBF interpolants, $\tilde{u}$, containing a shape parameter, c, guarantee:

$$\tilde{u}(\mathbf{x}_j, c) \quad = \quad u(\mathbf{x}_j), \qquad \text{for each } j = 1, \ldots, N. \tag{3.1}$$

Then, there exists a natural lower bound which limits the maximum error of each given RBF interpolation, as described:

$$\max_{\mathbf{x}\in\Omega} \{|\tilde{u}(\mathbf{x}, c) - u(\mathbf{x})|\} \quad \geq \quad \max_{\mathbf{x}_j\in\mathcal{X}} \left\{|\tilde{u}(\mathbf{x}_j, c) - u(\mathbf{x}_j)|\right\}. \tag{3.2}$$

In other words, the accuracy of a given RBF interpolant is always best at the sample points. This is a point which is well known, but often ignored as trivial.

Considering that shape parameter, c, can be adjusted to improve the accuracy of numerical solutions (Section 2.4, p. 14), if an optimal solution RBF interpolation exists, in terms of shape parameter, c, then the ideal c value should be located where:

$$\max_{\mathbf{x}\in\Omega} \{|\tilde{u}(\mathbf{x}, c) - u(\mathbf{x})|\} \quad \approx \quad \max_{\mathbf{x}_j\in\mathcal{X}} \left\{|\tilde{u}(\mathbf{x}_j, c) - u(\mathbf{x}_j)|\right\}. \tag{3.3}$$

Mathematically, when approximating a given surface, there will be no error at the given data points. However, due to the limited-precision of real implementation, there is always a residue error, $\mathbf{r}$, when performing an interpolation. The sensitivity of the residue error vector can be visualized via a simple approach. Let $\mathbf{r} = \mathbf{u} - \mathbf{A} \cdot \tilde{\boldsymbol{\alpha}}$ (where $\tilde{\mathbf{u}} = \mathbf{A}\tilde{\boldsymbol{\alpha}}$, and $\tilde{\boldsymbol{\alpha}}$ is the coefficient vector at machine precision). The vector, $\mathbf{e} = \boldsymbol{\alpha} - \tilde{\boldsymbol{\alpha}}$, then describes errors of the coefficient, yielding:

$$\mathbf{A}\mathbf{e} = \mathbf{A}\left(\boldsymbol{\alpha} - \tilde{\boldsymbol{\alpha}}\right) = \mathbf{A}\boldsymbol{\alpha} - \mathbf{A}\tilde{\boldsymbol{\alpha}} = \mathbf{u} - \mathbf{A}\tilde{\boldsymbol{\alpha}} = \mathbf{r}. \tag{3.4}$$

Computing the magnitude of $\mathbf{r}$ in any norm, we find that the residue, $\mathbf{r}$, of the RBF interpo-

lation is bounded by:

$$\begin{aligned} \|\mathbf{r}\| &= \|\mathbf{A}\mathbf{e}\| && (3.5)\\ &\leq \|\mathbf{A}\|\|\mathbf{e}\| && (3.6)\\ &\leq \|\mathbf{A}\|\|\mathbf{e}\|\frac{\|\boldsymbol{\alpha}\|}{\|\boldsymbol{\alpha}\|} && (3.7)\\ &\leq \|\mathbf{A}\|\|\mathbf{e}\|\frac{\|\mathbf{A}^{-1}\mathbf{u}\|}{\|\boldsymbol{\alpha}\|} && (3.8)\\ &\leq \|\mathbf{A}\|\|\mathbf{A}^{-1}\|\frac{\|\mathbf{e}\|}{\|\boldsymbol{\alpha}\|}\|\mathbf{u}\|. && (3.9) \end{aligned}$$

The magnitude of the residual errors in $\|\mathbf{r}\|$ can then be expressed as follows,

$$\|\mathbf{r}\| \leq \kappa(\mathbf{A})\,\|\mathbf{u}\|\frac{\|\mathbf{e}\|}{\|\boldsymbol{\alpha}\|}, \tag{3.10}$$

where $\kappa(\mathbf{A})$ is the condition number of the original interpolation matrix, $\mathbf{A}$. Because $\|\mathbf{u}\|$, and $\|\mathbf{e}\|/\|\boldsymbol{\alpha}\|$ are fixed values, (3.10) demonstrates that $\mathbf{r}$ is is directly influenced by the condition number, $\kappa(\mathbf{A})$.

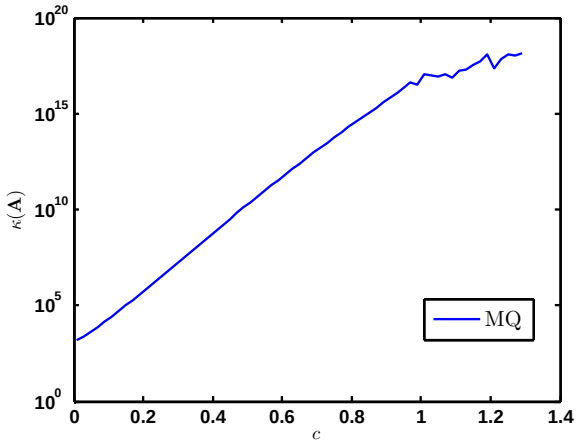
In order to illustrate the interaction between shape parameter, c, condition number, $\kappa(\mathbf{A})$, and maximum residue error, $\mathbf{r}_m$, in numerical implementations of MQ RBF interpolation, a 10 by 10 regular grid of interpolation points was sampled for test function:

$$u(x,y) = \exp(-2x+3y), \tag{3.11}$$

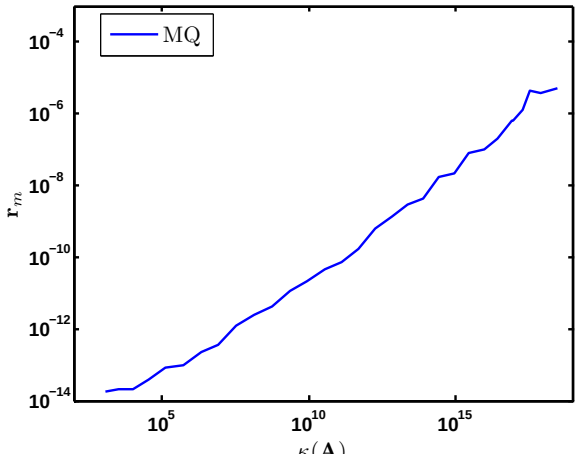
in the domain, $\Omega \in [0,1]^2$.

Figure 3.1 (a) shows condition number, $\kappa(\mathbf{A})$, versus shape parameter, c; while Figure 3.1 (b) shows maximum approximation error, E_m, versus the condition number, $\kappa(\mathbf{A})$. As can be seen in 3.1 (a), during MQ RBF interpolation, a larger shape parameter may result in a significantly ill-conditioned interpolation matrix. In addition, 3.1 (b) shows that maximum residue error grows exponentially as condition number, $\kappa(\mathbf{A})$, increases, as described in (3.10).

As (3.1) indicated, an RBF interpolant should perfectly intersect the given points, $\mathcal{X}$, in the domain Ω. Therefore, in numerical implementation we expect that the accuracy of an RBF interpolant with an optimal shape parameter should tend to machine precision. On the other hand, as we enlarge the shape parameter to achieve machine precision, the rapid growth of condition number, $\kappa(\mathbf{A})$, and consequently the rapid growth of residue error, $\|\mathbf{r}\|$, make this promised machine precision impossible to actually achieve.



(a) $\kappa(\mathbf{A})$ versus c



(b) $\mathbf{r}_\mathrm{m}$ versus $\kappa(\mathbf{A})$

Figure 3.1: *Error behavior for MQ RBF interpolation. See test function* (3.11). *Figure 3.1 (a)shows that condition number,* $\kappa(\mathbf{A})$*, grows exponentially as shape parameter, c increases; while Figure 3.1 (b) shows that maximum residue error grows exponentially as condition number,* $\kappa(\mathbf{A})$*, increases, as described in* (3.10).

Best MQ approximation with respect to c

In 1992, Madych [53], proposed a method to estimate and describe error behavior for MQ interpolation:

$$E = O(e^{ac}\lambda^{(c/h)}); \quad 0 < \lambda < 1, \quad a > 0. \tag{3.12}$$

As shown in (3.12), during MQ interpolation, error approaches zero, $E \to 0$, as the shape parameter approaches infinity. On the other hand, as shown by (3.2), the accuracy of MQ approximation is limited by increasing residue error as described in (3.10). Combining these

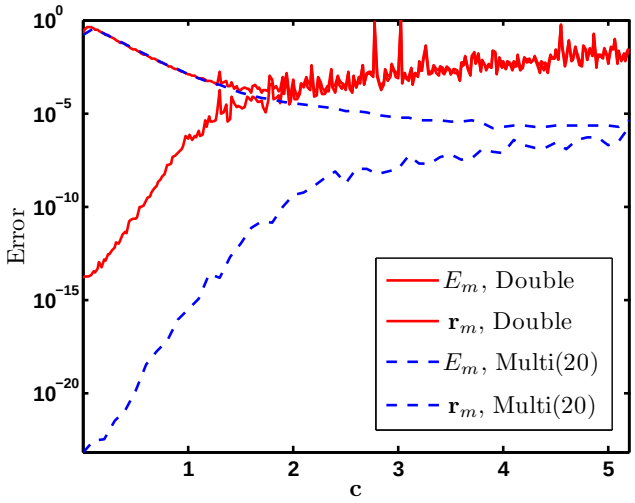


Figure 3.2: *Maximum residue error,* $\mathbf{r}_m$*, and maximum approximation error,* E_m*, solved by double- and multi-precision solvers. See test function* (3.11)*. For either precision of solver, as c increases, maximum approximation error goes down while maximum residue error goes up. The optimal value for c is therefore found where* $\mathbf{r}_m$ *and* E_m *approach, before the interpolation matrix becomes overly ill-conditioned.*

concepts, optimal c values for MQ RBF interpolation should occur where the maximum approximation error, E_m, approaches the maximum residue error, $\mathbf{r}_m$ as described in (3.3).

In order to visualize this hypothesized error behavior, MQ RBF test function (3.11) in the domain $[0,1]^2$ was sampled on a 10 by 10 regular grid; maximum residue error, $\mathbf{r}_m$, was calculated using the interpolation points, while maximum approximation error, E_m, was calculated using 40 by 40 evenly distributed points. Figure 3.2 shows both maximum residue error, $\mathbf{r}_m$, and maximum approximation error, E_m, plotted versus shape parameter, c, when solved by double- and multiple-precision arithmetic.

Two observations can be derived from the results shown in Figure 3.2:

1. **Error behavior and optimal c values**: Initially, maximum approximation error, E_{m}, of stationary MQ interpolations solved by different precision solvers decreases smoothly and exponentially as shape parameter, c, increases, just as Madych proposed; see (3.12). At the same time, the residue error, $\mathbf{r}_{\mathrm{m}}$, of MQ interpolations gets consistently worse as c increases, as predicted by (3.10). The two measures of error, E_{m} and $\mathbf{r}_{\mathrm{m}}$, as can be seen in the figure, stop significantly converging and optimal shape parameters occur at $c = 1.7$ and $c = 5$, respectively, for double- and multiple-precision solvers.

2. **Effect of limited-precision arithmetic:** Figure 3.2 shows that the maximum residue error, $\mathbf{r}_m$, of MQ interpolations solved by double- and multiple-precision both increase with nearly parallel curves (also shown in Figure 2.6, p.19). In other words, higher-precision solvers won't improve the rate of convergence of MQ approximations but will allow for larger shape parameters and more accurate numerical solutions.

3.2 RECV vs. LOOCV

Defining an Indicator, I(c), for Optimal Shape Parameters

Figure 3.2 shows that for different precision solvers, as c increases, maximum approximation error goes down while maximum residue error, $\mathbf{r}_m$, goes up. The optimal value for c is therefore found where E_m and $\mathbf{r}_m$ approach. In order to determine an optimal shape parameter, c, for RBF interpolation based on these observations, a shape parameter indicator, $(I)(c)$, is proposed to measure the approach of maximum approximation error, E_m, and maximum residue error, $\mathbf{r}_m$:

$$\begin{aligned} \mathrm{I}(c) &= \log_{10}(E_m) - \log_{10}(\mathbf{r}_m), \quad \text{where} \\ E_m &= \max_{\mathbf{x}\in\Omega} \{|\tilde{u}(\mathbf{x},c) - u(\mathbf{x})|\}, \\ \mathbf{r}_m &= \max_{\mathbf{x}_j\in\mathcal{X}} \left\{|\tilde{u}(\mathbf{x}_j,c) - u(\mathbf{x}_j), \quad j = 1,\ldots,N|\right\}. \end{aligned} \tag{3.13}$$

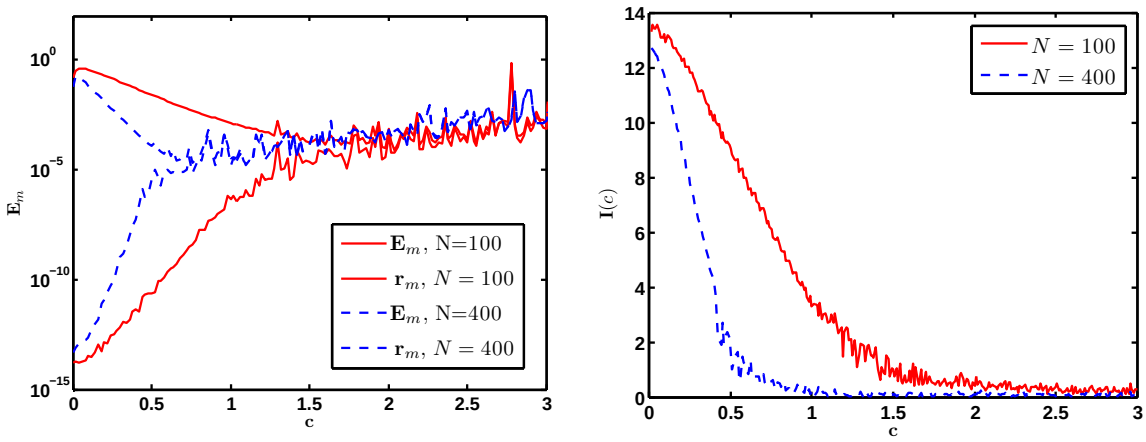
This indicator, I(c), allows the curves for E_m, and $\mathbf{r}_m$ to be plotted into a single nearly horizontally asymptotic curve. In this manner, the original optimization problem of defining an ideal shape parameter, c, is mapped to become a *root-finding* problem.

Figure 3.3 shows the error behavior of MQ interpolants for test function (3.11), constructed using regular grid interpolation points $\{\mathbf{x}_j\}_{j=1}^{100}$, $\{\mathbf{x}_j\}_{j=1}^{400}$ in a unit square domain $[0,1]^2$, solved by a double precision solver.

Figure 3.3 (a) shows that for both $N = 100$ and $N = 400$, as c increases, E_m goes down while $\mathbf{r}_m$ goes up. Thus, an optimal c value is found where $\mathbf{r}_m$ and E_m approach, before the interpolation matrix becomes overly ill-conditioned. Figure 3.3 (b) shows the shape parameter indicator, I(c), for the same example. The optimal c value is located where the error behavior represented by $\mathrm{I} = 1 \sim 2$ becomes horizontally asymptotic, at ~ 0.5 for $N = 400$ and ~ 1.5 for $N = 100$.

Strategies to Select a Shape Parameter for MQ Interpolation

Thus far, as demonstrated in Figure 3.3, by using the shape parameter indicator, I(c), defined in (3.13), the problem of finding an ideal shape parameter for MQ interpolation can be



(a) (b)

Figure 3.3: *Error behavior for an MQ RBF interpolation using* $N = 100$*, and* $N = 400$ *uniformly distributed points. Figure 3.3 (a) shows maximum residue error,* $\mathbf{r}_m$*, and maximum approximation error,* E_m*, for MQ interpolations of* (3.11) *in double precision. For both* $N = 100$ *and* $N = 400$*, as c increases,* E_m *goes down while* $\mathbf{r}_m$ *goes up. Thus, an optimal c value is found where* $\mathbf{r}_m$ *and* E_m *approach, before the interpolation matrix becomes overly ill-conditioned. Figure 3.3 (b) shows the shape parameter indicator,* $I(c)$*, for the same example. The optimal c value is located where the error behavior represented by* $I = 1 \sim 2$ *becomes horizontally asymptotic, at* ~ 0.5 *for* $N = 400$ *and* ~ 1.5 *for* $N = 100$*.*

mapped from an *optimization* problem to a *root-finding* problem. In practice, therefore, the cost function, Listing 3.1, can be solved using the built-in Matlab function `fzero`, to find an ideal shape parameter, c.

Equation (3.13) describes an ideal method for calculating E_m, taking into account an infinite number of data points. In the real world, this is impossible, and we must work with given functions and data sets of finite size. In this case, just as with (3.13), the given data set must be split into two sub sets. One set, $\mathcal{X} \in \Omega$, is used to construct an MQ approximation, while the other set, $\mathcal{V} \in \Omega \backslash \mathcal{X}$, is used to determine the accuracy of the approximation. Therefore, (3.13) should be re-written as follows:

$$\begin{aligned} \mathrm{I}(c) &= \log_{10}(E_m) - \log_{10}(\mathbf{r}_m), \quad \text{where} \\ E_m &= \max_{\mathbf{x}_j \in \mathcal{V}} \left\{ |\tilde{u}(\mathbf{x}_j) - u(\mathbf{x}_j)| \right\}, \\ \mathbf{r}_m &= \max_{\mathbf{x}_j \in \mathcal{X}} \left\{ |\tilde{u}(\mathbf{x}_j, c) - u(\mathbf{x}_j)| \right\}. \end{aligned} \tag{3.14}$$

In this study, two different strategies to determine optimal shape parameter, c, values for MQ RBF interpolation are proposed, both based on (3.14).

Strategy 1: Using the RECV Method to Determine Shape Parameter (RECV c)

To utilize (3.14) to determine an optimal shape parameter, c, value, let the *left-hand side* of the equation, $\mathrm{I}(c)$, be a positive constant. Within the near horizontally asymptotic curves, as shown in Figure 3.3b, there then exists a solution, respectively, for $\mathrm{I}(c) = 1$, 2, or any other positive value. The remainder of this work will examine the c values obtained by setting $\mathrm{I}(c) = 1$ or 2. Thus, the solution (or root) of the $\mathrm{I}(c)$ curves indicates a good shape parameter, c, for MQ interpolation, meaning c can be solved for using any *root-finding* method.

Algorithm 1 demonstrates a way in which to solve the root finding problem, (3.14), via the bisection method (*i.e.,* the binary search method). The bisection method is both simple and robust; however, it is also well known for being relatively slow.

This study, instead of using the bisection method shown in Algorithm 1, (3.14) is solved using the built-in Matlab function, `fzero`. For implementation, the cost function, Listing 3.1, can be called by `fzero` as follows,

```
c=fzero(@(c) CostSPI(c,rbf,DM,DMt,Z,Zt,sp),[minc,maxc],...
    optimset('TolX',1e-3));
```

where `DM` and `DMt` are distance matrices, with respect to interpolation points and training points, respectively, where `minc` and `maxc` define the range of interval for c, and where `Z` and `Zt` are the sample data and the given data sets, $\mathcal{X}$ and $\mathcal{V}$, respectively.

Algorithm 1 - Bisection method to find RECV c, see Equation (3.14)

Require: $c_L, c_U, I(c^*)$ `% define upper and lower bound of c, and value of I.`
  while $|c_U - c_L| \geq 1\mathrm{E}-3$ **do**
    $c^* = (c_L + c_U)/2$;
    Compute the errors of ε_m and $\mathbf{r}$
    if $\log_{10}(E_m) - \log_{10}(\mathbf{r}_m) < I(c^*)$ **then**
      $c_U = c^*$;
    else
      $c_L = c^*$;
    end if
  end while
  $c^* = c_L + c_U$;

Strategy 2: Combining RECV and LOOCV to Find Shape Parameter (Combined c)

In general, the shape parameter selected via Strategy 1 (RECV c) can overcome the local optimum issue and generate an accurate approximation for an MQ problem. However, this accurate MQ approximation is guaranteed only if the interpolation point dataset, $\mathcal{X}$, is large or dense enough to allow convergence between the maximum approximation error, E_m, and the maximum residue error, $\mathbf{r}_m$, as c increases. Essentially, if data sets are sparse, or low-density, Strategy 1 may yield slightly over-determined c values.

When using the LOOCV method, as shown in Table 2.4 (p. 20), choosing an interval within which to search for a shape parameter, c, is critical to find an optimal shape parameter. Therefore, Strategy 2 takes advantage of both the RECV and LOOCV methods to identify a good shape parameter, Combined c, using two steps:

1. Utilize RECV c (selected by Strategy 1) to be the upper bound of the c interval considered by the LOOCV method.

2. Select a shape parameter via the LOOCV method (see Section 2.5 in p.17) within this re-scaled interval, $[0,$ RECV $c]$.

Numerical results using RECV c and Combined c

In order to demonstrate the accuracy of MQ RBF interpolation using RECV c and Combined c (Matlab programs Listing 3.2 and 3.3), the MQ RBF interpolation problem described in Example 2.2.1 in p. 20 is reexamined in the interval $[0,100]$. As shown in Table 3.1, the results, in terms of maximum approximation error, E_m, indicate that both RECV c and Combined c serve as optimal shape parameter values for MQ RBF interpolation. In addition, two features can be derived from the results shown in Table 3.1.

1. For Strategy 1, the RECV method, for both $\mathrm{I}(c) = 1$ and $\mathrm{I}(c) = 2$, generates adequately accurate MQ approximations. However, RECV c values when $\mathrm{I}(c) = 1$ are slightly larger than RECV c values when $\mathrm{I}(c) = 2$.

2. For Strategy 2, comparing the numerical results in Tables 3.1 and 2.4, it is clear that the shape parameter yielded by combining the RECV and LOOCV methods successfully overcomes the local optimum issue (mentioned in [58]) for a wide interval range of c without the need to apply a global optimization method.

Table 3.1: *Maximum Approximation Error, E_m, of MQ RBF Interpolation using shape parameter RECV c ($I = 1$), RECV c ($I = 2$), or Combined c, selected from the interval* $[0, 100]$.

		Strategy 1 RECV c, I = 1		Strategy 1 RECV c, I = 2		Strategy 2 Combined c	
Precision	N	c	E_m	c	E_m	c	E_m
Double	100	1.82	5.28E-06	1.31	1.64E-05	1.39	1.17E-05
	400	0.58	7.56E-07	0.50	1.89E-06	0.39	5.74E-06
	900	0.34	2.30E-07	0.27	1.54E-06	0.27	1.66E-06
	1600	0.21	4.27E-07	0.19	6.72E-07	0.19	6.53E-07
Single	100	0.49	3.18E-04	0.33	1.16E-03	0.40	6.04E-04
	400	0.13	7.98E-04	0.11	1.15E-03	0.13	8.16E-04
	900	0.06	8.36E-04	0.04	1.22E-03	0.06	8.44E-04
	1600	0.04	2.42E-04	0.03	4.89E-04	0.04	2.48E-04

Listing 3.1: CostSPI.m

```
function [ceps] = CostSPI(c,rbf,DM,DMt,Z,Zt,sp)
A=rbf(c,DM);
Alpha = A\Z;
maxerr = norm(A*Alpha - Z, inf);
res = norm(rbf(c,DMt)*Alpha - Zt, inf);
ceps = log10(res) - log10(maxerr)-sp;
end
```

Listing 3.2: RBFInterpolation2DSpi.m

```
function []=RBFInterpolation2DSpi()
warning off all;
tic
% Define the Hardy's MQ.
rbf=@(c,r) sqrt(r.^2+c^2);
% Define unknown function which will be recovered.
%by MQ interpolation.
testfunction = @(x,y) sinc(2*x).*sinc(2*y);
% Define number of center points.
N = [100 400 900 1600];
% Define upper and lower bounded of shape parameter for optimization.
minc =0; sp=1;
% Generate test points.
[X,Y] = meshgrid(linspace(-0.5,0.5,sqrt(2500)));
test=[X(:) Y(:)]; clear X Y;
% Sample data in test poins.
zt = testfunction(test(:,1),test(:,2));
training = haltonseq(10,2); training=training-0.5;
ztrn = testfunction(training(:,1),training(:,2));
for i=1:length(N)
    % Generate center points.
    [X,Y] = meshgrid(linspace(-0.5,0.5,sqrt(N(i))));
    ctrs=[X(:) Y(:)]; clear X Y;
    % Create right hand side vector.
    z = testfunction(ctrs(:,1),ctrs(:,2));
    % Compute distance matrix.
    DM = DMatrix(ctrs,ctrs);
    DMtrn = DMatrix(training, ctrs);
    DMt = DMatrix(test,ctrs);
    % SPI selected an upper bound of range of c.
    c=fzero(@(c) CostSPI(c,rbf,DM,DMtrn,z,ztrn,sp),[0,100],...
        optimset('TolX',1e-3));
    % Solved the system of equations.
    A = rbf(c,DM);
    Alpha = A\z;
    % Evaluate the error at given test points.
    At = rbf(c,DMt);
    Error = At*Alpha-zt;
    Maxerr(i) = norm(Error, inf);
    fprintf('=======================================\n');
    fprintf('Number of interpolant: %i\n', N(i))
    fprintf('Maximum error: %e\n', Maxerr(i))
    fprintf('c = %f\n', c(i))
    fprintf('CPU time = %f\n', toc)
    fprintf('=======================================\n');
end
end
```

Listing 3.3: RBFInterpolation2DLoocvSpi.m

```
function []=RBFInterpolation2DLoocvSpi()
warning off all;
tic
% Define the Hardy's MQ.
rbf=@(c,r) sqrt(r.^2+c^2);
% Define unknown function which will be recovered.
%by MQ interpolation.
testfunction = @(x,y) sinc(2*x).*sinc(2*y);
% Define number of center points.
N = [100 400 900 1600];
% Define upper and lower bounded of shape parameter for optimization.
minc =0; sp=1;
% Generate test points.
[X,Y] = meshgrid(linspace(-0.5,0.5,sqrt(2500)));
test=[X(:) Y(:)]; clear X Y;
% Sample data in test poins.
zt = testfunction(test(:,1),test(:,2));
training = haltonseq(10,2); training=training-0.5;
ztrn = testfunction(training(:,1),training(:,2));
for i=1:length(N)
    % Generate center points.
    [X,Y] = meshgrid(linspace(-0.5,0.5,sqrt(N(i))));
    ctrs=[X(:) Y(:)]; clear X Y;
    % Create right hand side vector.
    z = testfunction(ctrs(:,1),ctrs(:,2));
    % Compute distance matrix.
    DM = DMatrix(ctrs,ctrs);
    DMtrn = DMatrix(training, ctrs);
    DMt = DMatrix(test,ctrs);
    % SPI selected an upper bound of range of c.
    maxc=fzero(@(c) CostSPI(c,rbf,DM,DMtrn,z,ztrn,sp),...
    [0,100],optimset('TolX',1e-3));
    % LOOCV for optimal c.
    [c(i),~] = fminbnd(@(c) CostLoocv(c,DM,rbf,z),minc,maxc,...
                  optimset('TolX',1e-3));
    % Solved the system of equations.
    A = rbf(c(i),DM);
    Alpha = A\z;
    % Evaluate the error at given test points.
    At = rbf(c(i),DMt);
    Error = At*Alpha-zt;
    Maxerr(i) = norm(Error, inf);
    fprintf('=======================================\n');
    fprintf('Number of interpolant: %i\n', N(i))
    fprintf('Maximum error: %e\n', Maxerr(i))
    fprintf('c = %f\n', c(i))
    fprintf('CPU time = %f\n', toc)
    fprintf('=======================================\n');
end
end
```

3.3 Modified Knot Insertion (MKI)

Knot insertion is a classic technique employed to enhance the accuracy of approximation in interpolation problems by adaptively increasing the number of data points. Since every RBF interpolant has a corresponding set of interpolation points (or knots), the accuracy of approximation should be increased by adding new centers to the existing dataset over multiple iterations.

The idea to use knot insertion for MQ RBF interpolation in $\mathbb{R}^2$ was first proposed by Franke, Nielson and Hagen [29]. Franke proposed a method, see algorithm 2 for use with a given data set, $\{\mathbf{x}_i, u_i\}_{i=1}^{M}$, with a large number of data points, M. Using Franke's method, the accuracy of MQ approximation conducted with a small shape parameter, c, is improved by adaptively adding knots in each iteration, at whichever data location last yielded highest error. In other words, an over-determined interpolation matrix is solved by applying a linear least squares fitting each iteration, over multiple iterations.

Algorithm 2 - Franke's Knot Insertion

(1) Let dataset $\{\mathbf{x}_i, u_i\}_{i=1}^{M}$, chosen initial knots $\mathcal{X} = \{\xi_j\}_{j=1}^{N}$, small value of c, and a maximum number of knot `maxk` to stop the iteration.
while $N \leq$ `maxk` **do**
(2) Solve the system of equations with least squares fitting

$$\sum_{j=1}^{N} \alpha_j \Phi(\|\mathbf{x}_i - \xi_j\|, c) = u(\mathbf{x}_i), \quad i = 1, \ldots, M.$$

(3) Find the data point $\mathbf{x}_\tau \in \mathcal{X} \backslash \Xi$ associated with the maximum residue

$$\|\mathbf{r}\|_\infty = \max_{1 \leq i \leq M} \left\{ |u(\mathbf{x}_i) - \sum_{j=1}^{N} \alpha_j \Phi(\|\mathbf{x}_i - \xi_j\|, c)| \right\}.$$

(4) Insert the data points $\mathbf{x}_\tau$ as knot in $\mathcal{X}$ and then $N = N + 1$.
end while

However, as shown in Figure 2.1 (e) (Example 2.2.1), an MQ approximation conducted with an optimal shape parameter, c, may, as Fornberg says, "behave badly at the end of an interval" [25].

In this thesis a new adaptive method, Modified Knot Insertion (MKI), is proposed (see Algorithm 3) in order to improve the boundary error behavior of MQ RBF interpolations. As opposed to the original Franke knot insertion method, rather than using a small fixed shape parameter value, the MKI method uses RECV c (Strategy 1) during each iteration.

Algorithm 3 - Modified Knot Insertion (MKI)

(1) Let dataset $\{\mathbf{x}_i, u_i\}_{i=1}^{M}$, chosen initial knots $\mathcal{X} = \{\xi_j\}_{j=1}^{N}$, and a maximum number of knot `maxk` to stop the iteration.
while $N \leq$ `maxk` **do**
(2) Compute a RECV c (Strategy 1) for MQ interpolant

$$u(\mathbf{x}) = \sum_{j=1}^{N} \alpha_j \Phi(\|\mathbf{x} - \xi_j\|, c)$$

(3) Solve the system of equations

$$\sum_{j=1}^{N} \alpha_j \Phi(\|\xi_i - \xi_j\|, c) = u(\xi_i), \quad i = 1, \ldots, N.$$

(4) Find the data point $\mathbf{x}_\tau \in \mathcal{X} \backslash \Xi$ associated with the maximum residue

$$\|\mathbf{r}\|_\infty = \max_{1 \leq i \leq M} \left\{ |u(\mathbf{x}_i) - \sum_{j=1}^{N} \alpha_j \Phi(\|\mathbf{x}_i - \xi_j\|, c)| \right\}.$$

(5) Insert the data points $\mathbf{x}_\tau$ as knot in $\mathcal{X}$ and then $N = N + 1$.
end if end while

To illustrate the differences in the adaptive points generated by Franke's knot insertion (Listing 3.4) and MKI (Listing 3.5), and the accuracy of results obtained by the two different methods in numerical implementations of MQ RBF interpolation, two test functions are examined within the domain $[0,1]^2$:

$$K1(x,y) = \sin(x) \cdot \cos(y), \tag{3.15}$$

$$K2(x,y) = \exp(-2x + 3y). \tag{3.16}$$

Figure 3.4 shows $4,225$ Halton points serve as given dataset in the domain $[0,1]^2$, which are used to test the efficacy of Franke's knot insertion and the MKI method with test functions $K1$ and $K2$ (see Figures 3.5 (a) and (b)). In Figures 3.5 (c) and (d) show the adaptive points for test functions $K1$ and $K2$, chosen using Franke's knot insertion with a shape parameter value ($c = 0.2$), while Figures 3.5 (e) and (f) show the adaptive points selected using the MKI method with RECV c, which is recalculated for each iteration.

A visual comparison of the two adaptive point sets shows that, while the adaptive points selected using Franke's knot insertion are clustered where the functions rapidly change value, and sparsely distributed in the rest of the domain, the MKI adaptive points are uniformly distributed throughout the domain and more densely distributed near the boundary.

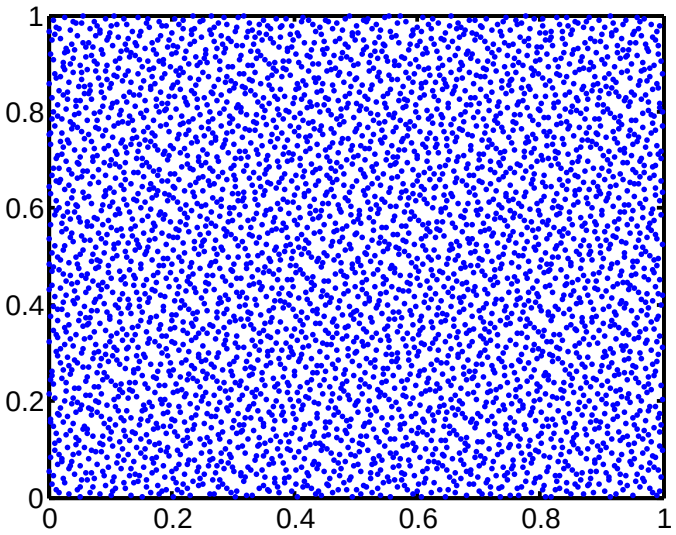
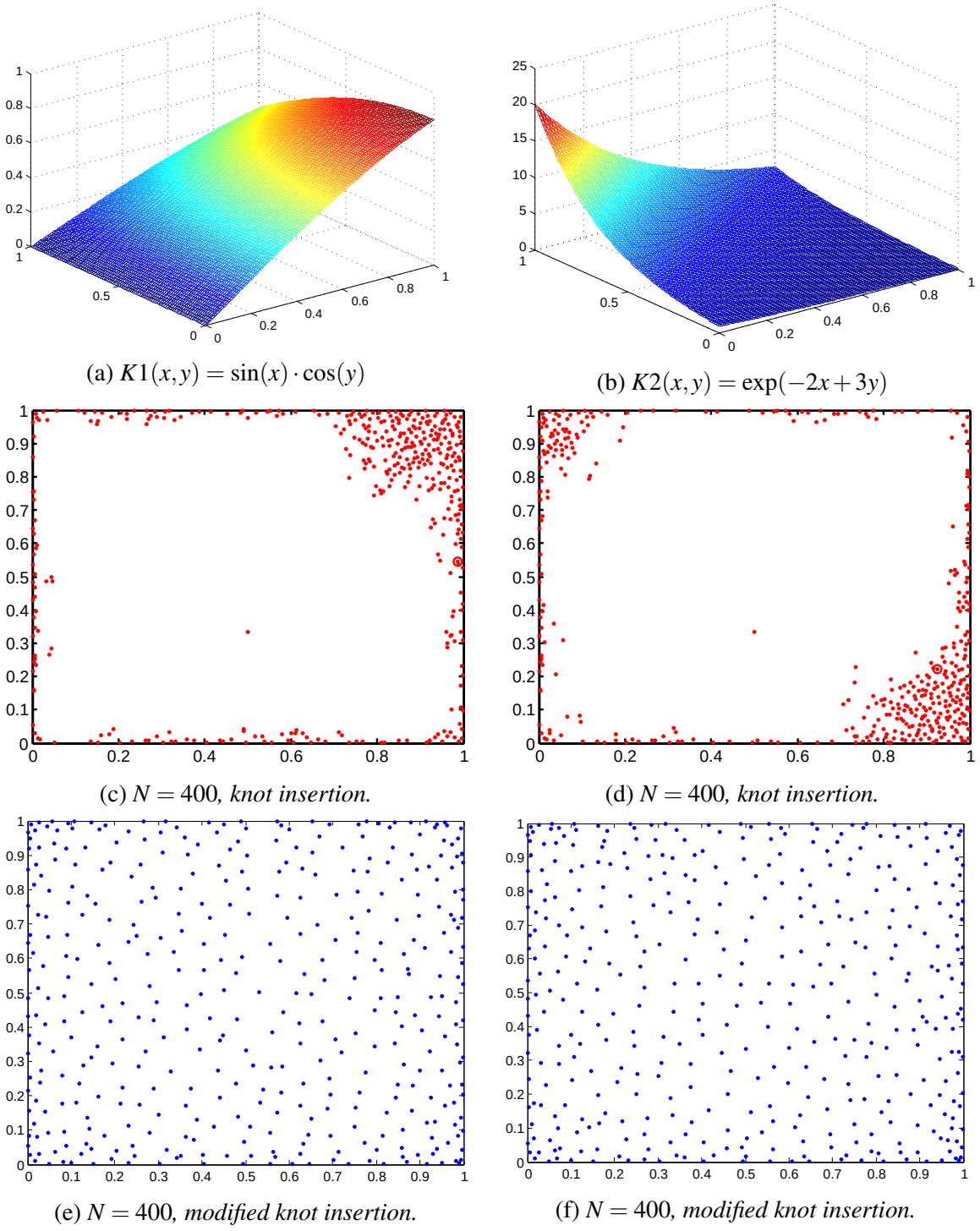


Figure 3.4: *4225 Halton points serve as given dataset in the unit square domain,* $[0,1]^2$. *Both knot insertion and MKI can be used to select a subset of data points from this given dataset, see Figures 3.5(c)-(f).*

Figure 3.6 shows the approximation error of MQ RBF interpolation at $\mathcal{X}$ for test functions $K1$ (3.15) and $K2$ (3.16), using RECV c with 400 Halton points, 400 regular grid points, and 400 MKI adaptive points. For both test functions, the MKI adaptive points yield results more accurate by at least one order of magnitude, compared to the other considered methods (see Figure 3.7), while also improve the boundary errors associated with regular grid points and Halton points.

