A CONVERGENCE STUDY OF SPECTRALLY MATCHED GRIDS IN THE PRESENCE OF NON-SMOOTH DATA AND ANISOTROPY

By
ADNAN H. SABUWALA

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY UNIVERSITY OF FLORIDA 2008
I dedicate this work to my parents Hatim A. Sabuwala and Fatema H. Sabuwala.
ACKNOWLEDGMENTS

There are a myriad of people that I know and to whom I would like to extend my thanks. First and foremost, I am immensely thankful and indebted to my advisor, Dr. Shari L. Moskow, for her continued support and constant mentoring. I would also like to thank her for her belief in me. This work would not have been possible without her guidance. I would also like to thank Vladimir Druskin for his invaluable suggestions and ideas from time to time. This work was supported by the National Science Foundation under grants SCREMS-0619080, DMS-0605021, DMS-0713833. Next, I would like to thank my parents for believing in me and being patient with me throughout these years. My knowledge is incomparable to their vast experience and it is with this experience that they have guided me all my life helping me tackle both academic and personal problems that life has thrown at me over these years. I am extremely grateful to my co-advisor, Dr. Jayadeep Gopalakrishnan, who extended his selfless support to me throughout my journey. I am thankful to Dr. William Hager, Dr. Sergei Pilyugin, and Dr. Babette Brumback for serving on my committee. I am also thankful to my beautiful wife, Alefiya, who showed great patience with me in my final stages of completing my doctoral studies. Finally, words are not enough to describe my gratitude for all the friends that I have made throughout my stay at Gainesville and I want to acknowledge each and every one of them for being my friend and making me feel at home far away from home!
TABLE OF CONTENTS

ACKNOWLEDGMENTS ......................................................... 4
LIST OF TABLES .............................................................. 7
LIST OF FIGURES .............................................................. 8
ABSTRACT ................................................................. 10

CHAPTER

1 INTRODUCTION ............................................................... 11

2 SPECTRALLY MATCHED GRIDS ........................................ 13
  2.1 Introduction ......................................................... 13
  2.2 Computing the Grids ................................................ 13
  2.3 An Attempt to Find an Equivalent Finite Element Method .... 18

3 SEMI-INFINITE SPECTRAL INTERVALS ................................... 23
  3.1 Introduction .......................................................... 23
  3.2 Motivation: An Elliptic Problem with Non-Smooth Data ......... 23
    3.2.1 The Problem .................................................... 23
    3.2.2 The Semidiscretization ...................................... 25
    3.2.3 Convergence Analysis ....................................... 27
  3.3 Remes Grids .......................................................... 29
    3.3.1 The Remes Algorithm ........................................ 29
    3.3.2 The Remes Grids .............................................. 31
    3.3.3 Convergence of Remes Grids ................................ 31
  3.4 Source Problem on a Square ....................................... 33
    3.4.1 Some Numerical Results .................................... 33
    3.4.2 Comparison to Padé-Chebyshev Grids ....................... 36

4 ANISOTROPY ............................................................... 39
  4.1 Introduction .......................................................... 39
  4.2 The 1-D Anisotropic Problem ..................................... 39
    4.2.1 Motivation ..................................................... 39
    4.2.2 The Two-Sided Anisotropic Problem on a Finite Interval .. 40
  4.3 The 2-D Anisotropic Problem ..................................... 52

