UNLV Retrospective Theses & Dissertations

1-1-2006

Developing meshless methods for partial differential equations

Arthur Jonathan Lee
University of Nevada, Las Vegas

Follow this and additional works at: https://digitalscholarship.unlv.edu/rtds

Repository Citation

Lee, Arthur Jonathan, "Developing meshless methods for partial differential equations" (2006). *UNLV Retrospective Theses & Dissertations*. 1958.
http://dx.doi.org/10.25669/ah6o-q1e3

This Thesis is protected by copyright and/or related rights. It has been brought to you by Digital Scholarship@UNLV with permission from the rights-holder(s). You are free to use this Thesis in any way that is permitted by the copyright and related rights legislation that applies to your use. For other uses you need to obtain permission from the rights-holder(s) directly, unless additional rights are indicated by a Creative Commons license in the record and/or on the work itself.

This Thesis has been accepted for inclusion in UNLV Retrospective Theses & Dissertations by an authorized administrator of Digital Scholarship@UNLV. For more information, please contact digitalscholarship@unlv.edu.

DEVELOPING MESHLESS METHODS FOR PARTIAL DIFFERENTIAL EQUATIONS

by

Arthur Jonathan Lee

Bachelor of Science, Mathematics
University of Nevada, Las Vegas
2004

A thesis submitted in partial fulfillment
of the requirements for the

Master of Science Degree in Mathematical Sciences
Mathematical Sciences Department
College of Sciences

Graduate College
University of Nevada, Las Vegas
May 2006

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

UMI Number: 1436771

INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

UMI®

UMI Microform 1436771

Copyright 2006 by ProQuest Information and Learning Company.

All rights reserved. This microform edition is protected against unauthorized copying under Title 17, United States Code.

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

UNLV
UNIVERSITY OF NEVADA LAS VEGAS

Thesis Approval
The Graduate College
University of Nevada, Las Vegas

April 14, 2006

The Thesis prepared by

Arthur Lee

Entitled

Developing Meshless Methods for Partial Differential Equations

is approved in partial fulfillment of the requirements for the degree of

MS in Mathematics

Examination Committee Chair

Dean of the Graduate College

Examination Committee Member

Examination Committee Member

Graduate College Faculty Representative

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

ABSTRACT

Developing Meshless Methods for Partial Differential Equations

by

Arthur Jonathan Lee

Dr. Jichun Li, Examination Committee Chair
Assistant Professor of Mathematics
University of Nevada, Las Vegas

In the past, the world of numerical solutions for Partial Differential Equations has been dominated by Finite Element Method, Finite Difference Method, and Boundary Element Method. These three methods all revolve around using a mesh or grid to solve their problems. This complicates problems with irregular boundaries and domains.

In this thesis, we develop methods for solving partial differential equations using Radial Basis Functions. This method is meshless, easy to understand, and even easier to implement.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

TABLE OF CONTENTS

ABSTRACT iii

LIST OF FIGURES v

ACKNOWLEDGEMENTS vi

CHAPTER 1 INTRODUCTION 1

CHAPTER 2 RADIAL BASIS FUNCTION 3

CHAPTER 3 INTERPOLATION AND APPROXIMATION 5

CHAPTER 4 ELLIPTIC PROBLEM 13

CHAPTER 5 IRREGULAR DOMAIN 19

CHAPTER 6 PARABOLIC PROBLEM 23

CHAPTER 7 HYPERBOLIC PROBLEM 26

CHAPTER 8 SECOND ORDER HYPERBOLIC PROBLEM 30

CHAPTER 9 DOMAIN SPLITTING TECHNIQUE 34

CHAPTER 10 CONCLUSION 39

APPENDIX SAMPLE PROGRAM 40

REFERENCES 43

VITA 45

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

LIST OF FIGURES

Figure 2 3-D graph of f(x,y) 6
Figure 3 Interpolation points........ 7
Figure 4 3-D Mapping of interpolation points........ 7
Figure 5 Interpolated function and error........ 9
Figure 6 144 interpolation points........ 10
Figure 7 Error when random points were used........ 11
Figure 9 3-D graph of u(x,y)........ 13
Figure 10 140 interpolation points........ 14
Figure 11 Error when gridded points were used........ 15
Figure 12 140 random interpolation points........ 16
Figure 13 Error when random points were used........ 17
Figure 15 3-D graph of u(x,y)........ 20
Figure 17 140 interpolation points........ 20
Figure 18 Error for irregular domain........ 22
Figure 19 3-D graph for u(x,y,t)........ 24
Figure 20 Max error verses time graph........ 25
Figure 21 3-D graph for u(x,y,t)........ 27
Figure 22 Interpolation points for hyperbolic problem........ 28
Figure 23 Max error verses time graph........ 29
Figure 24 3-D graph for u(x,y,t)........ 31
Figure 25 Max error verses time graph........ 33
Figure 26 3-D graph of u(x,y,t)........ 35
Figure 27 Interpolation points for the two domains........ 36
Figure 28 Error for domain splitting........ 37

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

ACKNOWLEDGEMENTS

I give my sincerest gratitude to Dr. Jichun Li, my thesis advisor. It is through his direction and support that has allowed me to finish my thesis. It has been his guiding light that has led me through the seas of academic confusion. Thank you for always being there to answer my questions.

I would also like to thank Dr. C. S. Chen for introducing me to meshless methods. It was his passion to this field that has inspired me in my research. He has taught me so much in the year that I had studied as his student. Thank you for your belief in me.

Also I would like to thank the UNLV Mathematical Sciences Department, the UNLV Graduate College, and my thesis committee members Dr. Douglas Burke, Dr. Zhonghai Ding, and Dr. Evangelos Yfantis.

Above all I would also like to thank Alicia Baker, Uchiha Itachi, and Edward Elric. Through the years of my academic journey, they have been my anchors. They have been my rock through the hard times and rough patches. It is through their faith and support that I have soared this high in my scholastic voyage. Lastly, thank you to Nara Shikamaru. I aspire one day to be even half the man you are.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER 1

INTRODUCTION

Partial differential equations (PDEs) exist in every science and engineering disciplinary. For example, we can find Maxwell's equations in electromagnetics; Navier-Stokes equations in fluid dynamics; and Richards' equations in unsaturated flow problems et al. For some simple model PDEs with simple geometry domains, we can find the exact solutions (i.e., the analytical solutions). However, for more complicated PDEs with complex geometry domains (which are very common for practical problems), finding the exact solutions is almost impossible. Hence looking for approximate solutions becomes very important and helpful. With the advancement of modern computer technology, finding approximate solutions (i.e., the numerical solutions) for all kinds of PDEs is possible.

In the past several decades, the method of choice for numerical solutions of PDEs in the world of science and engineering has been mainly restricted to the finite element methods (FEMs), the finite difference methods (FDMs), and the boundary element methods (BEMs). FDMs usually apply to regular shaped domains. FEMs and BEMs are good choices for complex geometry problems, but the meshing (i.e., creating a grid to be laid over the domain of the problem) is a very time-consuming process. Furthermore, the implementation of FEMs and BEMs is very complicated and it takes lots training time for people to grasp the techniques.

In recent years, there has been an increasing interest in developing the meshless or meshfree methods. Most meshfree methods [see, e.g., Atluri and Shen 2002, Belytschko et al

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

1996, Duarte and Oden 1996] are still based on finite element methods, hence it is still quite complicated. In 1990, Kansa [Kansa 1990] introduced a collocation method using radial basis functions (RBFs) for solving PDEs. A vital advantage of this meshless method over its predecessors is the ability to use amorphous nodes that neither need to be in a certain shape nor a certain pattern. Simply put, it provides flexibility to the input data that was unheard of prior to its discovery. Furthermore, since the nodes need not have structure, the level of complexity between using a perfectly rectangular domain and an abnormal amoeba like domain, for instance, would be the same. Additionally, because the formulation of 2-D and 3-D problems is very similar, these methods are very easy to learn and code. Since there is no meshing required, a few hundred nodes in the meshless method would be quite comparable to the thousands of nodes required for those meshing methods such as FEMs and BEMs. Seeing as the number of nodes has an exponential effect on the number of calculations needed, the meshless method is very computionally cost effective [Chen 2004]. In all, due to its simple implementation but with reasonable accuracy [Zerroukat, Power and Chen 1998, Li, Cheng and Chen 2003, Cheng et al 2003] this type meshless method becomes a very popular technique for solving different problems [see, e.g., Fasshauer 1999, Wong et al 1999, Li 2004].

In this thesis, we study this meshless method and implement it to various problems in a systematic way.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER 2

RADIAL BASIS FUNCTION

Before we begin with our discussion of meshless methods, we must first begin with RBFs. Introduced by R. L. Hardy [2] in 1968, RBFs were first used for geophysical surface-fitting. It was used to approximate the topography of a landscape from a set of known points and elevations. A major advantage with using RBFs was that the points on the grid did not need to be uniform in anyway. A random scattering of data points could be used just as easily as a uniform grid.

We define a radial basis function in two dimensions as the following:

$$\begin{gathered} \varphi : R^2 \rightarrow R \\ \varphi(x,y) = f(\|(x,y) - (x_i,y_i)\|) \end{gathered} \tag{1}$$

In our equation, (x_i,y_i) is simply a fixed point in which our radial basis function is associated with. From now on, we will denote $\|(x,y) - (x_i,y_i)\|$, the Euclidean norm, or the distance between point (x,y) and point (x_i,y_i), as the following:

$$r = \|(x,y) - (x_i,y_i)\| \tag{2}$$

Finally, $f(r)$ is simply a function such as r^3. For three dimensions, all that is needed is a change in r,the Euclidean norm, to three dimensions as such:

$$r = \|(x,y,z) - (x_i,y_i,z_i)\|$$

A list of radial basis functions are provided in Table 1.

The c parameter in the multiquadric and inverse multiquadric functions is a shape parameter represented as a positive real number. It has to be chosen for different problems to increase accuracy.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Name	$\varphi(r)$	Max. Dimensions
Polyharmonic (thin-plate) spline	$r^2 \log(r)$	∞
Polyharmonic spline	r^3	∞
Multiquadric	$\sqrt{c + r^2}$	∞
Inverse Multiquadric	$(c + r^2)^{-1/2}$	∞
Gaussian	e^{-r^2}	∞
Wendland's C^2 function	$(1 - r)_+^4(1 + 4r)$	3

Table 1. List of possible radial basis functions

A way of thinking about RBFs is that they are an enhanced metric that describes the distances between points in a way that is more suitable with PDEs.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER 3

INTERPOLATION AND APPROXIMATION

Next, we will see how RBFs work by interpolating a known function f from a set of n data points. These data points will be known as interpolation points. We will approximate f by creating $\hat{f}$, a linear combination of RBFs. We will have n RBFs corresponding directly to the n interpolation points we are given. Though various different RBFs can be used, for simplicity, we will use the Polyharmonic splines.

$$\begin{aligned} f(x,y) &= \hat{f}(x,y) \\ \hat{f}(x,y) &= \sum_{i=1}^{n} c_i \varphi_i(x,y) \\ \varphi_i(x,y) &= \varphi_i(r) \\ r &= \sqrt{(x-x_i)^2 + (y-y_i)^2} \end{aligned} \qquad 4$$

In $\hat{f}$, we have n unknowns $\{c_i\}$, that are the coefficients of our RBFs. To solve for these unknowns, we simply input the n interpolations points and their corresponding known values of f. In doing so, we get n equations of n unknowns. We get the following:

$$A\bar{c} = \bar{f} \qquad 5$$

A is the nxn matrix that corresponds our n unknown coefficients to our n equations. $\bar{c}$ is the vector of our unknowns. $\{c_i\}$. $\bar{f}$ is the vector of corresponding function values. As long as the matrix A is not singular, our unknowns coefficients are uniquely solvable and can be solved in the following way:

$$\bar{c} = A^{-1}\bar{f} \qquad 6$$

Since solving any inverse over 3x3 becomes quite troublesome and tedious, this is where the computer comes to save the day and solves for the coefficients in a fraction of the time.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

We will now use this method to reconstruct a two dimensional surface shown in Figure 2.

$$f(x,y) = -xye^{-(x^2+y^2)} \qquad 7$$

$$\Omega = [-1,1] \times [-1,1]$$

To approximate this function, we shall use a variation of the Polyharmonic Spline shown in Table 1. We will use the following radial basis function:

$$\varphi(x,y) = r^9 \qquad 8$$

As for the interpolation points, we will first use a grid of 144 interpolation points. This makes the $\Delta x = .2$ between two adjacent interpolation points. Later, we will vary n and randomize the position of the input points to show their affects. The position of the interpolation points on the xy-plane is shown in Figure 3. The 3-D placement of these points on the function can be seen in Figure 4.