Figure 3.8 shows the maximum approximation error, E_m of MQ RBF interpolation, as the number of adaptive points, N, increases, using either Franke's knot insertion with $c = 0.2$, or the MKI method with RECV c. For both test functions $K1$ and $K2$, the accuracy of MQ RBF interpolation using MKI adaptive points (dashed line) improves more rapidly than when using Franke's knot insertion (solid line).



(a) $K1(x,y) = \sin(x) \cdot \cos(y)$

(b) $K2(x,y) = \exp(-2x+3y)$

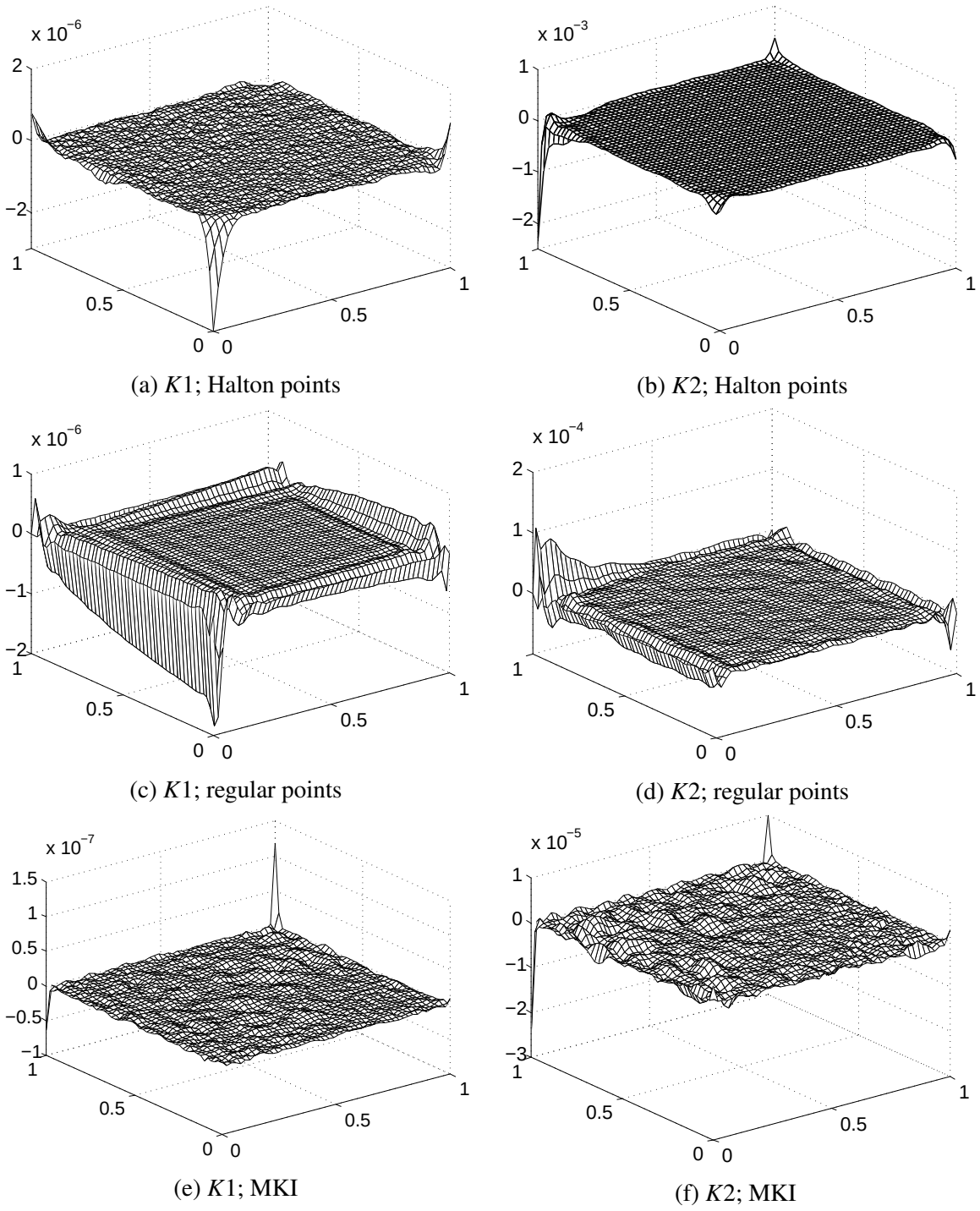
(c) $N = 400$, *knot insertion.*

(d) $N = 400$, *knot insertion.*

(e) $N = 400$, *modified knot insertion.*

(f) $N = 400$, *modified knot insertion.*

Figure 3.5: *Comparison of the adaptive points generated by Franke's knot insertion and Modified Knot Insertion (MKI). Figures (a) and (b) show the functions,* $K1$ (3.15) *and* $K2$ (3.16). *Figures (c) and (e) show the knot insertion and MKI data points for* (3.15)*, while Figures (d) and (f) show the knot insertion and MKI data points for* (3.16)*. The data points generated by the MKI method are uniformly distributed in the center of the domain, and more densely distributed towards to the boundary of the domain. This allows the MKI method to ameliorate the bad behave near boundary issues, see Figure 3.6.*



(a) $K1$; Halton points

(b) $K2$; Halton points

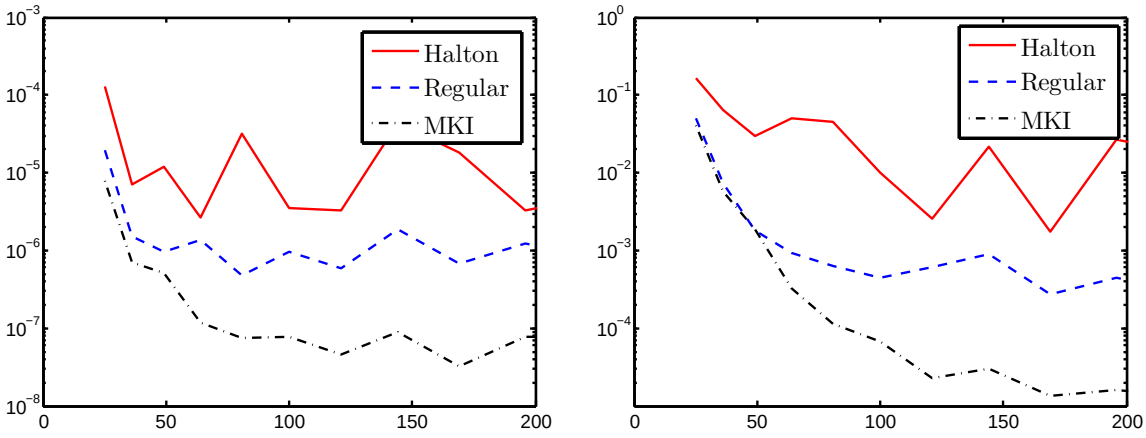
(c) $K1$; regular points

(d) $K2$; regular points

(e) $K1$; MKI

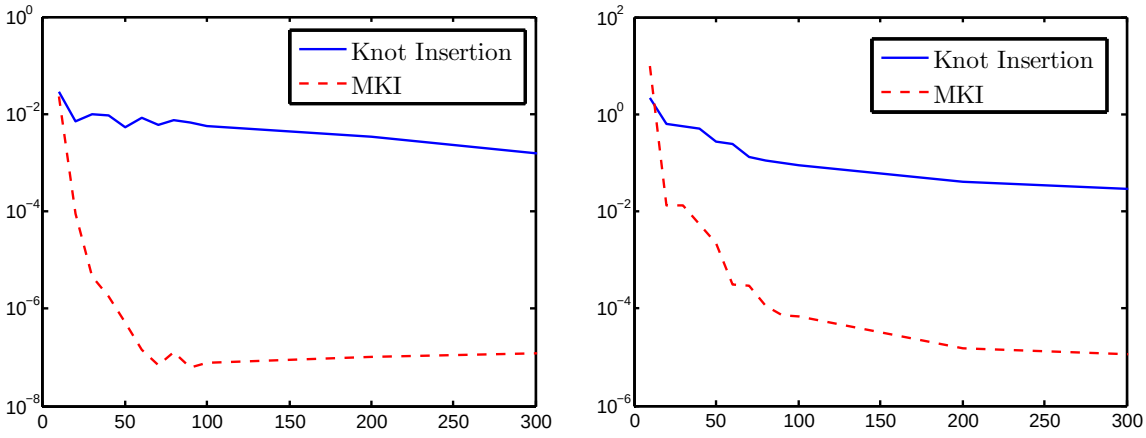
(f) $K2$; MKI

Figure 3.6: *Comparison of boundary error behavior when using adaptive points generated by different methods. For test functions* $K1$ (3.15) *and* $K2$ (3.16)*, Figures (a), (c), and (e) and Figures (b), (d), and (f) depict approximation error generated using* 400 *Halton points, regular grid points, and MKI adaptive points, respectively. For both test functions, the MKI adaptive points yield results more accurate by at least one order of magnitude, compared to the other considered methods (see Figure 3.7), while also improve the boundary errors associated with regular grid points and Halton points.*



(a) E_m *vs. N,* $K1$ (b) E_m *vs. N,* $K2$

Figure 3.7: *Maximum approximation error,* E_m*, versus number of data points, N, when using Halton points, Regular Grid points, and MKI adaptive points. For both test functions,* $K1$ (3.15) *and* $K2$ (3.16)*, the MKI adaptive points yield results more accurate by at least one order of magnitude, compared to the other considered methods.*



(a) $K1$ (b) $K2$

Figure 3.8: *Maximum approximation error,* E_m*, versus number of data points, N, for Franke's Knot Insertion and the MKI methods. For both test functions,* $K1$ (3.15) *and* $K2$ (3.16)*, when using Franke's Knot Insertion, increasing N causes accuracy to slowly improve; however, when using MKI to generate adaptive points, accuracy rapidly increases as N increases.*

Listing 3.4: KnotInsertion.m

```
function []=KnotInsertion()
warning off all;
maxnode =400; % criteria to stop iteration.
func = 1;
c=0.2; % shape parameter.
N=1; % inital points.
M=4225; % total number of points.
% MQ-------------------------------------------------------
rbf = @(c,r) sqrt(c^2+r.^2); % Hardy's MQ.
node = haltonseq(M,2);  % create 4225 Halton points.
%-----------------------------------------------------------
switch func
    case{1}
        testfunction = @(x,y) sin(x).*cos(y);
    case{2}
        testfunction = @(x,y) exp(-2*x+3*y);
    otherwise
        return
end
%-----------------------------------------------------------
z = testfunction(node,node);% right hand side.
J = false(M,1);
J(1:N) = true;
k=N; figure(2)
while (k<maxnode )
    k=k+1;
    DM = DMatrix(node,node(J,:));% distance matrix.
    A = rbf(c,DM); clear DM; % interpolation matrix(MxN)
    Alpha = A\z;
    absr = abs(A*Alpha-z); % residue.
    absr(J) = -Inf;
    [~,idx] = sort(absr,'descend');
    new = idx(1); J(new) = true;
    clf,
    plot(node(J,1),node(J,2),'b.',node(new,1),node(new,2),'ro')
    title(sprintf('adaptive points #%i',k)); pause(1e-5)
end
end
```

Listing 3.5: ModifiedKnotInsertion.m

```
function []=ModifiedKnotInsertion()
warning off all;
maxnode =400; % criteria to stop iteration
func = 1;
M = 4225; N=10; % total number of  points and initial points.
% MQ-----------------------------------------------
rbf = @(c,r) sqrt(c^2+(r).^2); % Hardy's MQ
% Nodes and samples
node = haltonseq(M,2);
%-------------------------------------------------
switch func
    case{1}
        testfunction = @(x,y) sin(x).*cos(y);
    case{2}
        testfunction = @(x,y) exp(-2*x+3*y);
    otherwise
        return
end
%-------------------------------------------------
J = false(M,1); Q = true(M,1);
index = 1:N;
J(index) = true; Q(index) = false;
k=N; figure(10)
while (k < maxnode )
    k=k+1;
    DM = DMatrix(node(J,:),node(J,:));
    DMt = DMatrix(node(Q,:),node(J,:));
    z = testfunction(node(J,1), node(J,2)); %right hand side.
    zt = testfunction(node(Q,1), node(Q,2));
    c=fzero(@(c) CostSPI(c,rbf,DM,DMt,z,zt,1),[0,200],...
        optimset('TolX',1e-3)); % chooing a RECV c for MQ
    A = rbf(c,DM);
    Alpha = A\z;
    DM = DMatrix(node,node(J,:));
    A = rbf(c,DM);
    absr = abs(testfunction(node(:,1), node(:,2))-A*Alpha);
    absr(J) = -Inf;
    [~,idx] = sort(absr,'descend');
    new = idx(1);
    J(new) = true; Q(new)=false;
    clf,
    plot(node(J,1),node(J,2),'b.',node(new,1),node(new,2),'ro')
    title(sprintf('adaptive points #%i',k)); pause(1e-5)
end
end
```

Chapter 4

NUMERICAL EXPERIMENTS UTILIZING THE RECV METHOD IN RBF INTERPOLATION PROBLEMS

MQ RBF approximation often yields significantly different numerical results depending on three factors: 1) value of the shape parameter, c; 2) number of interpolation points, N; and 3) distribution method of the interpolation points.

In addition, MQ RBF interpolation problems suffer from boundary issues. The well-known *knot insertion* adaptive point method (see Algorithm 2, p. 34) can be used to improve overall accuracy during MQ RBF interpolation. However, this work proposes a new adaptive point method, *Modified Knot Insertion* (MKI, see Algorithm 3, p. 35), which can be used to significantly improve the convergence rate of MQ RBF interpolation and reduce inaccuracy due to boundary issues.

In this chapter, eight test functions are utilized to demonstrate the efficacy of both the RECV method and the MKI method for MQ RBF interpolation.

Test functions $F1$–$F6$ are Franke's classic benchmark analytical functions [28, 58] in the computational domain, $[0,1]^2$:

$$\begin{aligned} F1(x,y) = \quad & \frac{3}{4}e^{(-1/4)\left((9x-2)^2+(9y-2)^2\right)} + \frac{3}{4}e^{-(1/49)(9x+1)^2-(1/10)(9y+1)^2} \\ & + \frac{1}{2}e^{-(1/4)\left((9x-7)^2+(9y-3)^2\right)} - \frac{1}{5}e^{-(9x-4)^2-(9y-7)^2}; \end{aligned} \tag{4.1}$$

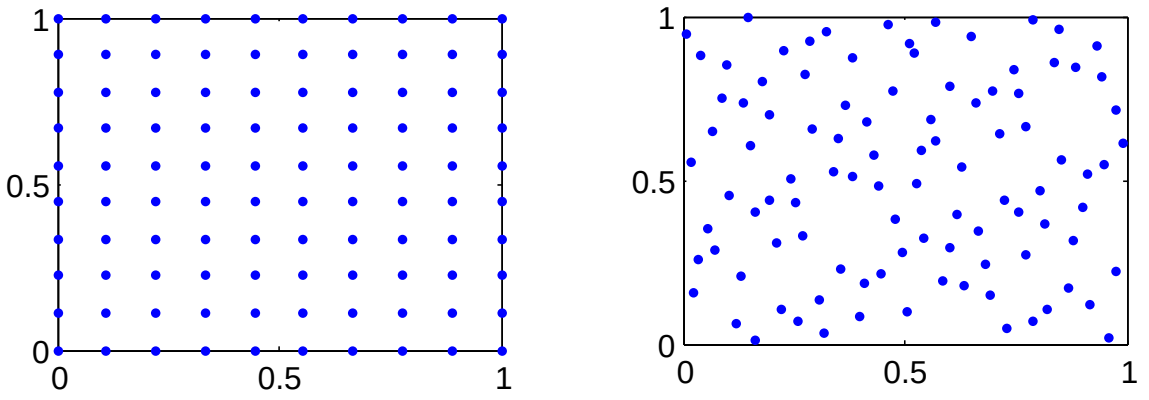
$$F2(x,y) = \quad \frac{1}{9}\left[\tanh\left(9y-9x\right)+1\right], \tag{4.2}$$

$$F3(x,y) = \quad \frac{1}{9}\left[64-81\left(\left(x-\frac{1}{2}\right)^2+\left(y-\frac{1}{2}\right)^2\right)\right]-\frac{1}{2}, \tag{4.3}$$

$$F4(x,y) = \quad \frac{1.25+\cos\left(5.4y\right)}{6\left[1+(3x-1)^2\right]}, \tag{4.4}$$

$$F5(x,y) = \quad \frac{1}{3}\exp\left[-\frac{81}{16}\left(\left(x-\frac{1}{2}\right)^2+\left(y-\frac{1}{2}\right)^2\right)\right], \tag{4.5}$$

$$F6(x,y) = \quad \frac{1}{3}\exp\left[-\frac{81}{4}\left(\left(x-\frac{1}{2}\right)^2+\left(y-\frac{1}{2}\right)^2\right)\right]. \tag{4.6}$$



(a) 100 *regular grid points* (b) 100 *Halton points*

Figure 4.1: *Uniform interpolation Points distributed in* $[0,1]^2$.

The seventh test function (4.7) is a smooth periodic function in the computational domain, $[-2,2]^2$. The eighth test function (4.8), is the built-in `peaks` function from the commercial software Matlab, solved in the computational domain, $[-3,3]^2$:

$$F7(x,y) = \sin(3x)\cdot\cos(3y), \tag{4.7}$$

$$F8(x,y) = 3(1-x)^2\exp\left[-x^2-(y+1)^2\right] - 10\left(\frac{x}{5}-x^3-y^5\right)\exp\left[-x^2-y^2\right] - \frac{1}{3}\exp\left[-(x+1)^2-y^2\right]. \tag{4.8}$$

The following notations are used in this chapter:

– $\mathcal{M}$: total available dataset points, $\Xi = \{\xi_i\}_{i=1}^{\mathcal{M}} \subseteq \mathcal{X}$;

– $\mathcal{N}$: the number of interpolation points, $\mathcal{X} = \{\mathbf{x}_j\}_{j=1}^{\mathcal{N}}$;

– $\mathcal{N}t$: the number of test points;

– c : the shape parameter of MQ;

– $\tilde{u}(x,y)$: MQ RBF approximation solution of RBF interpolation ;

– $u(x,y)$: the analytical solution of RBF interpolation;

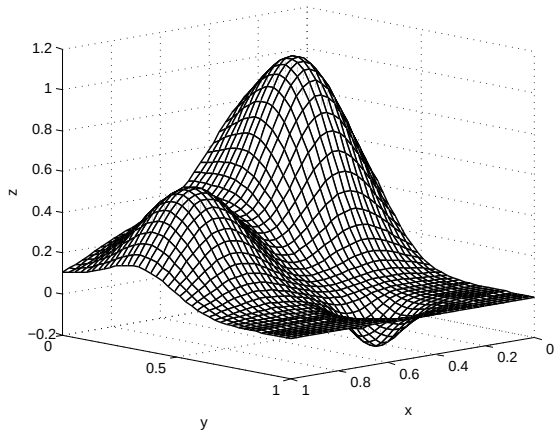
– $\mathbf{r}_m$: maximum residue error: the maximum error at the given points $\mathcal{X} := \{\mathbf{x}_j\}_{j=1}^{N}$:

$$\mathbf{r}_m = \max_{\mathbf{x}\in\mathcal{X}}\{|\tilde{u}(\mathbf{x}) - u(\mathbf{x})|\}.$$

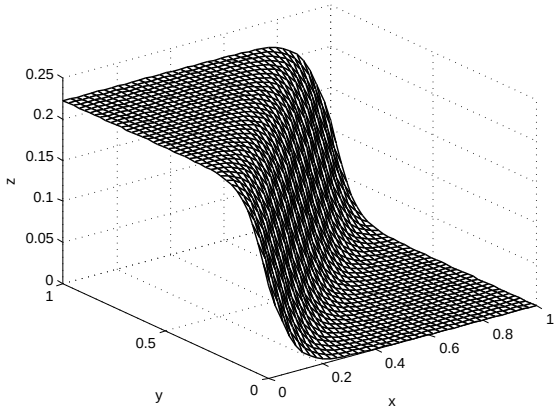
– E_m : maximum approximation error: the maximum error at the chosen test points:

$$E_m = \max_{1\le j\le Nt}\left\{|\tilde{u}(\mathbf{x}_j) - u(\mathbf{x}_j)|\right\}.$$

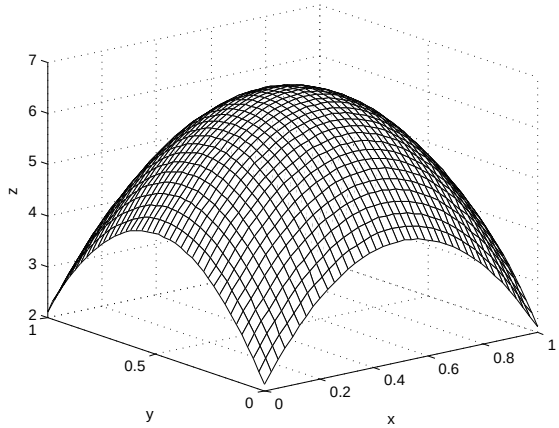
Computation were performed using MATLAB on a system with the following parameters: Windows 7 operating system (32 bits), Intel Core i7-2640M processor, 2.8 GHz CPU, and 3.49 GB memory.



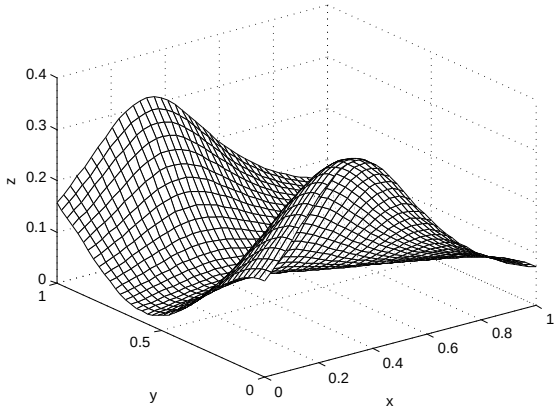
(a) F1



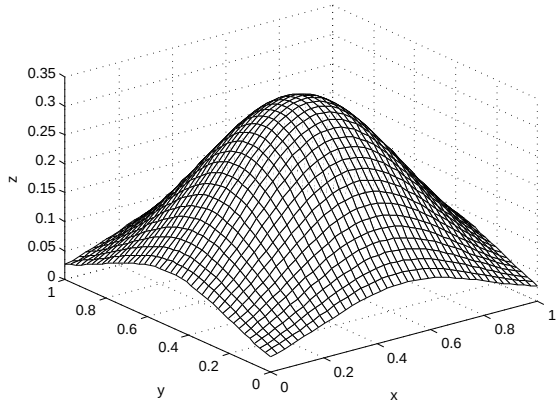
(b) F2



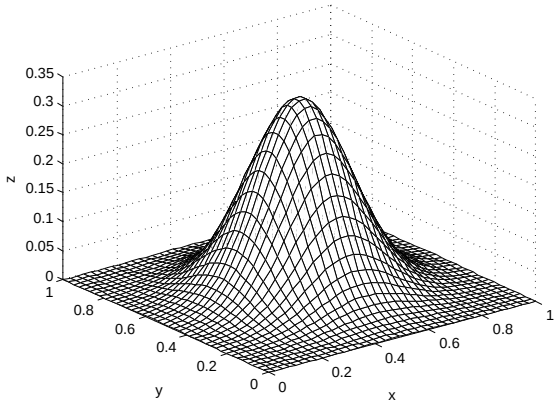
(c) F3



(d) F4

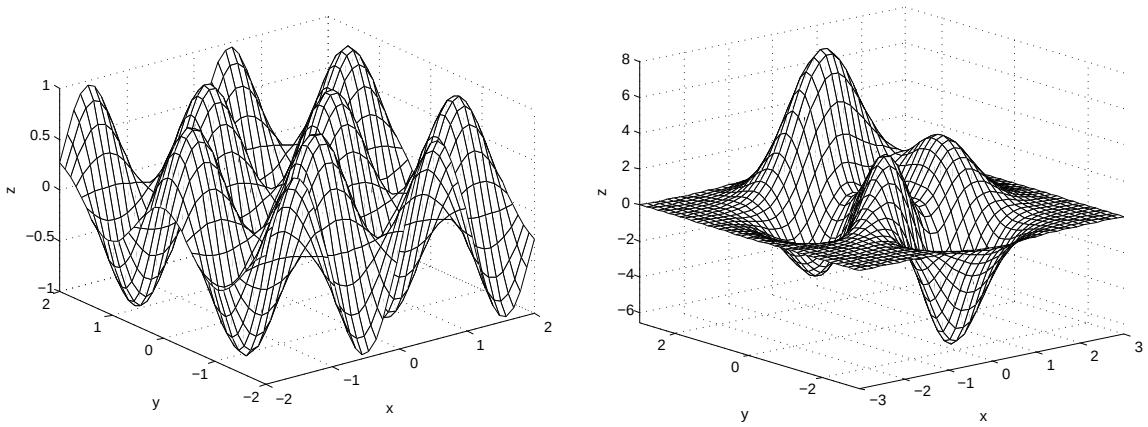


(e) F5



(f) F6

Figure 4.2: *Profiles of test functions* $F1$–$F6$.



(a) F7 (b) F8

Figure 4.3: *Profiles of test functions $F7$–$F8$.*

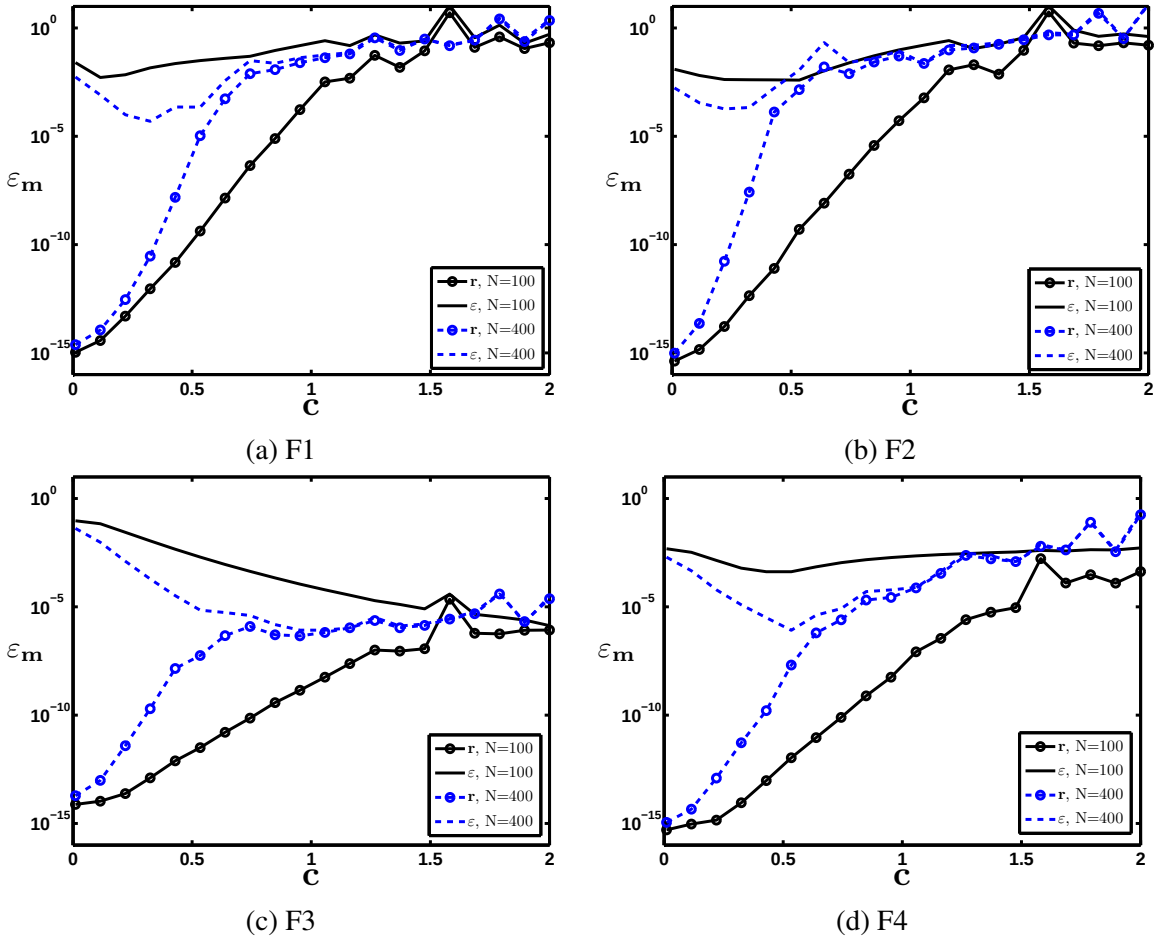
4.1 Numerical Comparison of RECV c and Combined c

In this section, tests are performed to examine the accuracy of interpolation with the different shape parameters introduced in this work, RECV c (I = 1), RECV c (I = 2), and Combined c. The recovery surfaces of test functions $F1$–$F8$ were generated using 100, 400, 900, and $1,600$ regular grid points, as shown in Figure 4.1 (a). Finding RECV c (Strategy 1, p.29), requires another point set (the validation set), which consists of Halton random (quasi-random) points[1] as shown in Figure 4.1 (b). Maximum approximation error, E_m, for MQ RBF interpolation was calculated using $2,500$ regular grid test points in domain Ω.

Figures 4.4 through 4.7 show maximum approximation error, E_m, versus shape parameter, c, for test functions $F1$–$F8$, with increasingly large test point sets. As can be seen in Figure 4.4, optimal shape parameters are quite different for the eight test functions when using lower numbers of collocation points, *i.e.*, $N = 100$. Noticeably, when $N = 100$, functions $F1$, $F2$, $F4$, $F6$ and $F8$ do not really follow the RECV method ((3.3)), which states that the optimal shape parameter should be located where $E_m \approx \mathbf{r}_m$. However, in greater density cases ($N = 400$, $N = 900$, and $N = 1,600$), the RECV method holds true, and an optimal shape parameter is found.

In other words, for MQ RBF interpolation, in lower density cases the shape parameter found by the RECV method, RECV c (Strategy 1), may be slightly larger than where the optimal shape parameter located; however, this issue can be eliminated by increasing the density of interpolation points.

[1] Halton points were generated using the program `haltonseq.m` created by Daniel Dougherty; the program can be downloaded from the website Matlab Central File Exchange.



(a) F1 (b) F2

(c) F3 (d) F4

Figure 4.4: *Figures (a)–(d) show maximum approximation error, E_m, and maximum residue error, $\mathbf{r}_m$, versus shape parameter, c, for test functions $F1$–$F4$, using either* 100 *or* 400 *regular grid points, N, in the domain* Ω.

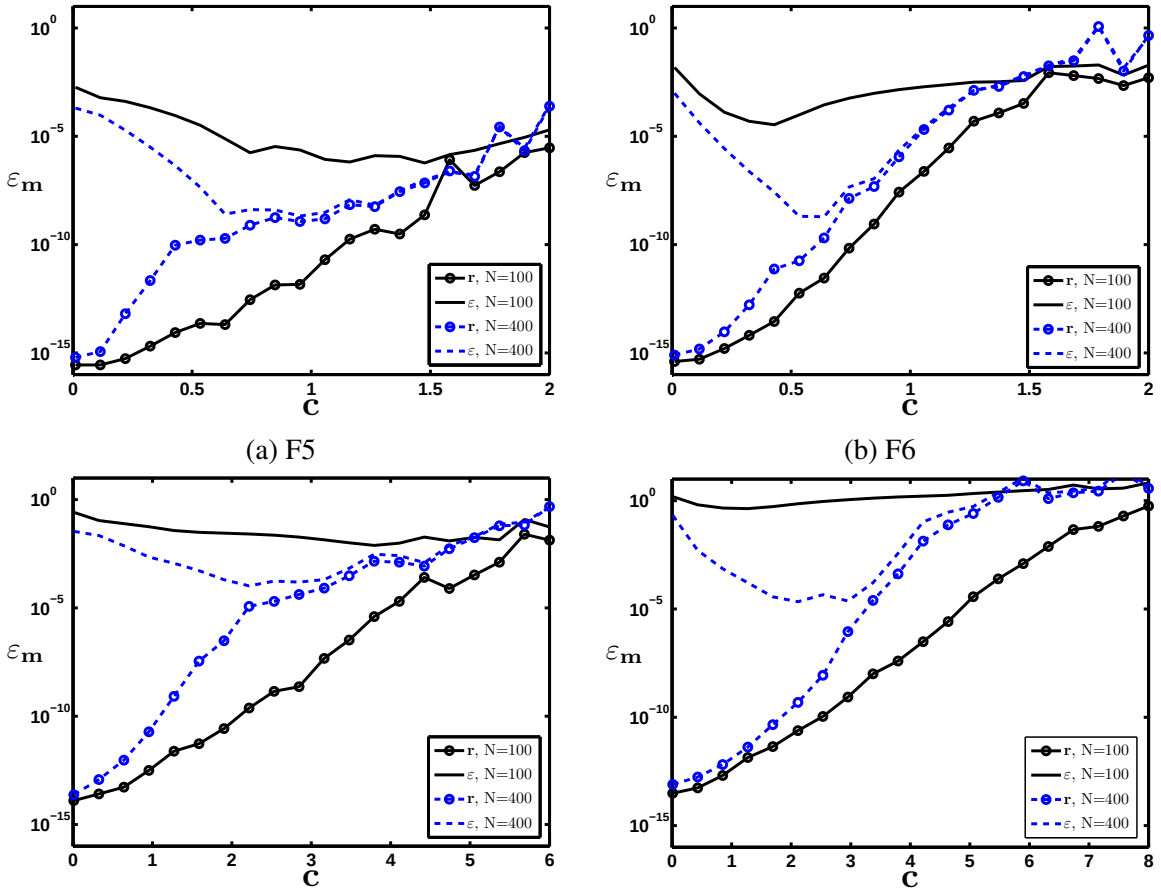
Table 4.1 shows absolute maximum approximation error, E_m, for MQ approximations with RECV c (I $= 1$), RECV c (I $= 2$), and Combined c, for test functions $F1$–$F8$, with increasingly large test point sets.

Three key points can be derived from the results shown in Table 4.1:

1. In higher density cases (e.g., $N = 900$, and $N = 1600$), the accuracy provided using RECV c (I $= 1$) was generally greater than that obtained using Combined c;

2. In lower density cases (e.g., $N = 100$, and $N = 400$), Combined c can generally provide higher accuracy than RECV c for wide intervals of $c = [0, 100]$.

3. In general, accuracy using RECV c (I $= 2$) is better than accuracy using Combined c or RECV c (I $= 1$) in higher density cases and lower density cases, respectively.