5 CONCLUSIONS AND FUTURE WORK ................................... 62

APPENDIX ............................................................... 65

REFERENCES ............................................................. 105
<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-1</td>
<td>Comparison of solution error magnitudes for the two-sided 1-d anisotropic problem using Remes grids over finite spectral interval for $\lambda = 1$</td>
<td>48</td>
</tr>
<tr>
<td>4-2</td>
<td>Comparison of solution error magnitudes for the two-sided 1-d anisotropic problem using Remes grids computed over semi-infinite spectral intervals for $\lambda = 10^5$</td>
<td>51</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-1</td>
<td>A plot of the staggered grid over the interval $[0, 0.5]$ for a $[4/5]$ Padé-Chebyshev rational approximation, $k = 5$.</td>
<td>18</td>
</tr>
<tr>
<td>2-2</td>
<td>A plot of the staggered grid over the interval $[0, 0.5]$ for a $[14/15]$ Padé-Chebyshev rational approximation, $k = 15$.</td>
<td>19</td>
</tr>
<tr>
<td>3-1</td>
<td>A plot of the error between the true impedance function and numerically computed rational approximation for $k = 7$.</td>
<td>32</td>
</tr>
<tr>
<td>3-2</td>
<td>A plot of the error between the true impedance function and numerically computed rational approximation for $k = 8$.</td>
<td>33</td>
</tr>
<tr>
<td>3-3</td>
<td>A plot of $\log(\text{abs(log error)})$ vs. $\log k$ for $k = 3, \ldots, 17$.</td>
<td>34</td>
</tr>
<tr>
<td>3-4</td>
<td>A plot of logarithm of the $L^2$ error vs. $k$ for $k = 3, \ldots, 16$ using the solution for $k = 17$ as a benchmark.</td>
<td>35</td>
</tr>
<tr>
<td>3-5</td>
<td>A comparison plot of logarithm of the $L^2$ error vs. $k$ for $k = 3, \ldots, 16$ using the solution using Remes grids for $k = 17$ as a benchmark and $M = 100,000$ uniform steps along the $y$-direction.</td>
<td>37</td>
</tr>
<tr>
<td>3-6</td>
<td>A plot of logarithm of the $L^2$ error vs. $\sqrt{k}$ for $k = 3, \ldots, 16$ using the solution using Remes grids for $k = 17$ as a benchmark and $M = 100,000$ uniform steps along the $y$-direction.</td>
<td>38</td>
</tr>
<tr>
<td>4-1</td>
<td>A comparison plot of the real parts of the true and numerically computed solutions for the two-sided 1-D anisotropic problem for $k = 6, \lambda = 1$.</td>
<td>43</td>
</tr>
<tr>
<td>4-2</td>
<td>A comparison plot of the imaginary parts of the true and numerically computed solutions for the two-sided 1-D anisotropic problem for $k = 6, \lambda = 1$.</td>
<td>44</td>
</tr>
<tr>
<td>4-3</td>
<td>A comparison plot of the magnitudes of the true and numerically computed solutions for the two-sided 1-D anisotropic problem for $k = 6, \lambda = 1$.</td>
<td>45</td>
</tr>
<tr>
<td>4-4</td>
<td>A comparison plot of the error between the true and numerically computed solutions for the two-sided 1-D anisotropic problem for $k = 6, \lambda = 1$.</td>
<td>46</td>
</tr>
<tr>
<td>4-5</td>
<td>A comparison plot of the real parts of the true and numerically computed solutions for the two-sided 1-D anisotropic problem for $k = 13, \lambda = 1$.</td>
<td>47</td>
</tr>
<tr>
<td>4-6</td>
<td>A comparison plot of the imaginary parts of the true and numerically computed solutions for the two-sided 1-D anisotropic problem for $k = 13, \lambda = 1$.</td>
<td>48</td>
</tr>
<tr>
<td>4-7</td>
<td>A comparison plot of the magnitudes of the true and numerically computed solutions for the two-sided 1-D anisotropic problem for $k = 13, \lambda = 1$.</td>
<td>49</td>
</tr>
</tbody>
</table>
4-8 A comparison plot of the error between the true and numerically computed solutions for the two-sided 1-D anisotropic problem for \( k = 13, \lambda = 1 \). ........................................ 50

4-9 Spectral behavior of the relative error in computing the numerical solution for the two-sided 1-D anisotropic problem. .................................................. 51

4-10 A comparison plot of the real parts of the true and numerically computed solutions for the two-sided 1-D anisotropic problem using Remes grids for \( k = 6, \lambda = 1 \). .................................................. 52

4-11 A comparison plot of the imaginary parts of the true and numerically computed solutions for the two-sided 1-D anisotropic problem using Remes grids for \( k = 6, \lambda = 1 \). .................................................. 53

4-12 A comparison plot of the magnitudes of the true and numerically computed solutions for the two-sided 1-D anisotropic problem using Remes grids for \( k = 6, \lambda = 1 \). .................................................. 54

4-13 A comparison plot of the error between the true and numerically computed solutions for the two-sided 1-D anisotropic problem using Remes grids for \( k = 6, \lambda = 1 \). .................................................. 55

4-14 Spectral behavior of the relative error in computing the numerical solution for the two-sided 1-D anisotropic problem using Remes grids. ................................. 56