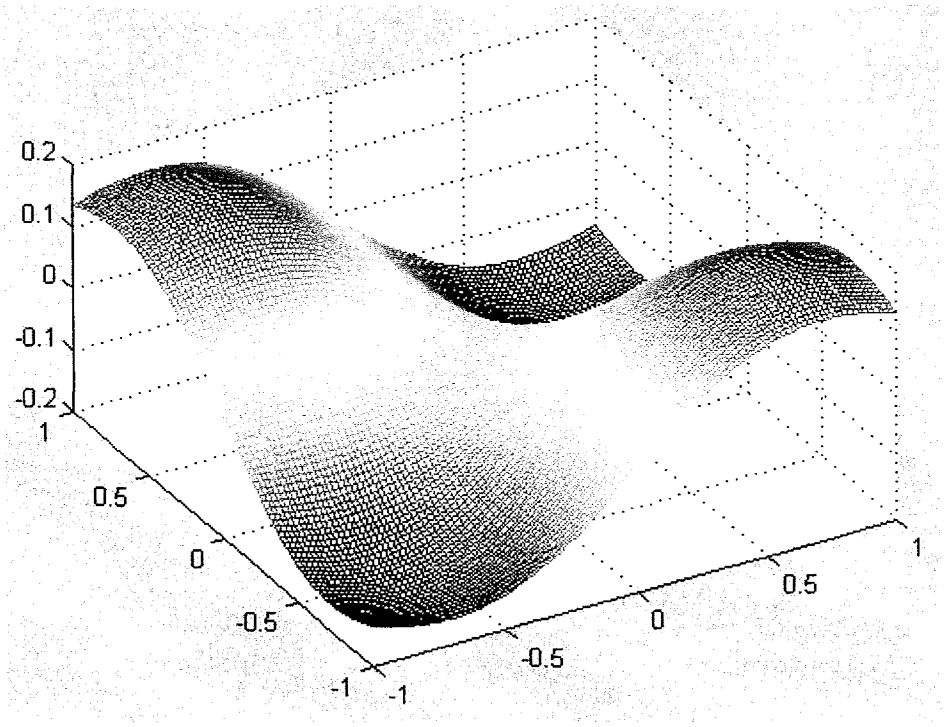


Figure 2. 3-D graph of $f(x,y) = -xye^{-(x^2+y^2)}$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

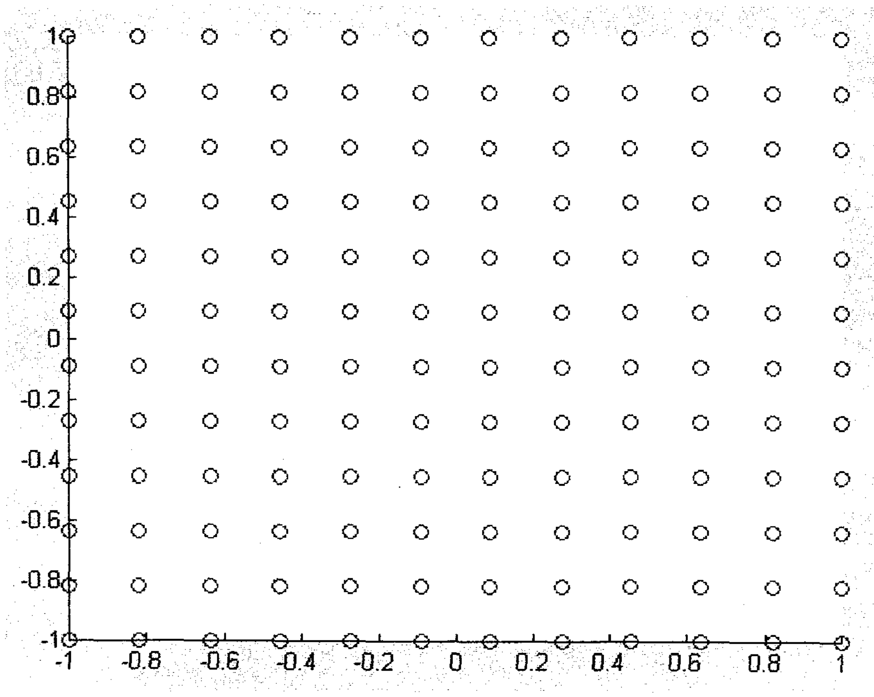


Figure 3. Position of 144 interpolation points on xy-plane

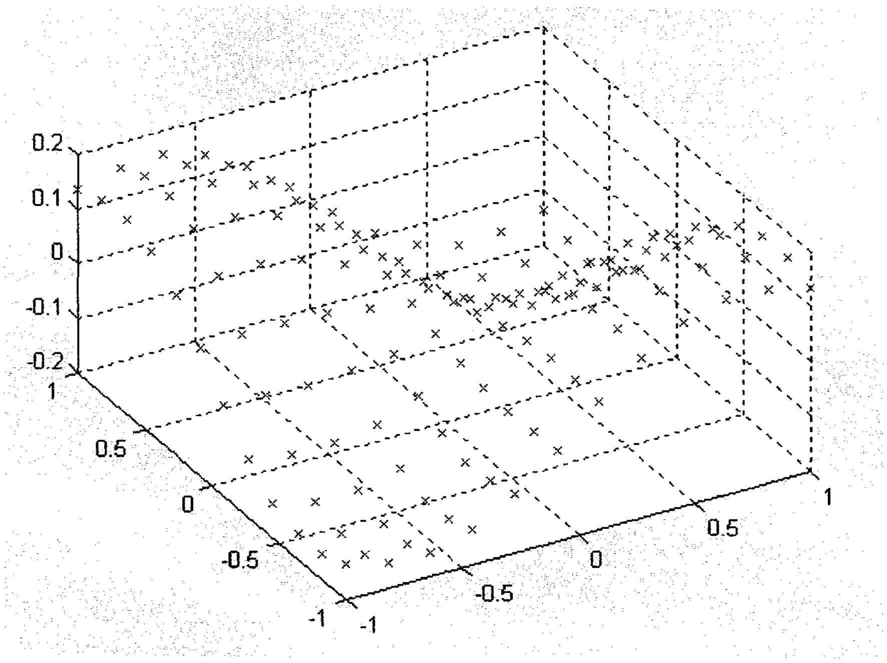


Figure 4. 3-D Mapping of interpolation points on function (7)

We now approximate *f* by formulating $\hat{f}$.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

$$f(x,y) = \hat{f}(x,y) \qquad 9$$

$$\hat{f}(x,y) = \sum_{i=1}^{144} c_i r^9$$

$$r(x,y) = \sqrt{(x-x_i)+(y-y_i)}$$

What we have is 144 unknown coefficients for our approximating function $\hat{f}$. To solve for this, we input our 144 interpolation points with corresponding f values. The first three points are shown below.

$$\sum_{i=1}^{144} c_i \sqrt{(-1-x_i)^2+(1-y_i)^2}^{\,9} = e^{-2} \qquad 10$$

$$\sum_{i=1}^{144} c_i \sqrt{(-.8-x_i)^2+(1-y_i)^2}^{\,9} = .8e^{-1.64}$$

$$\sum_{i=1}^{144} c_i \sqrt{(-.6-x_i)^2+(1-y_i)^2}^{\,9} = .6e^{-1.36}$$

All together, we have 144 equations with 144 unknowns. We can simplify this to the following.

$$A\bar{c} = \bar{f} \qquad 11$$

Solving for $\bar{c}$ in (11) we get.

$$\bar{c} = A^{-1}\bar{f} \qquad 12$$

Using the computer to solve for $\bar{c}$ and testing it for 10000 points we get the interpolated picture in Figure 5a. Since the error is very small, we can not see any significant difference between Figure 5a. and the actual function in Figure 2. It is for that reason that Figure 5b is also provided. This is the absolute error difference between the f and $\hat{f}$. The maximum error of the 10000 points shows to be 8.5673e-006. Another observation is that the maximum error seems to occur on the peripherals of the domain.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

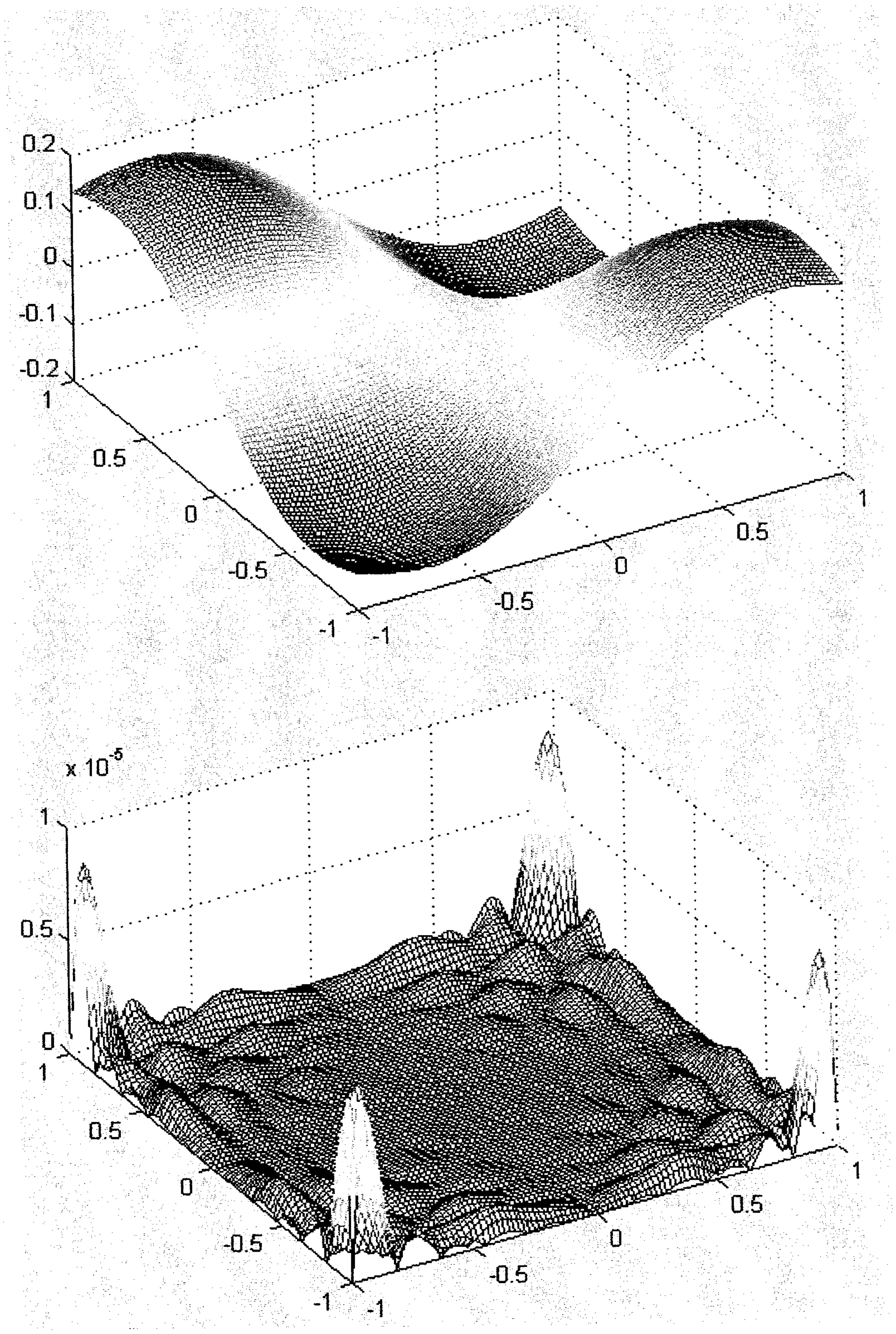


Figure 5. Interpolated function of $f(x,y) = -xye^{-(x^2+y^2)}$ on top. Error between f

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

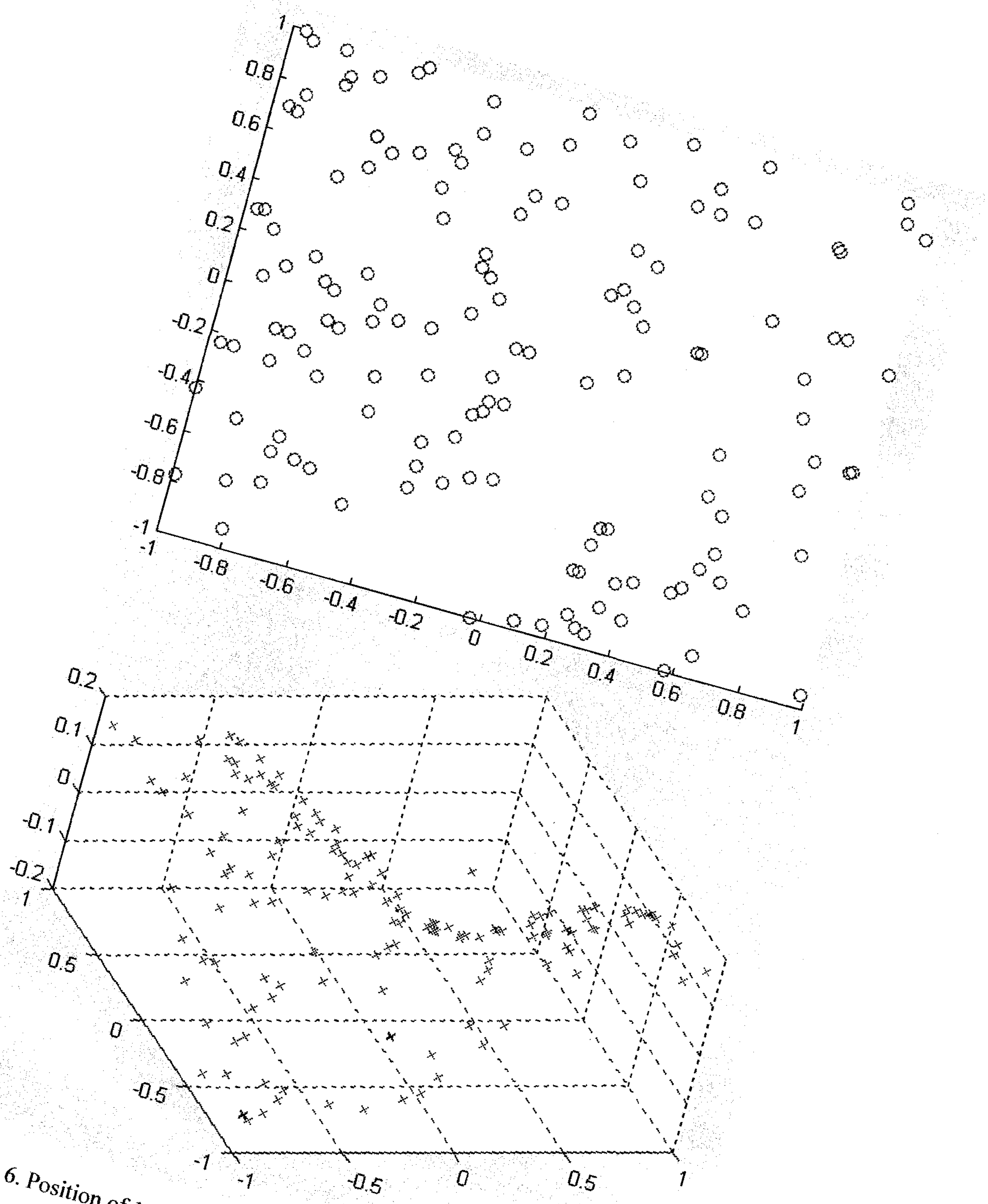


Figure 6. Position of 144 interpolation points on xy-plane is shown above. 3-D

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

From here, we shall now see if there is any difference between grid points and randomized points in our approximation. The position of the interpolation points on the xy-plane is shown in Figure 6a. The 3-D placement of these points on the function can be seen in Figure 6b.