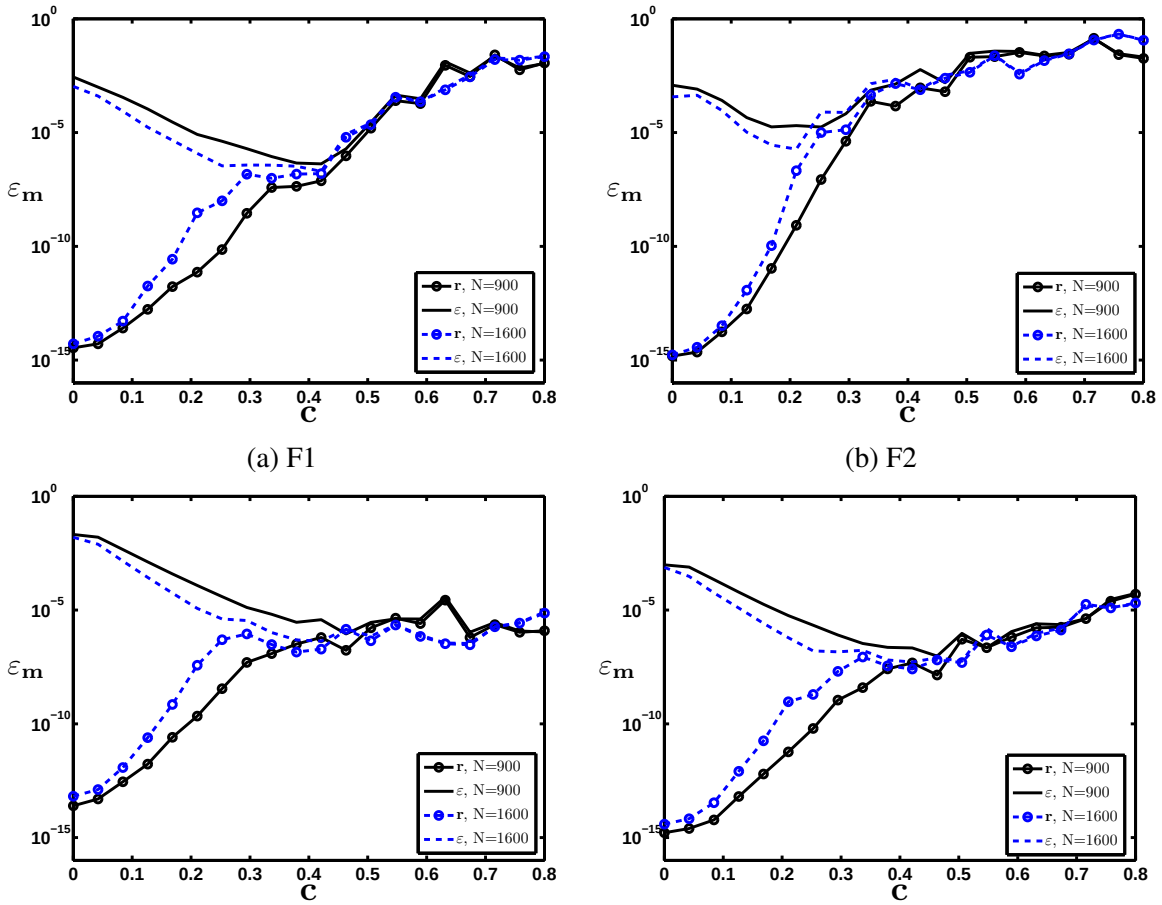


(a) F5 (b) F6

(c) F7 (d) F8

Figure 4.5: *Figures (a)–(d) show maximum approximation error, E_m, and maximum residue error, $\mathbf{r}_m$, versus shape parameter, c, for test functions F5–F8, using either* 100 *or* 400 *regular grid points, N, in the domain* Ω.

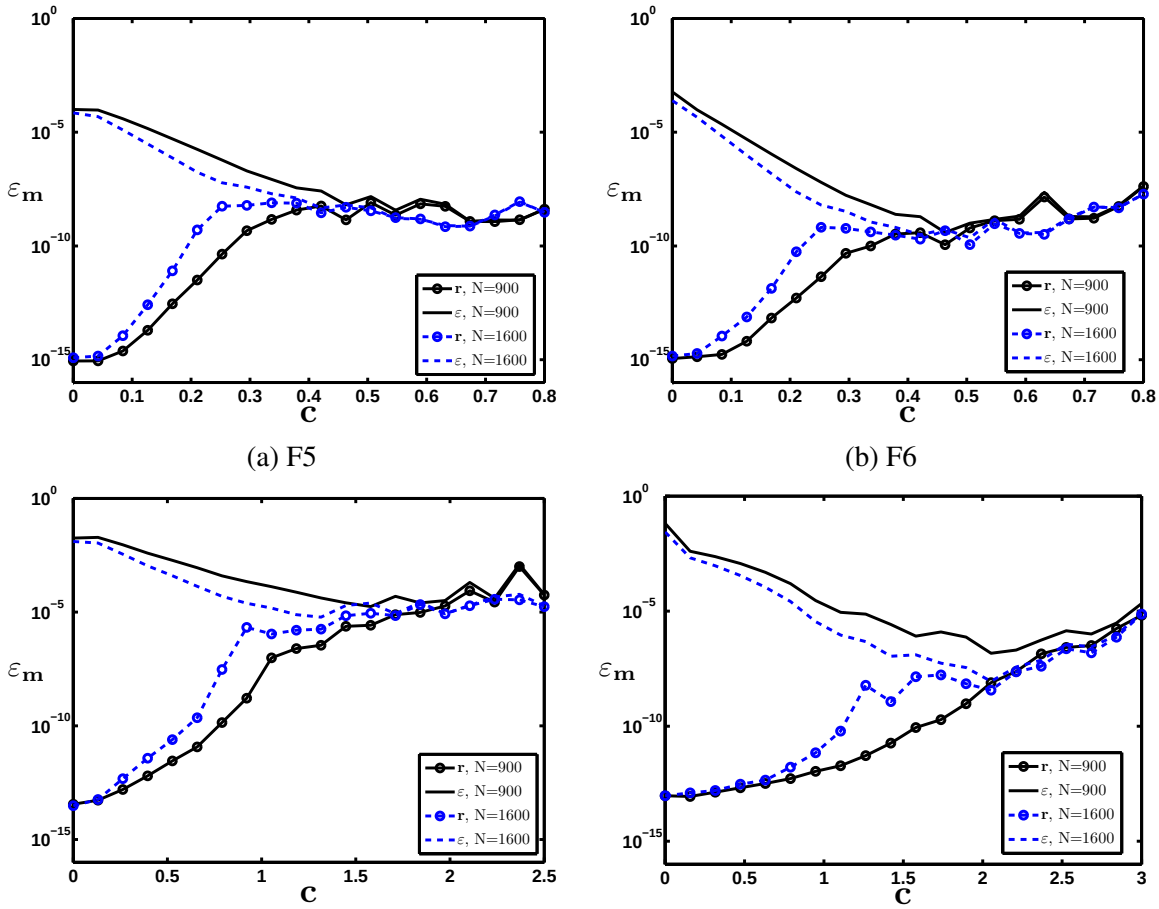
Ultimately, the accuracy of MQ interpolation demonstrated in Table 4.1 shows that either RECV c or Combined c can generally provide an accurate numerical solution. However, as shown in Table 4.2, for higher density datasets, the Combined c method may take more CPU time. Therefore, RECV c may be more suitable for higher density problems, in order to minimize computational costs, while Combined c may be better for lower density problems, to maximize accuracy.



(a) F1 (b) F2

(c) F3 (d) F4

Figure 4.6: *Figures (a)–(d) show maximum approximation error, E_m, and maximum residue error, $\mathbf{r}_m$, versus shape parameter, c, for test functions $F1$–$F4$, using either* 900 *or* $1,600$ *regular grid points, N, in the domain* Ω.



(a) F5 (b) F6

(c) F7 (d) F8

Figure 4.7: *Figures (a)–(d) show maximum approximation error, E_m, and maximum residue error, $\mathbf{r}_m$, versus shape parameter, c, for test functions F5–F8, using either* 900 *or* $1,600$ *regular grid points, N, in the domain* Ω.

Table 4.1: *Accuracy of MQ RBF Interpolation with Different Shape Parameters, RECV c ($I = 1$), RECV c ($I = 2$), and Combined c. Results were calculated in interval $c \in [0, 100]$, for test functions F1–F8, using various numbers of interpolation Points, N. Accuracy for RECV c ($I = 1$) and RECV c ($I = 2$) is usually better than for Combined c, but maximum approximation error, E_m, is color-coded to indicate when results are* **similar**, *or* **worse** *than those obtained with Combined c. Ultimately, either of the three methods obtain highly accurate numerical results, though each may eventually prove to be more suitable for specific situations.*

		RECV c (I = 1)		RECV c (I = 2)		Combined c	
$u(x)$	N	c	E_m	c	E_m	c	E_m
F1	100	1.229	2.99E-01	1.045	2.02E-01	0.355	1.69E-02
	400	0.555	3.02E-04	0.483	2.48E-04	0.424	2.17E-04
	900	0.370	8.08E-07	0.327	1.04E-06	0.259	3.72E-06
	1600	0.270	3.10E-07	0.229	6.38E-07	0.196	1.91E-06
F2	100	1.282	1.91E+00	1.117	2.74E-01	0.321	4.07E-03
	400	0.478	6.80E-03	0.378	3.47E-04	0.212	1.78E-04
	900	0.254	1.72E-05	0.235	1.87E-05	0.227	1.93E-05
	1600	0.197	3.59E-06	0.190	2.91E-06	0.191	2.91E-06
F3	100	1.552	5.73E-06	1.309	1.73E-05	1.151	3.67E-05
	400	0.563	9.34E-06	0.473	1.56E-05	0.425	3.48E-05
	900	0.329	6.49E-06	0.274	2.33E-05	0.269	2.35E-05
	1600	0.253	2.77E-06	0.207	1.40E-05	0.192	2.37E-05
F4	100	1.822	4.09E-03	1.536	3.47E-03	1.013	2.11E-03
	400	0.567	1.25E-06	0.515	1.07E-06	0.448	2.48E-06
	900	0.385	1.41E-07	0.328	4.03E-07	0.302	6.72E-07
	1600	0.272	1.35E-07	0.232	3.12E-07	0.216	5.13E-07
F5	100	1.809	4.64E-06	1.470	6.18E-07	1.006	1.48E-06
	400	0.624	3.25E-09	0.491	1.50E-07	0.554	2.65E-08
	900	0.283	4.19E-07	0.297	1.91E-07	0.266	3.81E-07
	1600	0.237	6.00E-08	0.216	1.50E-07	0.211	1.78E-07
F6	100	1.472	3.56E-03	1.242	3.16E-03	0.522	8.42E-05
	400	0.572	4.18E-09	0.485	6.36E-09	0.448	1.61E-08
	900	0.352	4.36E-09	0.286	2.07E-08	0.286	2.10E-08
	1600	0.273	2.01E-09	0.208	2.56E-08	0.201	3.54E-08
F7	100	5.282	2.33E-02	4.673	1.29E-02	3.272	1.23E-02
	400	2.244	1.04E-04	2.030	1.28E-04	1.834	2.53E-04
	900	1.450	3.05E-05	1.188	7.50E-05	0.736	5.55E-04
	1600	1.053	1.22E-05	0.852	3.57E-05	0.975	1.72E-05
F8	100	8.458	5.88E+00	6.737	3.72E+00	2.612	9.57E-01
	400	3.123	2.87E-05	2.780	1.14E-05	2.217	1.33E-05
	900	2.197	2.17E-07	1.901	6.79E-07	1.702	1.23E-06
	1600	1.567	7.02E-08	1.318	3.03E-07	1.228	6.10E-07

Table 4.2: *Computational time in seconds for interpolation with different shape parameters, RECV c ($I = 1$), RECV c ($I = 2$), and Combined c. Results were calculated in interval $c \in [0, 100]$, for test functions $F1$–$F8$, using various numbers of interpolation points, N. Results in this table correspond to the computations described in Table 4.1. In low density problems, Combined c is nearly as fast as RECV c ($I = 1$) and RECV c ($I = 2$), and is more stable (see Table 4.1); however, in high density problems, Combined c becomes less accurate than RECV c ($I = 1$) and RECV c ($I = 2$), and requires considerably more computational time.*

$u(x)$	N	RECV c (I = 1)	RECV c (I = 2)	Combined c
F_1	100	0.14	0.12	0.19
	400	0.20	0.20	0.75
	900	1.25	1.49	4.15
	1600	3.58	3.70	19.38
F_2	100	0.11	0.11	0.16
	400	0.21	0.24	0.52
	900	1.21	0.97	4.03
	1600	5.00	4.52	20.01
F_3	100	0.14	0.11	0.16
	400	0.21	0.20	0.69
	900	1.51	1.29	4.54
	1600	4.95	4.41	19.25
F_4	100	0.14	0.33	0.16
	400	0.18	0.16	0.70
	900	1.17	1.18	4.02
	1600	3.63	3.95	21.24
F_5	100	0.12	0.10	0.16
	400	0.26	0.21	0.66
	900	1.00	1.31	4.46
	1600	4.92	4.30	17.68
F_6	100	0.11	0.11	0.17
	400	0.19	0.20	0.66
	900	1.41	1.31	4.34
	1600	3.75	5.02	17.96
F_7	100	0.11	0.11	0.16
	400	0.19	0.18	0.88
	900	1.23	1.28	4.40
	1600	4.38	4.38	26.54
F_8	100	0.18	0.11	1.02
	400	0.22	0.18	0.67
	900	0.81	1.09	4.77
	1600	3.79	3.92	21.66

4.2 Generalizability of RECV Shape Parameters

Previously, within the field, Carlson and Foley [7] concluded that "the optimal shape parameter, c, for MQ RBF interpolation in $\mathbb{R}^2$ is most strongly influenced by the values of the function". While this may be somewhat true in low density cases (e.g., $N = 100$ and $N = 400$), the numerical results in Table 4.1 demonstrate that, in higher density cases (e.g., $N = 900$ and $N = 1600$), the different characteristics of individual functions become less and less significant. This in turn suggests that, so long as the interpolation points are shared and the fill distance, h, is sufficiently small, the RECV c values for individual functions may be functionally interchangeable. If this is true, then we should be able to select an RECV c value using any known function within the same computational domain, Ω, and using the same interpolation points, $\mathcal{X} := \{\mathbf{x}_j\}_{j=1}^{N}$, to solve a given problem with acceptably high accuracy.

To test this idea, Figure 4.8 shows test functions $F1$–$F8$, as shown in Figures 4.4 through 4.7, re-plotted in terms of $\mathrm{I}(c)$, as described in (3.14), for $N = 100$, 400, 900, and $1,600$ regular grid interpolation points. In this way we are able to compare the shape parameter indicator curves for each test function, and see that, while they can be a little different in lower density cases, for test functions $F1$–$F6$, which all share the same interpolation points, the curves become increasingly close as N increases. This supports the idea that function values have minimal influence when choosing RECV c via Strategy 1 for MQ RBF interpolation.

To test this theory, in the following numerical experiment, test functions $F1$–$F8$ are reexamined using RECV c values generated by two arbitrary known functions:

$$K1(x,y) = \quad \sin(x) + \cos(y), \tag{4.9}$$

$$K2(x,y) = \quad (1 - \tanh(10 * \sin(1.8 * x) - 10 * y - 2.6))/2. \tag{4.10}$$

The function $K1$ is a smooth trigonometry function which moves slowly up and down within the domain, while the $K2$ function, which is usually used to examine singular perturbation in boundary layer problem, is considerably more difficult to accurately approximate.

The RECV c values shown in Table 4.3 were chosen using Strategy 1 ($\mathrm{I} = 1$) with functions $K1$ and $K2$, (4.9) and (4.10), for the square domain matching that used with each of the eight test functions $F1$–$F8$. The $K1$ and $K2$ RECV c values were then utilized to approximate the test functions $F1$–$F8$. The results achieved using the $K1$ and $K2$ RECV c values (see Table 4.4) show that the accuracy of MQ RBF approximation using generic RECV c values from any function within the same domain, using the same interpolation points, can produce a very accurate solution in higher density cases. The results using the

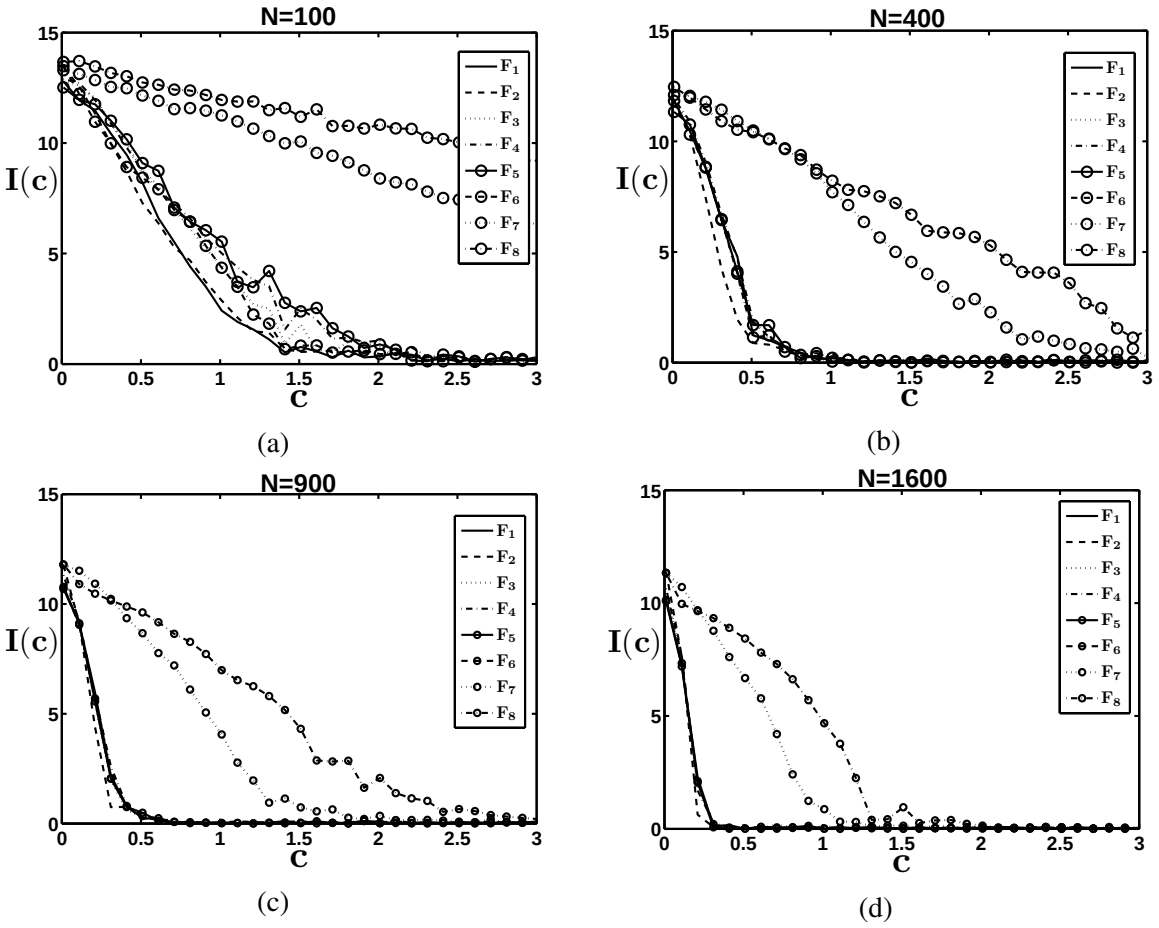


Figure 4.8: *Comparison of MQ RBF interpolation error behavior using various N uniformly distributed points. In this figure, the error behaviors shown in Figures 4.4 and 4.7 are re-plotted comparing the values of Shape Parameter Indicator (SPI), $I(c)$* (3.14)*, versus shape parameter, c. Note that for functions within the same domain, using the same interpolation points ($F1$–$F6$), the curves describing error behavior are increasingly similar as N becomes larger.*

$K1$ and $K2$ Combined c values show that using LOOCV method with the RECV method (i.e., re-scaling the LOOCV c interval to be $[0, \text{RECV } c]$) can successfully avoid the local optimum issue and choose an optimal shape parameter, even when using the Combined c from generic functions.

Table 4.3: *RECV c values generated from known functions, $K1$* (4.9) *and $K2$* (4.10)*, in various square domains. These generic RECV c values are used to test the generalizability of known function RECV c for application with unknown functions sharing the same domain and interpolation points (see Table 4.4).*

		RECV c	
Ω	N	$K1$	$K2$
$[0,1]^2$	100	1.636	1.248
	400	0.649	0.475
	900	0.394	0.280
	1600	0.244	0.198
$[-2,2]^2$	100	6.640	5.206
	400	2.502	2.034
	900	1.509	1.172
	1600	0.934	0.761
$[-3,3]^2$	100	9.325	7.030
	400	3.658	3.223
	900	2.207	1.675
	1600	1.489	1.116

Table 4.4: *Generalizability of Shape Parameter, c, values created with known functions. This table illustrates the maximum approximation error, E_m, of MQ RBF interpolation with generic Shape Parameter, c, values selected using known functions, $K1$ (4.9) and $K2$ (4.10), with various N values, in the same domain, sharing the same interpolation points (see Table 4.3). Note that for all cases, the MQ RBF approximation is accurate using generic c values generated with either known function.*

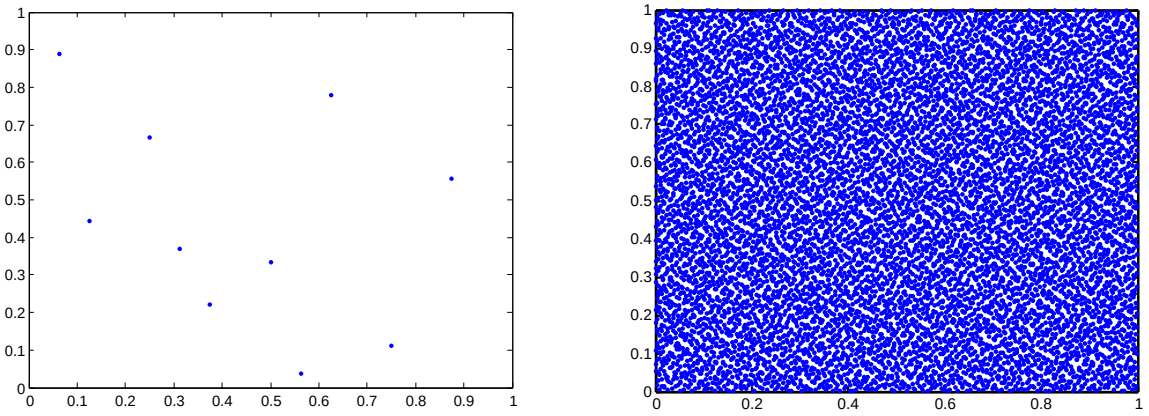
		RECV c ($K1$)	Combined c ($K1$)		RECV c ($K2$)	Combined c ($K2$)	
$u(x)$	N	E_m	c	E_m	E_m	c	E_m
F_1	100	1.98E+00	0.380	1.88E-02	3.42E-01	0.325	1.45E-02
	400	1.37E-03	0.429	2.24E-04	2.49E-04	0.422	2.14E-04
	900	1.16E-06	0.259	3.73E-06	2.94E-06	0.268	3.19E-06
	1600	3.18E-07	0.205	1.43E-06	1.64E-06	0.199	1.76E-06
F_2	100	6.14E-01	0.380	4.08E-03	4.37E-01	0.325	4.08E-03
	400	8.71E-02	0.212	1.78E-04	1.83E-02	0.212	1.78E-04
	900	1.31E-03	0.228	1.93E-05	2.06E-05	0.226	1.94E-05
	1600	9.09E-06	0.193	2.86E-06	2.63E-06	0.191	2.90E-06
F_3	100	7.87E-06	1.270	2.03E-05	2.52E-05	1.180	3.15E-05
	400	3.24E-06	0.512	9.67E-06	2.07E-05	0.420	3.67E-05
	900	9.55E-06	0.274	2.07E-05	2.25E-05	0.266	2.71E-05
	1600	2.72E-06	0.209	1.41E-05	1.79E-05	0.195	2.17E-05
F_4	100	5.39E-03	1.095	2.38E-03	2.76E-03	1.006	2.09E-03
	400	9.64E-07	0.433	2.96E-06	2.10E-06	0.442	2.67E-06
	900	2.41E-07	0.309	5.72E-07	1.34E-06	0.261	1.75E-06
	1600	1.53E-07	0.207	6.86E-07	8.28E-07	0.192	1.08E-06
F_5	100	2.49E-07	1.319	1.33E-06	1.09E-06	0.656	5.97E-06
	400	6.17E-09	0.512	7.29E-08	2.37E-07	0.429	5.25E-07
	900	9.77E-08	0.275	3.90E-07	3.52E-07	0.257	5.35E-07
	1600	3.86E-08	0.205	2.27E-07	2.56E-07	0.191	3.45E-07
F_6	100	2.21E-02	0.843	9.47E-04	2.81E-03	0.843	9.47E-04
	400	1.29E-09	0.512	3.10E-09	1.24E-08	0.436	2.26E-08
	900	8.17E-09	0.287	2.12E-08	3.28E-08	0.261	4.71E-08
	1600	3.84E-09	0.205	3.26E-08	3.81E-08	0.193	5.05E-08
F_7	100	1.25E-01	3.273	1.23E-02	1.27E-02	3.273	1.23E-02
	400	1.02E-04	1.961	1.68E-04	3.88E-04	0.905	2.77E-03
	900	3.84E-05	0.736	5.55E-04	1.57E-04	0.736	5.55E-04
	1600	1.18E-05	0.784	2.10E-04	5.72E-05	0.650	1.44E-04
F_8	100	5.59E+00	2.612	9.57E-01	3.52E+00	2.612	9.57E-01
	400	3.52E-04	2.224	1.44E-05	4.62E-05	2.214	1.28E-05
	900	2.17E-07	1.865	8.76E-07	8.52E-07	1.500	8.53E-07
	1600	6.21E-08	1.387	1.28E-07	9.11E-07	1.118	9.11E-07

4.3 Modified Knot Insertion

MQ approximation can be optimized in two ways: 1) by adjusting the value of the shape parameter [35, 28]; and 2) by utilizing adaptive points [18, 45].

Previously, Franke's knot insertion method was discussed (Algorithm 2, p. 34), and a new method, Modified Knot Insertion (MKI) was introduced (Algorithm 3, p. 35).

The following numerical experiments will explore the degree to which MKI adaptive points, when utilized with RECV c (Strategy 1, p. 29), can improve the accuracy of MQ RBF interpolants.



(a) 10 *initial points* (b) $10,000$ *Halton points*

Figure 4.9: *Left:* 10 *initial randomly selected points. Right:* $10,000$ *Halton points distributed in the unit square domain* $[0,1]^2$.

Density

An adaptive algorithm for interpolation usually provides a set of data points based on the characteristics of both the numerical method being utilized and the unknown function being recovered. For this reason, the adaptive points generated for different unknown functions differ from each other, with each set matching the specific type of problem being addressed (see Figures 3.5 (c) and (d), p. 37). However, when the MKI adaptive point method was introduced, it seemed that the MKI points generated for different test functions were quite similar in appearance, with uniform distribution in the center of the domain and greater density near the boundary (see Figures 3.5 (e) and (f), p. 37). In the following numerical experiment, we examine the density of MKI adaptive point distribution to determine if this observation is functionally accurate. Shape parameter values associated with Strategy 1 (p. 29), RECV c ($\mathrm{I}(c) = 1$), are utilized with test functions $F1$–$F8$ (p. 42),

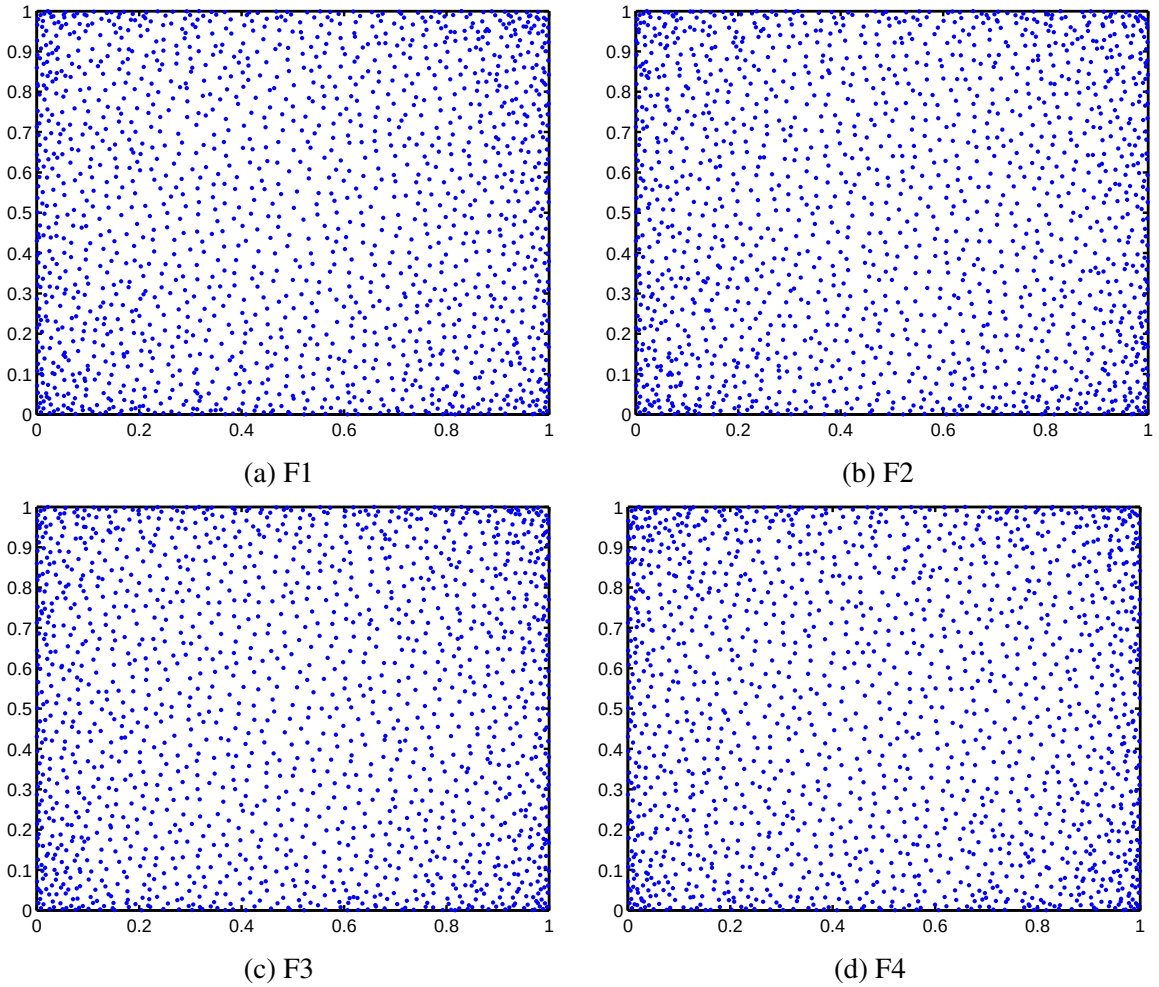
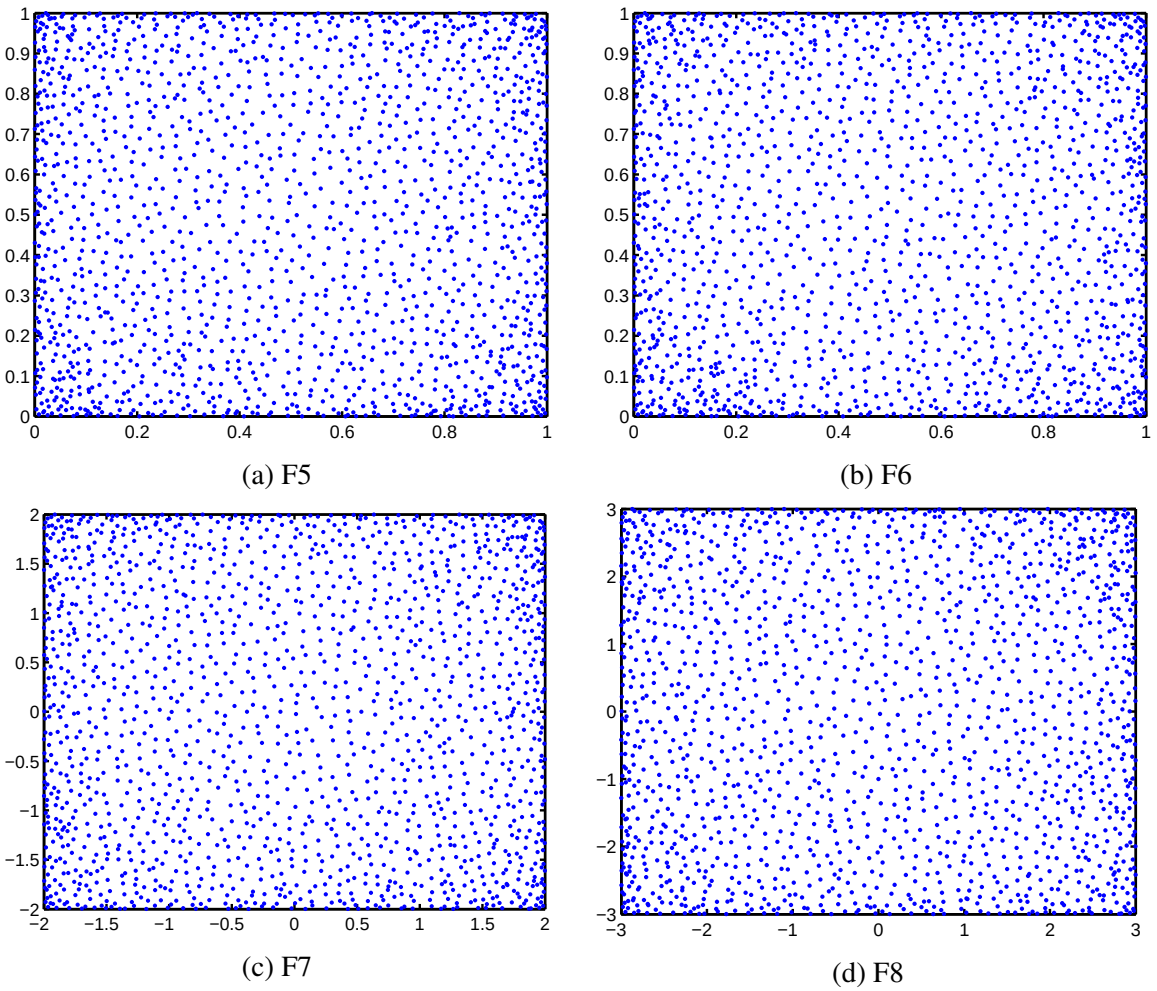


Figure 4.10: $1,600$ *adaptive points generated using the MKI method for test functions* $F1$–$F4$. *For each function, the MKI adaptive points are uniformly distributed in the center, but more densely distributed near the boundary of the domain.*

with MKI adaptive points generated using 10 randomly chosen initial points from $10,000$ Halton points, as depicted in Figure 4.9.

Figures 4.10 and 4.11 show 1600 MKI adaptive points for test functions $F1$–$F8$ in different square domains. As predicted, the MKI point sets for the different functions all look functionally identical, uniformly distributed in the center of the domain, and with greater density near the boundary.

To test the density distribution consistency of MKI adaptive points as N changes, Figures 4.12 and 4.13 show various N MKI points generated for test function $F1$ in the unit square domain $[0,1]^2$. For each value of N, the MKI adaptive points are uniformly distributed, with a higher density concentration of points near the boundary.



(a) F5 (b) F6

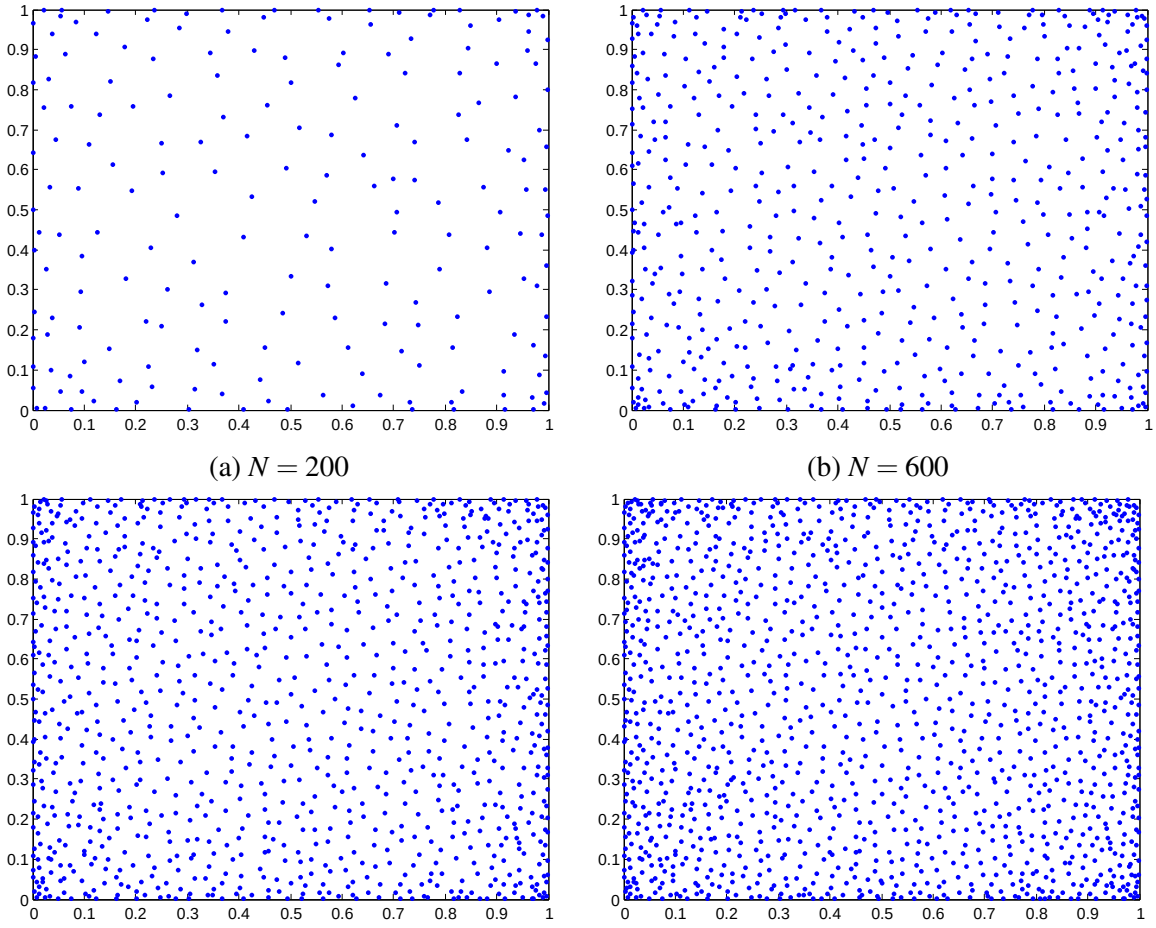
(c) F7 (d) F8

Figure 4.11: 1,600 *adaptive points generated using the MKI method for test functions* $F5$–$F8$. *For each function, the MKI adaptive points are uniformly distributed in the center, but more densely distributed near the boundary of the domain.*

Convergence rate

The RECV method determines optimal shape parameter, c, values utilizing the convergence of maximum residue error, $\mathbf{r}_m$, and maximum approximation error, E_M. While it was demonstrated previously that MKI adaptive points could achieve more accurate results than Halton uniformly distributed points (see Figures 3.6 and 3.7), the following numerical experiments examine the ways in which MKI adaptive points affect convergence for different test functions, when N is constant (see Figures 4.14 and 4.15) and when using increasingly large collocation point sets (see Figures 4.16 and 4.17).