4-15 Spectral behavior of the relative error in computing the numerical solution for the two-sided 1-D anisotropic problem using Remes grids and Padé-Chebyshev grids at \( x = 0 \) on a log-log scale. ........................................ 57

4-16 Spectral behavior of the relative error in computing the numerical solution for the two-sided 1-D anisotropic problem using Remes grids and Padé-Chebyshev grids at \( x = 1 \) on a log-log scale. ........................................ 58

4-17 A plot of logarithm of the \( L^2 \) error vs. \( k \) for \( k = 3, \ldots, 15 \) using the solution for \( k = 16 \) as a benchmark. .................................................. 59

4-18 A plot of logarithm of the \( L^2 \) error vs. \( k \) for \( k = 3, \ldots, 16 \) using the solution for \( k = 17 \) as a benchmark. .................................................. 60

4-19 A plot of the computed solution for the 2-d anisotropic problem with \( k = 6 \), Remes grid steps in the \( x \)-direction and \( M = 100 \) grid steps in the \( y \)-direction.  61
In this work, we present techniques that apply to receiver-targeted problems such as in geophysical exploration. In such applications, one wishes to construct an accurate image of the earth’s profile. One usually sets up a system of signal sources and receivers and the underlying pde’s are solved to obtain analytic solutions at the receiver locations. These are then compared to the received data and the guess for the earth’s profile is adjusted accordingly. One needs to solve these problems repeatedly and in an efficient manner. This calls for the use of non-uniform grids with some kind of spectral matching. In our work, we have analyzed the error convergence rate when such non-uniform spectrally matched grids are used for these receiver-targeted problems. We have also developed a new set of grids which we call Remes grids that prove to be extremely useful in problems over semi-infinite spectral intervals. The construction of these grids is outlined and so also their application to delta function signal source problems has been studied and analyzed to obtain the error convergence rate. Towards the end our work, we have applied these grids to anisotropic problems with the goal of studying their convergence rates.
CHAPTER 1
INTRODUCTION

Remote sensing is an extremely useful tool for scientists and engineers. It helps in several areas of exploration including geophysical exploration where scientists try to construct images of the earth’s profile. Typically, in geophysical exploration, one sets up a system of signal sources and receivers over an area of the earth’s surface whose image is desired. Signals are sent into the earth’s crust and the reflected signals are read by the receivers. Based on the received data one can construct an image of the earth’s profile. This involves solving certain set of partial differential equations (PDE’s) whose coefficients depend on the earth’s profile. Usually one starts with a guess for the earth’s profile, solves these equations, and then compares the analytical solution at the receiver locations with the received data to adjust the guess of the earth’s profile appropriately. As such, one needs to solve these equations repeatedly, quickly and accurately at the receiver locations. Conventional finite difference techniques of solving PDE’s are slow and yield a solution over the *entire domain*. However, we wish to compute relatively faster solutions that are very accurate only at the receiver locations. This *receiver-targeted* application has been investigated before where a non-uniform discretization of the domain is applied [2, 8, 9]. The idea is not merely to use a very fine refinement towards the locations of the signal sources and receivers but to choose grids which match the solution in the spectral domain. The foundation of this technique has been laid in [8, 9] where the exact construction of these grids has been detailed. It is based on a suitable rational approximation of the Neumann to Dirichlet map. Later on these grids have been analyzed further in [2, 12] where a simple idea of tensor product grids is used to solve multi-dimensional problems and the error convergence rate has been studied for the infinite spatial interval case. Anisotropic media present challenges in the application of these grids and these have been studied in further detail in [4]. Our current work is aimed primarily at analyzing the convergence rate of the error involved in approximating the solution when we use
these spectrally matched grids. We have also developed a new set of grids which we call *Remes grids*. These differ from the Padé-Chebyshev grids developed earlier in the rational approximation technique. These grids have been applied to a simple 2-D isotropic problem on a square and a convergence analysis is outlined in this work. The solution using these new set of grids has been compared to the solution using the traditional set of grids and we have shown numerically that the Remes grids perform much better for delta function signal sources. We have also exhibited numerical results for a 1-D anisotropic problem and propose to apply our Remes grids to solving a 2-D anisotropic problem on a square. Finally, an attempt has been made to find a finite element method which produces an equivalent system (in sense of system matrices) to the system obtained using finite difference techniques on the non-uniform discretized grid.
CHAPTER 2
SPECTRALLY MATCHED GRIDS

2.1 Introduction

In this chapter, we will describe the idea behind the construction of the spectrally matched grids and attempt to use them in order to construct an exact second order finite element formulation for our problem. The current chapter is divided into two main sections. In the first section, we will describe the theoretical background behind the construction of the spectrally matched Padé-Chebyshev grids. This will lead to a pseudocode for generation of these grids. We also describe this pseudocode in detail.