Using the computer to solve for $\bar{c}$ and testing it for 10000 points. As said before, since the error is very small, we shall omit the picture of the approximated function and simply plot the absolute error as shown in Figure 7. The maximum error of the 10000 points shows to be 0.0014. This error is not bad considering it is less than a percent error. As seen with gridded points, the error is maximal on the peripherals of the domain. The higher error in this case is due to the uneven distribution of points. As seen in Figure 6, we have significantly large regions where there are no points around. It is because of this that leads to the higher error. Gridded points are more accurate in most cases than randomized points. However, this method is about the flexibility to use either.

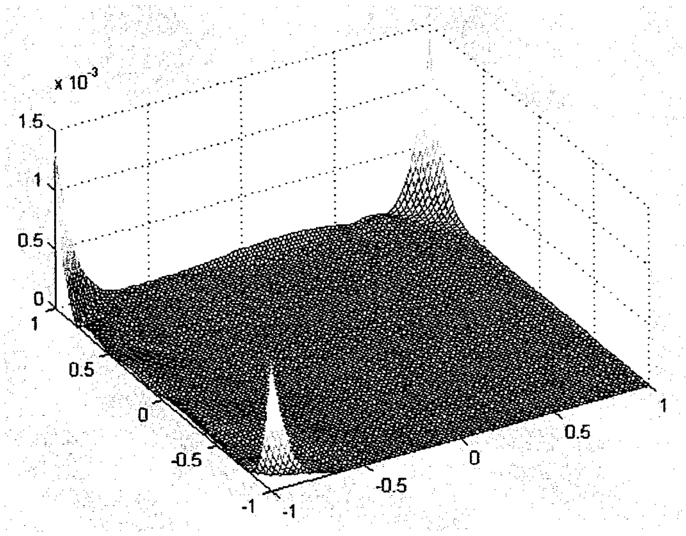


Figure 7. Error between f and $\hat{f}$ on bottom when

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Next, we analyze a couple different variations in number of points and between gridded and randomized points. In Table 8, we can see that the more points used, the more accurate the interpolation. However, it can also be seen that after a certain number, increasing the number of interpolation does less to improve the error as it did before.

n	Gridded	Δx	Randomized
64	4.94E-05	0.333	0.1512
100	2.43E-05	0.25	0.0634
144	8.57E-06	0.2	0.0014
169	5.64E-06	0.182	0.0011

Table 8. Gives the errors using various parameters in choosing interpolation points.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER 4

ELLIPTIC PROBLEM

We will now look into the Kansa's Method of solving PDEs. This will be done through an elliptic problem, such as the one in the following example (13).

$$
\begin{aligned}
f(x,y) &= \left[\frac{\partial}{\partial x}\left(2\frac{\partial u}{\partial x}\right)+\frac{\partial}{\partial y}\left(3\frac{\partial u}{\partial y}\right)\right] && \\
f(x,y) &= 2xye^{-(x^2+y^2)}(-6y+4x^2y-9+6y^2) && (x,y)\in\Omega \\
g(x,y) &= xye^{-(x^2+y^2)} && (x,y)\in\delta\Omega \\
\Omega &= [0,1]\times[0,1]
\end{aligned}
\qquad 13
$$

The exact solution for this problem is simply $u = xye^{-(x^2+y^2)}$. We can see the solution graphed in Figure 9. To solve this problem, we look at the right-hand side of 13a as simply an operator on the function u (14).

$$Lu = f(x,y) \qquad 14$$

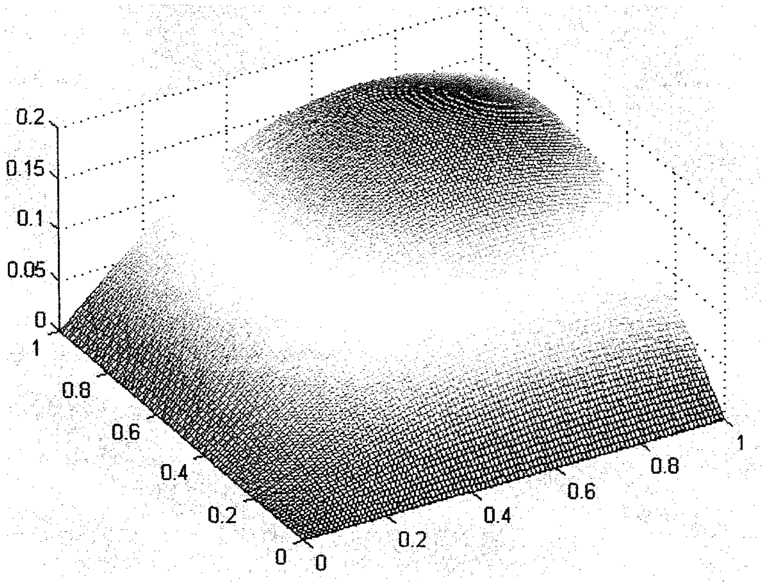


Figure 9. $u = xye^{-(x^2+y^2)}$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

We can now choose 40 points on the boundary of Ω and 100 points on the interior to be our interpolation points. We then approximate the solution for (13) to be (15).

$$u(x,y) = \hat{u}(x,y) \qquad 15$$

$$\hat{u}(x,y) = \sum_{i=1}^{140} c_i r^9$$

Let us assume the first 40 indices are the boundary points and the last 100 are the interior points. We can see in Figure 10. how the points are sparsed on the xy-plane. The circles represent the points on the boundary. The x's represent the points on the interior. We can see the $\Delta x = .091$.

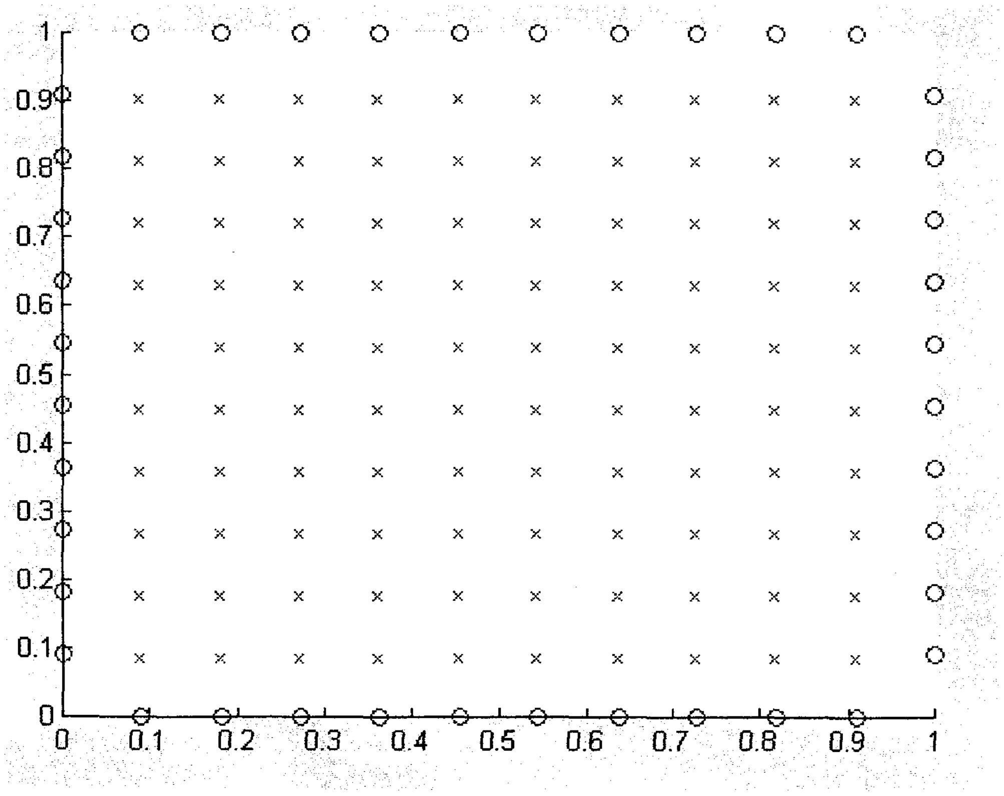


Figure 10. 140 Interpolation points on xy-plane.

To follow in the same approach we did with the approximation method, we have to come up with 140 equations. For the 40 points on the boundary, we know the solution of $u(x,y)$ to

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

be $g(x,y)$. So all we need to do is plug it in (16).

$$g(x,y) = \sum_{i=1}^{140} c_i r^9 \qquad 16$$

For the interior points, we use (14) and plug in what we know.

$$f(x,y) = \sum_{i=1}^{140} c_i L r^9 \qquad 17$$

Solving for the unknown coefficients as we did in the previous example, we get the error graph in Figure 11.

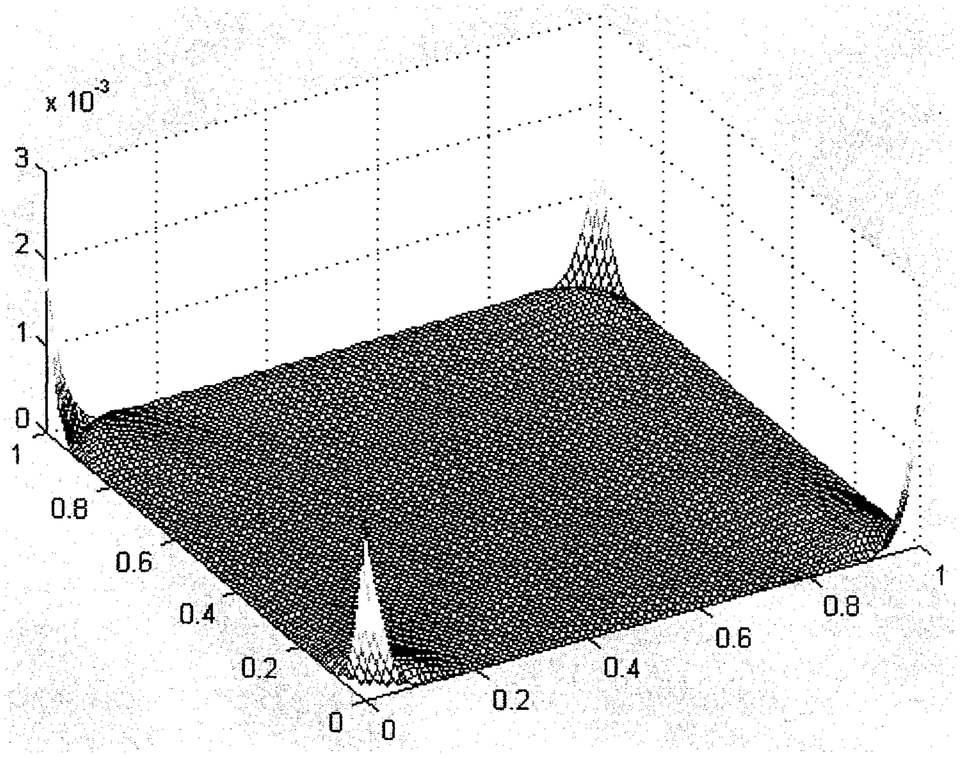


Figure 11. Difference between *u* and *û* when gridded points were used.

It can be seen that again, the highest errors occur at the sides of our domain. This error comes out to be .0021. Next, we will try randomizing the points. Let again choose 40 points on the boundary points of the problem and 100 points on the interior. We can see in Figure

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

12. how the points are sparsed on the xy-plane. The circles represent the points on the boundary. The x's represent the points on the interior.

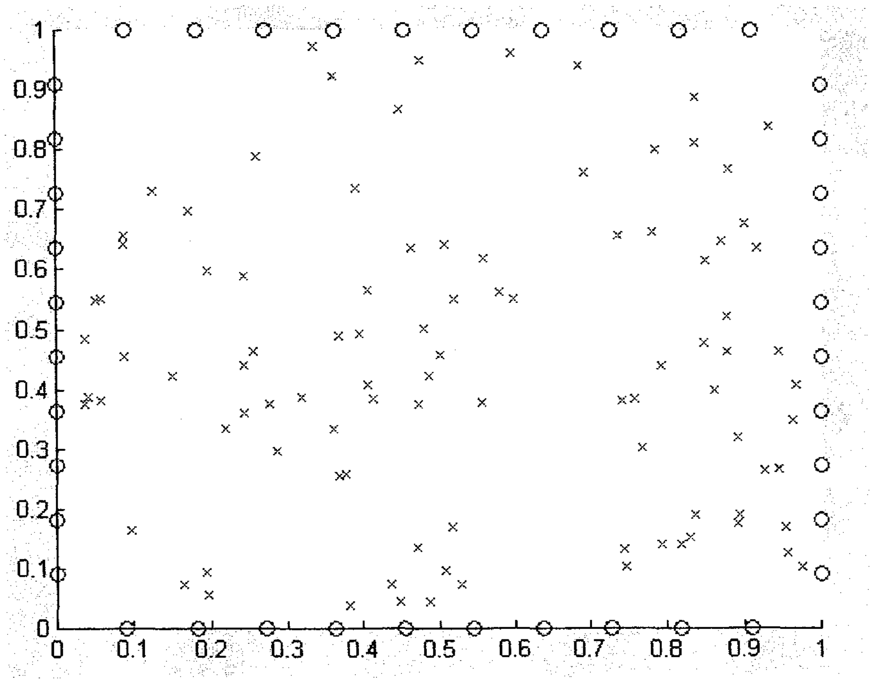


Figure 12. 140 random interpolation points on xy-plane.