Figures 4.14 and 4.15 show the error convergence behavior of MQ RBF interpolations constructed using either 600 MKI points or 600 Halton points for test functions $F1$–$F8$.



(a) $N = 200$ (b) $N = 600$

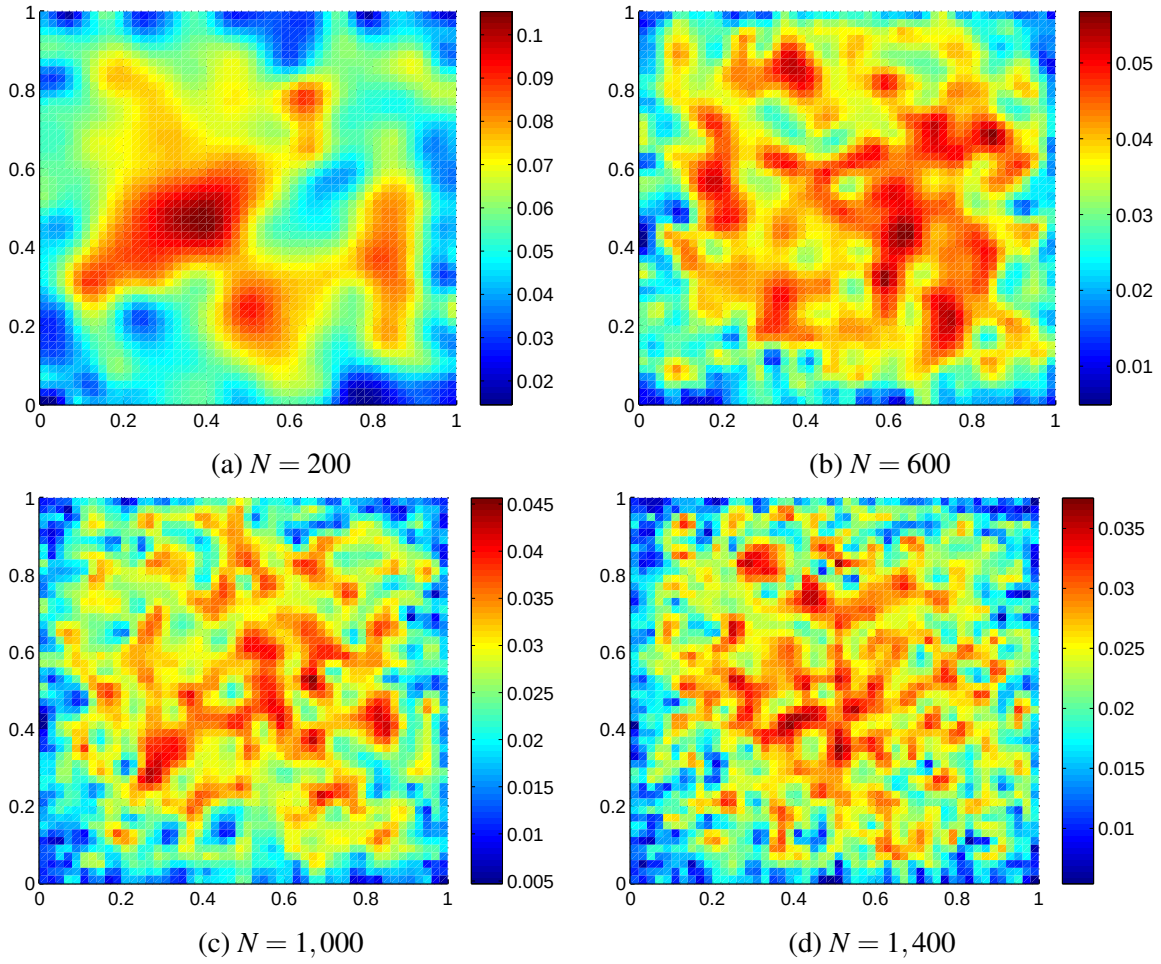
(c) $N = 1,000$ (d) $N = 1,400$

Figure 4.12: *Various numbers of adaptive points, N, generated using the MKI method for test function F*1. *For each case of N, the MKI adaptive points are uniformly distributed in the center, but more densely distributed near the boundary of the domain.*

Two details of significance can be derived from the results:

1. Convergence rates of E_m are faster using MKI adaptive points than when using Halton points, meaning MKI adaptive points can provide smaller RECV c values but more accurate MQ RBF approximations.

2. For both MKI adaptive points and Halton uniformly distributed points, the residue error, $\mathbf{r}_m$, curves are nearly overlapping. This indicates that using MKI adaptive points has little influence on the condition number, $\kappa(\mathbf{A})$, for the generated interpolation matrices. Thus, while MKI points are able to yield more accurate results than Halton points, they are unable to eliminate or reduce the condition number, $\kappa(\mathbf{A})$.

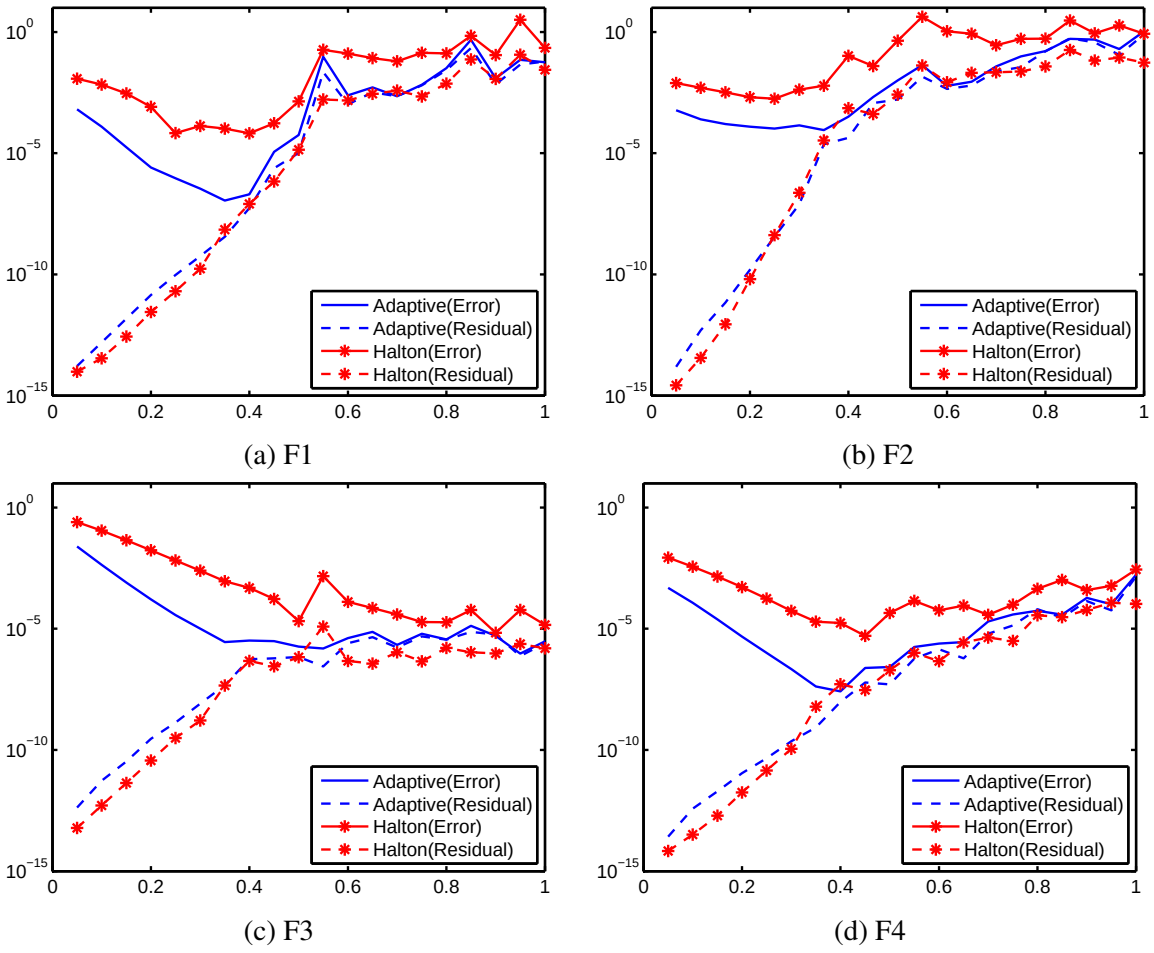


(a) $N = 200$ (b) $N = 600$

(c) $N = 1,000$ (d) $N = 1,400$

Figure 4.13: *Density of adaptive points generated using the MKI method. Figures (a)–(d) depict the density of the adaptive points generated using the MKI method for test function $F1$ with various numbers of points, N. For each value of N, the MKI adaptive points are uniformly distributed in the center, but more densely distributed near the edges of the domain.*

For test functions $F1$–$F8$, Figures 4.16 and 4.17 show maximum approximation error, E_m, of MQ RBF interpolation with RECV c and increasingly large sets of either MKI adaptive points or Halton points. Initially, with small numbers of interpolation points, the accuracy achieved using Halton points is similar to the accuracy achieved using MKI points. However, as N increases, the accuracy of interpolation using MKI points increases dramatically, quickly achieving far more accurate results. Table 4.5 shows the optimal solution of MQ RBF interpolation in term of N MKI adaptive points, and Halton points for eight test functions. To reach the best solution, the MQ approximation constructed by adaptive points only need coupled hundreds points which are much fewer than the Halton.



(a) F1 (b) F2

(c) F3 (d) F4

Figure 4.14: *Figures (a)–(d) show maximum approximation error, E_m, and maximum residue error, $\mathbf{r}_m$, versus shape parameter, c, for test functions $F1$–$F4$, using 600 Halton uniformly distributed points or MKI adaptive points. Because it is controlled by condition number, $\kappa(\mathbf{A})$, $\mathbf{r}_m$ is similar for both point distribution methods; however, E_m is less when using MKI adaptive points, allowing faster and closer Error-Residue convergence, and revealing optimal c values.*

Generalizability of MKI Adaptive Points

As demonstrated in Figures 4.10–4.11 (p. 57 and 58), the adaptive points generated by the MKI method show very similar patterns of distribution for different test functions. In Figures 4.16–4.17, error behavior is plotted for test functions $F1$–$F8$, using increasingly large sets of either MKI adaptive points or Halton uniformly distributed points; results show that MKI adaptive points allow for far more accurate approximations, with quicker convergence (see Table 4.5). Finally, to test the generalizability of MKI adaptive points, test functions $F1$–$F8$ were approximated using both their own set of MKI adaptive points

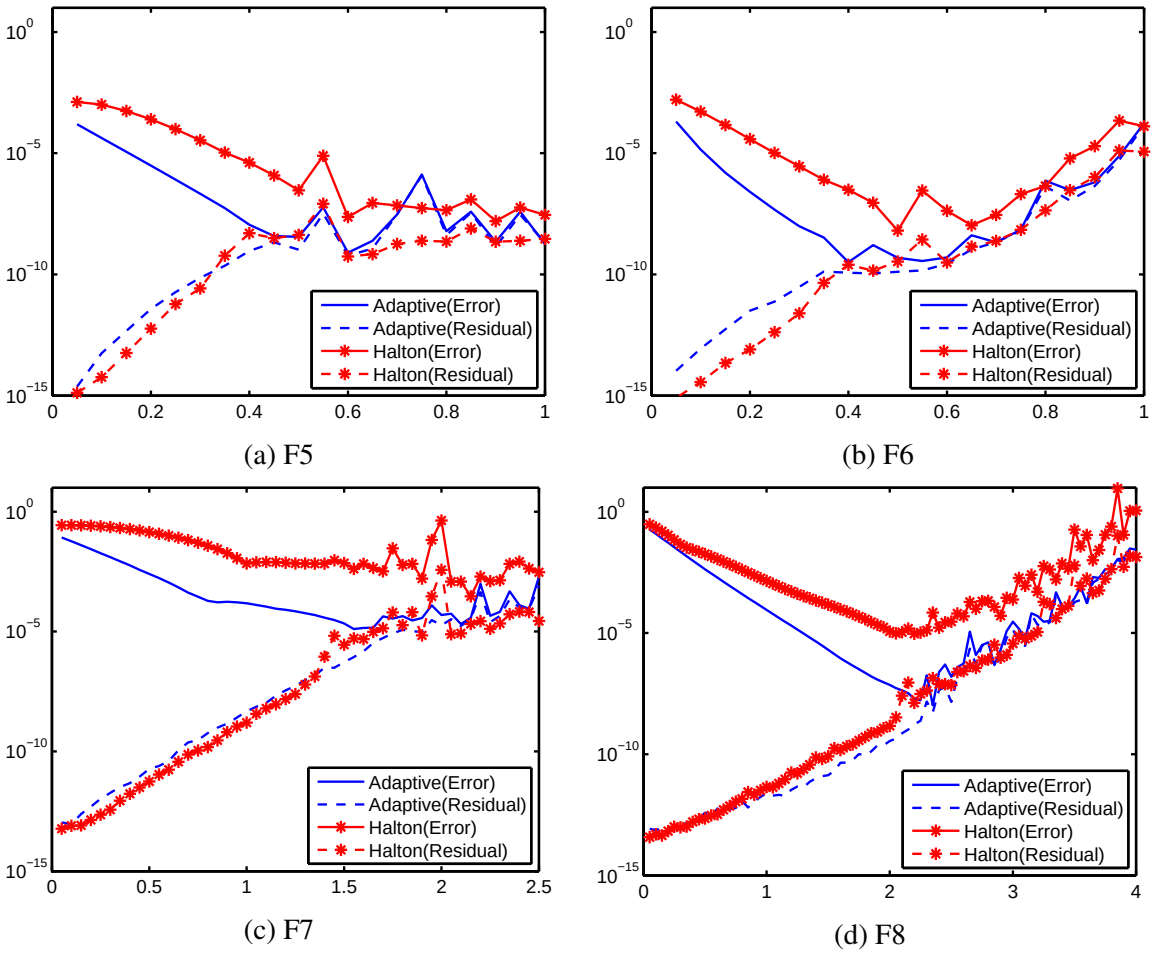


Figure 4.15: *Figures (a)–(d) show maximum approximation error, E_m, and maximum residue error, $\mathbf{r}_m$, versus shape parameter, c, for test functions $F5$–$F8$, using 600 Halton uniformly distributed points or MKI adaptive points. Because it is controlled by condition number, $\kappa(\mathbf{A})$, $\mathbf{r}_m$ is similar for both point distribution methods; however, E_m is less when using MKI adaptive points, allowing faster and closer Error-Residue convergence, and revealing optimal c values.*

and the MKI point sets generated for each of the other test functions (see Figures 4.18 and 4.19); results show that the accuracy and convergence rates of MQ RBF interpolation are similar, for test functions $F1$–$F8$, when using any of the MKI point sets generated for test functions $F1$–$F8$. This means that, when performing MQ RBF interpolation, the MKI adaptive point set utilized can be generated by any known smooth function, rather than the unknown function, and still result in highly accurate numerical solutions.

In addition, the MKI adaptive points generated for MQ interpolation might also be suitable for approximating both MQ derivatives and MQ integrations. Figures 4.20–4.21

show the maximum approximation error, E_m, of interpolations utilizing Hardy's MQ (see Table 2.2, p.12), the Laplacian of MQ, $\phi(r) = \Delta\Phi(r)$, (see Table 5.1, p. 73):

$$\phi(r) = \left(r^2 + 2c^2\right) / \left(\sqrt{r^2 + c^2}\right)^3, \tag{4.11}$$

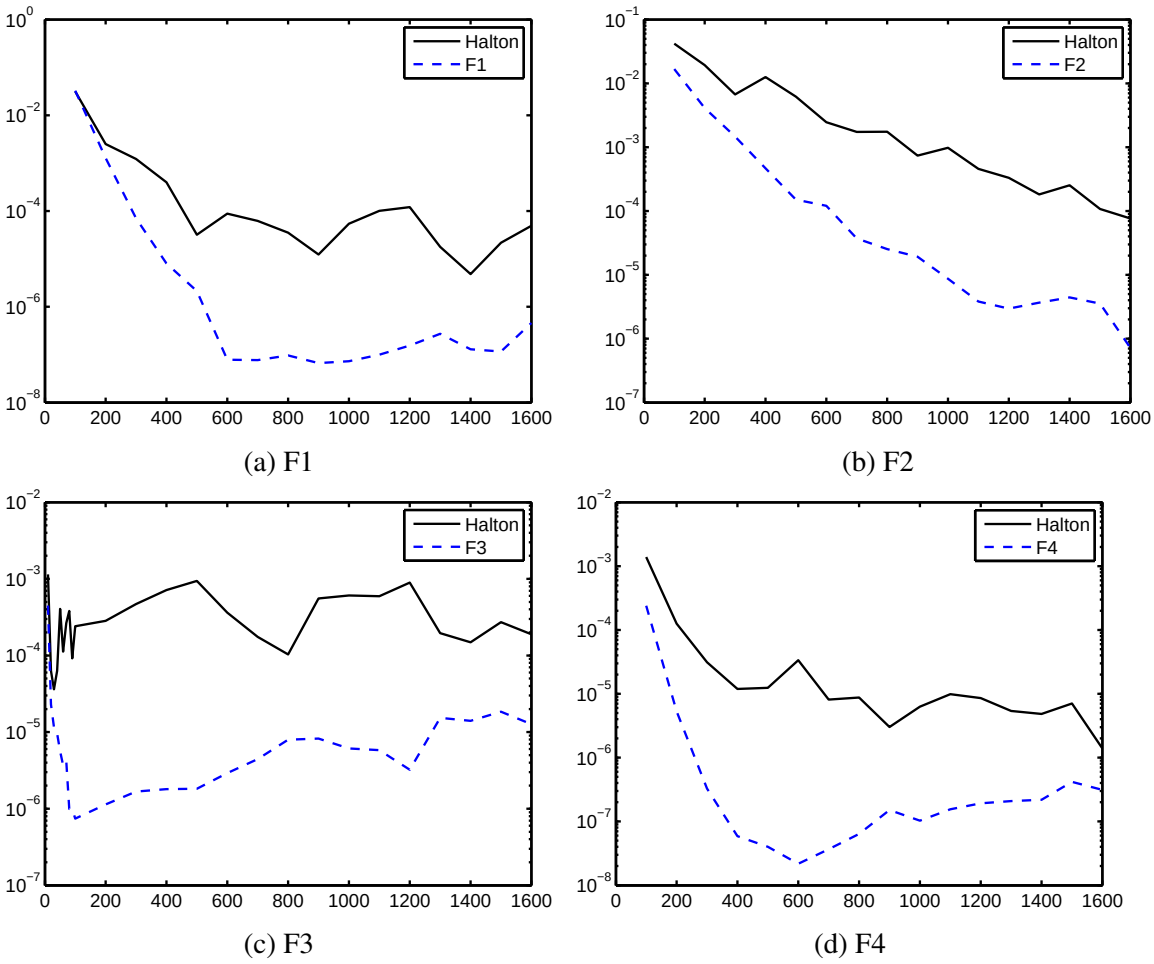
and an integrated MQ, $\Phi(r) = \Delta\Psi(r)$, (see Table 5.1):

$$\Psi(r) = \frac{1}{9}\left(r^2 + 4c^2\right)\left(\sqrt{r^2 + c^2}\right) - \frac{c^3}{3}\ln\left(c + \sqrt{r^2 + c^2}\right), \tag{4.12}$$

for test functions $F1$–$F8$. Results show the adaptive points generated by MKI are just as effective in providing highly accurate results and quick convergence rates when interpolating MQ derivatives and MQ integrations, as they are when interpolating MQs.

Table 4.5: *MKI adaptive points versus Halton uniformly distributed points: optimal solutions for test functions $F1$–$F8$, using RECV c, in terms of maximum approximation error, E_M, and interpolation points, N. This table summarizes the results plotted in Figures 4.16 and 4.17, showing that, in general, far fewer MKI points are needed to increase the rate of convergence between E_m and N, to achieve more accurate optimal MQ RBF approximations.*

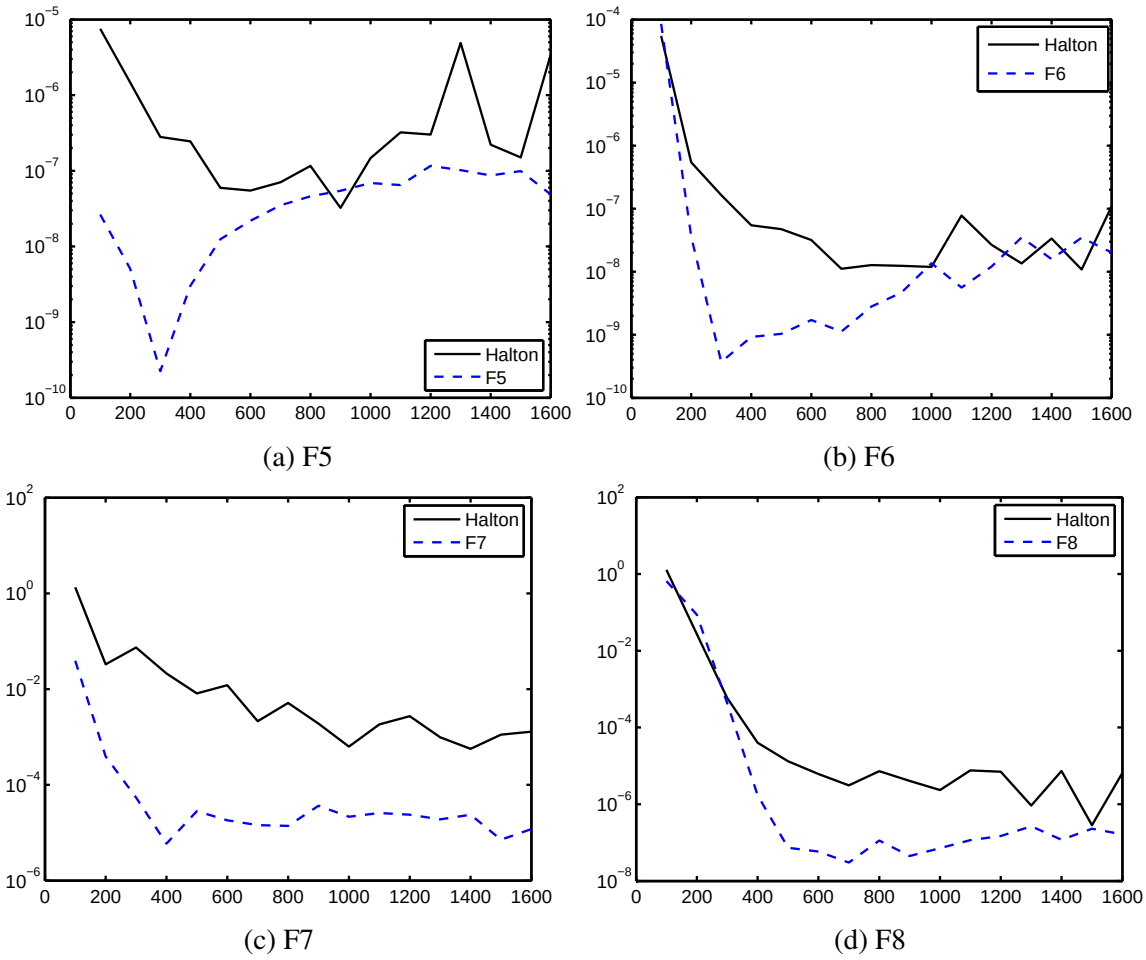
	MKI points		Halton points	
$u(x)$	N	E_m	N	E_m
F1	600	7.87E-08	1400	4.84E-06
F2	1600	7.18E-07	1600	7.65E-05
F3	100	6.78E-07	1000	8.71E-06
F4	600	1.39E-08	1600	4.81E-07
F5	300	2.25E-10	900	3.24E-08
F6	300	3.69E-10	700	1.11E-08
F7	400	5.91E-06	1400	5.66E-04
F8	500	7.31E-08	1500	2.85E-07



(a) F1 (b) F2

(c) F3 (d) F4

Figure 4.16: *Figures (a)–(d) show maximum approximation error, E_m, versus number of interpolation points, N, for test functions $F1$–$F4$, using RECV c and either Halton uniformly distributed points or MKI adaptive points. As is demonstrated, when using RECV c and MKI adaptive points, increasing N quickly leads to very accurate MQ RBF approximations.*



(a) F5 (b) F6

(c) F7 (d) F8

Figure 4.17: *Figures (a)–(d) show maximum approximation error, E_m, versus number of interpolation points, N, for test functions F5–F8, using RECV c and either Halton points or MKI adaptive points. As is demonstrated, when using RECVc and MKI adaptive points, increasing N quickly leads to very accurate MQ RBF approximations.*

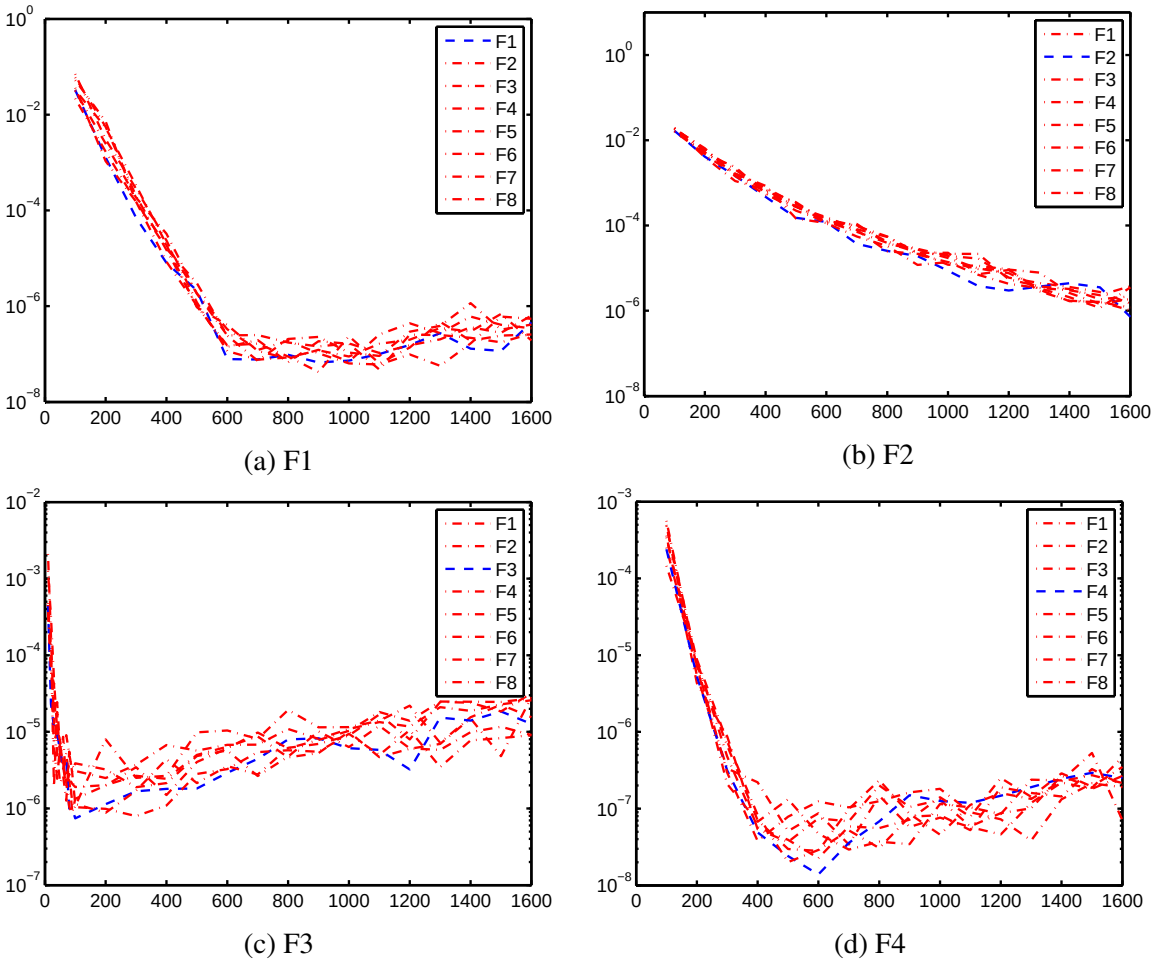
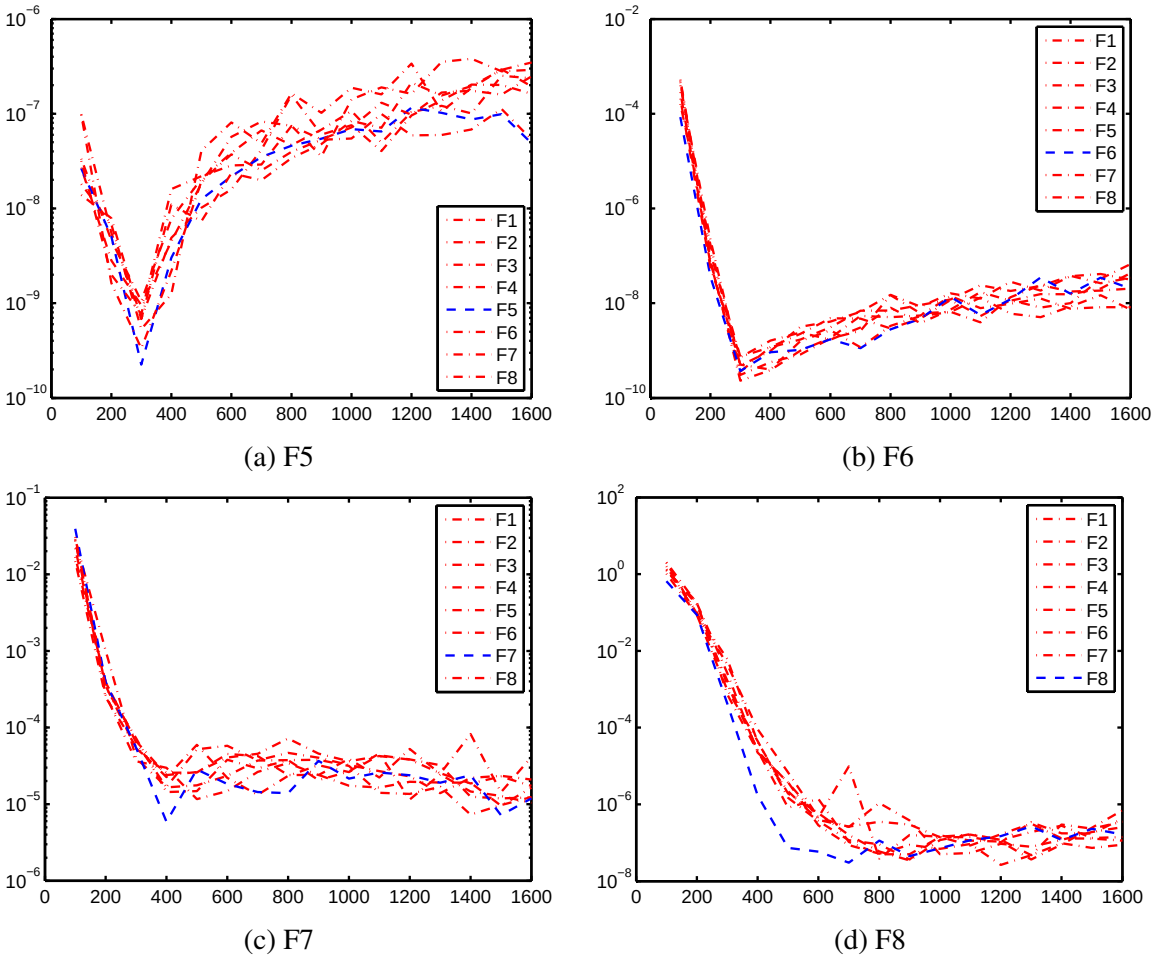


Figure 4.18: *Generalizability of unknown function MKI Adaptive Point Sets for other unknown fuctions within the same domain. Figures (a)–(d) plot the convergence rate of MQ RBF interpolation for Test Functions* $F1$–$F4$*, solved with their own MKI adaptive points and the point sets of each other Test Function. As shown, convergence rate is similar enough to consider the different MKI adaptive point sets to be functionally interchangeable.*



(a) F5 (b) F6

(c) F7 (d) F8

Figure 4.19: *Generalizability of unknown function MKI Adaptive Point Sets for other unknown fuctions within the same domain. Figures (a)–(d) plot the convergence rate of MQ RBF interpolation for Test Functions F5–F8, solved with their own MKI adaptive points and the point sets of each other Test Function. As shown, convergence rate is similar enough to consider the different MKI adaptive point sets to be functionally interchangeable.*

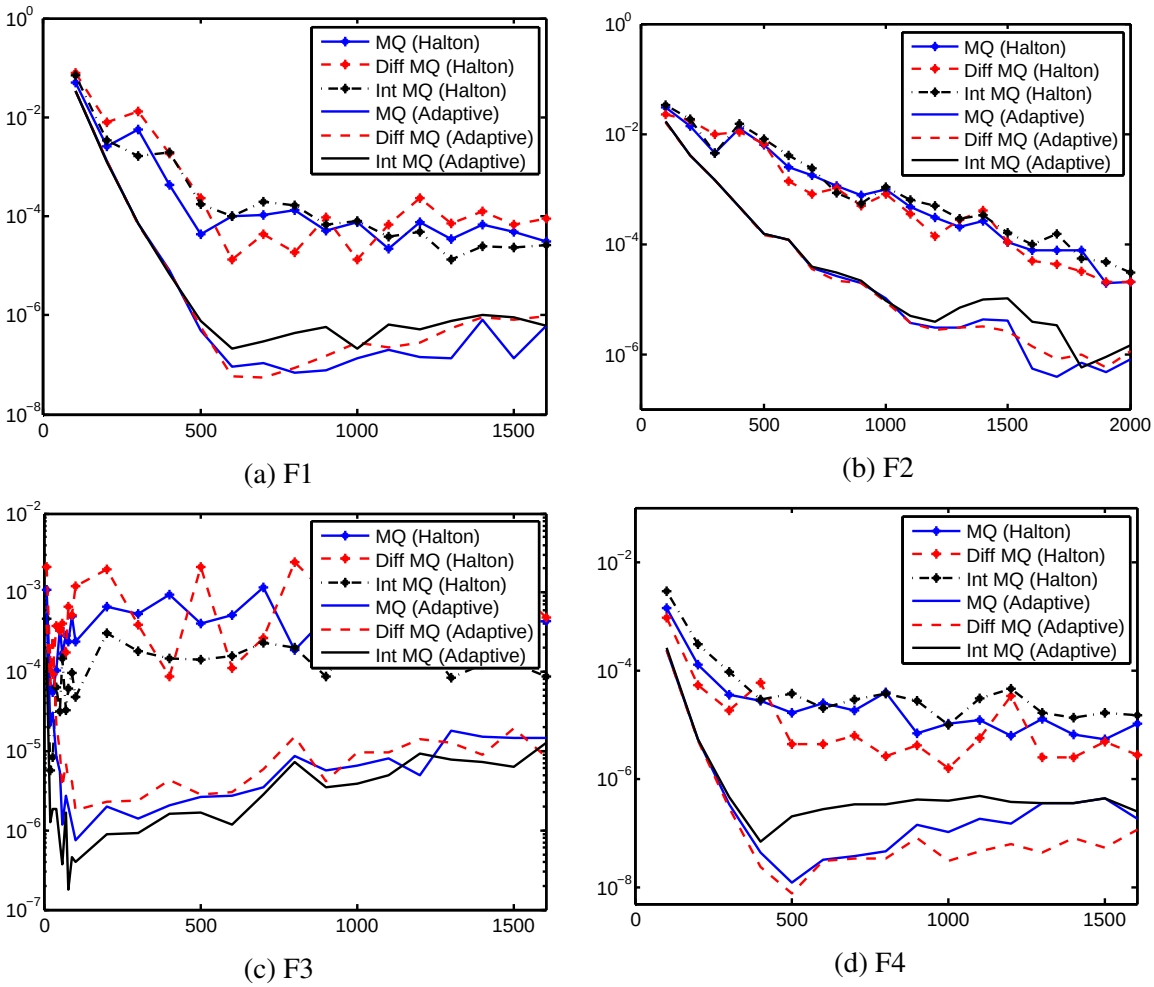


Figure 4.20: *MKI adaptive points versus Halton Points: Convergence rates when interpolating MQs, MQ derivatives, and MQ integrations. Figures (a)–(d) depict the differing convergence rates when interpolating an MQ* (2.3)*, an MQ derivative* (4.11)*, and an MQ integration* (4.12) *using either MKI adaptive points or Halton points generated for the MQ for test functions F*1*–F*4*. Using the MKI method, the adaptive points are acceptably effective for derivatives or integrations, while Halton points generated for an MQ yield highly unstable results when used with derivatives or integrations of the MQ.*