In the second section, we attempt to formulate our problem in variational form using a second order formulation. This is an attempt to obtain an exactly equivalent finite element method that uses these spectrally matched grids.

2.2 Computing the Grids

In this section, we will describe the various steps involved in the computation of the spectrally matched grids. A pseudocode for their generation is also listed. Towards the end of this section, we will present examples of grids computed using the technique described below.

We shall begin with a very simple 1-D Helmholtz equation to explain the theory behind the computation of spectrally matched grids. Consider the 1-D Helmholtz equation on the spatial interval \([0, L]\), \(L > 0\), with the prescribed boundary conditions:

\[
\lambda u - u_{xx} = 0, \quad x \in [0, L], \quad L > 0
\]

\[
u_x(0) = -1, \quad u(L) = 0
\]

We define the impedance function of problem (2–1) to be the solution at the left end-point \(x = 0\). It is easy to see that the solution to problem (2–1) is given by,
\( u(x) = \frac{(e^{2L\sqrt{\lambda}} \cdot e^{-\sqrt{\lambda}x} - e^{\sqrt{\lambda}x})}{\sqrt{\lambda} \left( e^{2L\sqrt{\lambda}} + 1 \right)} \)  \hspace{1cm} (2-2)

So, the impedance function is given by,

\[ f(\lambda) = u(0) = \frac{\tanh(L\sqrt{\lambda})}{\sqrt{\lambda}}. \] \hspace{1cm} (2–3)

We wish to approximate \((2–1)\) by a two-point finite difference scheme using non-uniform \textit{spectrally matched} grids. In particular, we will define the solution \(u\) at “potential” nodes \(x_i, \ i = 1, \cdots, k+1\), with \(x_1 = 0\) and the “derivatives” \(u_x\) at the derivative nodes \(\hat{x}_i, \ i = 0, \cdots, k\) with \(\hat{x}_0 = 0\). Corresponding to the location of the potential nodes, we get a first set of grid steps which we will call the \textit{primary grid steps}. In a similar manner, the location of the derivative nodes give rise to a second set of grid steps which we will call the \textit{dual grid steps}. Thus, define the primary grid sizes to be \(h_i = x_{i+1} - x_i, \ i = 1, \cdots, k\) and the dual grid sizes to be \(\hat{h}_i = \hat{x}_i - \hat{x}_{i-1}, \ i = 1, \cdots, k\).

Our goal is to determine the values for \(h_i, \hat{h}_i\) which lead to certain desired spectral approximation properties. Rewriting \((2–1)\) using this scheme, we get the following finite difference problem:

\[
\begin{align*}
\lambda u_i - \frac{1}{h_i} \left( \frac{u_{i+1} - u_i}{h_i} - \frac{u_i - u_{i-1}}{h_{i-1}} \right) &= 0, & i = 2, \cdots, k, \\
\lambda u_1 - \frac{1}{h_1} \left( \frac{u_2 - u_1}{h_1} \right) &= \frac{1}{h_1}, & u_{k+1} = 0. 
\end{align*}
\] \hspace{1cm} (2–4)

Note that here we have implemented the Neumann boundary condition at the left end-point \(x = 0\) as a \textit{ghost point} condition.

Equation \((2–4)\) reveals that the FD solution at \(x = 0, u_1, \) is a discrete rational function of \(\lambda, \ f_k(\lambda)\) see for example [10]. This rational function depends on the parameters \(h_i, \hat{h}_i\) that are yet to determined. Also recall that the solution to the continuous problem \((2–1)\) at \(x = 0\) was given by a continuous function of \(\lambda\) as in \((2–3)\).
In order to determine \( h_i, \hat{h}_i \), the idea is extremely simple. We wish to optimize the location of the grid points, that is, the values of \( h_i, \hat{h}_i \), so that the overall absolute error \( |f - f_k| \) in approximating the solution at \( x = 0 \) is minimized over the spectral interval of interest, say \([\lambda_1, \lambda_2]\). This can be done in two basic steps. As a first step, we will choose a suitable rational approximation over \([\lambda_1, \lambda_2]\) to our impedance function such that the error is close to optimal. Then, in the second step, we will determine the values of the parameters \( h_i, \hat{h}_i \) based on our rational approximation. First, let us take a closer look at the theory behind computing these parameter values by revisiting our FD approximation \([2]\).