Using (16) and (17) as our equations, we can solve for the unknown coefficients and get the error graph in Figure 13.

It can be seen that again, the highest errors occur at the sides of our domain. This error comes out to be .0019. One should notice that in our previous example of interpolation, the gridded points did better than the randomized points when the circumstances were the same. It seems that as the problems increase in difficulty, the positioning of the points becomes less of a factor when talking about the accuracy.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

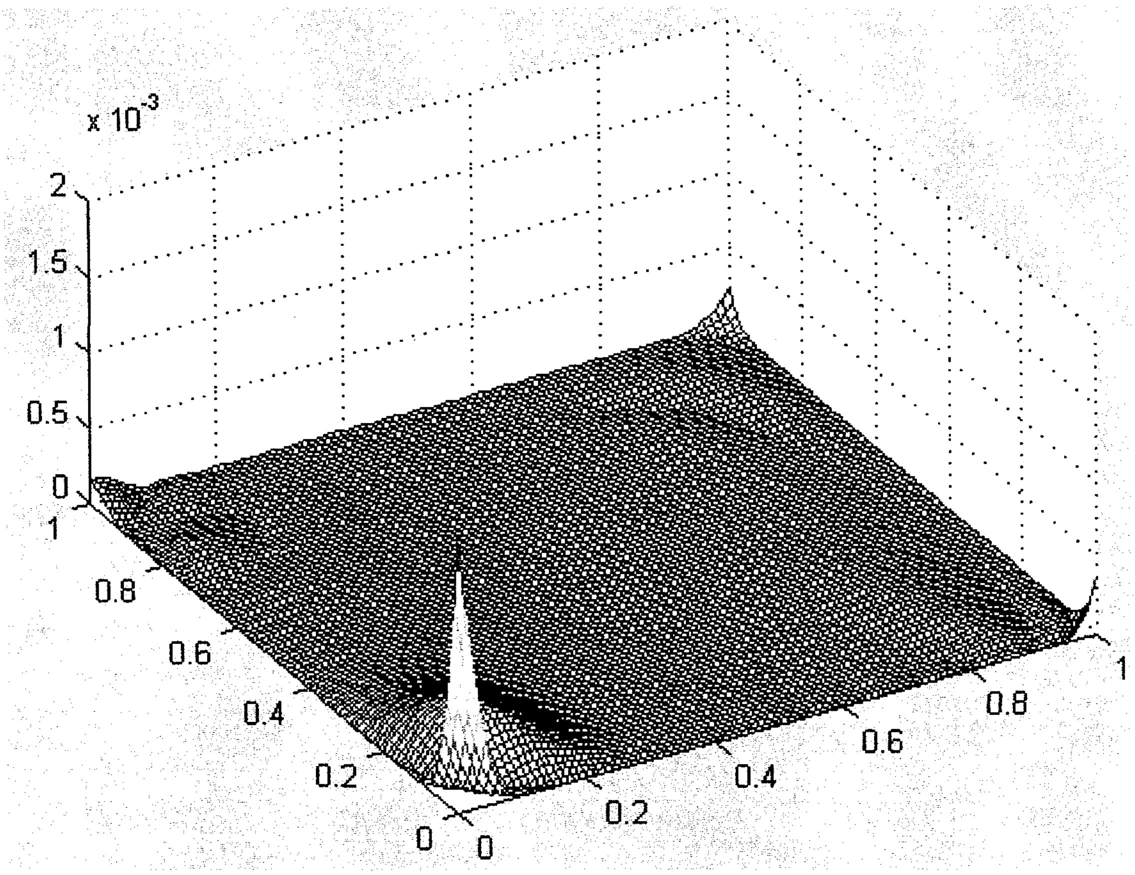


Figure 13. Difference between u and $\hat{u}$ when random points were used.

Next, we analyze a couple different variations in number of interior points and between gridded and randomized points. In Table 14, we can see that the more points used, the more accurate the solution is. Another interesting fact is that randomizing points does not have such a negative affect as it did with our previous example. On three of the tests, when the number of points used is fewer, randomizing points actually did slightly better than gridded. However, since the points are random, the error ranges a little as well. As long as the points decently cover the domain, it seems the error is pretty good.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

n	m	Gridded	Δx	Randomized
40	64	0.0053	0.333	0.0028
40	100	0.0021	0.25	0.0019
40	144	8.65E-004	0.2	0.0011
40	169	6.00E-004	0.182	3.54E-004

Table 14. Gives the errors using various parameters in choosing interpolation points.

We next look to see if changing the number of boundary points helps the error at all. In Table 15, we can see our results.

n	m	Gridded
20	100	0.0105
40	100	0.0021
60	100	3.41E-004
80	100	5.30E-005
100	100	6.85E-005

Table 16. Gives the errors using various parameters in choosing boundary points.

Here we can see that increasing the number of boundary points decreases the error as well. However, we see there is an optimal number of points, in which increasing beyond it simply reduces the accuracy.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER 5

IRREGULAR DOMAIN

Up until now, we've been dealing with rectangular domains and Dirichlet boundary conditions. We shall redo the elliptic problem with an odd looking domain and both Dirichlet and Neumann boundary conditions such as in the following example(18).

$$f(x,y) = \left[\frac{\partial}{\partial x}\left(2\frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(3\frac{\partial u}{\partial y}\right) \right] \qquad 18$$

$$f(x,y) = 2xye^{-(x^2+y^2)}(-6y + 4x^2y - 9 + 6y^2) \qquad (x,y) \in \Omega$$

$$g_1(x,y) = xye^{-(x^2+y^2)} \qquad (x,y) \in \delta\Omega 1$$

$$\frac{\partial u}{\partial n} = -ye^{-(x^2+y^2)} + 2x^2ye^{-(x^2+y^2)} \qquad (x,y) \in \delta\Omega_2$$

$$\delta\Omega_1 = \{(x,y) : x = r\cos\theta, y = r\sin\theta, r = e^{\sin\theta}\sin^2(2\theta) + e^{\cos\theta}\cos^2(2\theta), -\pi/2 \le \theta \le \pi/2\}$$

$$\delta\Omega_2 = \{(x,y) : x = 0, -1 \le y \le 1\}$$

The exact solution for this problem is simply $u = xye^{-(x^2+y^2)}$. We can see the solution graphed in Figure 16. To solve this problem, we look at the right-hand side of 18a as simply an operator on the function u (19).

$$Lu = f(x,y) \qquad 19$$

We can now choose 30 points on the boundary of Ω_1, 10 points on Ω_2 and 100 points on the interior to be our interpolation points. We then approximate the solution for (18) to be (20).

$$u(x,y) = \hat{u}(x,y) \qquad 20$$

$$\hat{u}(x,y) = \sum_{i=1}^{140} c_i r^9$$

Let us assume the first 30 are of boundary Ω_1, our next 10 are from Ω_2 and the last 100 are the interior points. We can see in Figure 17. how the points are sparsed on the xy-plane.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

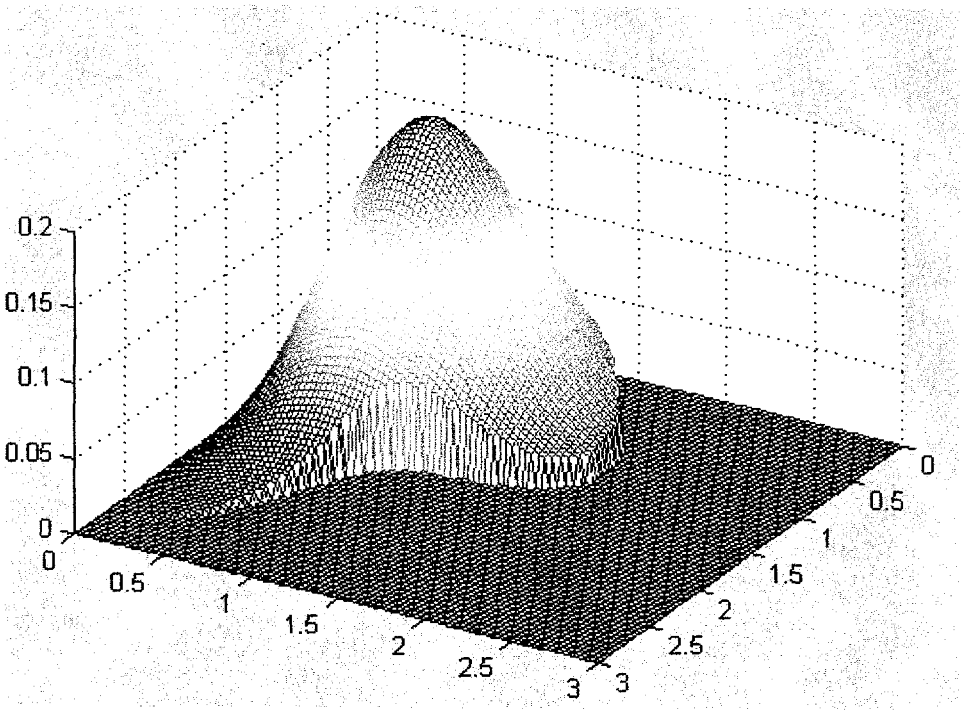


Figure 16. $u = xye^{-(x^2+y^2)}$

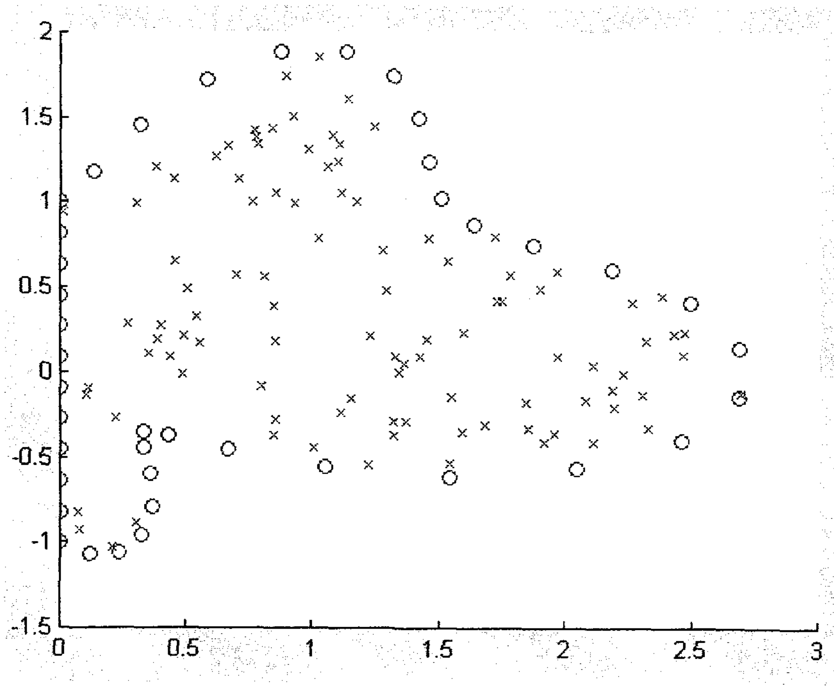


Figure 17. 140 Interpolation points on xy-plane.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

To follow in the same approach we did with the approximation method, we have to come up with 140 equations. For the 30 points on boundary Ω_1, we know the solution of $u(x,y)$ to be $g(x,y)$. So all we need to do is plug it in (21).

$$g(x,y) = \sum_{i=1}^{140} c_i r^9 \tag{21}$$

For the next 10 points on boundary Ω_2, we know what the directional derivative is. All we need to do is take the directional derivative of the RBF as well. Since the side is a straight line, it is simply the negative partial of x. So, we just plug in what we know to get (22).

$$\frac{\partial u}{\partial n} = \sum_{i=1}^{140} c_i \frac{\partial (r^9)}{\partial n} \tag{22}$$

For the interior points, we use (23) and plug in what we know.

$$f(x,y) = \sum_{i=1}^{140} c_i L r^9 \tag{23}$$

Using (21), (22), and (23), we have our 140 equations. Solving for the unknown coefficients, we get the error graph in Figure 18.