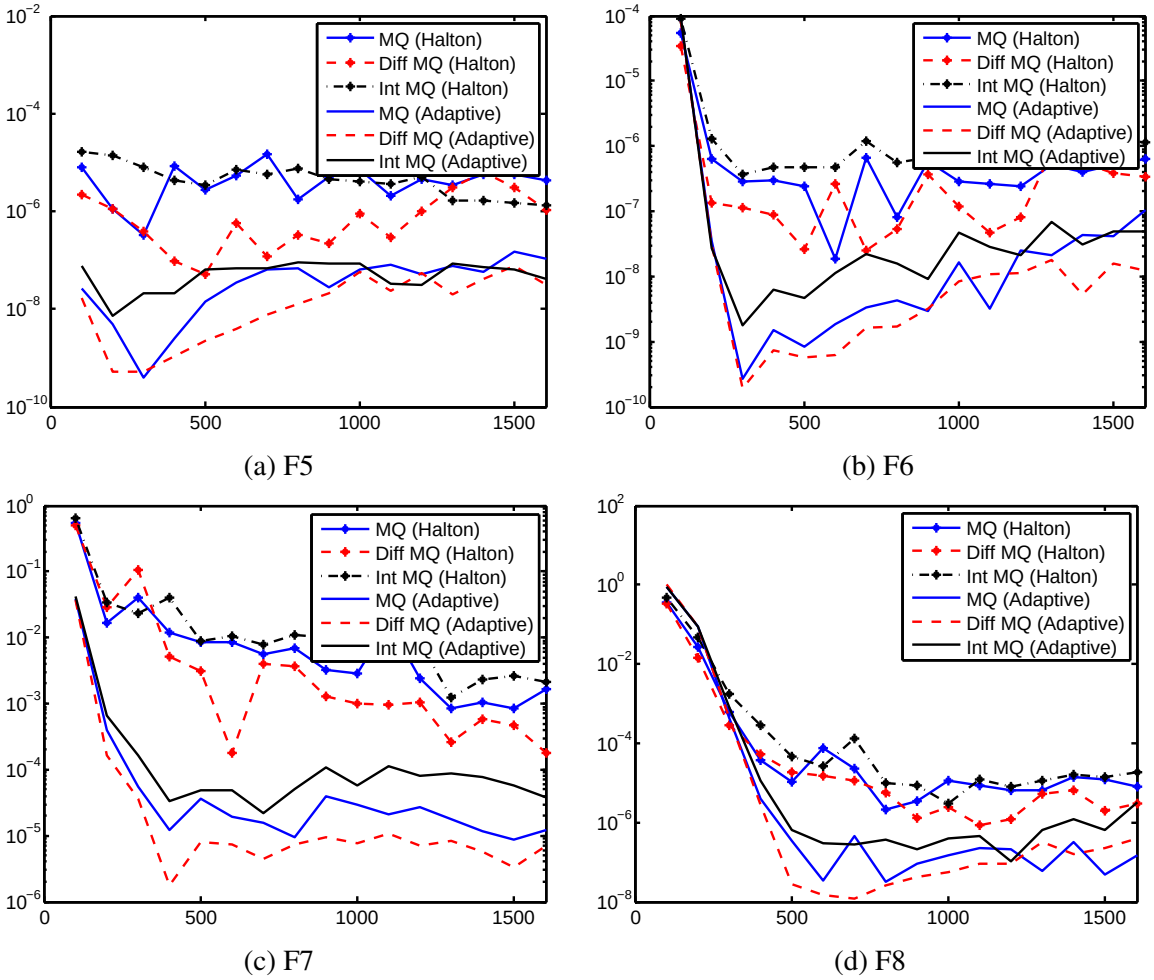


Figure 4.21: *MKI adaptive points versus Halton Points: Convergence rates when interpolating MQs, MQ derivatives, and MQ integrations. Figures (a)–(d) depict the differing convergence rates when interpolating an MQ* (2.3)*, an MQ derivative* (4.11)*, and an MQ integration* (4.12) *using either MKI adaptive points or Halton points generated for the MQ for test functions F5–F8. Using the MKI method, the adaptive points are acceptably effective for derivatives or integrations, while Halton points generated for an MQ yield highly unstable results when used with derivatives or integrations of the MQ.*

Chapter 5

UTILIZING THE RECV METHOD WITH KBC MESHLESS METHODS TO SOLVE TIME-INDEPENDENT PDE PROBLEMS

This chapter briefly introduces two major domain-type kernel-based meshless collocation methods using MQ RBFs: the RBF collocation method (Kansa's method), and the Method of Approximate Particular Solutions (MAPS). For meshless methods utilizing MQ RBFs, shape parameter, c, still plays an important role in determining the accuracy of the method. However, in PDE problems, inside of the domain and the boundary are described by the different functions of data. The examples in this chapter are designed to demonstrate that interpolation and PDE problems which share the same RBFs and data points also share the same optimal shape parameter values. In other words, the shape parameter values selected by the RECV method for RBF interpolation problems are also suitable for use in PDE problems, and can provide very accurate numerical results.

While examples in the previous chapter demonstrated that the adaptive points selected by the MKI method are generally suitable for use with MQ RBFs, the examples in this chapter will show that utilizing the MKI method to select adaptive points can also improve the numerical results when solving PDE problems.

5.1 Kernal-Based Collocation Methods

In the early 1990s, E. Kansa [43, 44], while working on a numerical solution to the Navier-Stokes fluid dynamics equations, proposed a meshless method using Hardy's MQ RBF. Kansa's method later became known as the RBF Collocation Scheme (Section 2.2). While no one has yet been able to provide mathematical explanation for the efficacy of Kansa's method, it provides a powerful method to solve PDE problems.

Because numerical evidence [59] has shown that RBF integrations can be more stable and accurate than regular RBF derivatives, Chen and his coworkers [8, 9] developed a new meshless method called the Method of Approximate Particular Solution (MAPS) which utilized RBF integrations to approximate the particular solution for PDEs while simultaneously satisfying the boundary conditions. Thus far, Kansa's method and MAPS, which are both considered to be meshless Kernal-Based Collocation (KBC) methods, have been applied successfully to solve numerous linear and nonlinear PDEs [46, 50, 51, 57, 69].

Both Kansa's method and MAPS are domain-type KBC methods, which need both domain data and boundary data. Consider an elliptic PDE on a bounded domain, $\Omega \subset \mathbb{R}^2$, with a smooth boundary, Γ. Then, the given function in the domain, $f(x,y)$, and the function on the boundary, $g(x,y)$, are described as follows,

$$k\Delta u + o(x,y)\frac{\partial u}{\partial x} + p(x,y)\frac{\partial u}{\partial y} + q(x,y)\,u = f(x,y), \quad (x,y) \in \Omega, \tag{5.1}$$

$$\mathcal{B}u = g(x,y), \quad (x,y) \in \Gamma, \tag{5.2}$$

where $o(x,y)$, $p(x,y)$, and $q(x,y)$ are given functions, k is a constant, $\mathcal{B}$ is a boundary differential operator, and Δ is the Laplacian, which is defined as follows in 2-D space,

$$\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}.$$

Just as with RBF interpolation (Section 2.1), KBC meshless methods assume that the function, u, is the PDE solution in the domain, $\overline{\Omega} := \Omega \cup \Gamma$, which can be approximated,

$$\tilde{u}(x,y) := \sum_{j=1}^{N} \alpha_j \Phi\left(r_j\right), \quad (x,y) \in \Omega, \tag{5.3}$$

where $r_j = \|(x,y) - (x_j,y_j)\|$ represents the distance between any two given points, $\left\{(x_j,y_j)\right\}_{j=1}^{N}$, which are labeled 'centers' or 'trial points'.

In addition, in KBC methods, the Laplacian, Δu, as well as $\partial u/\partial x$, and $\partial u/\partial y$, are approximated by corresponding kernels with shared coefficients, expressed as follows,

$$\Delta\tilde{u}(x,y) := \sum_{j=1}^{N} \alpha_j \Delta\Phi\left(r_j\right) = \sum_{j=1}^{N} \alpha_j \phi\left(r_j\right), \tag{5.4}$$

$$\frac{\partial\tilde{u}}{\partial x}(x,y) := \sum_{j=1}^{N} \alpha_j \frac{\partial\Phi}{\partial x}\left(r_j\right), \tag{5.5}$$

$$\frac{\partial\tilde{u}}{\partial y}(x,y) := \sum_{j=1}^{N} \alpha_j \frac{\partial\Phi}{\partial y}\left(r_j\right), \tag{5.6}$$

We can then finish the substitution of an MQ RBF into the original elliptic PDE problem by plugging (5.3)–(5.6) back into (5.1) and (5.2), which then yields,

$$f(x,y) \approx \sum_{j=1}^{N} \alpha_j \left(\phi(r_j) + o(x,y)\frac{\partial\Phi}{\partial x}(r_j) + p(x,y)\frac{\partial\Phi}{\partial y}(r_j) + q(x,y)\Phi(r_j)\right), \tag{5.7}$$

$$g(x,y) \approx \sum_{j=1}^{N} \alpha_j \mathcal{B}\Phi(r_j). \tag{5.8}$$

In real implementation, utilizing the collocation technique allows us to discretize the PDEs by imposing finite conditions [5, 21, 64], described,

$$h(x_j,y_j) = f(x_j,y_j), \quad (x_j,y_j) \in \Omega, \quad j = 1,2,\ldots,\mathcal{N}i, \tag{5.9}$$

$$\mathcal{B}u(x_j,y_j) = g(x_j,y_j), \quad (x_j,y_j) \in \Gamma, \quad j = 1,2,\ldots,\mathcal{N}b, \tag{5.10}$$

where

$$h(x_j,y_j) = \Delta u(x_j,y_j) + o(x_j,y_j)\frac{\partial u(x_j,y_j)}{\partial x} + p(x_j,y_j)\frac{\partial u(x_j,y_j)}{\partial y} + q(x_j,y_j)\,u(x_j,y_j),$$

$\mathcal{N}_i$ denotes the number of points inside the domain and $\mathcal{N}_b$ is number of points on the boundary. Furthermore, N denotes the total number of points; i.e., $N = \mathcal{N}i + \mathcal{N}b$. By collocation technique in (5.9) and (5.10) implied (5.7) and (5.8) yields

$$\sum_{j=1}^{N} \alpha_j \Psi(r_{jk}) = f(x_k,y_k), \quad k = 1,\ldots,\mathcal{N}_i, \tag{5.11}$$

$$\sum_{j=1}^{N} \alpha_j \mathcal{B}\Phi(r_{jk}) = g(x_k,y_k), \quad k = \mathcal{N}_i+1,\ldots,N, \tag{5.12}$$

where

$$\Psi(r_{jk}) = \phi(r_{jk}) + o(x_k,y_k)\frac{\partial \Phi(r_{jk})}{\partial x} + p(x_k,y_k)\frac{\partial \Phi(r_{jk})}{\partial y} + q(x_k,y_k)\Phi(r_{jk}), \tag{5.13}$$

and $r_{jk} = \|(x_k,y_k) - (x_j,y_j)\|$. Then (5.11) and (5.12) can be written in matrix notation

$$\mathbf{M}\boldsymbol{\alpha} = \mathbf{b}, \tag{5.14}$$

where $\mathbf{b} = [f(x_1,y_1),\ldots,f(x_{\mathcal{N}_i},y_{\mathcal{N}_i}), g(x_{\mathcal{N}_i+1},y_{\mathcal{N}_i+1}),\ldots,g(x_N,y_N)]^T$, $\boldsymbol{\alpha} = [\alpha_1,\ldots,\alpha_N]^T$ are undetermined coefficients, and

$$\mathbf{M} = \begin{bmatrix} \Psi(\|(x_1,y_1)-(x_1,y_1)\|) & \cdots & \Psi(\|(x_1,y_1)-(x_N,y_N)\|) \\ \vdots & \ddots & \vdots \\ \Psi(\|(x_{\mathcal{N}_i},y_{\mathcal{N}_i})-(x_1,y_1)\|) & \cdots & \Psi(\|(x_{\mathcal{N}_i},y_{\mathcal{N}_i})-(x_N,y_N)\|) \\ \mathcal{B}\Phi(\|(x_{\mathcal{N}_i+1},y_{\mathcal{N}_i+1})-(x_1,y_1)\|) & \cdots & \mathcal{B}\Phi(\|(x_{\mathcal{N}_i+1},y_{\mathcal{N}_i+1})-(x_N,y_N)\|) \\ \vdots & \ddots & \vdots \\ \mathcal{B}\Phi(\|(x_N,y_N)-(x_1,y_1)\|) & \cdots & \mathcal{B}\Phi(\|(x_N,y_N)-(x_N,y_N)\|) \end{bmatrix}$$

is a N by N unsymmetric matrix. If the matrix $\mathbf{M}$ is invertible, the coefficients $\boldsymbol{\alpha}$ can be easily obtained by solving (5.14) associated with standard matrix solver.

In the early 2000s, Hon and Schaback [38] demonstrated that, unlike symmetric matrices, such as **A** (see p. 6), for unsymmetric matrices, such as matrix **M**, proof of invertibility can not be established because there can be no guarantee that they are nonsingular. However, thus far, unsymmetric collocation techniques have been used and shown to approximate many different applications with acceptable accuracy. Building on this, Wu [67] developed a symmetric collocation technique which has been proven to have no singularity inside the interpolation matrix [41]. Comparing symmetric and unsymmetric collocation techniques, unsymmetric techniques require a weaker assumption of regularity and can be applied to more problems. Additionally, Fasshauer [24] demonstrated that the numerical results obtained from unsymmetric collocation are usually superior to those obtained via symmetric collocation. For this reason, this work focuses on meshless methods utilizing unsymmetric collocation techniques.

Table 5.1: *Corresponding MQ kernels for Kansa's method and the MAPS in* $\mathbb{R}^2$.

Kansa's method	$\Phi:$	$\sqrt{r^2+c^2}$
	$\frac{\partial\Phi}{\partial x}:$	$\frac{x}{\sqrt{r^2+c^2}}$
	$\frac{\partial\Phi}{\partial x}:$	$\frac{y}{\sqrt{r^2+c^2}}$
	$\phi:$	$\frac{\left(r^2+2c^2\right)}{\left(\sqrt{r^2+c^2}\right)^3}$
MAPS	$\Phi:$	$\frac{1}{9}\left(r^2+4c^2\right)\left(\sqrt{r^2+c^2}\right)-\frac{c^3}{3}\ln\left(c+\sqrt{r^2+c^2}\right)$
	$\frac{\partial\Phi}{\partial x}:$	$x\cdot\left(\frac{3r^2+6c^2}{9\sqrt{r^2+c^2}}-\frac{c^3}{3\sqrt{r^2+c^2}\left(\sqrt{r^2+c^2}+c\right)}\right)$
	$\frac{\partial\Phi}{\partial y}:$	$y\cdot\left(\frac{3r^2+6c^2}{9\sqrt{r^2+c^2}}-\frac{c^3}{3\sqrt{r^2+c^2}\left(\sqrt{r^2+c^2}+c\right)}\right)$
	$\phi:$	$\sqrt{r^2+c^2}$

5.2 Suitability of the KBC Method Utilizing RECV c and MKI Adaptive Points

In the RBF literature, Kanas's method and MAPS have been shown to be capable for solving PDEs [32, 43, 44]. However, due to the trade-off principle (Section 2.4, p. 14), the numerical solutions can be significantly improved depending on: 1) the number of collocation points; 2) the distribution method of collocation points; 3) the choice of the shape parameter, c.

As the numerical results demonstrated in Section 4.2 (see p. 52), the RECV c is interchangeable as long as the MQ approximations constructed within the same collocation points, and the fill distance, h, of the collocation points is sufficiently small. Moreover, Section 4.3 (see p. 56) shows MKI adaptive points allow for far more excellent convergence and more accurate numerical results for individual functions for MQ approximation.

The primary purpose of this section lies in proving the effectiveness of the RECV c for Kansa's method and MAPS in PDEs. Additionally, to demonstrated the suitability of using MKI adaptive points in PDEs, Kansa's method and the MAPS will be constructed by either regular grid collocation points or MKI adaptive collocation points in the following examples.

In this section, the ratio of regular grid interior points with equally spaced boundary points are test manually in order to obtained better numerical result, and the ratio of adaptive collocation points (interior and boundary points) are generated depending on the MKI method.

Three benchmark PDEs (Poisson equation and convection-diffusion-reaction equations) are examined in two irregular domains. Respectively, the RECV c and MKI adaptive points are selected and generated by a smooth periodic function,

$$K(x,y) = \sin(x) + \cos(y). \tag{5.15}$$

To evaluate the accuracy of the MQ approximation, the error measurement, such as the maximum approximation error (E_m), and relative approximation error (E_{L^2}), are employed and are defined as follows:

$$E_m = \max_{1 \le j \le Nt} \left\{ \left| u(x_j) - \tilde{u}(x_j) \right| \right\}, \tag{5.16}$$

$$E_{L^2} = \left(\frac{\sum_{j=1}^{Nt} \left(u(x_j) - \tilde{u}(x_j) \right)^2}{\sum_{j=1}^{Nt} u^2(x_j)} \right)^{1/2}, \tag{5.17}$$

where Nt is the number of testing points, u is analytical solution and $\tilde{u}$ is approximate solution.

Computation were performed using MATLAB on a system with the following parameters: Windows 7 operating system (32 bits), Intel Core i7-2640M processor, 2.8 GHz CPU, and 3.49 GB memory.

Example 5.2.1. Consider the following boundary value problem with the Dirichlet boundary condition,

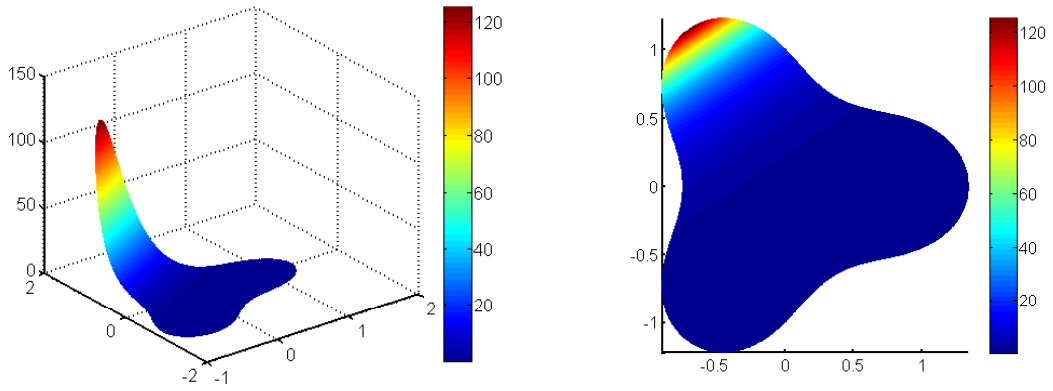
$$\Delta u(x,y) = f(x,y), \quad (x,y) \in \Omega, \tag{5.18}$$

$$u(x,y) = g(x,y), \quad (x,y) \in \Gamma, \tag{5.19}$$

where $f(x,y)$ and $g(x,y)$ are given functions according to the following analytical solution:

$$u(x,y) = \exp(-2x+3y). \tag{5.20}$$

The exact solution is depicted in Figure 5.1.



(a) $\mathbb{R}^3$ (b) $\mathbb{R}^2$

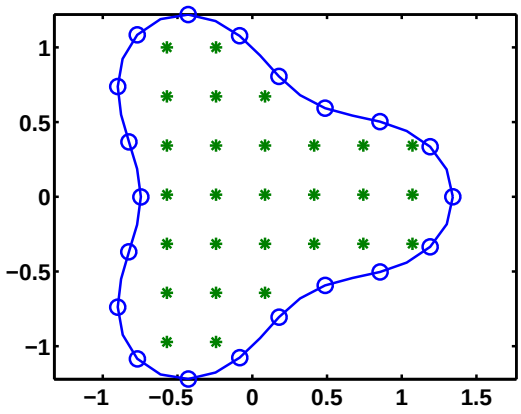
Figure 5.1: *The analytical solution, u (see* (5.20)*), to the boundary value problem described in Example 5.2.1.*

Figure 5.2 shows example regular grid interior points with equally spaced boundary points, and MKI adaptive collocation points in the computational domain $\Omega \cup \Gamma$ which is defined by the following parametric equation:

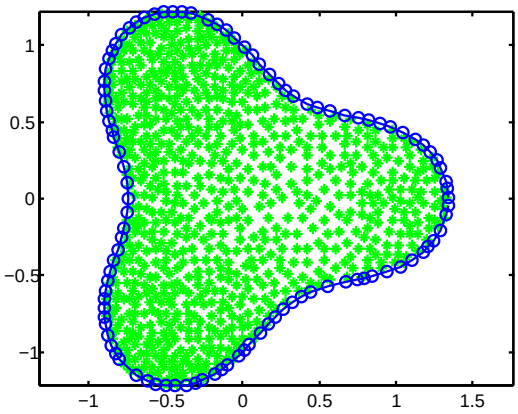
$$\Omega = \{r(\theta)\cos(\theta), r(\theta)\sin(\theta) : \theta \in [0, 2\pi)\}, \tag{5.21}$$

where

$$r(\theta) = \left(\cos(3\theta) + \left(2 - \sin^2(3\theta)\right)^{1/2}\right)^{1/3}. \tag{5.22}$$



(a) *Regular grid interior, and uniform distributed boundary points in* $\overline{\Omega}$.



(b) *MKI adaptive collocation points in* $\overline{\Omega}$.

Figure 5.2: *Regular grid points with equally spaced boundary points, and MKI adaptive points in the computational domain,* $\overline{\Omega} = \Omega \cup \Gamma$*, for Example 5.2.1. Figures 5.2 (a) and (b) depict example regular grid interior points with equally spaced boundary points and MKI adaptive collocation points, respectively.*

Chapter 4 demonstrated that the RECV c and MKI adaptive collocation points can significantly improve the accuracy of RBF approximation in interpolation problem. For PDEs, the KBC methods which adopted the RBF collocation approach, such as Kansa's method, and the MAPS have been successfully applied to solve linear and nonlinear problems in physics; the property trade-off principle is also inherited from RBF interpolation (see Sec. 2.4, p. 14).

This example is designed to demonstrated the effectiveness of utilizing RECV c, and MKI adaptive collocation points suitability for Kansa's method and the MAPS employ MQ as a basis function within PDE problem. To compare the suitability of KBC methods with MKI adaptive points in PDEs, Kansa's method and the MAPS in this example is solved using N regular grid points, or N MKI adaptive points[1] (*i.e.,* N = 100, 400, and 900) (see Figures 5.2 (a)–(b)) for both Kansa's Method and the MAPS.

Figures 5.3–5.4 show the approximation error of Kansa's method and the MAPS using RECV c with 400 regular grid collocation points, and 400 MKI adaptive points. For both methods, MKI adaptive points yield results more accurate by eliminate the boundary error issue which inherited from RBF interpolation (see Sec. 4.3, p. 38).

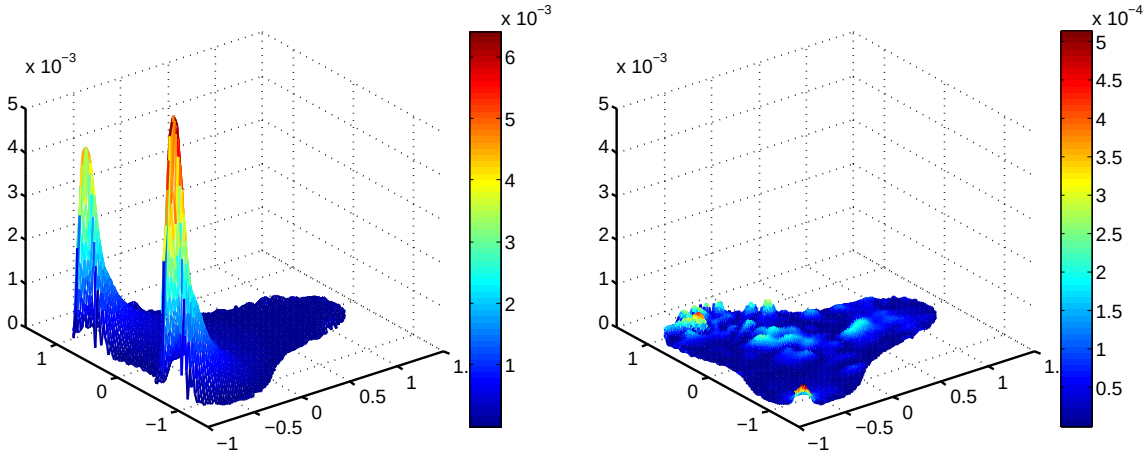
Figures 5.5–5.6 show the relative approximation error, E_{L^2}, convergence behavior for

[1] In this example, MKI adaptive points are generated from a given initial dataset which included $10,138$ regular grid interior points and $1,000$ equally spaced boundary points.

RBF interpolation and Kansa's method, and for RBF interpolation and the MAPS, respectively, as N increases with either regular grid points or MKI adaptive points plotted versus shape parameter, c. In these figures, the solid lines indicate the error of RBF interpolation's solution to the analytical solution, (5.20), using kernel Φ (corresponding to Table 5.1) within the same set of collocation points as dash lines (*i.e.,* Kansa's methods and the MAPS) did. As the figures show the dash lines are almost parallel to the solid lines (RBF interpolation), and the optimal shape parameters are located in the similar places as RBF interpolation did. In other words, the RECV c value selected by RBF interpolation within the same computational domain, and using the same set of collocation points in this example for Kansa's method, and the MAPS can obtain an acceptably high accuracy.

Figures 5.3–5.4 show the approximation errors of Kansa's method and the MAPS, respectively. The approximation in these figures are either constructed by 310 regular grid interior points with 90 equally spaced boundary points or 400 MKI adaptive points. In this boundary value problem, the large boundary error which in the case of regular grid points (subfigure (a), [25]) are significantly improved by utilizing MKI adaptive points (subfigure (b)).

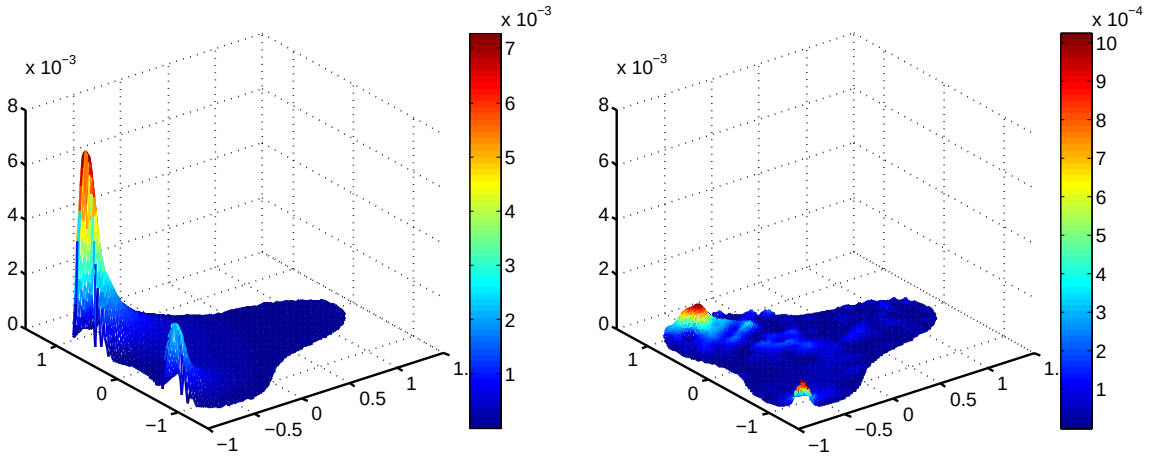
Comparing Table 5.2, Table 5.3, Figure 5.5 and Figure 5.6 show the RECV c chosen from the interval $c \in [0, 100]$ are successfully avoiding the local optimum issue and selecting an acceptab approximation when using either Kansa's method or the MAPS.



(a) Regular grid points; $N = 400$.

(b) MKI adaptive points; $N = 400$.

Figure 5.3: *Comparison of boundary error behavior for Kansa's method using either regular grid interior points with equally spaced boundary points, or MKI adaptive collocation points for the boundary value problem described in Example 5.2.1.*



(a) Regular grid points; $N = 400$.

(b) MKI adaptive points; $N = 400$.

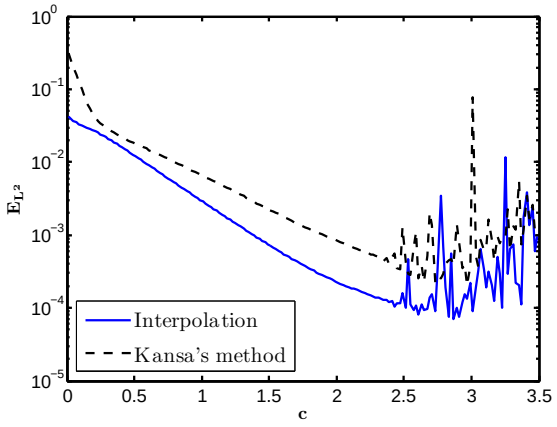
Figure 5.4: *Comparison of boundary error behavior for the MAPS using either regular grid interior points with equally spaced boundary points, or MKI adaptive collocation points.*

	Kansa's method (regular grid points)				Kansa's method (MKI adaptive points)			
N	Nb	$RECV\ c$	E_m	E_{L^2}	Nb	$RECV\ c$	E_m	E_{L^2}
100	40	2.54	5.58E-002	4.14E-004	28	2.52	6.99E-001	3.23E-003
400	90	1.00	4.82E-002	2.81E-004	65	0.84	5.98E-004	4.00E-006
900	108	0.54	1.65e-003	9.91E-006	103	0.47	1.76E-004	2.39E-006

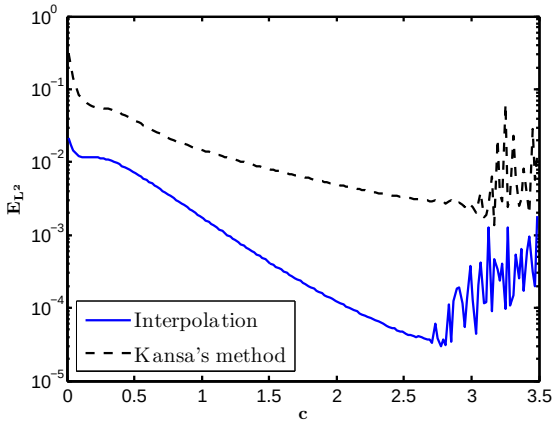
Table 5.2: *Testing RECV c with Kansa's method using either MKI adaptive collocation points or regular grid collocation points. For the PDE problem (see Example 5.2.1), this table depicts maximum approximation error, E_m, and relative approximation error, E_{L^2}, with either N regular grid collocation points or N MKI adaptive collocation points in the irregular domain when using Kansa's method with RECV c chosen from the interval* $c \in [0, 100]$.

	MAPS (regular grid points)				MAPS (MKI adaptive points)			
N	Nb	$RECV\ c$	E_m	E_{L^2}	Nb	$RECV\ c$	E_m	E_{L^2}
100	40	2.43	1.20E-001	1.05E-003	28	1.90	5.70E-001	3.21E-003
400	90	0.75	7.24E-003	4.20E-005	65	0.61	1.18E-003	7.34E-006
900	108	0.41	2.55E-003	1.20E-005	103	0.33	2.44E-004	2.14E-006

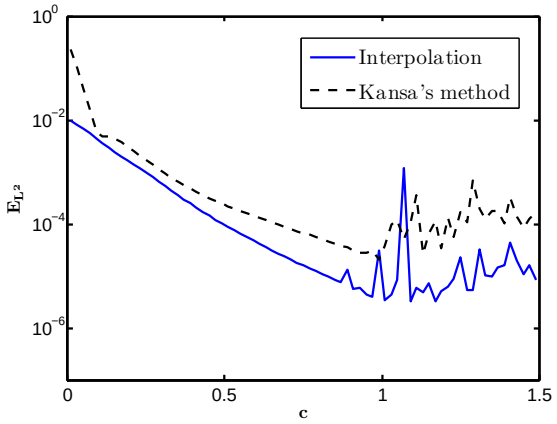
Table 5.3: *Testing RECV c with MAPS using either MKI adaptive collocation points or regular grid collocation points. For the PDE problem (see Example 5.2.1), this table depicts maximum approximation error, E_m, and relative approximation error, E_{L^2}, with either N regular grid collocation points or N MKI adaptive collocation points in the irregular domain when using the MAPS with RECV c chosen from the interval* $c \in [0, 100]$.



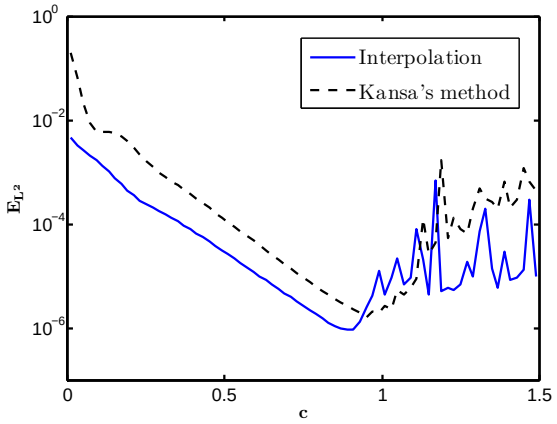
(a) Regular grid points; $N = 100$.



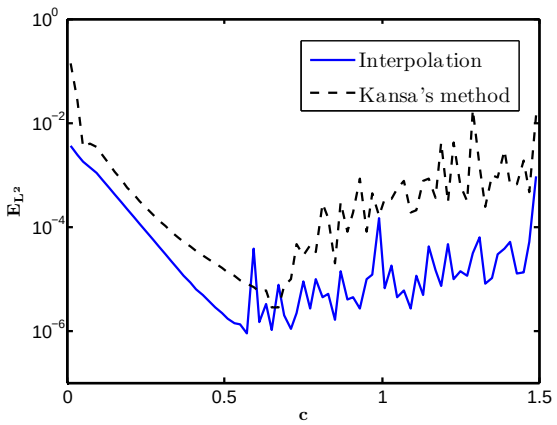
(b) MKI adaptive points; $N = 100$.



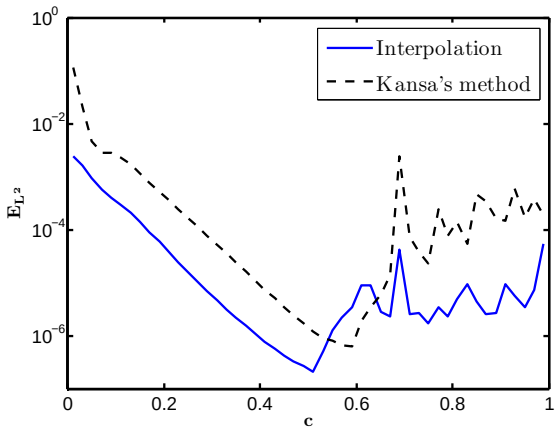
(c) Regular grid points; $N = 400$.



(d) MKI adaptive points; $N = 400$.