The FD approximation \((2-4)\) can be rewritten more compactly in matrix form as

\[
\lambda u - Su = \hat{h}_i^{-1}e_1, \quad (2-5)
\]

where \( u = (u_1, \cdots, u_k)^T \) and \( S \) is a system matrix whose entries depend on \( h_i, \hat{h}_i \). It is easy to see that \( S \) is not symmetric and so we make a suitable transformation to make it symmetric. If we introduce a new variable \( w_i = \hat{h}_i^{1/2}u_i, \ i = 1, \cdots, k \), then we can write \((2-5)\) as

\[
\lambda w - Hw = \hat{h}_i^{-1/2}e_1, \quad w = (w_1, \cdots, w_k)^T. \quad (2-6)
\]

where \( H \) is now a symmetric tridiagonal system matrix of the form

\[
H = \begin{pmatrix}
\alpha_1 & \beta_1 & \cdots & 0 \\
\beta_1 & \alpha_2 & \beta_2 & \\
\vdots & \ddots & \ddots & \\
0 & \cdots & \beta_{k-1} & \alpha_k
\end{pmatrix}, \quad (2-7)
\]

where
\[ \alpha_1 = - (\hat{h}_1 h_1)^{-1}, \quad \beta_1 = \left( h_1 \sqrt{\hat{h}_1 \hat{h}_2} \right)^{-1}, \]

\[ \alpha_i = - \hat{h}_i^{-1} \left( \frac{1}{\hat{h}_i} + \frac{1}{\hat{h}_{i-1}} \right), \quad \beta_i = \left( h_i \sqrt{\hat{h}_i \hat{h}_{i+1}} \right)^{-1} \quad i = 2, \ldots, k. \]  

(2.8)

Suppose the eigenvectors of \( H \) are \( s_i \) and the corresponding eigenvalues are \( \theta_i \), then we can write \( H = LDL^T \) using eigenvalue decomposition, where \( D = \text{diag}\{\theta_i\} \) and \( L = [s_1, \ldots, s_k]^T \) is the orthogonal matrix of eigenvectors. We can now solve for \( \mathbf{w} \) and hence, \( u_1 \), using the above decomposition. Multiplying (2–6) by \( L^T \) on the left and using \( H = LDL^T \) combined with \( L^T L = I \), we get,

\[ u_1 = f_k(\lambda) = \hat{h}_1^{-1} e_1^T L(\lambda I - D)^{-1} L^T e_1. \]  

(2–9)

where \( I \) is the corresponding identity matrix.

If we rearrange (2–9), then we see that

\[ f_k(\lambda) = \sum_{i=1}^{k} \frac{y_i}{\lambda - \theta_i}, \]  

(2–10)

is a partial fraction of \([k - 1/k]\) form of our rational function with \( y_i = s_i^2 / \hat{h}_1, \quad i = 1, \ldots, k \) and \( s_i \) are the first components of the eigenvectors \( s_i \).

The above discussion then suggests the following pseudocode for the computation of the spectrally matched grids.

**Pseudocode for computing the grids:**

**Step 1:** Compute a rational approximation to the continuous impedance function:

In the first step, we compute a Padé-Chebyshev rational approximation \([6]\) of a \([k - 1/k]\) form to our impedance function \( f(\lambda) = u(0) = \frac{\tanh(L \sqrt{\lambda})}{\sqrt{\lambda}} \) over a desired spectral interval, say \([\lambda_1, \lambda_2]\). We achieve this in MAPLE by using the `chebpade` command under the `numapprox` package. The result is a rational function of the form \([k - 1/k]\) and
so we proceed to compute a partial fraction decomposition of the form \((2–10)\) from which we can extract the values for \(y_i, \theta_i\).