It can be seen that again, the highest errors occur at the sides of our domain. This error comes out to be 5.04e-004. This is amazing considering that the error is actually better than having a rectangular boundary with similar conditions. However, from now on, we will be using rectangular domains with Dirichlet boundary condition. It is obvious to see that changing the boundary does not change the implementation method. Changing the boundary condition from Dirichlet to a different boundary condition however, will complicate the program slightly. This section is simply to show the flexibility of using RBFs.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

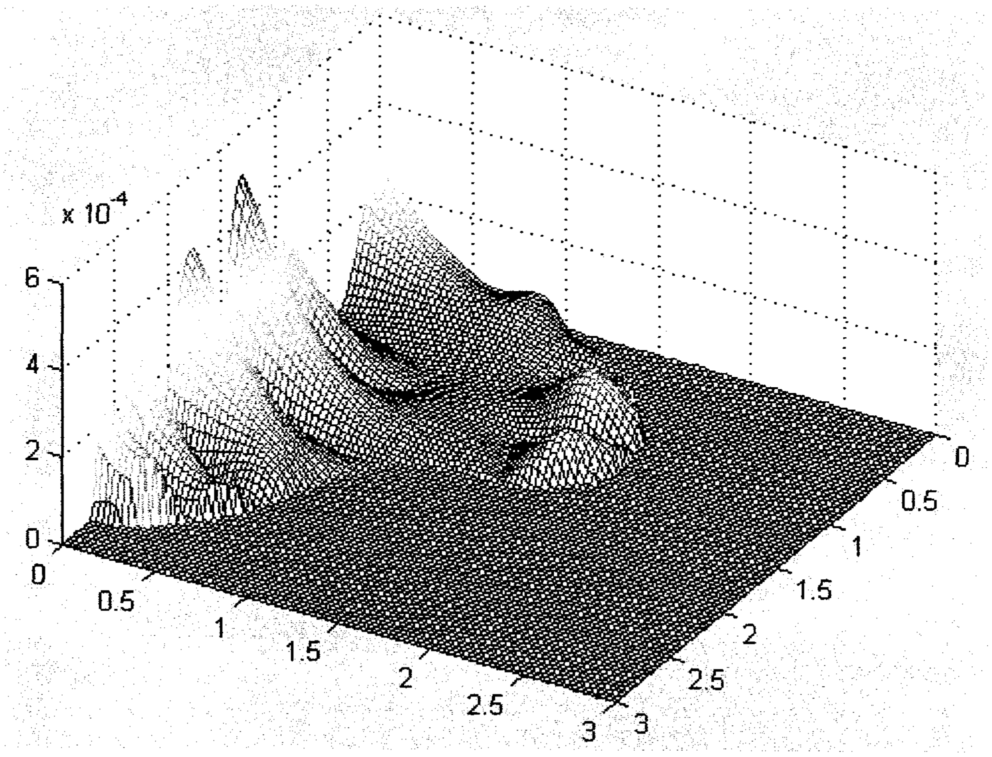


Figure 18. Difference between u and $\hat{u}$.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER 6

PARABOLIC PROBLEM

Next, we will try our hand at a parabolic problem. The main difference here is that time adds another dimension to our answer. We shall do a parabolic problem, such as in the following example(24).

$$\frac{\partial u}{\partial t} = f(x,y,t) + \left[\frac{\partial}{\partial x}\left(2\frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(3\frac{\partial u}{\partial y}\right) \right] \tag{24}$$

$$\begin{aligned} f(x,y,t) &= xye^{-(x^2+y^2)}(-30\cos t + 8x^2\cos t + 12y^2\cos t + \sin t) && (x,y) \in \Omega \\ g(x,y,t) &= xye^{-(x^2+y^2)}\cos t && (x,y) \in \delta\Omega \\ u(x,y,0) &= xye^{-(x^2+y^2)} && (x,y) \in \Omega \\ \Omega &= [-1,1]\times[-1,1] \end{aligned}$$

The exact solution for this problem is simply $u = xye^{-(x^2+y^2)}\cos t$. We can see the solution graphed in Figure 19 for various timesteps.

Unlike our previous problems, a major issue we have in this problem is that we have a time variable. More accurately, we have a partial time variable, $\frac{\partial u}{\partial t}$. Since $\frac{\partial u}{\partial t}$ is simply the change of u over time, we can approximate $\frac{\partial u}{\partial t}$ to the following:

$$\frac{\partial u}{\partial t} \approx \frac{u_{n+1} - u_n}{\Delta t} \approx f(x,y,t) + \left[\frac{\partial}{\partial x}\left(2\frac{\partial u_{n+1}}{\partial x}\right) + \frac{\partial}{\partial y}\left(3\frac{\partial u_{n+1}}{\partial y}\right) \right] \tag{25}$$

Where u_n is the equation u at time $n * \Delta t$. What we have done is introduced a time-stepping method for solving this equation. Though this gets rid of the partial with respect to time, it introduces a new parameter into the error: Δt. To minimize error, we will pick a small Δt, such as Δt =.005. In our next step, we will take the partials of the right hand side of (25) to get the following equation:

$$\frac{u_{n+1} - u_n}{\Delta t} \approx f(x,y,t) + 2u_{n+1,xx} + 3u_{n+1,yy} \tag{26}$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

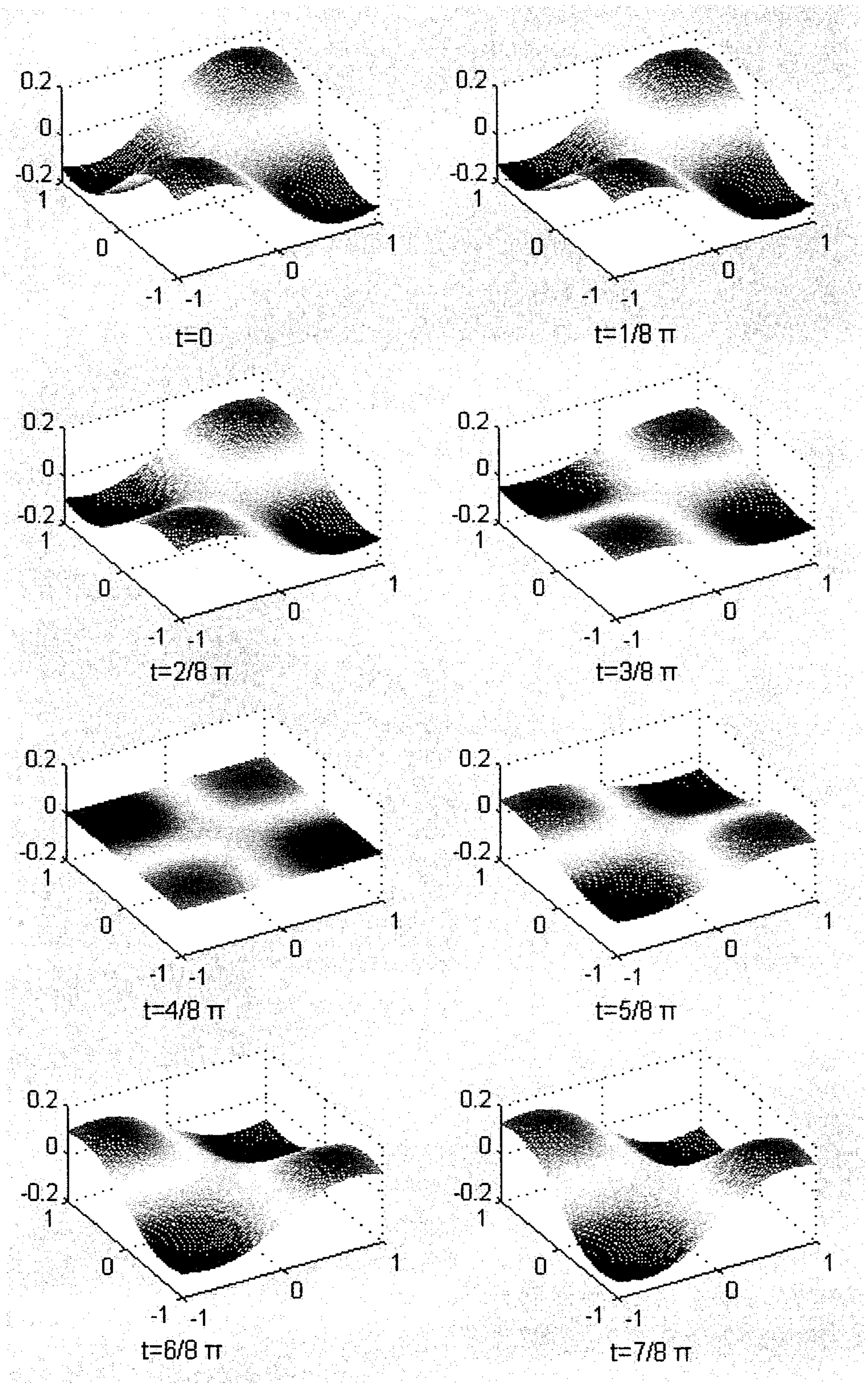


Figure 19. $u = xye^{-(x^2+y^2)}\cos t$ plotted for various t's.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

At this point, the equation still seems quite complicated. However, if we move all the terms with u_{n+1} in them to one side and take the rest to the other, we will get the following:

$$u_{n+1} - \Delta t[2u_{n+1,xx} + 3u_{n+1,yy}] \approx u_n + \Delta t f(x,y,t) \qquad 27$$

Though it may not seem like it, we have reduced this mess into a solvable problem.

$$\begin{aligned} L(u_{n+1}) &\approx u_n + \Delta t f(x,y,t) && (x,y) \in \Omega \\ g(x,y,t) &= xye^{-(x^2+y^2)}\cos t && (x,y,t) \in \delta\Omega \\ u(x,y,0) &= xye^{-(x^2+y^2)} && (x,y) \in \Omega \end{aligned} \qquad 28$$

In this, $L(u_{n+1})$ is simply an operator acting on u_{n+1}. Since $u(x,y,0)$ is u_0, and $f(x,y,t)$ is known, we can solve for u_1. From there, through a process much like induction, we can get to any time by stepping through all the previous steps. We shall use 40 boundary points, 100 interior points (giving us $\Delta x = 0.182$), and a time step of .005. Thus, to get to 5, we have to go through 1000 timesteps. We can see maximum error at each timestep in Figure 20.

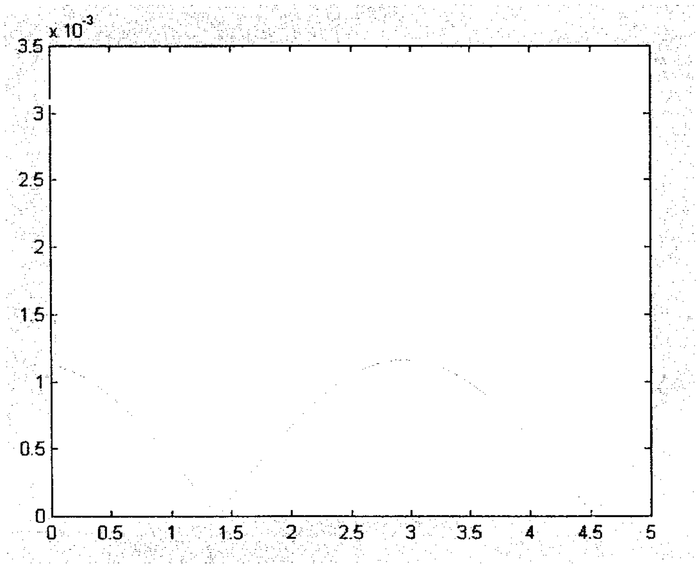


Figure 20. Max Error vs Time for time-stepping.

We can see that the error for this is periodic with respect to time.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER 7

HYPERBOLIC PROBLEM

Next, we will try our hand at a hyperbolic problem. The main difference here is that we have a once partial, rather the twice partial. Also, we will only have two sides of the boundary for our boundary conditions. We shall do a hyperbolic problem, such as in the following example(29).

$$
\begin{aligned}
f(x,y,t) &= \frac{\partial u}{\partial t} + 2\frac{\partial u}{\partial x} + 3\frac{\partial u}{\partial y} && && (29)\\
f(x,y,t) &= 6\cos(2x+y-t) && (x,y) \in \Omega\\
g(x,y,t) &= \sin(2x+y-t) && (x,y) \in \delta\Omega\\
u(x,y,0) &= \sin(2x+y) && (x,y) \in \Omega\\
\Omega &= [-1,1]\times[-1,1]\\
\delta\Omega &= \{(x,y) : (x = 0,-1 \le y \le 1) \cup (y = 0,-1 \le x \le 1)\}
\end{aligned}
$$