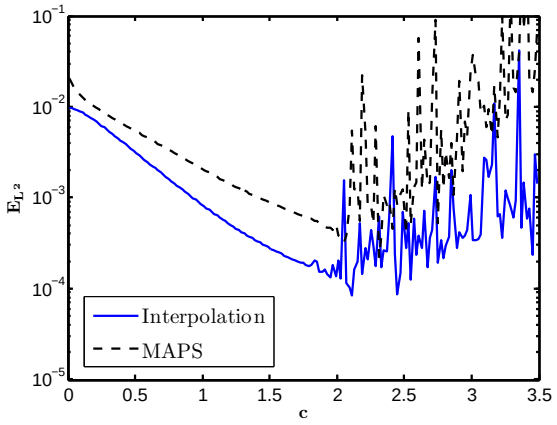


(e) Regular grid points; $N = 900$.

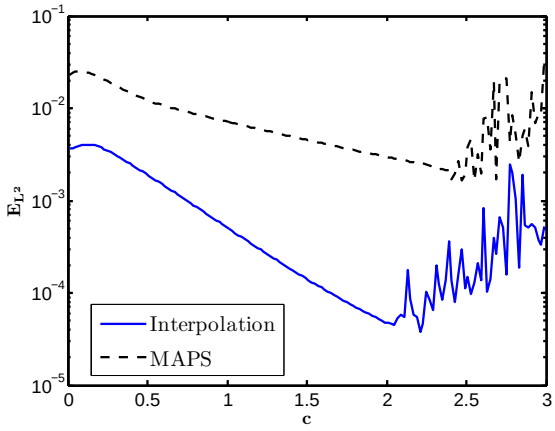


(f) MKI adaptive points; $N = 900$.

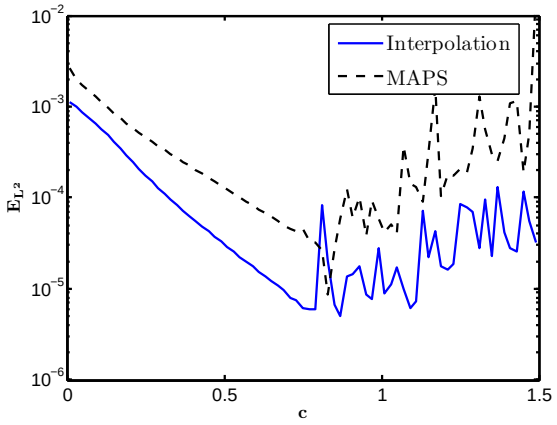
Figure 5.5: *Suitability of RECV c and MKI adaptive points with Kansa's method for boundary value problem described in Example 5.2.1. Figures (a)–(f) show the relative approximation error, E_{L^2}, convergence behavior for RBF interpolation and Kansa's method as N increases with either regular grid points or MKI adaptive points plotted versus shape parameter, c.*



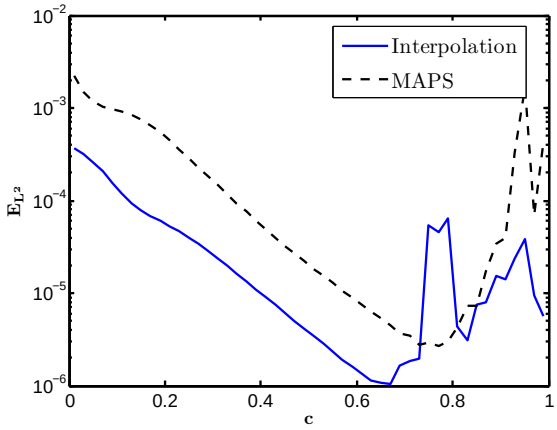
(a) Regular grid points; $N = 100$.



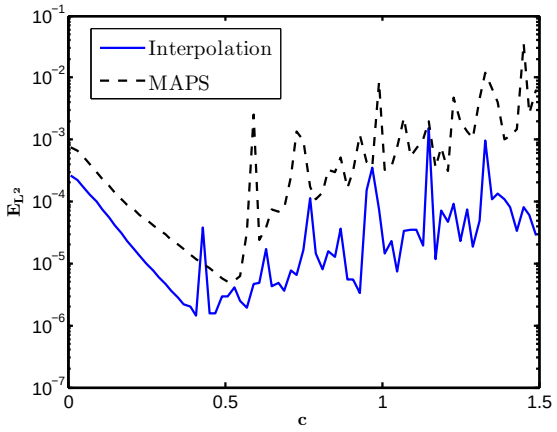
(b) MKI adaptive points; $N = 100$.



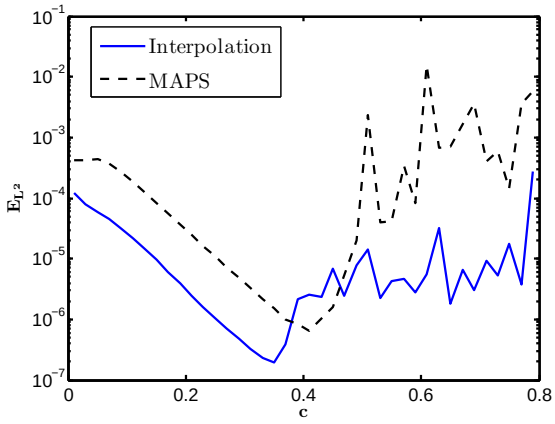
(c) Regular grid points; $N = 400$.



(d) MKI adaptive points; $N = 400$.



(e) Regular grid points; $N = 900$.



(f) MKI adaptive points; $N = 900$.

Figure 5.6: Suitability of RECV c and MKI adaptive points with the MAPS for boundary value problem described in Example 5.2.1. Figures (a)–(f) show the relative approximation error, E_{L^2}, convergence behavior for RBF interpolation and the MAPS as N increases with either regular grid points or MKI adaptive points plotted versus shape parameter, c.

Example 5.2.2. Consider the *convection-diffusion-reaction* equation is a combination of the diffusion and convection equations with the Dirichlet boundary condition:

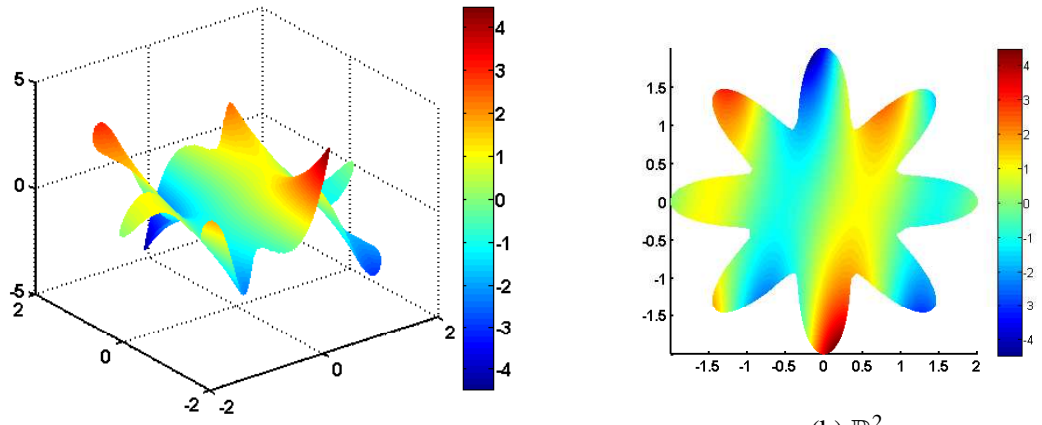
$$\Delta u + \left(x^2 + y^2\right) u + y\cos(y)\frac{\partial u}{\partial x} + \sinh(x)\frac{\partial u}{\partial y} = f(x,y), \quad (x,y) \in \Omega, \tag{5.23}$$

$$u = g(x,y), \quad (x,y) \in \Gamma, \tag{5.24}$$

where $f(x,y)$ and $g(x,y)$ are given functions depends on the following analytical solution:

$$u(x,y) = \sin(\pi x)\cosh(y) - \cos(\pi x)\sinh(y). \tag{5.25}$$

The exact solution is depicted in Figure 5.7.



(a) $\mathbb{R}^3$ (b) $\mathbb{R}^2$

Figure 5.7: *The analytical solution, u (see* (5.25)*), to the convection-diffusion-reaction equation described in Example 5.2.2.*

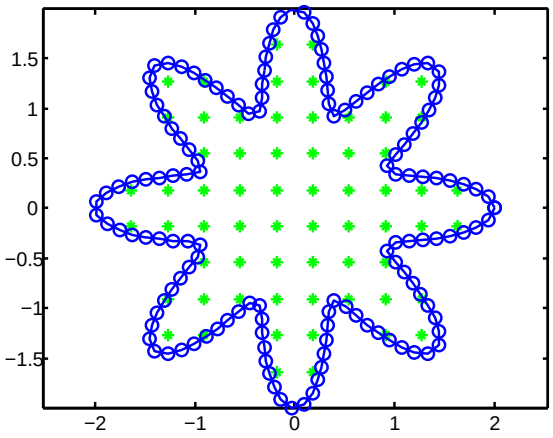
Figure 5.8 shows example regular grid interior points with equally spaced boundary points and MKI adaptive collocation points in the computational domain $\Omega \cup \Gamma$ which is defined by the following parametric equation:

$$\Omega = \{(x,y) \mid \quad x = r(\theta)\cos(\theta), y = r(\theta)\sin(\theta) : \theta \in [0, 2\pi)\}, \tag{5.26}$$

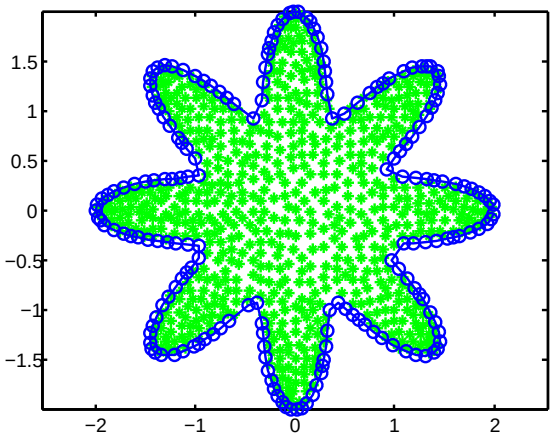
where

$$r(\theta) = 1 + \cos^2(4\theta). \tag{5.27}$$

To compare the difference of convergence behavior in RBF interpolation and solved PDE utilized the KBC methods, Example 5.2.1 examined the simplest elliptic PDE – Poisson's



(a) *Regular grid interior, and uniform distributed boundary points in* $\overline{\Omega}$.



(b) *MKI adaptive collocation points in* $\overline{\Omega}$.

Figure 5.8: *For the convection-diffusion-reaction equation described in Example 5.2.2 in the computational domain* $\overline{\Omega} = \Omega \cup \Gamma$, *Figures 5.8 (a) and (b) depict example regular grid interior points with equally spaced boundary points and MKI adaptive collocation points, respectively. Various numbers of regular grid or adaptive points (*N = 100, 400, *and* 900*) were tested with Kansa's Method (see Table 5.4) and MAPS (see Table 5.5).*

equation with Dirichlet boundary condition. Figures 5.5–5.6 showed the optimal value shape parameter for the KBC methods is highly correlated to the RBF interpolation when utilizing the same kernel, Φ. Additionally, comparing the numerical results in Tables 5.2–5.3 with Figures 5.5–5.6 showed the shape parameter, c, found by RECV method are close to the optimal value can be obtained.

Figures 5.9 and 5.10 show maximum approximation error, E_m, of the KBC methods as N increase with either regular grid collocation points or MKI adaptive collocation points plotted versus shape parameter, c. In lower density cases (*i.e.,* $N = 100$, see subfigure (a)–(b)) show the numerical solution instability of the KBC methods occurs when the value of shape parameter, c, is small. In other words, the numerical solution stability of the KBC methods also can be influenced by the computational complexity of PDE problem that involved complication of PDE, boundary condition, and computational domain. Fortunately, this instability issue because of computational complexity can be eliminated as N increases with either regular grid points or MKI adaptive points for both Kansa's method and the MAPS in this PDE problem.

Usually the instability issue can be a serious problem for selecting an optimal shape parameter for the KBC methods. However, combined the numerical results in Table 5.4, Table 5.5, Figure 5.9 and Figure 5.10 show the shape parameter found by the RECV

method can generally avoid the local optimum issues which caused by ill-conditioned and computational complexity of PDE problem with acceptable accuracy. In addition, Figures 5.11–5.12 show the approximation errors of Kansa's method and the MAPS with either 400 regular grid interior points or 400 MKI adaptive points. These figures demonstrate that MKI adaptive points (see subfigure (b)) can effectively diminish the large boundary errors when using regular grid points (see subfigure (a).

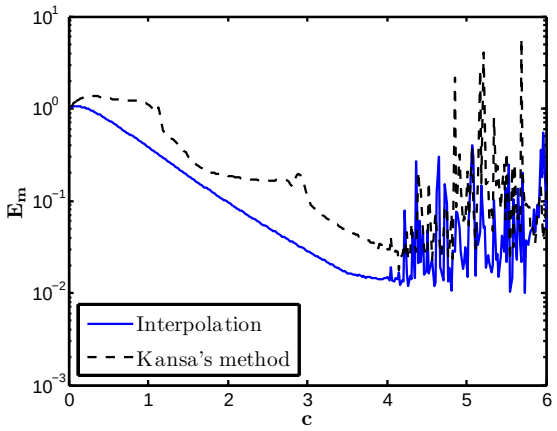
For the suitability of MKI adaptive points for KBC methods, Figures 5.11–5.12 show the convergence rates of E_m is faster when using MKI adaptive collocation points than when using regular grid collocation points, meaning the accuracy of KBC methods can provide a far more accurate numerical solution.

	Kansa's method (regular grid points)				Kansa's method (MKI adaptive points)			
N	Nb	c	E_m	E_{L^2}	Nb	c	E_m	E_{L^2}
100	36	4.24	7.43E-002	9.77E-003	47	4.14	1.01E-002	1.28E-003
400	139	1.44	4.11E-004	2.47E-005	111	1.37	5.65E-005	1.08E-005
900	295	0.93	1.36E-004	6.10E-006	188	0.77	3.75E-005	3.65E-006

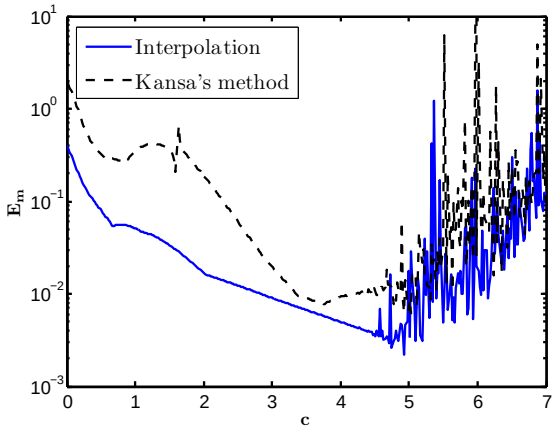
Table 5.4: *Testing RECV c with Kansa's method using either MKI adaptive collocation points or regular grid collocation points. For convection-diffusion-reaction equation (see Example 5.2.2), this table shows maximum approximation error, E_m, and relative approximation error, E_{L^2}, with either N regular grid collocation points or N MKI adaptive collocation points in the irregular domain Ω when using Kansa's method with RECV c chosen from the interval $c \in [0, 100]$.*

	MAPS (regular grid points)				MAPS (MKI adaptive points)			
N	Nb	c	E_m	E_{L^2}	Nb	c	E_m	E_{L^2}
100	36	3.45	2.73E-002	3.78E-003	47	3.73	9.74E-003	1.56E-003
400	139	1.07	7.60E-004	4.05E-005	111	1.01	8.72E-005	1.88E-005
900	295	0.59	2.19E-004	1.29E-005	188	0.55	5.77E-005	4.04E-006

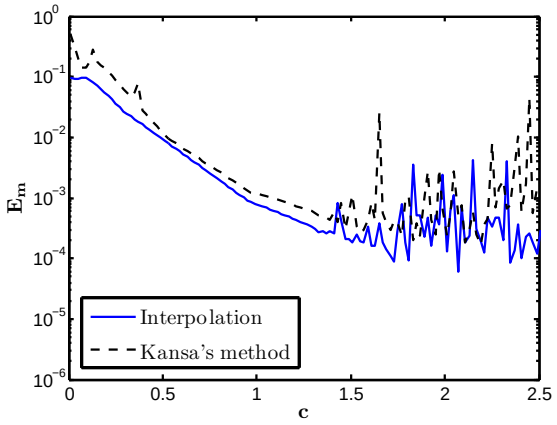
Table 5.5: *Testing RECV c with the MAPS using either MKI adaptive collocation points or regular grid collocation points. For convection-diffusion-reaction equation (see Example 5.2.2), this table shows maximum approximation error, E_m, and relative approximation error, E_{L^2}, with either N regular grid collocation points or N MKI adaptive collocation points in the irregular domain Ω when using the MAPS with RECV c chosen from the interval $c \in [0, 100]$.*



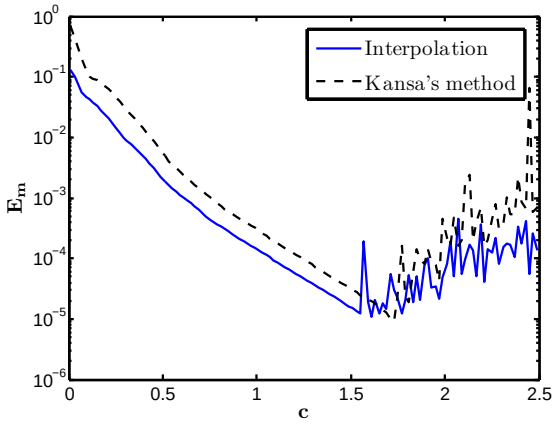
(a) Regular grid points; $N = 100$.



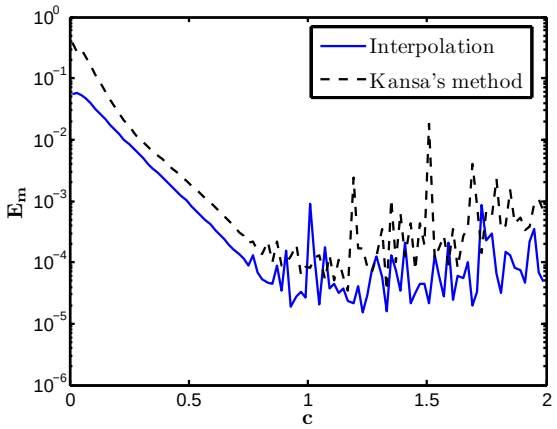
(b) MKI adaptive points; $N = 100$.



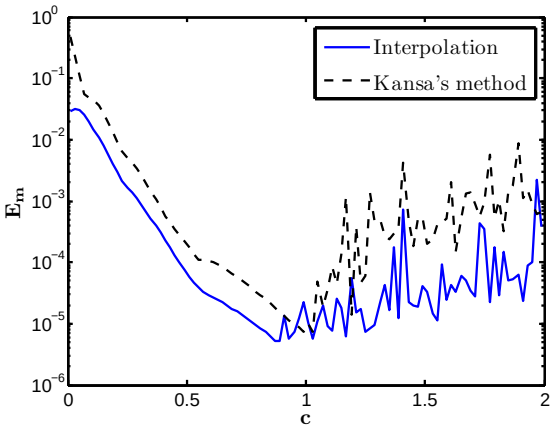
(c) Regular grid points; $N = 400$.



(d) MKI adaptive points; $N = 400$.

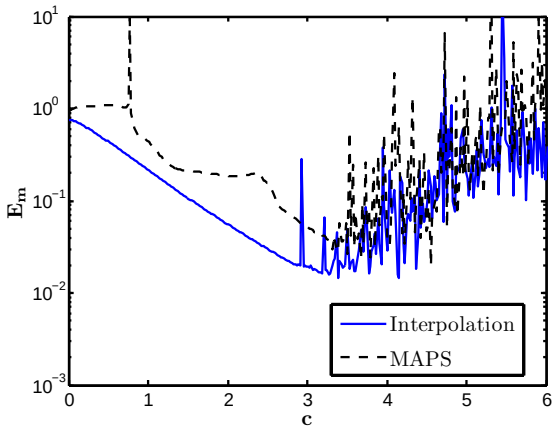


(e) Regular grid points; $N = 900$.

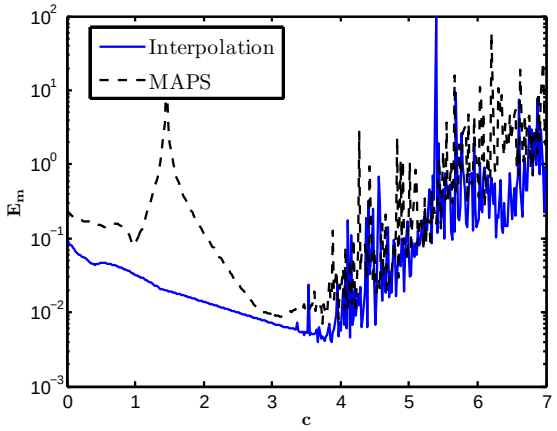


(f) MKI adaptive points; $N = 900$.

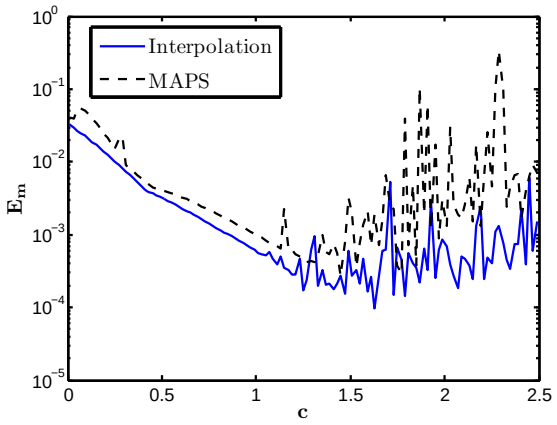
Figure 5.9: *Suitability of RECV c and MKI Adaptive Points with Kansa's Method for convection-diffusion-reaction equation. For the analytical solution* (5.25) *to the convection-diffusion-reaction equation problem described in Example 5.2.2 in domain* Ω*, Figures (a)-(f) show maximum approximation error,* E_m*, convergence behavior for MQ RBF interpolation (solid line) and Kansa's method (dash line) as N increases with either regular grid points or MKI adaptive points plotted versus shape parameter, c.*



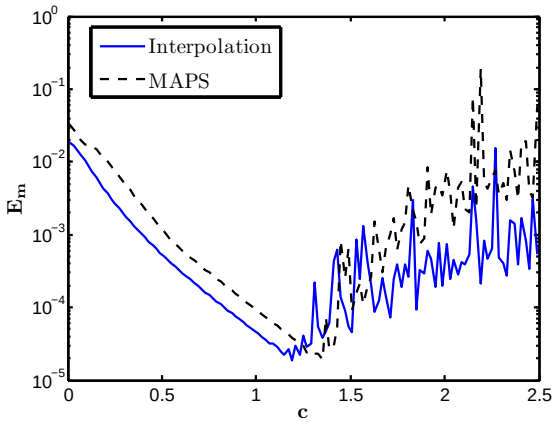
(a) Regular grid points; $N = 100$.



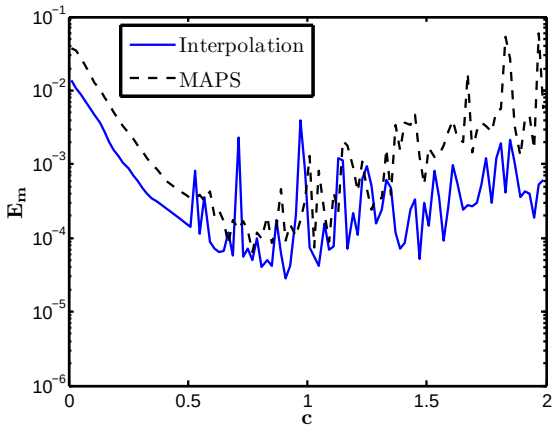
(b) MKI adaptive points; $N = 100$.



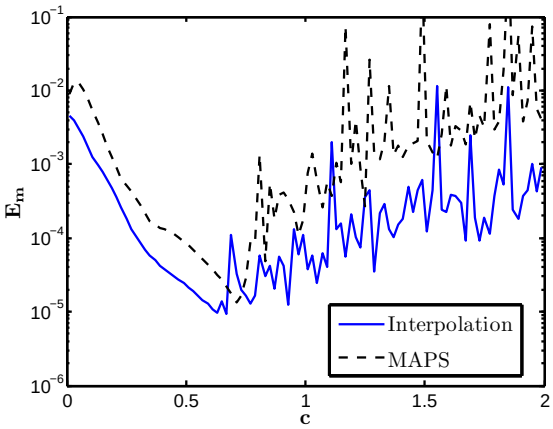
(c) Regular grid points; $N = 400$.



(d) MKI adaptive points; $N = 400$.

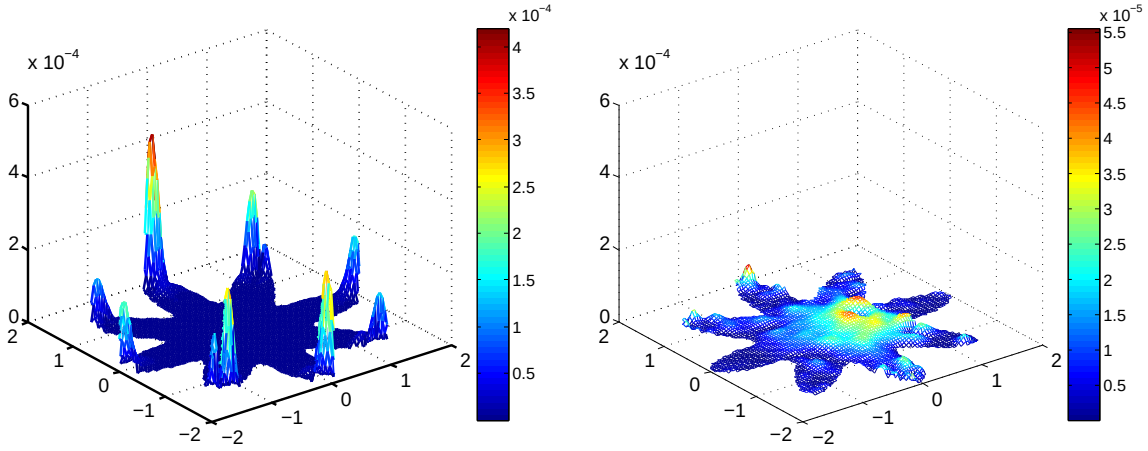


(e) Regular grid points; $N = 900$.



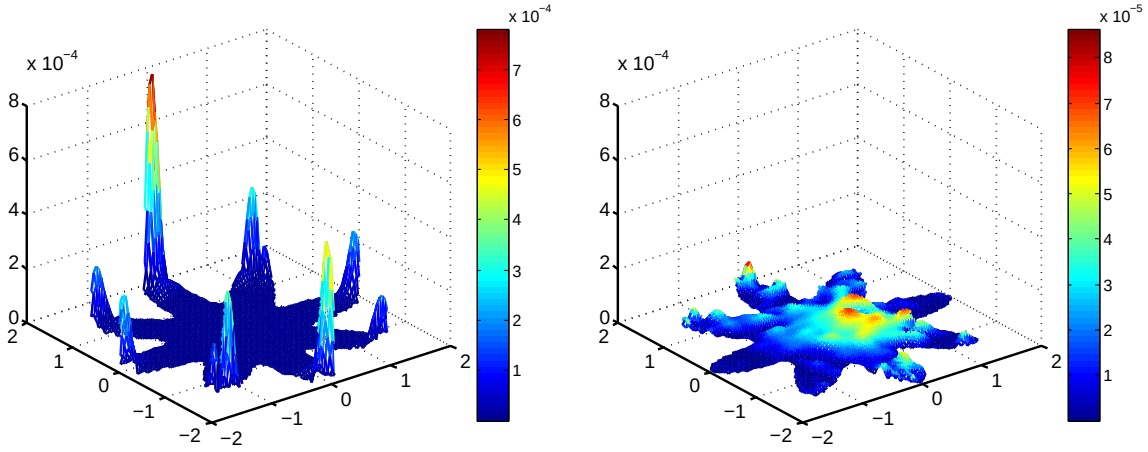
(f) MKI adaptive points; $N = 900$.

Figure 5.10: *Suitability of RECV c and MKI adaptive points with MAPS for convection-diffusion-reaction equation. For the analytical solution* (5.25) *to the convection-diffusion-reaction equation problem described in Example 5.2.2 in domain* Ω*, Figures (a)-(f) show maximum approximation error,* E_m*, convergence behavior for MQ RBF interpolation (solid line) and MAPS (dash line) as N increases with either regular grid points or MKI adaptive points plotted versus shape parameter, c.*



(a) Regular grid points; $N = 400$. (b) MKI adaptive points; $N = 400$.

Figure 5.11: *Comparison of boundary error behavior for Kansa's method using either regular grid interior points with equally spaced boundary points, or MKI adaptive collocation points to the convection-diffusion-reaction equation problem described in Example 5.2.2 in domain* Ω.



(a) Regular grid points; $N = 400$. (b) MKI adaptive points; $N = 400$.

Figure 5.12: *Comparison of boundary error behavior for the MAPS using either regular grid interior points with equally spaced boundary points, or MKI adaptive collocation points to the convection-diffusion-reaction equation problem described in Example 5.2.2 in domain* Ω.

Example 5.2.3. Consider the following convection-diffusion equation,

$$\Delta u + \left(x^2 y\right) u + \left(y^2 + \cos\left(x\right)\right) \frac{\partial u}{\partial x} - y \sin\left(x\right) \frac{\partial u}{\partial y} = f(x,y), \quad (x,y) \in \Omega, \tag{5.28}$$

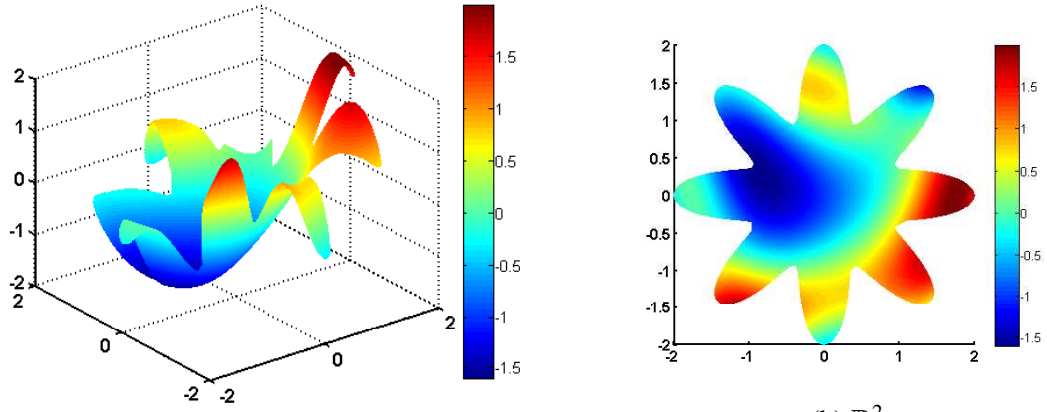
$$u = g(x,y), \quad (x,y) \in \Gamma^D, \tag{5.29}$$

$$\frac{\partial u}{\partial \mathbf{n}} = \nabla g(x,y) \cdot \overrightarrow{n}, \quad (x,y) \in \Gamma^N, \tag{5.30}$$

where $f(x,y)$ and $g(x,y)$ are given functions depends on the following analytical solution:

$$u(x,y) = \sin\left(y^2 + x\right) - \cos\left(y - x^2\right) \tag{5.31}$$

is examined in the bounded domain Ω is the same as that shown in Example 5.2.2 with boundary $\Gamma = \Gamma_D \cup \Gamma_N$ and $\Gamma_D \cap \Gamma_N = \emptyset$ as Figure 5.13 depicted.



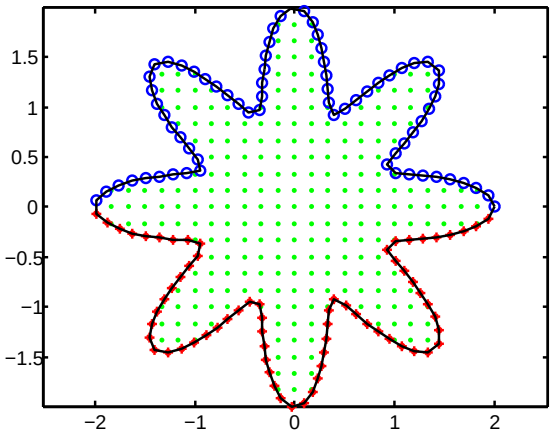
(a) $\mathbb{R}^3$ (b) $\mathbb{R}^2$

Figure 5.13: *The analytical solution* (5.31) *to the convection-diffusion-reaction equation described in Example 5.2.3.*

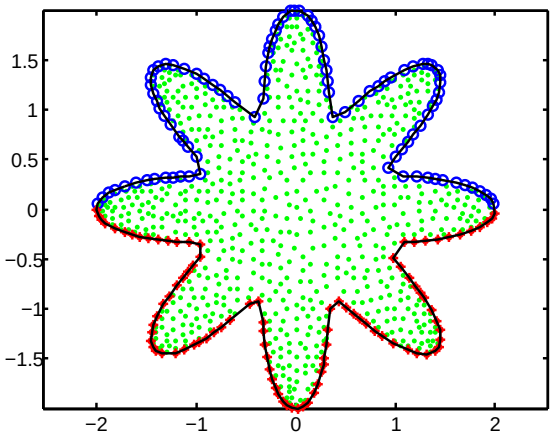
Figure 5.14 depicts example of regular grid interior points with equally spaced boundary points and MKI adaptive points. In the figure, the points "$\cdot$" indicated interior points, "$\circ$" are Dirichlet boundary data points, and "$*$" are Neumann boundary data points.

Examples 5.2.1 and 5.2.2 demonstrate that the RECV method can effectively select an optimal value of shape parameter, RECV c, for KBC methods with fairly high accuracy. For MKI adaptive points, it can significantly improve the large boundary error issue for KBC methods that using regular grid points.

This example considers a slightly challenge PDE problem than Example 5.2.2 in computational complexity, *i.e.,* a similar type of PDE with Dirichlet boundary condition (5.29) and Neumann boundary condition (5.30) in the same computational domain (5.26).



(a) *Regular grid interior, and uniform distributed boundary points in* $\overline{\Omega}$.



(b) *MKI adaptive collocation points in* $\overline{\Omega}$.

Figure 5.14: *For the convection-diffusion-reaction equation described in Example 5.2.3 in the computational domain* $\overline{\Omega} = \Omega \cup \Gamma$, *Figures 5.14 (a) and (b) depict example regular grid interior points with uniformly distributed boundary points and adaptive collocation points generated via the MKI method, respectively. Various numbers of regular grid or adaptive points (*$N = 100, 400,$ *and* 900*) were tested with Kansa's Method (see Table 5.6) and MAPS (see Table 5.7). Note, in the figures, the points "*$\cdot$*" indicated interior points, "*$\circ$*" are Dirichlet boundary data points, and "*$*$*" are Neumann boundary data points.*

Figures 5.15–5.16 show maximum approximation error, E_m, versus shape parameter, c, as N increases with either regular grid points or MKI adaptive points for KBC methods. Subfigures (b), (d), and (f) show the instability because of the computational complexity are significantly improved as N increases with MKI adaptive points. However, subfigures (a), (c), and (e) depict that the instability issue is getting worse as N increases with regular grid points [2].

For the suitability of using RECV c for KBC method, the numerical results show in Figure 5.15, Figure 5.16, Table 5.6, and Table 5.7 demonstrate that the RECV c avoid the local optimum issue that caused by ill-conditioning and computational complexity of PDE problem with acceptable accuracy for both regular grid collocation points and MKI adaptive collocation points.

For Kansa's method and the MAPS using MKI adaptive points, three details of significance can be derived from the results shown in Examples 5.2.1 through 5.2.3:

1. Figures 5.5–5.6 (see Example 5.2.1), Figures 5.9–5.10 (see Example 5.2.2), and Figures 5.15–5.16 (see Example 5.2.3) demonstrate that MKI adaptive points can

[2]The numerical results plotted in subfigures (a), (c), and (e) are the optimal result that is selected from the several different ratios of points inside the domain (interior points) and on the boundary (boundary points).

provide more accurate numerical results upto two order of magnitude than regular grid points.

2. Figures 5.3–5.4 (see Example 5.2.1), Figures 5.11–5.12 (see Example 5.2.2), and Figures 5.17–5.18 (see Example 5.2.3) show MKI adaptive points can significantly improve the large boundary error issue that occurs when using regular grid points.

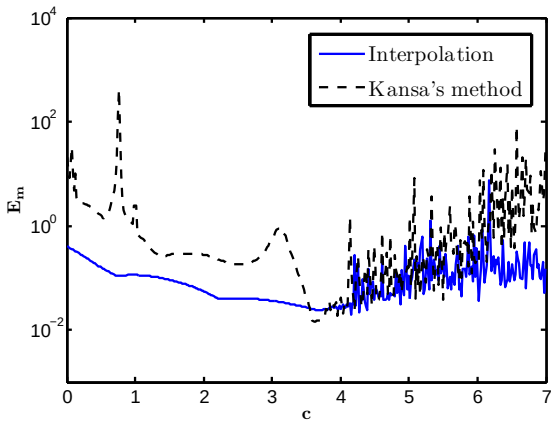
3. Figures 5.15–5.16 (see Example 5.2.3) show MKI adaptive can effectively improve the instability caused by computational complexity of PDE problem on determining the ratio of interior points and boundary points.