**Step 2: Determining the values of \(h_i, \hat{h}_i\):**

Once we know the values of \(y_i, \theta_i\), then we reconstruct the matrix \(H\) by solving the inverse spectral problem. We use a \(k\)-step recursive Lanczos algorithm \([14]\) with reorthogonalization to avoid loss of orthogonality of the Lanczos vectors in finite precision arithmetic. Note that we assume a normalization \(\sum_{i=1}^{k} s_i^2 = 1\) and compute \(\hat{h}_1 = \frac{1}{\sum_{i=1}^{k} y_i}\) and \(s_i^2 = \hat{h}_1 y_i\) to get the first components of the eigenvectors which along with the eigenvalues \(\theta_i\) are used in reconstructing \(H\).

We then use Equations \((2–8)\) recursively to determine the values of \(h_i\) and \(\hat{h}_i\) for \(i = 1, \cdots, k\). It is easy to see that

\[
\begin{align*}
    h_1 &= -(\hat{h}_1 \alpha_1)^{-1}, \\
    \hat{h}_i &= (\beta_{i-1} h_{i-1}^2 \hat{h}_{i-1})^{-1}, \\
    h_i &= -\left(\alpha_i \hat{h}_i + \frac{1}{\hat{h}_{i-1}}\right)^{-1} \quad i = 2, \cdots, k.
\end{align*}
\]

All of the above calculations are done in MATLAB.

We compute these grids for several values of \(k\) over the spatial interval \(x = 0\) to \(x = 0.5\) and spectral interval \(\lambda = 1\) to \(\lambda = 100\). Figures \((2-1)\) and \((2-2)\) show the corresponding Padé-Chebyshev grids for \(k = 5\) and \(k = 15\) grid steps. Note that the grids are actually staggered even though this was not one of the requirements imposed when we were constructing these grids.

It can be shown that the convergence of the Padé-Chebyshev rational approximation is superexponential \([9]\). Finally, the spectral interval of interest may contain some resonances \(\xi_i\) of \(f(\lambda)\). If \(n, 0 \leq n \leq k\) are the number of such resonances, then we prescribe the first \(n\) terms of \(f_k(\lambda)\) to contain the resonances, that is, we look for a rational approximation of the form
Staggered grid points for $k=5$ on $[0, 0.5]$ using a $[4/5]$ Padé-Chebyshev rational approximation

\[ f_k(\lambda) = 2 \sum_{i=1}^{n} \frac{1}{\lambda - \xi_i} + \sum_{i=n+1}^{k} \frac{y_i}{\lambda - \theta_i} \quad (2-12) \]

instead of (2–10), see for example [2].

2.3 An Attempt to Find an Equivalent Finite Element Method

In the earlier section, we have seen that approximating a standard three-point finite difference operator on a non-equidistant grid computed by optimization of the grid with respect to the error of the Neumann-to-Dirichlet boundary impedance leads to exponential convergence at the prescribed boundary points. However, finite element methods (unlike finite difference techniques) are more suitable when handling complex geometries. As
such, it would be desirable to ask the question, “Is there a finite element method which is equivalent to the finite difference scheme?” In this section, we attempt to mathematically formulate the problem and then try various approaches in order to answer this question.

Consider the continuous 1-D Helmholtz problem on \([0, 1]\) with Dirichlet boundary conditions:

\[
\begin{align*}
  u'' - \lambda u &= 0 \\
  u'(0) &= -1, \quad u(1) = 0
\end{align*}
\]  

(2–13)
Previously, we have seen that a finite difference scheme for the above problem on an
optimal grid of length $k$ which is computed by a $[k - 1/k]$ Padé-Chebyshev rational
approximation of the boundary impedance function for (2–13) led to exponential
convergence at the left end-point. The resulting scheme was staggered and consisted
of a system of primary and dual grid lines given respectively by $x = x_j$ and $x = \hat{x}_j$. This
finite difference approximation led to a $k \times k$ system matrix for (2–13). One of the key
properties of this matrix is that it is a tridiagonal matrix that is symmetric with respect
to the $\hat{h}$-weighted norm.

In this section, we formulate (2–13) in variational form. This will enable us to develop
a finite element technique which will be equivalent (in the sense of system matrices) to the
finite difference scheme that has been previously developed.