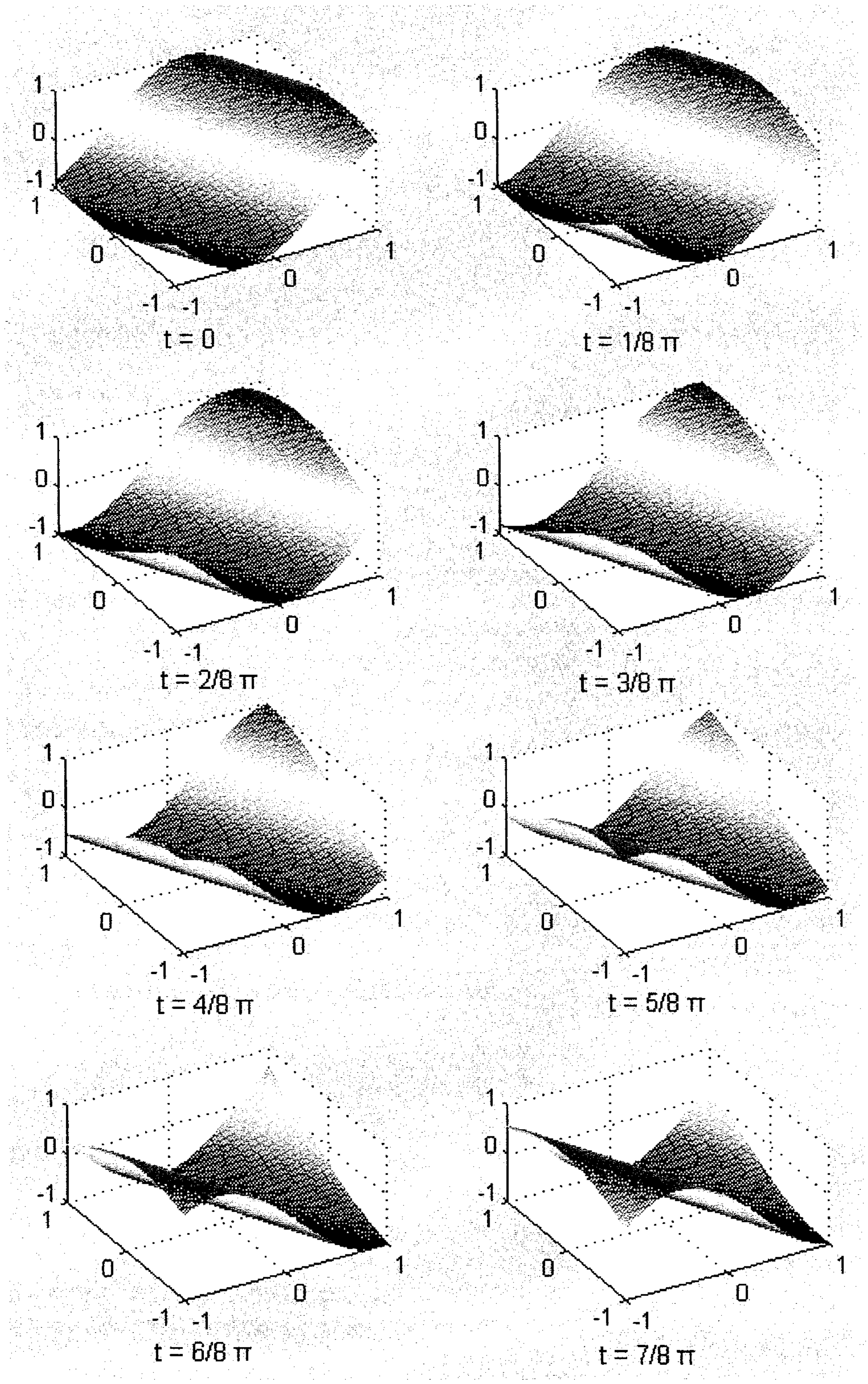
The exact solution for this problem is simply $u = \sin(2x+y-t)$. We can see the solution graphed in Figure 21 for various timesteps.

Just as with our previous problem, a major issue we have in this problem is that we have a time variable. So just as we did in the previous problem, we will approximate $\frac{\partial u}{\partial t}$. We can approximate $\frac{\partial u}{\partial t}$ to the following:

$$
\frac{\partial u}{\partial t} \approx \frac{u_{n+1} - u_n}{\Delta t} \approx f(x,y,t) - 2\frac{\partial u_{n+1}}{\partial x} - 3\frac{\partial u_{n+1}}{\partial y} \qquad (30)
$$

Where u_n is the equation u at time $n * \Delta t$. What we have done is introduced a time-stepping method for solving this equation. Though this gets rid of the partial with respect to time, it introduces a new parameter into the error: Δt. To minimize error, we will pick a small Δt, such as $\Delta t = .005$. At this point, the equation still seems quite complicated.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.



t = 0

t = 1/8 π

t = 2/8 π

t = 3/8 π

t = 4/8 π

t = 5/8 π

t = 6/8 π

t = 7/8 π

Figure 21. $u = \sin(2x + y - t)$ plotted for various t's.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

However, if we move all the terms with u_{n+1} in them to one side and take the rest to the other, we will get the following:

$$u_{n+1} + \Delta t[2u_{n+1,x} + 3u_{n+1,y}] \approx u_n + \Delta t f(x,y,t) \qquad 31$$

Though it may not seem like it, we have reduced this mess into a solvable problem.

$$\begin{aligned} L(u_{n+1}) &\approx u_n + \Delta t f(x,y,t) && (x,y) \in \Omega \\ g(x,y,t) &= \sin(2x + y - t) && (x,y,t) \in \delta\Omega \\ u(x,y,0) &= \sin(2x + y) && (x,y) \in \Omega \end{aligned} \qquad 32$$

In this, $L(u_{n+1})$ is simply an operator acting on u_{n+1}. Since $u(x,y,0)$ is u_0, and $f(x,y)$ is known, we can solve for u_1. From there, through a process much like induction, we can get to any time by stepping through all the previous steps. We shall use 21 boundary points, 144 interior points (giving us $\Delta x = 0.154$), and a time step of .005. As we can see in Figure 22, we can only put boundary points on the two sides we know the conditions for. The boundary points are in circles and the interior points are in x's. Doing the problem, we can see the maximum error at each timestep in Figure 23.

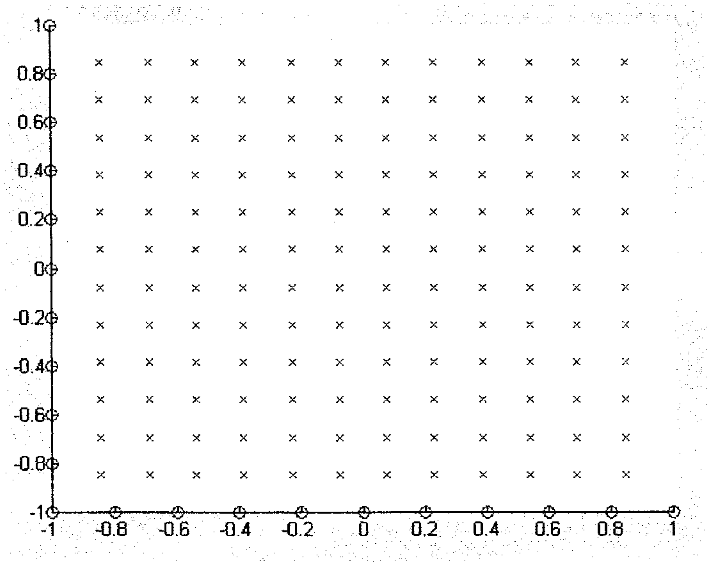


Figure 22. Interpolation points on xy-plane.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

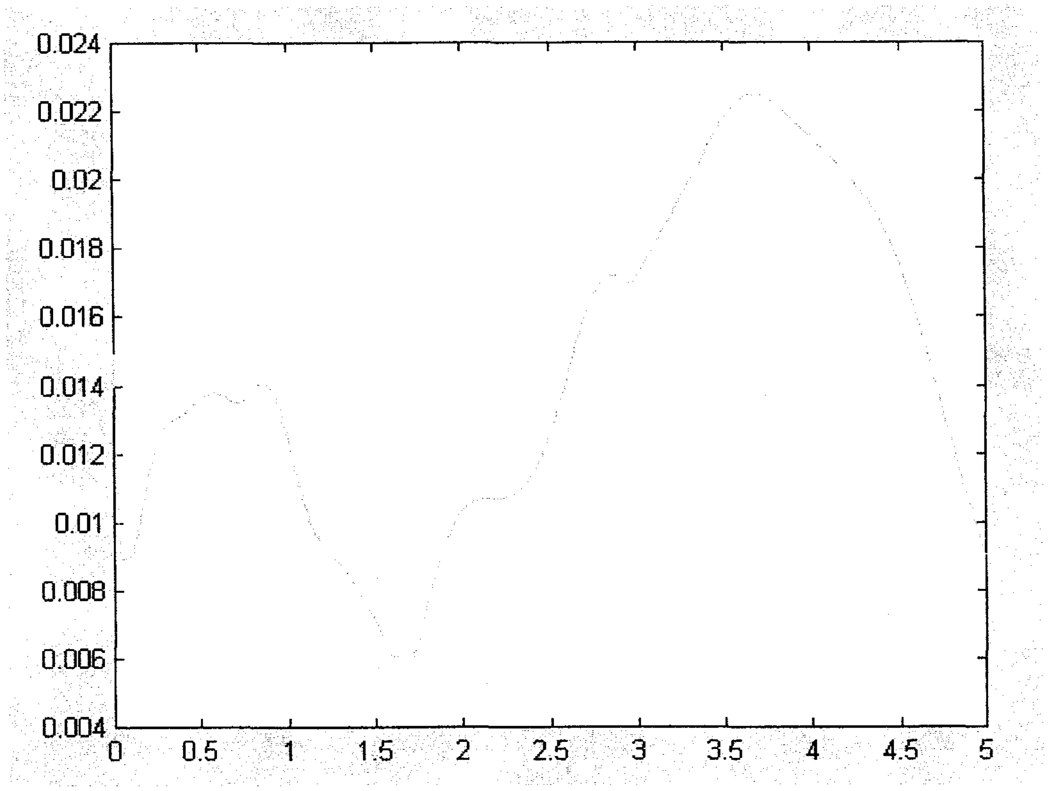


Figure 23. Max Error vs Time for time-stepping.

Since *u* is a sin function, the values range from -1 to 1. Having a maximum error through the 5 seconds as .022 is synonymous to a 2% error, which is very good.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER 8

SECOND ORDER HYPERBOLIC PROBLEM

Next, we will try our hand at a second ordered hyperbolic problem. The difference between this problem and the previous is that the partials are now twice. We shall do a second ordered hyperbolic problem, such as in the following example(29).

$$\frac{\partial^2 u}{\partial t^2} = f(x,y,t) + 2\frac{\partial^2 u}{\partial x^2} + 3\frac{\partial^2 u}{\partial y^2} \tag{33}$$

$$\begin{aligned}
f(x,y,t) &= 13\sin(2x+y-t) && (x,y) \in \Omega \\
g(x,y,t) &= \sin(2x+y-t) && (x,y) \in \delta\Omega \\
u(x,y,0) &= \sin(2x+y) && (x,y) \in \Omega \\
u_t(x,y,0) &= -\cos(2x+y) && (x,y) \in \Omega \\
\Omega &= [-1,1]\times[-1,1] \\
\delta\Omega &= \{(x,y) : (x=0,-1 \le y \le 1) \cup (y=0,-1 \le x \le 1)\}
\end{aligned}$$

The exact solution for this problem is simply $u = \sin(2x+y-t)$. We can see the solution graphed in Figure 24 for various timesteps.

Here, we have a second order time derivative. However, we can still approximate this. Let us take three time steps u_{n+1}, u_n, and u_{n-1}. We can then approximate $\frac{\partial u_{n+1}}{\partial t}$ and $\frac{\partial u_n}{\partial t}$:

$$\begin{aligned}
\frac{\partial u_{n+1}}{\partial t} &\approx \frac{u_{n+1}-u_n}{\Delta t} \\
\frac{\partial u_n}{\partial t} &\approx \frac{u_n - u_{n-1}}{\Delta t}
\end{aligned} \tag{34}$$

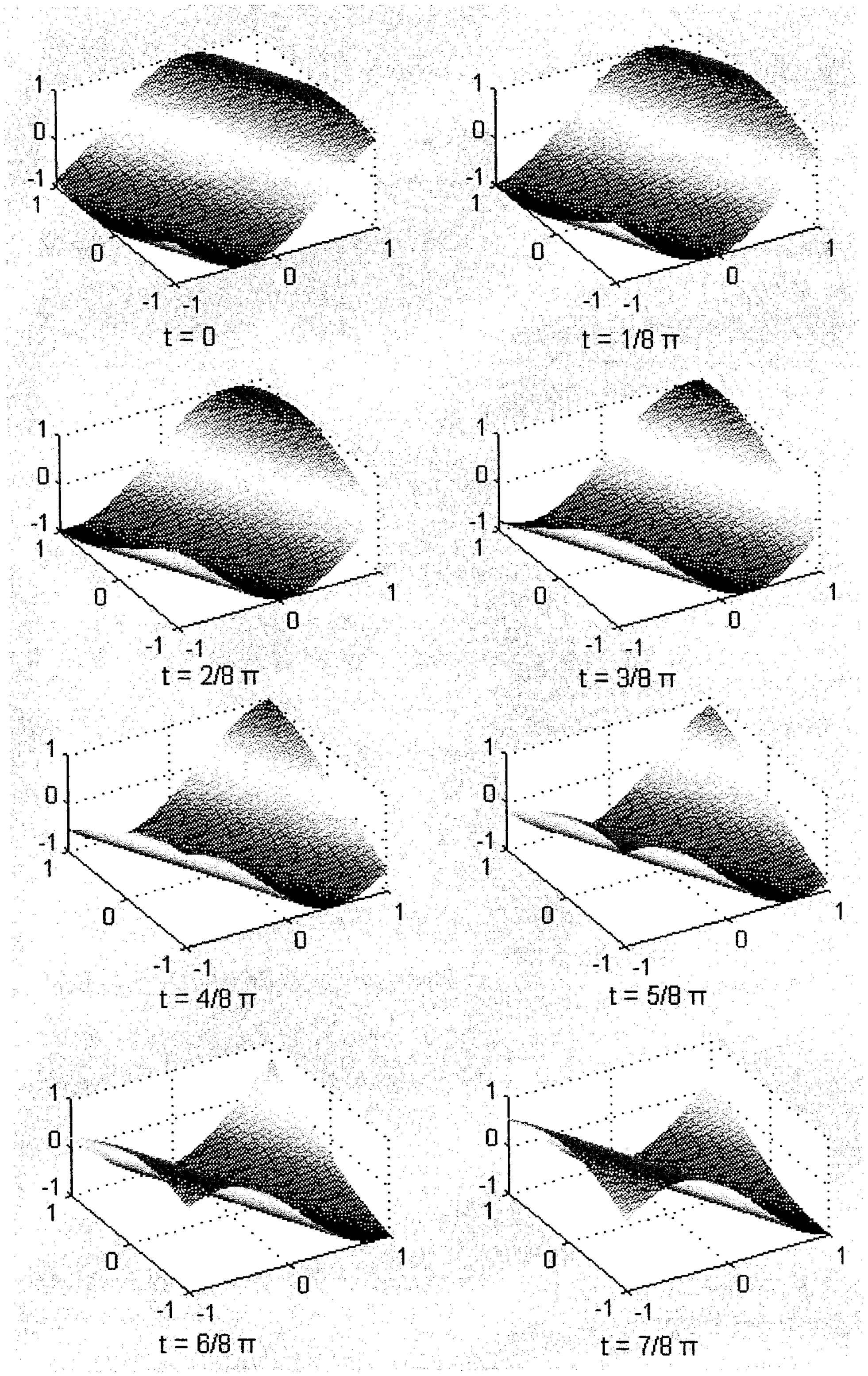
From there, we can approximate $\frac{\partial^2 u}{\partial t^2}$ in a similar manner:

$$\frac{\partial^2 u_{n+1}}{\partial t^2} \approx \frac{\frac{\delta u_{n+1}}{\delta t} - \frac{\delta u_n}{\delta t}}{\Delta t} = \frac{\frac{u_{n+1}-u_n}{\Delta t} - \frac{u_n - u_{n-1}}{\Delta t}}{\Delta t} = \frac{u_{n+1} - 2u_n + u_{n-1}}{\Delta t^2} \tag{35}$$