	Kansa's method (regular grid points)				Kansa's method (MKI adaptive points)			
N	Nb	c	E_m	E_{L^2}	Nb	c	E_m	E_{L^2}
100	36	4.24	2.37E-002	3.70E-003	47	4.16	5.04E-002	1.49E-002
400	139	1.44	7.67E-003	1.16E-003	111	1.36	4.25E-004	8.63E-005
900	295	0.93	2.41E-003	2.60E-004	188	0.79	6.37E-005	1.35E-005

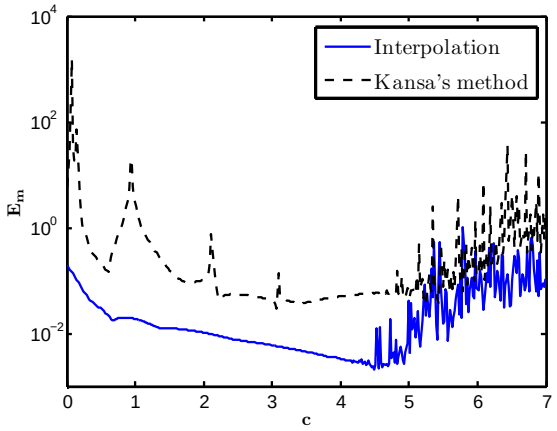
Table 5.6: *Testing RECV c with Kansa's method using either MKI adaptive collocation points or regular grid collocation points. For PDE problem (see Example 5.2.3), this table depicts maximum approximation error, E_m, and relative approximation error, E_{L^2}, with either N regular grid collocation points or N MKI adaptive collocation points for Kansa's method with RECV c chosen from the interval $c \in [0, 100]$.*

	MAPS (regular grid points)				MAPS (MKI adaptive points)			
N	Nb	c	E_m	E_{L^2}	Nb	c	E_m	E_{L^2}
100	36	3.45	7.88E-002	1.22E-002	47	3.62	4.89E-002	1.43E-002
400	139	1.07	2.82E-002	3.77E-003	111	1.05	4.25E-004	1.16E-004
900	295	0.59	6.00E-003	7.57E-004	188	0.57	5.05E-005	1.24E-005

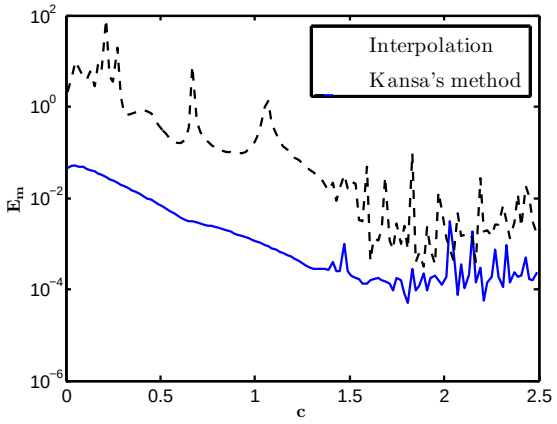
Table 5.7: *Testing RECV c with the MAPS using either MKI adaptive collocation points or regular grid collocation points. For PDE problem (see Example 5.2.3), this table depicts maximum approximation error, E_m, and relative approximation error, E_{L^2}, with either N regular grid collocation points or N MKI adaptive collocation points for the MAPS with RECV c chosen from the interval $c \in [0, 100]$.*



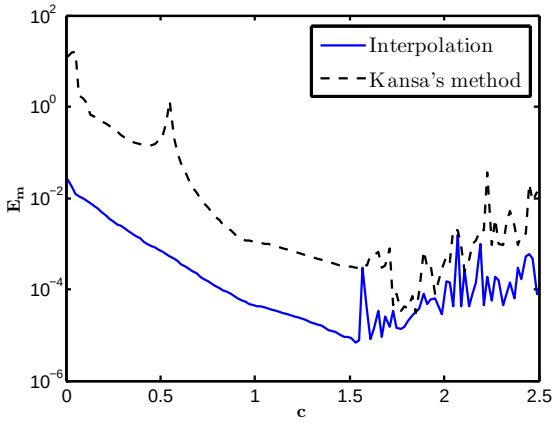
(a) Regular grid points; $N = 100$.



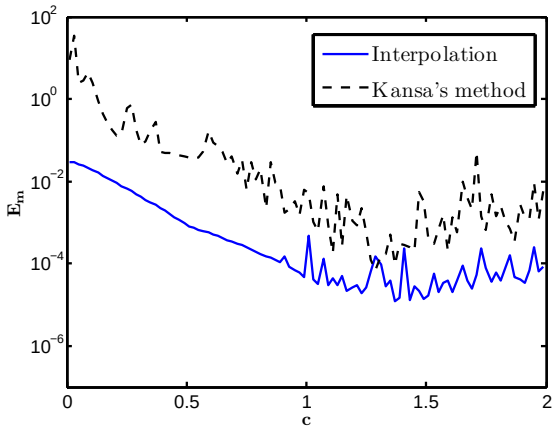
(b) MKI adaptive points; $N = 100$.



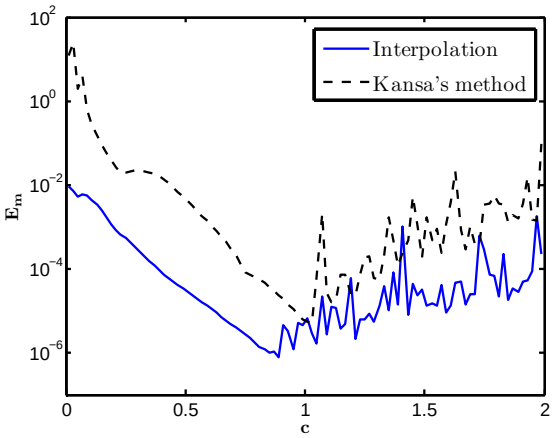
(c) Regular grid points; $N = 400$.



(d) MKI adaptive points; $N = 400$.

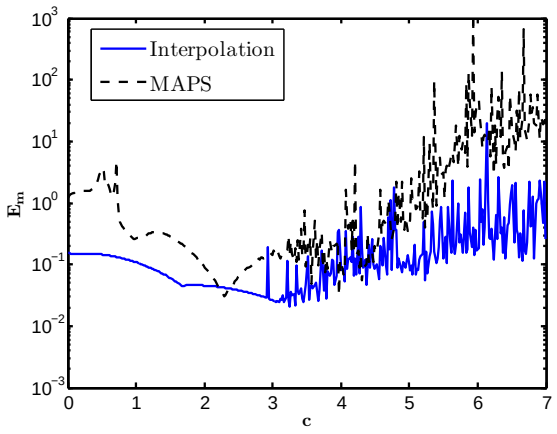


(e) Regular grid points; $N = 900$.

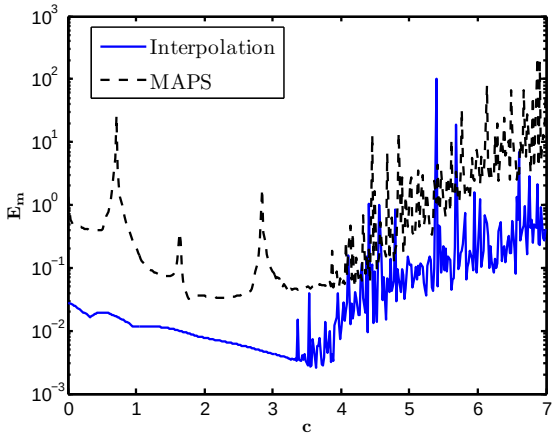


(f) MKI adaptive points; $N = 900$.

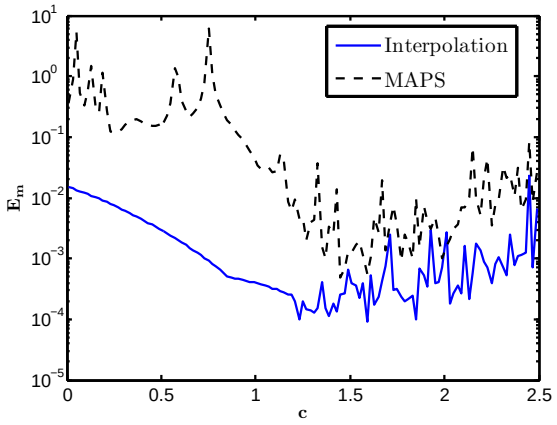
Figure 5.15: *Suitability of RECV c and MKI Adaptive Points with Kansa's Method for convection-diffusion-reaction equation. For the analytical solution* (5.31) *to the convection-diffusion-reaction equation problem described in Example 5.2.3 in domain* Ω*, Figures (a)-(f) show the Relative Approximation Error,* E_{L^2}*, Convergence Behavior for MQ RBF interpolation (solid line) and Kansa's approximation method (dash line) as N increases with either regular grid points or MKI adaptive points plotted versus shape parameter, c.*



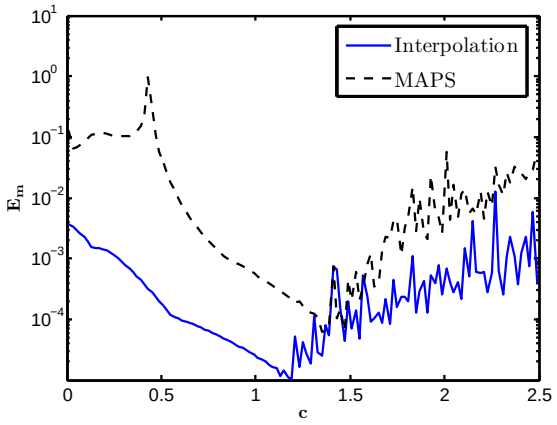
(a) Regular grid points; $N = 100$.



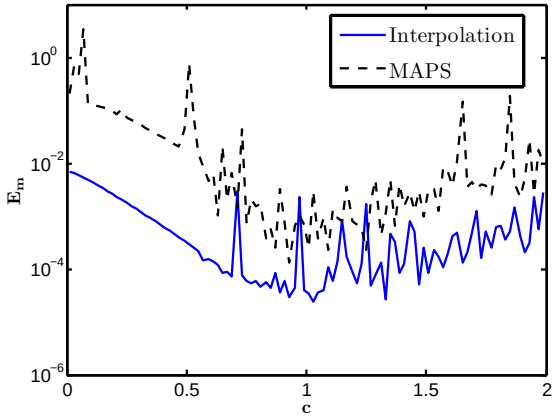
(b) MKI adaptive points; $N = 100$.



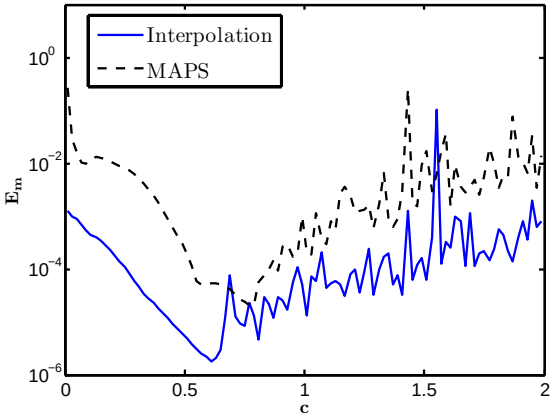
(c) Regular grid points; $N = 400$.



(d) MKI adaptive points; $N = 400$.

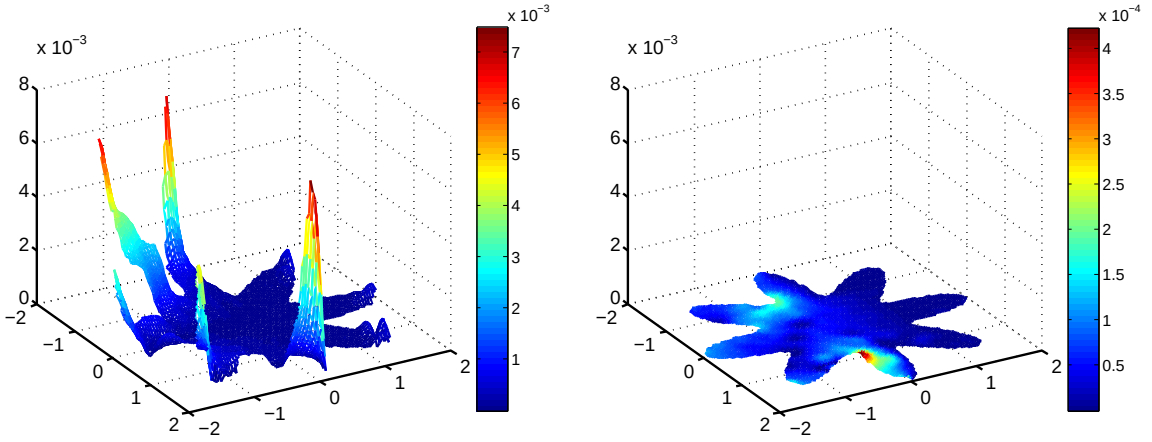


(e) Regular grid points; $N = 900$.



(f) MKI adaptive points; $N = 900$.

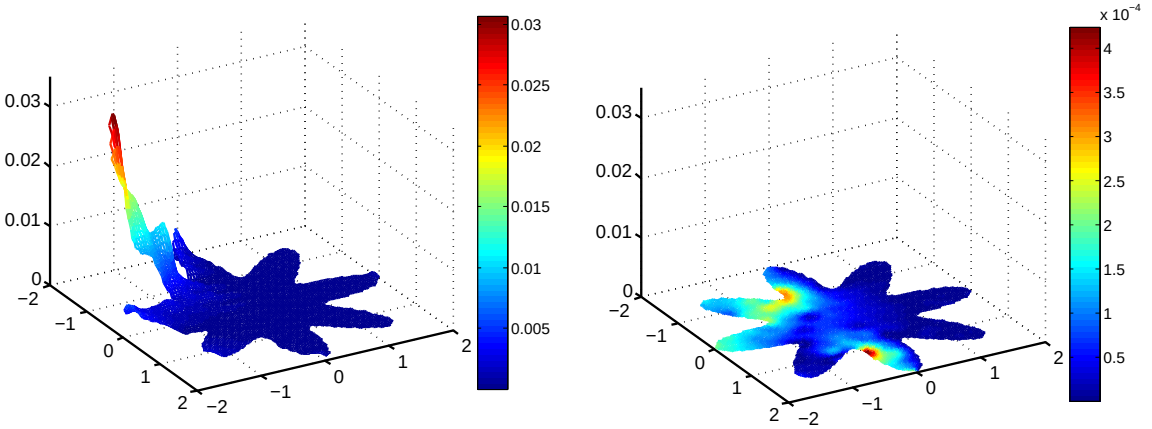
Figure 5.16: *Suitability of RECV c and MKI Adaptive Points with MAPS for convection-diffusion-reaction equation. For the analytical solution* (5.31) *to the convection-diffusion-reaction equation problem described in Example 5.2.3 in domain* Ω*, Figures (a)-(f) show the Relative Approximation Error,* E_{L^2}*, Convergence Behavior for MQ RBF interpolation (solid line) and approximation using MAPS (dash line) as N increases with either regular grid points or MKI adaptive points plotted versus shape parameter, c.*



(a) Regular grid points; $N = 400$.

(b) MKI adaptive points; $N = 400$.

Figure 5.17: *Comparison of boundary error behavior for Kansa's method using either regular grid interior points with equally spaced boundary points, or MKI adaptive collocation points to the convection-diffusion-reaction equation problem described in Example 5.2.3 in domain* Ω.



(a) Regular grid points; $N = 400$.

(b) MKI adaptive points; $N = 400$.

Figure 5.18: *Comparison of boundary error behavior for the MAPS using either regular grid interior points with equally spaced boundary points, or MKI adaptive collocation points to the convection-diffusion-reaction equation problem described in Example 5.2.3 in domain* Ω.

Chapter 6

UTILIZING THE RECV METHOD WITH THE HOUBOLT METHOD IN TIME-DEPENDENT WAVE EQUATIONS

There were many numerical methods proposed for solving hyperbolic-type partial differential equations, such as wave equations, which govern many different physical problems. For instance, water wave propagation in water bodies, the stress wave in an elastic solid and sound wave propagation in a medium. But, the development of accurate and efficient numerical methods remains an important and challenging work in the field of scientific computing.

This chapter briefly introduces the Houbolt method which is a third order finite difference scheme for time-dependent problems [39, 62, 66, 72], and then demonstrates how to couple the Kansa's method and MAPS with the Houbolt method to reduce the given wave equation to Poisson-type equation. At each time step, the MAPS, and the Kansa's method have been applied to solve the Poisson-type equation. Several tests are performed to demonstrate the stability and accuracy of utilizing the KBC method coupled with the Houbolt method.

6.1 The Wave Equation

The wave equation (6.1) is a second-order linear PDE which describes physical phenomena, and is important for solving problems in physics, electromagnetics, acoustics, and fluid dynamics.

Let Ω be a bounded domain in $\mathbb{R}^d$ with boundary $\Gamma = \Gamma_D \cup \Gamma_N$ and $\Gamma_D \cap \Gamma_N = \emptyset$. Then, the wave equation is as follows:

$$u_{tt}(\mathbf{x},t) = k^2 \Delta u(\mathbf{x},t) - f(\mathbf{x},t), \mathbf{x} \in \Omega,\ t > 0, \tag{6.1}$$

with boundary conditions:

$$\begin{aligned} u(\mathbf{x},t) &= h_D(\mathbf{x},t), \quad \mathbf{x} \in \Gamma_D,\ t > 0, \\ \frac{\partial u}{\partial \mathbf{n}}(\mathbf{x},t) &= h_N(\mathbf{x},t), \quad \mathbf{x} \in \Gamma_N,\ t > 0, \end{aligned} \tag{6.2}$$

and initial conditions:

$$u(\mathbf{x},0) = I_1(\mathbf{x}), \quad \mathbf{x} \in \Omega, \tag{6.3}$$

$$u_t(\mathbf{x},0) = I_2(\mathbf{x}), \quad \mathbf{x} \in \Omega, \tag{6.4}$$

where Δ is the Laplacian, k is a constant, t is the time variable, $\mathbf{n}$ is the outward unit vector normal to Γ and $h_D(\mathbf{x},t)$, $h_N(\mathbf{x},t)$, $I_1(\mathbf{x})$, $I_2(\mathbf{x})$, and $f(\mathbf{x},t)$ are known functions.

Many numerical methods have been developed for solving wave equations. In general, various numerical schemes are utilized to first remove the time dependent variable.

6.2 The Houbolt Method

The Houbolt method is a third order time marching scheme to transform a given wave equation into a series of Poisson-type equations. To remove the time variable, the wave equation (6.1) and its boundary conditions (6.2) are reformulated as follows:

$$\Delta u(\mathbf{x},t) = \frac{1}{k^2} u_{tt}(\mathbf{x},t) + \frac{1}{k^2} f(\mathbf{x},t), \qquad \mathbf{x} \in \Omega,\ t > 0, \tag{6.5}$$

$$\mathcal{B}u(\mathbf{x},t) = h(\mathbf{x},t), \qquad \mathbf{x} \in \Gamma,\ t > 0, \tag{6.6}$$

where $\mathcal{B}$ is the boundary operator. The time domain of the wave equation is discretized by the Houbolt finite difference method [39, 62, 72] which is required to solve the Taylor series expansions as follows:

$$u^n \approx u^{n+1} - (\delta t)\, u_t^{n+1} + \frac{(\delta t)^2}{2} u_{tt}^{n+1} - \frac{(\delta t)^3}{6} u_{ttt}^{n+1} \tag{6.7}$$

$$u^{n-1} \approx u^{n+1} - (2\,\delta t)\, u_t^{n+1} + \frac{(2\,\delta t)^2}{2} u_{tt}^{n+1} - \frac{(2\,\delta t)^3}{6} u_{ttt}^{n+1} \tag{6.8}$$

$$u^{n-2} \approx u^{n+1} - (3\,\delta t)\, u_t^{n+1} + \frac{(3\,\delta t)^2}{2} u_{tt}^{n+1} - \frac{(3\,\delta t)^3}{6} u_{ttt}^{n+1} \tag{6.9}$$

where $u^n = u(\mathbf{x},t^n)$, $\delta t = t^{n+1} - t^n$, $u_t^{n+1} = \partial u^{n+1}/\partial t$, $u_{tt}^{n+1} = \partial^2 u^{n+1}/\partial t^2$, and $u_{ttt}^{n+1} = \partial^3 u^{n+1}/\partial t^3$. After solving (6.7), (6.8), and (6.9), the Houbolt method yields:

$$u_t^{n+1} \approx \frac{1}{6\,\delta t}(11\,u^{n+1} - 18\,u^n + 9\,u^{n-1} - 2\,u^{n-2}), \tag{6.10}$$

$$u_{tt}^{n+1} \approx \frac{1}{\delta t^2}(2\,u^{n+1} - 5\,u^n + 4\,u^{n-1} - u^{n-2}). \tag{6.11}$$

Then, substituting (6.11) into (6.5) and (6.6) gives us:

$$\Delta u^{n+1} - \frac{2}{k^2\,\delta t^2} u^{n+1} = \frac{1}{c^2\,\delta t^2}(-5\,u^n + 4\,u^{n-1} - u^{n-2}) + \frac{1}{k^2} f(\mathbf{x},t^{n+1}), \quad \mathbf{x} \in \Omega, \tag{6.12}$$

$$Bu\left(\mathbf{x},t^{n+1}\right) = h\left(\mathbf{x},t^{n+1}\right), \qquad \mathbf{x} \in \Gamma. \tag{6.13}$$

In order to fully implement the Houbolt method, we need to know the initial values of the first three time steps. To accomplish this, the Euler method will be implemented (6.22) with very tiny steps to obtain the initial values of these three time steps. After that, the right hand sides of (6.12) and (6.13) are known and the Houbolt method can be started. Meanwhile, the left hand side of equations can be discretized by a Kernel-based collocation method.

6.3 Combining the KBC Methods and the Halbolt Method

In order to solve time discretized wave equations, (6.12) and (6.13), using the KBC method, let dataset $\mathcal{X} = \{\mathbf{x}_j\}_{j=1}^{N}$ be N distinct collocation points in $\overline{\Omega}$ of which $\{\mathbf{x}_j\}_{j=1}^{Ni}$ are in Ω and $\{\mathbf{x}_j\}_{j=Ni+1}^{N}$ are in Γ. The main idea of the KBC method is to approximate the exact solution $u(\mathbf{x}, t_{n+1})$ at each time step by kernel, Φ, as follows:

$$\Delta\tilde{u}(\mathbf{x}, t_{n+1}) = \sum_{j=1}^{N} \alpha_j^{n+1} \Delta\Phi(r_j) = \sum_{j=1}^{N} \alpha_j^{n+1} \phi(r_j), \tag{6.14}$$

$$\tilde{u}(\mathbf{x}, t_{n+1}) = \sum_{j=1}^{N} \alpha_j^{n+1} \Phi(r_j), \tag{6.15}$$

$$\frac{\partial\tilde{u}}{\partial\mathbf{n}}(\mathbf{x}, t_{n+1}) = \sum_{j=1}^{N} \alpha_j^{n+1} \frac{\partial\Phi}{\partial\mathbf{n}}(r_j), \tag{6.16}$$

where $r_j = \|\mathbf{x} - \mathbf{x}_j\|$, $j = 1, 2, ..., N$. Therefore, at each time step (6.12) and (6.13) can be approximated as follows:

$$\phi_i^{n+1} - \frac{2}{k^2\,\delta t^2}\Phi_i^{n+1} = \frac{1}{k^2\,\delta t^2}(-5\,\Phi^n + 4\,\Phi^{n-1} - \Phi^{n-2}) + \frac{1}{k^2} f(\mathbf{x}_i)^{n+1},\ i = 1, ..., Ni \tag{6.17}$$

$$B\Phi_i^{n+1} = h\left(\mathbf{x}_i, t^{n+1}\right), \qquad i = Ni+1, ..., N \tag{6.18}$$

where

$$\Phi_i^{n+1} = \sum_{j=1}^{m} \alpha_j^{n+1} \Phi(r_{ij}), \qquad r_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|, \tag{6.19}$$

$$\phi_i^{n+1} = \sum_{j=1}^{m} \alpha_j^{n+1} \phi(r_{ij}), \qquad r_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|. \tag{6.20}$$

The above linear system can be written as the following matrix form:

$$\begin{bmatrix} \phi - \dfrac{2}{k^2\,\delta t^2}\Phi \\ B\,\Phi \end{bmatrix}_{N\times N} \begin{bmatrix}\ \alpha^{n+1}\ \end{bmatrix}_{N\times 1} = \begin{bmatrix} \dfrac{1}{k^2\,\delta t^2}\left[-5\,\Phi^n + 4\,\Phi^{n-1} - \Phi^{n-2}\right] + \dfrac{1}{k^2} f^{n+1} \\ h\left(\mathbf{x}, t^{n+1}\right) \end{bmatrix}_{N\times 1} \tag{6.21}$$

Note that the $N \times N$ matrix in (6.21) remains unchange for each time step. Hence, we need only one matrix inversion for all the time steps. As a result, the solution process is very efficient.

The Houbolt method is a third order multi-steps scheme to deal with the time dependent problem. For solving the first two steps Φ^1 and Φ^2, we need the numerical data Φ^0, Φ^{-1}, and Φ^{-2}. Notice that Φ^0 is the given initial condition in (6.3), and the Φ^{-1}, and Φ^{-2} can

be obtained using the Euler method associated with initial conditions in (6.3) and (6.4) as follows:

$$\begin{cases} \Phi^{-1} = I_1(\mathbf{x}) - \delta t \cdot I_2(\mathbf{x}) \\ \Phi^{-2} = I_1(\mathbf{x}) - 2\,\delta t \cdot I_2(\mathbf{x}) \end{cases} \tag{6.22}$$

Note: here the symbols Φ and ϕ please referred to Table 5.1[43, 44, 8, 9].

6.4 Numerical Results

In order to demonstrate the efficacy and consistency of the KBC methods, four numerical examples are considered in this section. To validate the numerical accuracy of the solution u fairly to different test functions, the error measurement, relative approximation error (E_{L^2}), is employed and is defined as follows:

$$E_{L^2} = \left(\frac{\sum_{j=1}^{Nt} \left(u(x_j) - \tilde{u}(x_j)\right)^2}{\sum_{j=1}^{Nt} u^2(x_j)} \right)^{1/2} \tag{6.23}$$

where the Nt is the number of testing points, u and $\tilde{u}$ are exact and approximate solution respectively, are employed. The test points are randomly selected. For the notation in all the numerical examples, we denote N as the total number of interpolation points which include the interior and boundary points, Ni the number of interior points, c the shape parameter of MQ. In wave equation, the shape parameter, c, can be determined through the given initial condition which is solving the MQ interpolation as previous chapter.

Computation were performed using MATLAB on a system with the following parameters: Windows 7 operating system (32 bits), Intel Core i7-2640M processor, 2.8 GHz CPU, and 3.49 GB memory.

Example 6.4.1. In the first example, we consider the wave equation in a unit square as follows:

$$u_{tt}(x,y,t) = \Delta u(x,y,t) + f(x,y,t), \qquad (x,y) \in \Omega,\ t > 0. \tag{6.24}$$

where

$$f(x,y,t) = (2x(1-x) + 2y(1-y) - x(1-x)y(1-y))\cos(t). \tag{6.25}$$

The initial and boundary condition are given as follows:

$$u(x,y,0) = 1 + x(1-x)y(1-y), \qquad (x,y) \in \Omega, \tag{6.26}$$

$$u_t(x,y,0) = 0, \qquad (x,y) \in \Omega, \tag{6.27}$$

$$u(x,y,t) = x(1-x)y(1-y)\cos(t), \qquad (x,y) \in \Gamma. \tag{6.28}$$

The analytical solution is given as follows:

$$u(x,y,t) = 1 + x(1-x)y(1-y)\cos(t). \tag{6.29}$$

For the numerical implementation utilized 400 evenly distributed interior and boundary interpolation points in $\overline{\Omega}$ and set the time step $\delta t = 0.02$. In the Figure 6.1, we show the relative errors E_{L^2} of both methods. No significant difference in term of accuracy has been observed. In Table 6.1, the maximum relative errors of the MAPS is similar to the Kansa's method when we extended the simulation to a much larger time domain $t \in [0, 100]$.

For the results showed in Figure 6.1 and Table 6.1, we observe that the accumulated error which is commonly happen over a long simulation period is apparently not occurred for an extended period of time. This is significant for solving time-dependent problems, in particular for the wave equations. This is an indication that both methods are not only accurate but also stable. In the time-dependent problems, the stability of the computational algorithm is as important as the accuracy and efficiency.

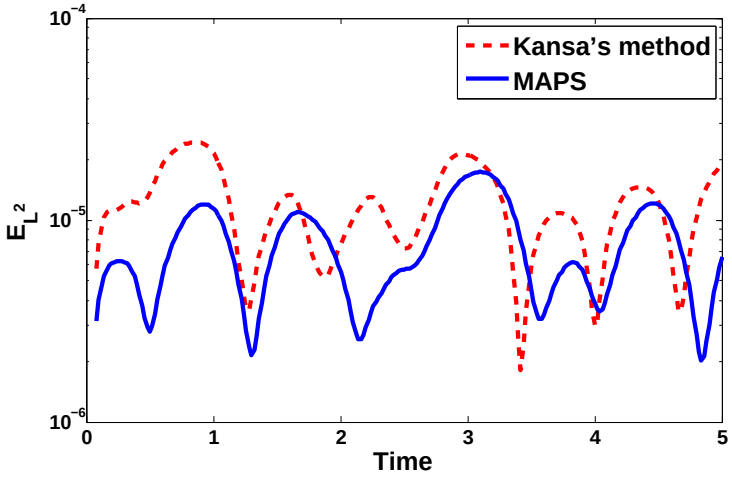


Figure 6.1: *Relative approximation error of the MAPS and the Kansa's method, E_{L^2}, using* 400 *collocation points for the wave equation described in Example 6.4.1.*

		MAPS		Kansa	
N	Ni	c	E_{L^2}	c	E_{L^2}
196	144	0.5	4.4×10^{-5}	0.64	5.5×10^{-5}
289	225	0.37	2.6×10^{-5}	0.47	3.4×10^{-5}
400	324	0.30	1.7×10^{-5}	0.39	2.6×10^{-5}
529	441	0.25	1.4×10^{-5}	0.32	2.5×10^{-5}

Table 6.1: *Relative approximation error of the MAPS and the Kansa's method, E_{L^2}, using various number of interpolation points for wave equation described in Example 6.4.1. Note that for the time step, $\delta t = 0.02$ within the time interval $t \in [0,100]$.*

Example 6.4.2. In this example, we investigate the homogeneous wave equation in irregular domain which is considered in [33]. The wave problem can be written as follows:

$$u_{tt}(x,y,t) = \Delta u(x,y,t), \qquad (x,y) \in \Omega,\ t > 0. \tag{6.30}$$

$$u(x,y,0) = 3, \qquad (x,y) \in \Omega, \tag{6.31}$$

$$u_t(x,y,0) = \frac{\sqrt{2}\pi}{10}\cos\left(\frac{\pi x}{10}\right)\cos\left(\frac{\pi y}{10}\right), \qquad (x,y) \in \Omega, \tag{6.32}$$

$$u(x,y,t) = 3+\cos\left(\frac{\pi x}{10}\right)\cos\left(\frac{\pi y}{10}\right)\sin\left(\frac{\sqrt{2}\pi t}{10}\right), \qquad (x,y) \in \Gamma. \tag{6.33}$$

The analytical solution is given by

$$u(x,y,t) = 3+\cos\left(\frac{\pi x}{10}\right)\cos\left(\frac{\pi y}{10}\right)\sin\left(\frac{\sqrt{2}\pi t}{10}\right) \tag{6.34}$$

The irregular domain (see Figure 6.2) is defined as follows:

$$\partial\Omega = \{(r(\theta)\cos(\theta), r(\theta)\sin(\theta)) : \theta \in [0,2\pi)\}$$

where

$$r(\theta) = 10\left[\cos(4\theta) + \left(\frac{18}{5} - \sin^2(4\theta)\right)^{1/2}\right]^{1/3}$$

In the numerical implementation, we choose 400 evenly distributed interior and boundary points in $\overline{\Omega}$ with the time step $\delta t = 0.02$ for both methods. Figure 6.3 (top) shows that the E_{L^2} error over a long period of time ($t \in [0,120]$). We observe that the errors are stably oscillating between 10^{-5} and 10^{-4} without accumulating the error. In Figure 6.3 (top) we

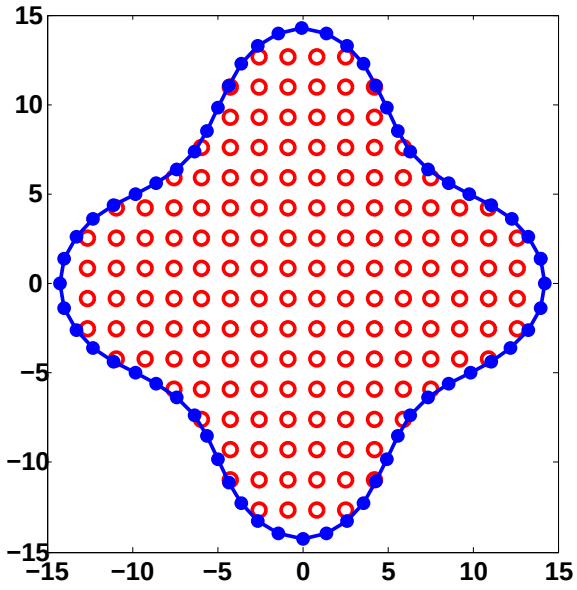


Figure 6.2: *Profile of the computational domain (Cassini) described in Example 6.4.2.*

observe that the error of the MAPS method is half order more accurate than the Kansa's method. In Table 6.2, we observe that the MAPS requires only $Nt = 200$ to reach a very high accuracy in a large domain and a long period of time $t \in [0, 120]$. The MAPS converges rapidly and hence very efficient. Furthermore, when comparing the results with the MFS-MPS and FEM as shown in Figure 6.3 (bottom) [33], the MAPS is apparently much superior in term of accuracy and stability. From Table 6.2, the MAPS is slightly better than the Kansa's method.