Let us begin with a second-order continuous variational formulation where we
multiply (2–13) by a test function $v \in V$ and integrate by parts over $[0, 1]$. Here

$$V = \{v \mid v \text{ is continuous, } v' \text{ is piecewise continuous on } [0, 1], \, v(1) = 0\} \quad (2–14)$$

This yields the following equation

$$\int_0^1 u'v' \, dx + \lambda \int_0^1 uv \, dx = v(0) \quad (2–15)$$

which can be rewritten as

$$\langle u', v' \rangle + \lambda \langle u, v \rangle = v(0) \quad (2–16)$$

where $\langle v, w \rangle = \int_0^1 vw \, dx$. Thus, a continuous second-order variational formulation can be stated as
Find \( u \in V \) such that \( \forall \ v \in V \)

\[
\langle u', v' \rangle + \lambda \langle u, v \rangle = v(0)
\]

(2–17)

We now present a discrete second-order variational formulation corresponding to the above approach. First, let us make a few notational definitions. Let \( I_j = [x_j, x_{j+1}] \) and \( \hat{I}_j = [\hat{x}_{j-1}, \hat{x}_j], \ \forall \ j = 1, \cdots, k \) be a partition of \([0, 1]\) using the system of primary and dual grids as derived for our finite difference scheme. Let \( \{ \phi_j \}_{j=1}^k \) be the set of standard basis hat functions where \( \phi_j(x_i) = \delta_{ij} \) and \( \phi_j \) is linear on each \( I_i \). Here \( \delta_{ij} \) denotes the standard Kronecker delta. Similarly, let \( \{ \hat{\phi}_j \}_{j=1}^k \) be the set of standard basis hat functions where \( \hat{\phi}_j(\hat{x}_i) = \delta_{ij} \) and \( \hat{\phi}_j \) is linear on each \( \hat{I}_i \). Now, define \( V_h \subset V \) to be the following subspace:

\[
V_h = \{ v \mid v \text{ is linear on each } \hat{I}_j, \text{ and } v \text{ is continuous on } [0, 1], v(1) = 0 \}
\]

(2–18)

and, \( U_h \subset V \) to be:

\[
U_h = \{ u \mid u \text{ is linear on each } I_j, \text{ and } u \text{ is continuous on } [0, 1], u(1) = 0 \}
\]

(2–19)

Note that \( \phi_j \in U_h \) and \( \hat{\phi}_j \in V_h \) and they form the basis for the respective spaces. Thus, a second order discrete variational formulation for (2–17) is

Find \( u_h \in U_h \) such that \( \forall \ v_h \in V_h \)

\[
\langle u'_h, v'_h \rangle + \lambda \langle u_h, v_h \rangle = v_h(0)
\]

(2–20)

Since \( \{ \hat{\phi}_j \}_{j=1}^k \) and \( \{ \phi_j \}_{j=1}^k \) form a basis for \( V_h \) and \( U_h \) respectively, we can write
\[ u_h(x) = \sum_{j=1}^{k} \xi_j \phi_j(x) \quad (2-21) \]

\[ v_h(x) = \sum_{j=1}^{k} \eta_j \hat{\phi}_j(x) \]

Combining (2–20) and (2–21) and using \( v_h = \hat{\phi}_i, \forall i = 1, \cdots, k \), we get the following set of equations:

\[
\sum_{j=1}^{k} \left[ \xi_j \langle \phi_j', \hat{\phi}_i' \rangle + \lambda \xi_j \langle \phi_j, \hat{\phi}_i \rangle \right] = \hat{\phi}_i(0); \quad \forall i = 1, \cdots, k.
\] (2–22)

The above set of equations can be written in matrix form leading to a \( k \times k \) system matrix. However, if we compute \( \langle \phi_j', \hat{\phi}_i' \rangle \) and \( \langle \phi_j, \hat{\phi}_i \rangle \) (\( \forall i = 1, \cdots, k \)), we find that the resulting system matrix is not tridiagonal (unlike the second order system matrix for the finite difference scheme). As such, we have not yet been able to find a second order finite element formulation (2–20) which yields exactly the finite difference formulation for (2–13). Most likely we will need to use a first order formulation. This is a subject of ongoing research. Note here that it is not necessary to have the matrices exactly alike – what we need is for the solutions to be the same.
CHAPTER 3
SEMI-INFINITY SPECTRAL INTERVALS