After we have approximated $\frac{\partial^2 u}{\partial t^2}$, we can simply plug that into (33).

$$\frac{\partial^2 u}{\partial t^2} \approx \frac{u_{n+1} - 2u_n + u_{n-1}}{\Delta t^2} \approx f(x,y,t) + 2\frac{\partial u_{n+1}}{\partial x} + 3\frac{\partial u_{n+1}}{\partial y} \tag{36}$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.



t = 0

t = 1/8 π

t = 2/8 π

t = 3/8 π

t = 4/8 π

t = 5/8 π

t = 6/8 π

t = 7/8 π

Figure 24. $u = \sin(2x + y - t)$ plotted for various t's.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Where u_n is the equation u at time $n * \Delta t$. What we have done is introduced a time-stepping method for solving this equation. Though this gets rid of the partial with respect to time, it introduces a new parameter into the error: Δt. To minimize error, we will pick a small Δt, such as $\Delta t = .005$. At this point, the equation still seems quite complicated. However, if we move all the terms with u_{n+1} in them to one side and take the rest to the other, we will get the following:

$$u_{n+1} + \Delta t^2[2u_{n+1,x} + 3u_{n+1,y}] \approx 2u_n - u_{n-1} + \Delta t^2 f(x,y,t) \qquad 37$$

As you can see, after we have two timesteps done, we can timestep to anytime. The problem is that we are given u_0 but we are not given u_1. However, we are given $u_{0,t}$. We can then get u_1 in the following manner:

$$\begin{aligned} \frac{\partial u_0}{\partial t} &\approx \frac{u_1 - u_0}{\Delta t} \\ u_1 &\approx u_0 - \Delta t * u_{0,t} \end{aligned} \qquad 37$$

By Though it may not seem like it, we have reduced this mess into a solvable problem.

$$\begin{aligned} L(u_{n+1}) &\approx 2u_n - u_{n-1} + \Delta t f(x,y,t) && (x,y) \in \Omega \\ g(x,y,t) &= \sin(2x + y - t) && (x,y,t) \in \delta\Omega \\ u_0(x,y) &= \sin(2x + y) && (x,y) \in \Omega \\ u_1(x,y) &= \sin(2x + y) - \Delta t * -\cos(2x + y) && (x,y) \in \Omega \end{aligned} \qquad 38$$

In this, $L(u_{n+1})$ is simply an operator acting on u_{n+1}. Since u_0,u_1 , and $f(x,y)$ are known, we can solve for u_2. From there, through a process much like induction, we can get to any time by stepping through all the previous steps. We shall use 40 boundary points, 100 interior points (giving us $\Delta x = 0.182$), and a time step of .005. Thus, to get to 25, we have to go through 5000 timesteps. One thing different is the way we scatter the points. As we can see in Figure 25, we can only put boundary points on the two sides we know the conditions for. The boundary points are in circles and the interior points are in x's. Doing the problem, we can see the maximum error at each timestep in Figure 23.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

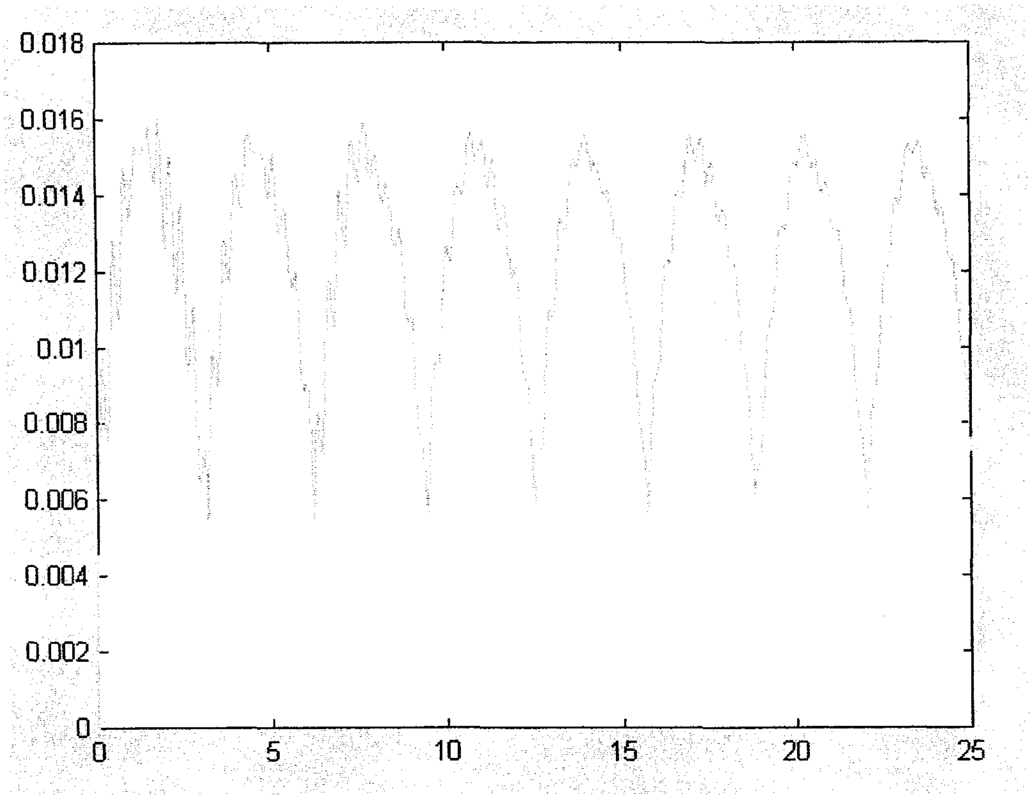


Figure 25. Max Error vs Time for time-stepping.

Since *u* is a sin function, the values range from -1 to 1. Having a maximum error through the 25 seconds as .016 is synonymous to a 2% error, which is very good.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER 9

DOMAIN SPLITTING TECHNIQUE

Sometimes while doing problems, some require a huge number of interpolation points. Let's say that number is n. Since we solve a system of n variables and n linear equations, when n becomes too great, we run into the problem that the program will take too long. If the algorithm we use to solve our matrix is Gaussian elimination, then our time complexity is $O(n^3)$. This means that the length of time to finish this algorithm is proportional to the cube of the number of points. Similarly, LU-decomposition is at a time complexity of $O(n^2)$. Thus, an option would be to split the problem into two separate problems that have less interpolation points each. Let us take the following example:

$$\begin{aligned} f(x,y) &= \left[\frac{\partial}{\partial x}\left(2\frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(3\frac{\partial u}{\partial y}\right)\right] \\ f(x,y) &= 2xye^{-(x^2+y^2)}(-6y+4x^2y-9+6y^2) & (x,y) \in \Omega \\ g(x,y) &= xye^{-(x^2+y^2)} & (x,y) \in \delta\Omega \\ \Omega &= [0,1]\times[0,1] \end{aligned} \tag{39}$$

The exact solution for this problem is simply $u = xye^{-(x^2+y^2)}\cos t$. We can see the solution graphed in Figure 26 for various timesteps.

Let us assume we have 4- boundary points and 324 interior data points in our domain. That is 18 by 18 in the interior. If this number of points were too great, we could split the domain into two smaller parts, with an overlapping section. We can then look at the two sections as two separate problems. So, let us split up the domain into two sections:

$$\begin{aligned} \Omega_1 &= [0,.6]\times[0,1] \\ \Omega_2 &= [.4,1]\times[0,1] \end{aligned} \tag{40}$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

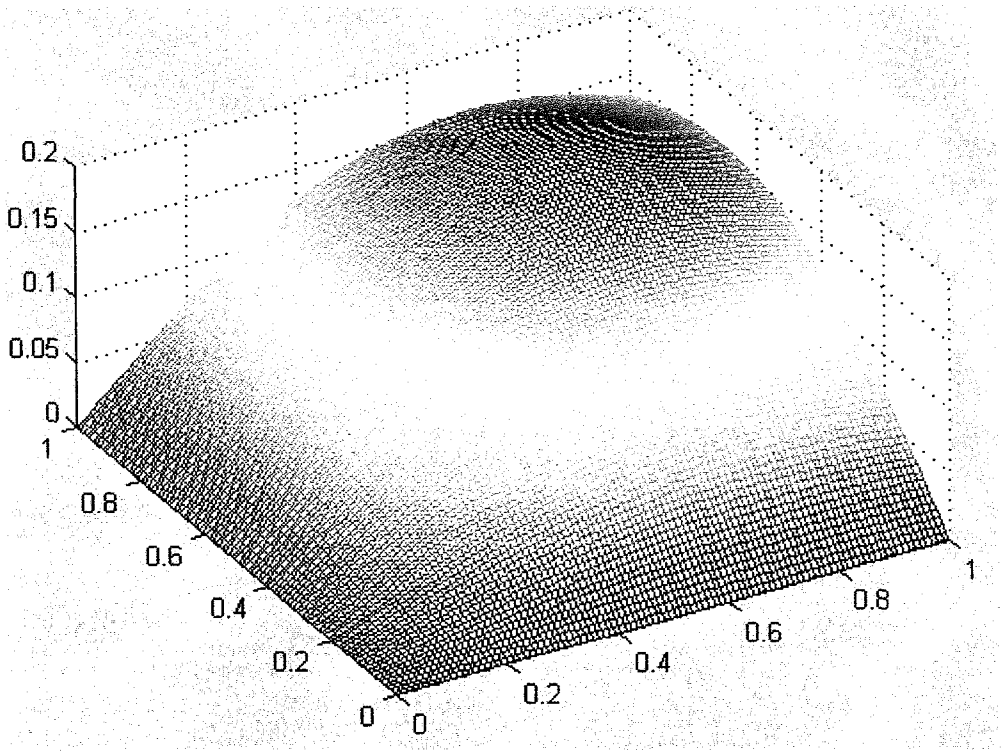


Figure 26. $u = xye^{-(x^2+y^2)}$

We now have two domains with 32 boundary points and 180 interior points. All the points are taken from the original problem except for the boundary points that split original domain. Let us call this section the following:

$$\begin{aligned} \delta\Omega_{n1} &= \{(x,y) : x = .4, 0 \le y \le 1\} \\ \delta\Omega_{n2} &= \{(x,y) : x = .6, 0 \le y \le 1\} \end{aligned} \tag{41}$$

We now have the following two problems:

$$\begin{aligned} L(a) &= f(x,y,t) && (x,y) \in \Omega_1 \\ g(x,y) &= xye^{-(x^2+y^2)} && (x,y) \in \delta\Omega_1/\delta\Omega_{n1} \\ g(x,y) &= b(x,y) && (x,y,t) \in \delta\Omega_{n1} \\ L(b) &= f(x,y,t) && (x,y) \in \Omega_2 \\ g(x,y) &= xye^{-(x^2+y^2)} && (x,y) \in \delta\Omega_2/\delta\Omega_{n2} \\ g(x,y) &= a(x,y) && (x,y,t) \in \delta\Omega_{n2} \end{aligned} \tag{42}$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