		MAPS		Kansa	
N	Ni	c	E_{L^2}	c	E_{L^2}
200	136	6.8	4.5×10^{-5}	9.0	1.4×10^{-4}
300	205	5.3	4.4×10^{-5}	7.6	7.0×10^{-5}
400	268	2.8	4.8×10^{-5}	3.9	9.9×10^{-5}
500	328	2.3	6.9×10^{-5}	3.8	9.8×10^{-5}
1400	976	1.3	5.5×10^{-5}	2.1	1.2×10^{-4}

Table 6.2: *Relative approximation error of the MAPS and the Kansa's method, E_{L^2}, using various number of interpolation points for wave equation described in Example 6.4.2. Note that for the time step, $\delta t = 0.02$ within the time domain $t \in [0, 120]$.*

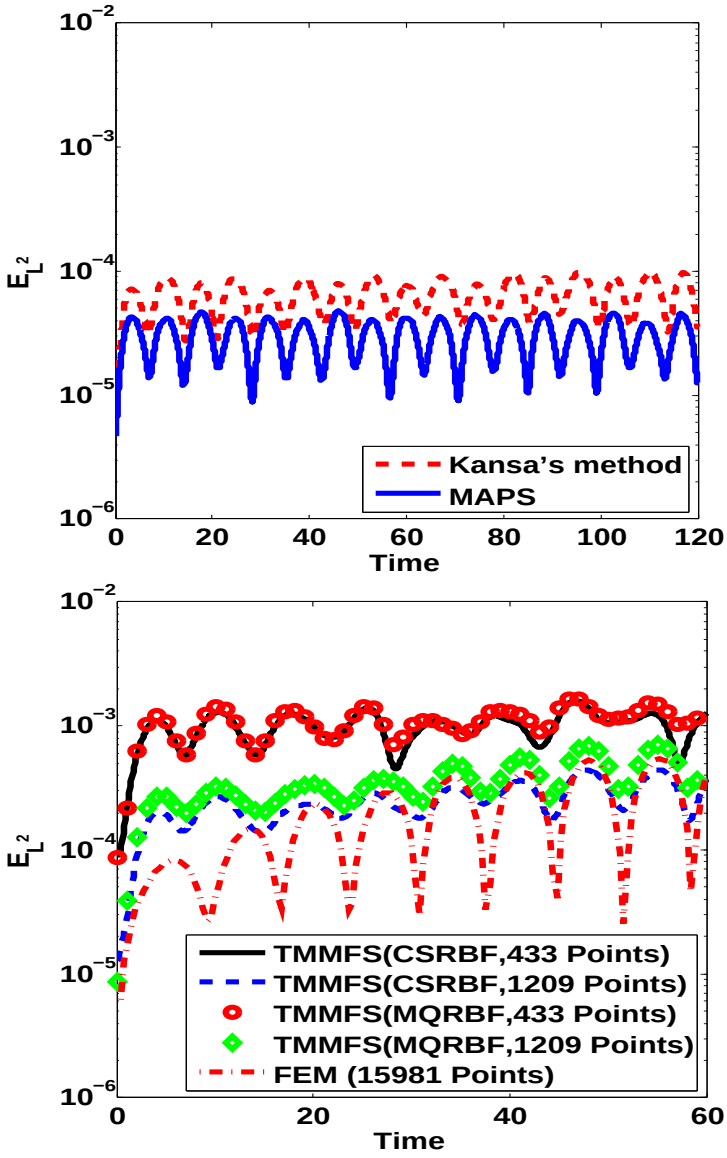


Figure 6.3: *Top: Relative approximation error of the MAPS and the Kansa's method, E_{L^2}, using* 400 *collocation points . Bottom: Relative approximation error of TMMFS and FEM [33].*

Example 6.4.3. In this example we consider the same wave equation as in the last example but with more complicate domain and without symmetry. The profile of the domain is shown in Figure 6.4. The parametric equation of the boundary of the domain is as follows:

$$\partial\Omega = \{(r(\theta)\cos(\theta), r(\theta)\sin(\theta)) : \theta \in [0, 2\pi)\}$$

where

$$r(\theta) = e^{\sin(\theta)}\sin^2(2\theta) + e^{\cos(\theta)}\cos^2(2\theta).$$

In the numerical implementation, we choose $\delta t = 0.02$ for the Houbolt method. In general,

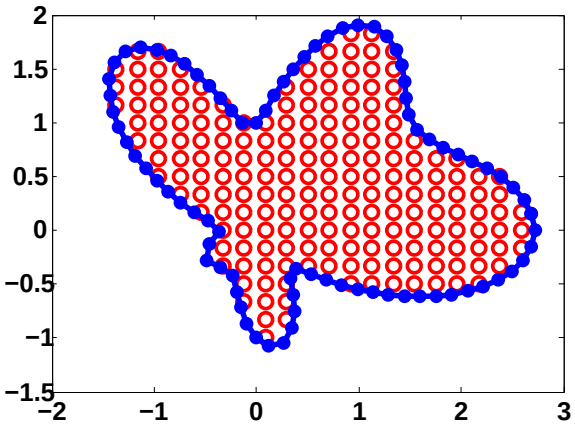


Figure 6.4: The profile of computational domain (Amoeba) in Example 6.4.3.

the error will oscillate when time goes by for wave equation. But in the case, Figure 6.5 shows that the error is coincidentally stable in the time domain $t \in [0, 120]$. In Table 6.3, we observe that the MAPS is slightly more accurate than the Kansa's method. Both methods converge rapidly to its potential. Therefore, the numerical error for the the MAPS is more stable and a half order more accurate than the Kansa's method in this example of irregular domain.

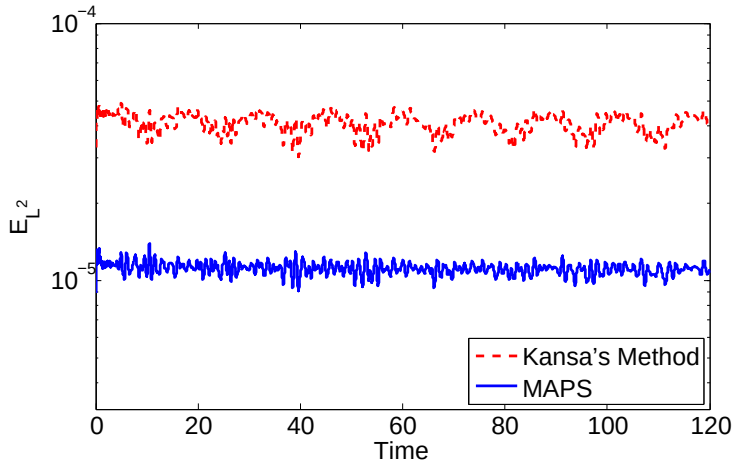


Figure 6.5: *Relative approximation error of the MAPS and the Kansa's method,* E_{L^2}*, using* 400 *collocation points for the wave equation described in Example 6.4.3.*

		MAPS		*Kansa*	
N	Ni	c	E_{L^2}	c	E_{L^2}
100	64	0.05	4.6×10^{-4}	0.21	1.6×10^{-3}
200	145	0.1	1.5×10^{-4}	0.19	4.8×10^{-4}
300	237	0.12	1.0×10^{-4}	0.25	3.3×10^{-4}
400	317	0.22	1.4×10^{-5}	0.35	4.9×10^{-5}

Table 6.3: *Relative approximation error of the MAPS and the Kansa's method, E_{L^2}, using various number of interpolation points for wave equation described in Example 6.4.3. Note that for the time step, $\delta t = 0.02$ within the time domain $t \in [0,120]$.*

Example 6.4.4. In this example, we consider the following wave equation

$$u_{tt}(x,y,t) = \Delta u(x,y,t), \qquad (x,y) \in \Omega,\ t > 0 \tag{6.35}$$

$$u(x,y,0) = 2 + \sin(\frac{\pi x}{4})\sin(\frac{\pi y}{4}), \qquad (x,y) \in \Omega, \tag{6.36}$$

$$u_t(x,y,0) = 0, \qquad (x,y) \in \Omega, \tag{6.37}$$

$$u(x,y,t) = 2, \qquad (x,y) \in \partial\Omega, \tag{6.38}$$

where Ω is a L shape domain (see Figure 6.6). The analytical solution is given as follows:

$$u(x,y,t) = 2 + \sin(\frac{\pi x}{4})\sin(\frac{\pi y}{4})\cos(\frac{\sqrt{2}\pi t}{4}) \tag{6.39}$$

This example intend to test the effectiveness of both methods for the non-smooth and irregular domain. For the numerical implementation, the time step $\delta t = 0.02$ is chosen for the Houbolt method for $t \in [0,120]$. In Figure 6.7, we choose $Nt = 408$ for both the Kansa's method and the MAPS which has the clear advantage with respect to the FEM. The Figure 6.7 and the Table 6.4 are the results of the Kansa's, MAPS, and FEM methods. In Table 6.4, we observe that the MAPS requires only $Nt = 225$ to reach the accuracy of 10^{-4} while the Kansa's method requires $Nt = 408$ to have such accuracy. For the FEM, the convergent rate is much slower. The MAPS has the clear advantages in term of accuracy, convergent rate, and stability.

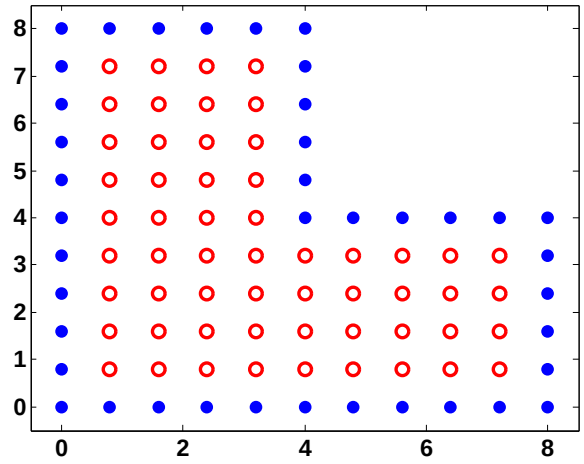


Figure 6.6: The profile of the non-smooth Domain (L shape) in Example 6.4.4.

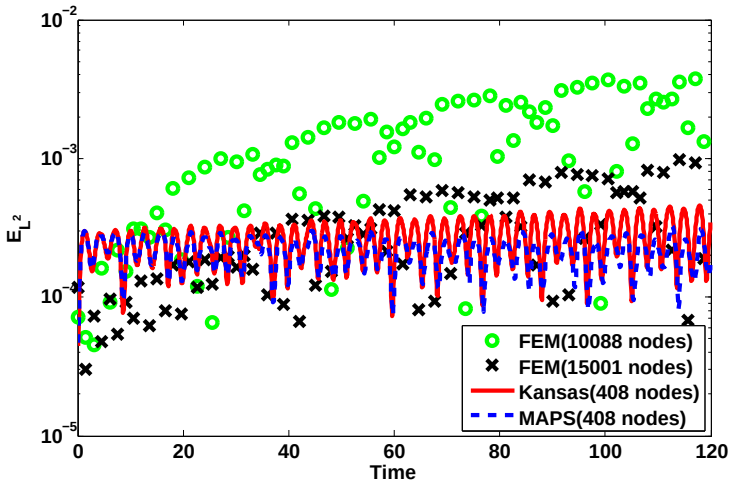


Figure 6.7: *Relative approximation error of the MAPS and the Kansa's method,* E_{L^2}*, using* 408 *collocation points for the wave equation described in Example 6.4.4.*

		MAPS		*Kansa*		*FEM*	
N	Ni	c	E_{L^2}	c	E_{L^2}	N	E_{L^2}
96	56	3.3	1.5×10^{-2}	4.2	2.9×10^{-2}	1027	2.2×10^{-2}
225	161	2.1	4.1×10^{-4}	2.95	1.9×10^{-3}	5004	2.6×10^{-3}
408	320	1.28	3.3×10^{-4}	2.1	4.6×10^{-4}	10088	1.8×10^{-4}
736	616	0.68	2.9×10^{-4}	1.35	3.0×10^{-4}	15001	3.5×10^{-4}

Table 6.4: *Relative approximation error of the MAPS and the Kansa's method,* E_{L^2}*, using various number of interpolation points for wave equation described in Example 6.4.4. Note that for the time step,* $\delta t = 0.02$ *within the time interval* $t \in [0, 120]$.

Chapter 7

CONCLUSIONS AND FUTURE WORK

7.1 Conclusions

This dissertation introduces two significant contributions, the Residue–Error Cross Validation (RECV) method, and the Modified Knot Insertion (MKI) method. The RECV method is a new strategy to select an optimal shape parameter, c, for MQ RBF approximation. The MKI method, on the other hand, is a new strategy for choosing adaptive collocation points for MQ RBF approximation.

Numerical experimentation demonstrated that, by using the RECV method, c values which yield highly accurate numerical results can be obtained with minimal computational costs. The specific approximation error of MQ RBF interpolation utilizing RECV c values was explored exhaustively, demonstrating the pros and cons of using the RECV method for different situations such as low-density or high-density problems, with variously large or small data sets, and with simple or complex functions.

The accuracy of interpolation utilizing RECV c was found to rely on the fill distance, h. In general, RECV c was found to provide highly accurate approximations in higher density point sets. However, RECV c could be over-determined in lower density point sets, particularly for singular functions.

In order to overcome this issue, the accuracy of approximation utilizing the RECV method was further improved for small data sets by combining the RECV method with the *Leave–One–Out* Cross Validation (LOOCV) method. In this 'combined' method, the RECV c for a function was utilized as the upper bound of the search interval considered by the LOOCV method. Combining RECV with LOOCV was found to further increase stability and accuracy for low-density problems.

A particular advantage of the RECV method is that it can quickly find an optimal c value in situations where the local optimum issue must otherwise be overcome by computationally expensive means, such as by using the *Moore-Penrose* pseudoinverse algorithm[1] [23]. By avoiding more expensive strategies and instead combining the RECV method and the LOOCV method, computational costs can be reduced by almost 100 times.

[1] a built-in Matlab algorithm, `pinv; computation is based on Singular Value Decomposition (SVD); any singular values less than tol are truncated.`

The second significant contribution of this dissertation was the introduction of the MKI method, an adaptive strategy developed by modifying Franke's knot insertion method [cite Franke]. Numerical experimentation demonstrated that MKI adaptive points yield far more accurate results than either regular grid points or Halton quasi-random uniformly distributed points. Specifically, MKI adaptive points significantly increase the convergence rate of interpolations and overcome the boundary error issues associated with MQ RBF approximation. Additionally, numerical experimentation demonstrated that the MKI method can be used with MQ integrations and MQ derivatives, as well as with MQs.

An important feature of both the RECV method and the MKI method relates to the generalizability of both RECV c and MKI adaptive points:

1. Regarding the generalizability of RECV c, numerical experimentation demonstrated that optimal shape parameter, c, values depend, not on specific functions, as was previously believed [7], but on the domain size, the number of interpolation points, and the method by which the points are selected. It was demonstrated that the c value selected by the RECV method, using any known smooth function, could be utilized for any unknown function within the same domain, using the same interpolation points, to achieve highly accurate approximations.

2. Regarding the generalizability of MKI adaptive points, numerical experimentation demonstrated that MKI adaptive points generated for any known smooth function within the same domain could be utilized to interpolate an unknown function, resulting in highly accurate approximations.

In addition to improving interpolation accuracy and efficiency for MQ RBF interpolation, the RECV method and the MKI method can both be applied to PDE problems with both Kansa's method and MAPS (the Method of Approximate Particular Solution), two well-known domain-type meshless Kernal–Based Colocation (KBC) methods. Numerical experiments demonstrated that RECV c and MKI adaptive points improve the convergence rate of approximations using both Kansa's method and MAPS, and smooth the oscillation of accuracy caused by complex boundary conditions.

7.2 Future Work

The primary goal of the work in this dissertation, the development and testing of the RECV method and the MKI adaptive point method, was to enable practitioners to more easily perform MQ RBF interpolations and utilize KBC meshless methods with PDE problems, without facing difficulty in choosing an optimal shape parameter values or in selecting ideal collocation points.

While both the RECV method and the MKI method are powerful and effective, several areas still remain where further research could refine or find new applications for the techniques, or utilize the underlying principles to develop useful methods for other situations where convergence might point to ideal constant values.

One potential problem with the MKI method is CPU usage, which can increase exponentially as more points are called for. For example, MKI, when run on the system utilized in this research [footnote], takes 30 seconds to generate 100 adaptive points, but takes 30 minutes to generate 200 adaptive points. Therefore, more research should be done to try to streamline or refine the MKI method for use with larger data sets.

In addition, several other topics seem to merit additional investigation:

- Currently the MKI method adds one point to the set of adaptive points being generated with each iteration. A more efficient algorithm, perhaps adding multiple points with each iteration, could significantly reduce required CPU usage.

- It may be possible to combine the MKI method with other adaptive algorithms to generate more useful point sets for singular functions.

- While the RECV method and the MKI method have proven to be effective at improving accuracy and reducing computational costs with both Kansa's method and MAPS, both the RECV method and the MKI method should be tested for suitability with other domain-type KBC methods for use with PDEs.

- In the past two decades, more than a hundred publications, *e.g.* [16, 19, 23] have explored the trade-off principle in global KBC methods. However, shape parameters needed in global domains are not the same as those needed in local domains. In designing the RECV method, an indicator was proposed to select ideal shape parameters. Further research should be done exploreing ways to use the RECV method to redefine the local domain and choose an optimal shape parameter for local KBC methods.

- Both the RECV method and the MKI method should be tested for suitability with the Method of Fundamental Solutions (MFS), for use with PDEs.

- Madych and Nelson [53, 54] proposed that the convergence rate of KBC methods should yield a given bound. However, numerical implementation showed the proposed bound may never be reached due to the ill-conditioned nature of interpolation matrices; i.e., the accuracy of approximation becomes increasingly unstable as the condition number increases. It may be possible to utilize the RECV method, along with an appropriate conditioning method, to aleviate this problem.

Appendix A
Derivatives and Particular Solutions of MQ

We provide detailed derivation of all formulas as commonly used differential operators listed in Table 5.1 for MQ in $\mathbb{R}^2$.

A.1 Derivatives for Kansa's Method

For Kansa's method, the derivatives of MQ, $\Phi(r,c) = \sqrt{r^2+c^2}$, following the chain rule implies:

$$\begin{aligned} \frac{\partial \Phi(r)}{\partial x} &= \frac{\partial r}{\partial x} \cdot \frac{d\Phi(r)}{dr} \\ &= \frac{x}{r} \cdot \frac{d\Phi(r)}{dr} = \frac{x}{\sqrt{r^2+c^2}}, \end{aligned} \tag{A.1}$$

Similarly,

$$\begin{aligned} \frac{\partial \Phi(r)}{\partial y} &= \frac{\partial r}{\partial y} \cdot \frac{d\Phi(r)}{dr} \\ &= \frac{y}{r} \cdot \frac{d\Phi(r)}{dr} = \frac{y}{\sqrt{r^2+c^2}}. \end{aligned} \tag{A.2}$$

The second order derivatives are given by:

$$\begin{aligned} \frac{\partial^2 \Phi(r)}{\partial x^2} &= \frac{\partial}{\partial x}\left(\frac{x}{r} \cdot \frac{d\Phi(r)}{dr}\right) \\ &= \frac{\partial}{\partial x}\left(\frac{x}{r}\right) \cdot \frac{d\Phi(r)}{dr} + \left(\frac{x}{r}\right) \cdot \frac{\partial}{\partial x}\left(\frac{d\Phi(r)}{dr}\right) \\ &= \frac{r^2-x^2}{r^3}\frac{d\Phi(r)}{dr} + \frac{x^2}{r^2}\frac{d^2\Phi(r)}{dr^2} \\ &= \frac{y^2+c^2}{(r^2+c^2)^{3/2}}, \end{aligned} \tag{A.3}$$

as well as

$$\frac{\partial^2 \Phi(r)}{\partial y^2} = \frac{y^2+c^2}{(r^2+c^2)^{3/2}}, \tag{A.4}$$

and the Laplace differential operator or Laplacian

$$\phi(r) = \frac{\partial^2 \Phi(r)}{\partial x^2} + \frac{\partial^2 \Phi(r)}{\partial y^2} = \frac{r^2+2c^2}{(r^2+c^2)^{3/2}}. \tag{A.5}$$

A.2 Particular Solution for MAPS

In $\mathbb{R}^2$, let MQ RBF $\phi(r) = \sqrt{r^2+c^2}$. Then for MAPS, the Laplacian in radial coordinates can be expressed as:

$$\phi(r) = \Delta\Phi = \frac{1}{r}\frac{d}{dr}\left(r\frac{d\Phi(r)}{dr}\right). \tag{A.6}$$

Then,

$$\frac{1}{r}\frac{d}{dr}\left(r\frac{d\Phi(r)}{dr}\right) = \sqrt{r^2+c^2}. \tag{A.7}$$

This implied the $\Phi(r)$ can be calculated as follows:

$$\Phi(r) = \int \frac{1}{r}\cdot\left(\int r\cdot\sqrt{r^2+c^2}\cdot dr\right)\cdot dr, \tag{A.8}$$

$$= \frac{(4c^2+r^2)\cdot\sqrt{r^2+c^2}}{9} - \frac{c^3}{3}\ln\left(\frac{2c+2\sqrt{r^2+c^2}}{c^4 r}\right) + C_1\ln r + C_2, \tag{A.9}$$

$$= \frac{(4c^2+r^2)\cdot\sqrt{r^2+c^2}}{9} - \frac{c^3}{3}\ln\left(c+\sqrt{r^2+c^2}\right) + \frac{c^3}{3}\ln(\frac{c^4}{2})$$
$$+ \frac{c^3}{3}\ln(r) + C_1\ln r + C_2, \tag{A.10}$$

where C_1, C_2 are constants. In order to eliminate the singularity of $\Phi(r)$ at $r = 0$. we choose

$$C_1 = -\frac{c^3}{3}, \quad C_2 = -\frac{c^3}{3}\ln(\frac{c^4}{2}).$$

Then, the particular solution of $\phi(r)$ can be expressed by:

$$\Phi(r) = \frac{(4c^2+r^2)\cdot\sqrt{r^2+c^2}}{9} - \frac{c^3}{3}\ln\left(c+\sqrt{r^2+c^2}\right). \tag{A.11}$$

Furthermore,

$$\begin{aligned}
\frac{\partial\Phi}{\partial x} = \frac{\partial r}{\partial x}\frac{d\Phi}{dr} &= x\cdot\left(\frac{3r^2+6c^2}{9\sqrt{r^2+c^2}} - \frac{c^3}{3\sqrt{r^2+c^2}\left(\sqrt{r^2+c^2}+c\right)}\right) \\
\frac{\partial\Phi}{\partial y} = \frac{\partial r}{\partial y}\frac{d\Phi}{dr} &= y\cdot\left(\frac{3r^2+6c^2}{9\sqrt{r^2+c^2}} - \frac{c^3}{3\sqrt{r^2+c^2}\left(\sqrt{r^2+c^2}+c\right)}\right)
\end{aligned} \tag{A.12}$$

Appendix B
Additional Computer Program

B.1 Hardy's shape parameter

Listing B.1: MQHardyShapeParameter.m

```
function []=MQHardyShapeParameter()
tic
% Define Hardy's MQ.
rbf = @(c,r) sqrt(r.^2 + c^2);
% Define unknown function which will be recovered.
%by MQ interpolation.
testfunction = @(x,y) sinc(2*x).*sinc(2*y);
% Define number of center points.
N = [100 400 900 1600];
% Generate test points.
[X,Y] = meshgrid(linspace(-0.5,0.5,sqrt(2500)));
test=[X(:) Y(:)]; clear X Y;
% Sample data in test poins.
zt = testfunction(test(:,1),test(:,2));
for i=1:length(N)
    % Generate center points.
    [X,Y] = meshgrid(linspace(-0.5,0.5,sqrt(N(i))));
    ctrs=[X(:) Y(:)]; clear X Y;
    % Create right hand side vector.
    z = testfunction(ctrs(:,1),ctrs(:,2));
    % Compute distance matrix.
    DM = DMatrix(ctrs,ctrs);
    DMt = DMatrix(test,ctrs);
    % Hardy's optimal c.
    c = 0.815/(N(i)-1);
    % Solved the system of equations.
    A = rbf(c,DM);
    Alpha = A\z;
    % Evaluate the error at given test points.
    At = rbf(c,DMt);
    Error = At*Alpha-zt;
    Maxerr(i) = norm(Error, inf);
    fprintf('=======================================\n');
    fprintf('Number of interpolant: %i\n', N(i))
    fprintf('Maximum error: %e\n', Maxerr(i))
    fprintf('c = %f\n', c)
    fprintf('CPU time = %f\n', toc)
    fprintf('=======================================\n');
end
end
```

B.2 Franke's shape parameter

Listing B.2: MQFrankeShapeParameter.m

```
function []=MQFrankeShapeParameter()
tic
% Define Hardy's MQ.
rbf = @(c,r) sqrt(r.^2 + c^2);
% Define unknown function which will be recovered.
%by MQ interpolation.
testfunction = @(x,y) sinc(2*x).*sinc(2*y);
% Define number of center points.
N = [100 400 900 1600];
% Generate test points.
[X,Y] = meshgrid(linspace(-0.5,0.5,sqrt(2500)));
test=[X(:) Y(:)]; clear X Y;
% Sample data in test poins.
zt = testfunction(test(:,1),test(:,2));
for i=1:length(N)
    % Generate center points.
    [X,Y] = meshgrid(linspace(-0.5,0.5,sqrt(N(i))));
    ctrs=[X(:) Y(:)]; clear X Y;
    % Create right hand side vector.
    z = testfunction(ctrs(:,1),ctrs(:,2));
    % Compute distance matrix.
    DM = DMatrix(ctrs,ctrs);
    DMt = DMatrix(test,ctrs);
    % Hardy's optimal c.
    c = sqrt(2/0.64/N(i));
    % Solved the system of equations.
    A = rbf(c,DM);
    Alpha = A\z;
    % Evaluate the error at given test points.
    At = rbf(c,DMt);
    Error = At*Alpha-zt;
    Maxerr(i) = norm(Error, inf);
    fprintf('=======================================\n');
    fprintf('Number of interpolant: %i\n', N(i))
    fprintf('Maximum error: %e\n', Maxerr(i))
    fprintf('c = %f\n', c)
    fprintf('CPU time = %f\n', toc)
    fprintf('=======================================\n');
end
end
```

BIBLIOGRAPHY

[1] F Afiatdoust and M Esmaeilbeigi. Optimal variable shape parameters using genetic algorithm for radial basis function approximation. *Ain Shams Engineering Journal*, 2015.

[2] P Alotto and MA Nervi. An efficient hybrid algorithm for the optimization of problems with several local minima. *International Journal for Numerical Methods in Engineering*, 50(4):847–868, 2001.

[3] D. Braess. *Finite Elements. Theory, Fast Solvers and Applications in Solid Mechanics*. Cambridge University Press, 2001.

[4] S. C. Brenner and L. R. Scott. *The Mathematical Theory of Finite Element Methods, Second edition*. Springer, 2002.

[5] M. D. Buhmann. *Radial Basis Functions, Theory and Implementations*. Cambridge University Press, 2003.

[6] Ralph E. Carlson and Thomas A. Foley. The parameter r< sup> 2</sup> in multiquadric interpolation. *Computers & Mathematics with Applications*, 21(9):29–42, 1991.

[7] Ralph E. Carlson and Thomas A. Foley. The parameter {R2} in multiquadric interpolation. *Computers & Mathematics with Applications*, 21(9):29 – 42, 1991.

[8] C. S. Chen, C. M Fan, and P. H. Wen. The method of particular solutions for solving elliptic problems with variable coefficients. *The International Journal for Numerical Methods in Biomedical Engineering*, 8:545–559, 2011.

[9] C. S. Chen, C. M Fan, and P. H. Wen. The method of particular solutions for solving certain partial differential equations. *Numerical Methods for Partial Differential Equations*, 28:506–522, 2012. doi: 10.1002/num.20631.

[10] C. S. Chen, M. A. Golberg, and Y. F. Rashed. A mesh free method for linear diffusion equations. *Numerical Heat Transfer, Part B*, 33:469–486, 1998.

[11] C. S. Chen and Y. F. Rashed. Evaluation of thin plate spline based particular solutions for Helmholtz-type operators for the DRM. *Mech. Res. Comm.*, 25:195–201, 1998.

[12] J. T. Chen, Y. T. Lee Lee, S. S. R. Yu, and S. C. Shieh. Equivalence between the trefftz method and the method of fundamental solution for the annular green's function using the addition theorem and image concept. *Engineering Analysis with Boundary Elements*, 33(5):678–688, 2009.

[13] J. T. Chen, C. S. Wu, Y. T. Lee, and K. H. Chen. On the equivalence of the trefftz method and method of fundamental solutions for laplace and biharmonic equations. *Computers and Mathematics with Applications*, 53(6):851–879, 2007.

[14] W. Chen. Symmetric boundary knot method. *Engineering Analysis with Boundary Elements*, 26:489–494, 2002.

[15] Y. W. Chen, C.S. Liu, and J. R. Chang. Applications of the modified trefftz method for the laplace equation. *Engineering Analysis with Boundary Elements*, 33(2):137–146, 2009.

[16] A. H.-D. Cheng. Multiquadric and its shape parameter – a numerical investigation of error estimate, condition number, and round-off error by arbitrary precision computation. *Engineering Analysis with Boundary Elements*, 36(2):220(20), 2012-02-01.

[17] T. A. Driscoll and B. Fornberg. Interpolation in the limit of increasingly flat radial basis functions. *Computers & Mathematics with Applications*, 43(3âĂŞ5):413 – 422, 2002.

[18] Tobin A Driscoll and Alfa RH Heryudono. Adaptive residual subsampling methods for radial basis function interpolation and collocation problems. *Computers & Mathematics with Applications*, 53(6):927–939, 2007.

[19] M Esmaeilbeigi and MM Hosseini. A new approach based on the genetic algorithm for finding a good shape parameter in solving partial differential equations by kansaâĂŹs method. *Applied Mathematics and Computation*, 249:419–428, 2014.

[20] G Fairweather and A. Karageorghis. The method of fundamental solution for elliptic boundary value problems. *Advances in Computatonal Mathematics*, 9:69–95, 1998.

[21] G. E. Fasshauer. *Meshfree Approximation Methods with MATLAB*, volume 6 of *Interdisciplinary Mathematical Sciences*. World Scientific, 2007.

[22] G. E. Fasshauer and M. McCourt. Stable evaluation of gaussian radial basis function interpolants. *SIAM Journal on Scientific Computing*, 34(2):A737–A762, 2012.

[23] G. E. Fasshauer and Jack G Zhang. On choosing optimal shape parameters for rbf approximation. *Numerical Algorithms*, 45(1-4):345–368, 2007.

[24] Gregory E Fasshauer. Solving partial differential equations by collocation with radial basis functions. In *Proceedings of Chamonix*, volume 1997, pages 1–8. Citeseer, 1996.

[25] B Fornberg, TA Driscoll, G Wright, and R Charles. Observations on the behavior of radial basis function approximations near boundaries. *Computers & Mathematics with Applications*, 43(3):473–490, 2002.

[26] B. Fornberg and G. Wright. Stable computation of multiquadric interpolants for all values of the shape parameter. *Comput. Math. Appl.*, 48(5-6):853–867, 2004.

[27] Bengt Fornberg and Julia Zuev. The runge phenomenon and spatially variable shape parameters in rbf interpolation. *Computers & Mathematics with Applications*, 54(3):379 – 398, 2007.

[28] R. Franke. Scattered data interpolation: tests of some methods. *Mathematics of Computation*, 48:181–200, 1982.

[29] Richard Franke, GM Nielson, and H Hagen. *Repeated knots in least squares multiquadric functions*. Springer, 1995.

[30] M. A. Golberg. The method of fundamental solutions for poissonâĂŹs equations. *Engineering Analysis with Boundary Elements*, 16(3):205–213, 1995.

[31] M. A. Golberg and C. S. Chen. The method of fundamental solutions for potential, Helmholtz and diffusion problems. In M.A. Golberg, editor, *Boundary Integral Methods: Numerical and Mathematical Aspects*, pages 103–176. WIT Press, 1998.

[32] M. A. Golberg, C.S. Chen, and S. Karur. Improved multiquadric approximation for partial differential equations. *Engineering Analysis with Boundary Elements.*, 18(1):9–17, 1996.

[33] M. H. Gu, C. M. Fan, and D. L. Young. The method of fundamental solutions for the multi-dimensional wave equations. *Journal of Marine Science and Technology*, 19(6):586–595, 2011.

[34] M. H. Gu., D. L. Young, and C. M. Fan. The method of fundamental solutions for one-dimensional wave equations. *CMC: Computers, Materials & Continua*, 11(3):185–208, 2009.

[35] R. L. Hardy. Multiquadric equations of Topography and other irregular surfaces. *Journal of Geophysical Research*, 176:1905–1915, 1971.

[36] R. L. Hardy. Geodetic applications of multiquadric analysis. *AVN Allg. Vermess. Nachr.*, 79:389–406, 1972.

[37] Y. C. Hon. Typhoon surge in Tolo Harbour of Hong Kong - an approach using finite element method with quadrilateral elements and parallel processing technique. *Chinese J. Num. Math. Appl.*, 15(4):21–33, 1993.

[38] Y. C. Hon and R. Schaback. On unsymmetric collocation by radial basis functions. *J. Appl. Math. Comp.*, 119:177–186, 2001.

[39] J. C. Houbolt. A recurrence matrix solution for the dynamic response of elastic aircraft. *Journal of the Aeronautical Sciences*, 17:540–550, 1950.

[40] Alexander Hrennikoff. Solution of problems of elasticity by the framework method. *Journal of applied mechanics*, 8(4):169–175, 1941.

[41] A. Iske. Reconstruction of functions from generalized Hermite-Birkhoff data. In C.K. Chui and L.L. Schumaker, editors, *Approximation Theory VIII, Vol. 1*, pages 257–264. World Scientific, Singapore, 1995.

[42] Fangming Jiang, MÃşnica S.A. Oliveira, and Antonio C.M. Sousa. Mesoscale sph modeling of fluid flow in isotropic porous media. *Computer Physics Communications*, 176:471–480, 2007.

[43] E. J. Kansa. Multiquadrics - a scattered data approximation scheme with applications to computational fluid dynamics - I. *Comput. Math. Applic.*, 19(8/9):127–145, 1990.

[44] E. J. Kansa. Multiquadrics - a scattered data approximation scheme with applications to computational fluid dynamics - II. *Comput. Math. Applic.*, 19(8/9):147–161, 1990.

[45] G. Kosec and B. Šarler. H-adaptive local radial basis function collocation meshless method. *Computers Materials and Continua*, 26(3):227, 2011.

[46] L. H. Kuo, M. H. Gu, D. L. Young, and C. Y. Lin. Domain type kernel-based meshless methods for solving wave equations. *CMC-COMPUTERS MATERIALS & CONTINUA*, 33(3):213–228, 2013.

[47] V. D. Kupradze and M. A. Aleksidze. The method of functional equations for the approximate solution of certain boundary value problems. *U.S.S.R. Computational Mathematics and Mathematical Physics*, 4:82–126, 1964.

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

[49] E. Larsson and B. Fornberg. Theoretical and computational aspects of multivariate interpolation with increasingly flat radial basis functions. *Comput. Math. Appl.*, 49(1):103–130, 2005.

[50] C. K. Lee, X. Liu, and S. C. Fan. Local multiquadric approximation for solving boundary value problems. *Computational Mechanics*, 30:396–409, 2003.

[51] N. A. Libre, A. Emdadi, E. J. Kansa, M. Rahimian, and M. Shekarchi. A stabilized rbf collocation scheme for neumann type boundary value problems. *Comput. Model. Eng. Sci.*, 24:61–80, 2008.

[52] W. R. Madych. Curves and surfaces. chapter Error estimates for interpolation by generalized splines, pages 297–306. Academic Press Professional, Inc., San Diego, CA, USA, 1991.

[53] W. R. Madych. Miscellaneous error bounds for multiquadric and related interpolators. *Computers and Mathematics with Applications*, 24(12):121–138, 1992.

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

[55] R. Mathon and R. L. Johnston. The approximate solution of elliptic boundary-value problems by fundamental solutions. *SIAM J. Numer. Anal.*, 14:638–650, 1977.

[56] J. M. Melenk and I. Babuska. The partition of unity finite element method: Basic theory and applications. *Comput. Meths. Appl. Mech. Engrg.*, 139:289–314, 1996.

[57] G. J. Moridis and Kansa E. J. The Laplace transform multiquadric method: A highly accurate scheme for the numerical solution of linear partial differential equations. *J. Appl. Sci. Comp.*, 1:375–407, 1994.

[58] S. Rippa. An algorithm for selecting a good value for the parameter c in radial basis function interpolation. *Advances in Computational Mathematics*, 11:193–210, 1999.

[59] S. A. Sarra. Integrated multiquadric radial basis function approximation methods. *Comput. Math. Appl.*, 51:1283–1296, 2006.

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

[61] Robert Schaback. *Multivariate Interpolation and Approximation by Translates of a Basis Function*. World Scientific Publishing Co., Inc, 1995.

[62] A. Soroushian and J. Farjoodi. âĂIJa unified starting procedure for the houbolt method. *Communications in Numerical Methods in Engineering*, 24(1):1–13, 2008.

[63] Lloyd N Trefethen and David Bau III. *Numerical linear algebra*, volume 50. Siam, 1997.

[64] H. Wendland. *Scattered Data Approximation*. Cambridge University Press, 2005.

[65] J. Wertz, E. J. Kansa, and L. Ling. The role of the multiquadric shape parameters in solving elliptic partial differential equations. *Comput. Math. Appl.*, 51(8):1335–1348, 2006.

[66] T. Y. Wu. *A Study on Time-Discontinuous Galerkin Finite Element Method for Elastodynamic Problems*. PhD thesis, Chung Yuan Christian University, Chung Li, Taiwan, 2001.

[67] Z. M. Wu. Hermite–Birkhoff interpolation of scattered data by radial basis functions. *Approximation Theory Appl.*, 8/2:1–10, 1992.

[68] Z. M. Wu and R. Schaback. Local error estimates for radial basis function interpolation of scattered data. *IMA Journal of Numerical Analysis*, 13:13–27, 1993.

[69] Guangming Yao, Joseph Kolibal, and C. S. Chen. A localized approach for the method of approximate particular solutions. *Comput. Math. Appl.*, 61(9):2376–2387, May 2011.

[70] J. Yoon. Spectral approximation orders of radial basis function interpolation on the Sobolev space. *SIAM J. Math. Anal.*, 33(4):946–958 (electronic), 2001.

[71] D. L. Young, K. H. Chen, and C. W. Lee. Novel meshless method for solving the potential problems with arbitrary domain. *Journal of Computational Physics*, 209(1):290–321, 2005.

[72] D. L. Young, M. H. Gu, and Fan C. M. The time-marching method of fundamental solutions for wave equations. *Engineering Analysis with Boundary Elements*, 33(12):1411–1425, 2008.

[73] D.L. Young, C. M. Fan, S. P. Hu, and S. N. Atluri. The eulerianâĂŞlagrangian method of fundamental solutions for two-dimensional unsteady burgersâĂŹ equations. *Engineering Analysis with Boundary Elements*, 32(5):395 – 412, 2008.