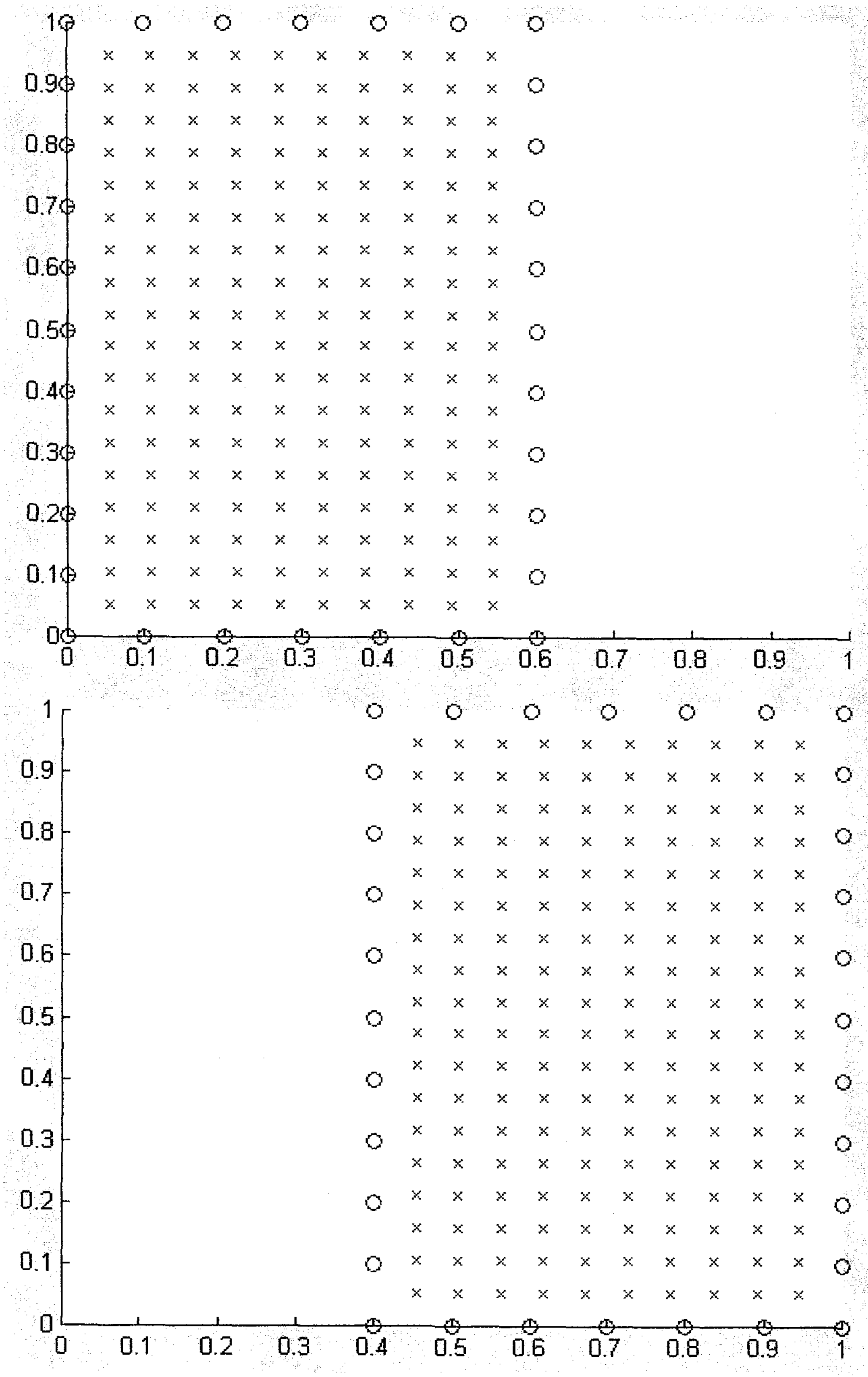


Figure 27. The top are the interpolation points for domain 1. The bottom are the

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

As you can see, the two problems feed off the answer of the other. So thus, what we can do is assume the value of on $\delta\Omega_{n1}$ In this, $L(u_{n+1})$ is zero. We then solve for $a(x,y)$. We then use that and solve for $b(x,y)$. We can then repeat those steps until the value of $a(x,y)$ and $b(x,y)$. Now, if we run this problem with 32 boundary points on each domain, 180 interior points, and iterate until the difference in the center region is below 1e-004, we get the error graph in Figure 28. We just use $a(x,y)$ or $b(x,y)$ to solve for their respective regions. As for the center, we can use either.

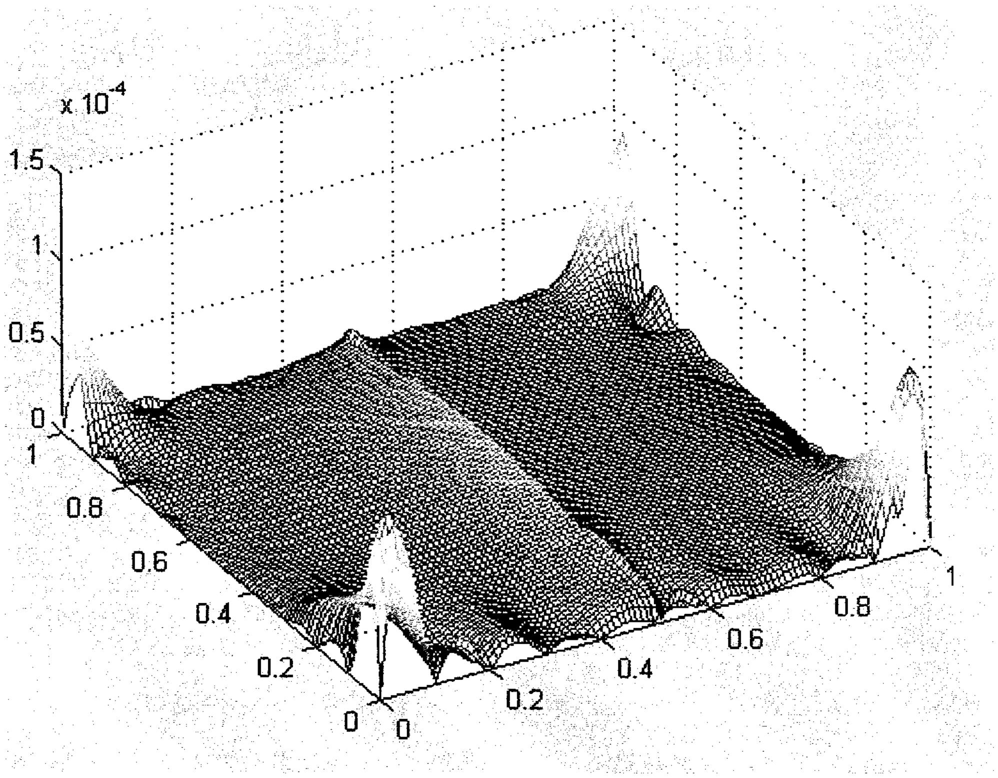


Figure 28. Error of a(x,y) and b(x,y) compared to the exact soluion.

The maximum difference in the center region was 3.365844e-005. The maximum error was 1.028746e-004, which happened on the boundaries. To do this, it took five iterations. If we were to run this problem without domain splitting, with 40 boundary points and 324

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

interior points, the maximum error would of been 6.136312e-005, which isn't even twice as better. Keep in mind that for such low numbers, it is possible to do both. However, when the number of points gets too large, domain splitting will the only option.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER 10

CONCLUSION

As we have gone through the various problems, we can see that alot of the various problems can be done using RBFs. We have gone through elliptic, parabolic, hyperbolic, and second-order hyperbolic. In all these cases, we can see that the algorithms used can be easily understood and programmed. Furthermore, the boundaries of the domains in these problems need not be presented in any special way. Because the algorithms only care about how far the points are away from each other and not how they are placed, this gives this method incredible flexibility. Also, when the number of points becomes so large that it will hinder the time to calculate the answer, a simple domain splitting technique can be used to reduce that number. So it seems that RBFs have massive potential in the field of PDEs.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

APPENDIX I

SAMPLE PROGRAM

```
function elliptic()
warning off; clear all; close all;

n=100; m=100; mm=10;
p=create_rect(n,0,0,1,1);
p2=create_grid(m,0,0,1,1);
%p2=create_randgrid(m,0,0,1,1);
for i=1:m p(i+n,1)=p2(i,1); p(i+n,2)=p2(i,2); end

for i=1:n z(i,1)=g(p(i,1),p(i,2)); end
for i=n+1:n+m z(i,1)=f(p(i,1),p(i,2)); end

for i=1:n
  for j=1:n+m
    r=sqrt((p(i,1)-p(j,1))^2+(p(i,2)-p(j,2))^2);
    matrix(i,j)=r^9;
  end
end
for i=1+n:n+m
  for j=1:n+m
    r=sqrt((p(i,1)-p(j,1))^2+(p(i,2)-p(j,2))^2);
        matrix(i,j)=r^9;
    matrix(i,j)=Lr(r,p(i,1),p(i,2),p(j,1),p(j,2));
  end
end

lam=matrix\z;

[x1,y1]=meshgrid(0:1/(mm-1):1);
zexp=zeros(mm,mm);
maxerr=0;

for i=1:mm
  for j=1:mm
    for k=1:n+m
      r=sqrt((p(k,1)-x1(i,j))^2+(p(k,2)-y1(i,j))^2);
      zexp(i,j)=zexp(i,j)+lam(k)*r^9;
```

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

```
    end
    zact(i,j)=u(x1(i,j),y1(i,j));
    zerr(i,j)=abs(zexp(i,j)-zact(i,j));
    if(abs(zexp(i,j)-zact(i,j))>maxerr) maxerr=abs(zexp(i,j)-zact(i,j)); end
  end
end
maxerr

%//////////////////////////////////////////
%END//////////////////////////////////////
%//////////////////////////////////////////
function [answer]=u(x,y)
answer=x*y*exp(-x^2-y^2);
return
function [answer]=g(x,y)
answer=x*y*exp(-x^2-y^2);
return
function [answer]=f(x,y)
answer=2*x*y*exp(-x^2-y^2)*(-6*y+4*x^2*y-9+6*y^2);
return
function [answer]=Lr(r,x,y,xo,yo)
answer=9*r^5*(14*x^2*y-28*y*x*xo+14*y*xo^2+2*r^2*y+24*y^2-
48*y*yo+24*yo^2+3*x^2-6*x*xo+3*xo^2);
return
%//////////////////////////////////////////
%create grid-----------------------------------
function [m]=create_grid(n,x1,y1,x2,y2)
nn=sqrt(n);
for i=1:nn
  for j=1:nn
    m(i+(j-1)*nn,1)=x1+(x2-x1)*(i)/(nn+1);
    m(i+(j-1)*nn,2)=y1+(y2-y1)*(j)/(nn+1);
  end
end
return
%//////////////////////////////////////////
%create random grid----------------------------
function [m]=create_randgrid(n,x1,y1,x2,y2)
for i=1:n
  m(i,1)=x1+rand(1)*(x2-x1);
  m(i,2)=y1+rand(1)*(y2-y1);
end
return
%//////////////////////////////////////////
%create rectangle------------------------------
function [m]=create_rect(n,x1,y1,x2,y2)
```

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

```
nn=n/4;
for i=1:nn          %top,left,bottom,right
  m(i,1)=x1+(x2-x1)*i/(1+nn);
  m(i,2)=y2;
  m(i+nn,1)=x2;
  m(i+nn,2)=y1+(y2-y1)*i/(1+nn);
  m(i+nn*2,1)=x1+(x2-x1)*i/(1+nn);
  m(i+nn*2,2)=y1;
  m(i+nn*3,1)=x1;
  m(i+nn*3,2)=y1+(y2-y1)*i/(1+nn);
end
return
```

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

REFERENCES

[1] Atluri, S.N., Shen. S. The Meshless Local Petrov-Galerkin Method. Tech Science Press, California, 2002.

[2] Belytschko, T., et al. Meshless methods: An overview and recent developments. Computer Methods in Applied Mechanics and Engineering. 139 (1996), 3-47.

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

[4] Chen, C.S. Meshless Methods for Scientific Computing, lecture notes, 2004.

[5] Cheng, A. H.-D.; Golberg, M. A.; Kansa, E. J.; Zammito, G. Exponential convergence and h-c multiquadric collocation method for partial differential equations. Numer. Methods Partial Differential Equations 19 (2003), no. 5, 571--594.

[6] Duarte, C.A., Oden, J.T. H-p clouds - an h-p meshless method. Numer. Methods Partial Dfferential Equations 12 (1996), 673-705.

[7] Fasshauer, Gregory E. Solving differential equations with radial basis functions: multilevel methods and smoothing. Radial basis functions and their applications. Adv. Comput. Math. 11 (1999), no. 2-3, 139--159.

[8] Kansa, E. J. Multiquadrics---a scattered data approximation scheme with applications to computational fluid-dynamics. I. Surface approximations and partial derivative estimates. Comput. Math. Appl. 19 (1990), no. 8-9, 127--145.

[9] Li, Jichun. A radial basis meshless method for solving inverse boundary value problems. Comm. Numer. Methods Engrg. 20 (2004), no. 1, 51--61.

[10] Li, Jichun; Cheng, A.H.D., Chen, C.S. A comparison of efficiency and error convergence of multiquadratic collocation method and finite element method. Engineering Analysis with Boundary Elements 27 (2003), 251-257.

[11] Li, Jichun; Hon, Y. C. Domain decomposition for radial basis meshless methods. Numer. Methods Partial Differential Equations 20 (2004), no. 3, 450--462.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

[12] Li, Jichun, Hon, Y.C., Chen, C.S. Numerical comparisons of two meshless methods using radial basis functions. Engineering Analysis with Boundary Elements 26 (2002), 205-225.

[13] Wendland, Holger Scattered data approximation. Cambridge Monographs on Applied and Computational Mathematics, 17. Cambridge University Press, Cambridge, 2005.

[14] Wong, A. S. M.; Hon, Y. C.; Li, T. S.; Chung, S. L.; Kansa, E. J. Multizone decomposition for simulation of time-dependent problems using the multiquadric scheme. Comput. Math. Appl. 37 (1999), no. 8, 23--43.

[15] Zerroukat, M., Power, H., Chen, C.S. A numerical method for heat transfer problem using collocation and radial basis functions. International Journal for Numerical Methods in Engineering 42 (1998), 1263-1278.

[16] Zhou, X.; Hon, Y. C.; Li, Jichun. Overlapping domain decomposition method by radial basis functions. Appl. Numer. Math. 44 (2003), no. 1-2, 241--255.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Vita

Graduate College
University of Nevada, Las Vegas

Arthur Jonathan Lee

Home Address:
2475 Golden Arrow Dr.
Las Vegas, Nevada 89121

Degrees:
Bachelor of Science, Mathematics 2004

Thesis Title: Developing Meshless Methods for Partial Differential Equations

Thesis Examination Committee:
Chairperson, Dr. Jichun Li, Ph. D.
Committee Member, Dr. Doug Burke, Ph. D.
Committee Member, Dr. Zhonghai Ding, Ph. D.
Graduate Faculty Representative, Dr. Evangelos Yfantis, Ph. D.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.