3.1 Introduction

In the previous chapter we looked at how the spectrally matched grids are constructed for a simple 1-D problem over a finite spectral interval of interest. In this chapter, we wish to analyze and study the case where we have non-smooth boundary data with possibly semi-infinite spectrum. In particular, we will be interested in a delta function boundary data case. This chapter is divided into three main sections. In the first section, we will look at a simple 2-D Laplace’s equation with non-smooth boundary data. We will analyze the error convergence rate for this problem. In the next section, we will present a conjecture for the error convergence rate in the case of a finite spatial interval and semi-infinite spectral interval using a different set of grids called Remes grids. Finally, in the last section, we will present several numerical results on a 2-D problem illustrating the convergence of Remes grids and their comparison with Padé-Chebyshev grids.

3.2 Motivation: An Elliptic Problem with Non-Smooth Data

3.2.1 The Problem

Consider the following boundary value problem for Laplace’s equation on the rectangle \((0, L) \times (0, \tilde{L})\):

\[
-w_{xx} - w_{yy} = 0 \quad (3-1)
\]

\[-w_x(0, y) = \phi(y), \quad w(L, y) = 0,
\]

\[w \quad \tilde{L}-\text{periodic in } y.\]

To begin with, let us assume that the data has a bounded spectrum, that is,

\[\phi(y) = \sum_{j=-m}^{m} a_j e^{i\omega_j y}, \quad (3-2)\]
where \( \omega_j = 2j\pi/\tilde{L} \). Furthermore suppose that \( \phi \in L^2[0, \tilde{L}] \) (or alternatively, that \( a_j \in l^2 \)). Then, using the Fourier method (separation of variables), we can obtain the Dirichlet data exactly as follows [4]:

\[
w(0, y) = \sum_{j=\pm m} f\big(\omega_j^2\big) a_j e^{i\omega_j y} \tag{3–3}
\]

where \( f \) is the impedance function defined by

\[
f(\lambda) = \frac{\tanh \left( L\sqrt{\lambda} \right)}{\sqrt{\lambda}} \tag{3–4}
\]

Below, I outline a quick proof of this result.

**Proof.** Let us assume that \( w_j(x, y) \) solves (3–1) and write it by separation of variables as

\[
w_j(x, y) = g_j(x)h_j(y) \tag{3–5}
\]

Then, it follows from Equation (3–1) that

\[
\frac{g_j''}{g_j} = \frac{-h_j''}{h_j} = \omega_j^2 \text{ (say)} \tag{3–6}
\]

So, we can write

\[
g_j(x) = A_j \cosh(\omega_j x) + B_j \sinh(\omega_j x) \tag{3–7}
\]

\[
h_j(y) = C_j e^{i\omega_j y} + D_j e^{-i\omega_j y}
\]

Since \( w_j \) is \( \tilde{L} \)-periodic in \( y \), it follows that \( \omega_j = 2j\pi/\tilde{L} \), for \( j = 0, 1, \ldots \). Further, since \( w_j(L, y) = 0 \), it follows that \( B_j = -A_j \coth(\omega_j L) \). Hence, we have

\[
w(x, y) = \sum_{j=0}^{\infty} A_j \left[ \cosh(\omega_j x) - \coth(\omega_j L) \sinh(\omega_j x) \right] \cdot \left( C_j e^{i\omega_j y} + D_j e^{-i\omega_j y} \right) \tag{3–8}
\]
Using the other boundary condition, namely, $-w_x(0, y) = \phi(y)$, we can solve for $A_j C_j$ and $A_j D_j$ whence we get (3–3).

### 3.2.2 The Semidiscretization

Now, we introduce a semidiscretization (in the $x$-direction) of (3–1) on a system of primary and dual grid lines given respectively by $x = x_j$ and $x = \hat{x}_j$ as described in Chapter 2. This yields the following semidiscretized version of (3–1):

\[
- \hat{d}d W - W_{yy} = 0
\]  

\[-(dW)|_0(y) = \phi(y), \quad W_{k+1}(y) = 0
\]

\[W_j(y) \text{ $L$-periodic in } y, \quad j = 1, \ldots, k
\]

We apply the Fourier method again to (3–9) to obtain [4]

\[W_1(y) = \sum_{j=-m}^{m} f_k(\omega_j^2)a_j e^{i\omega_j y}\]  

(3–10)

where $f_k$ is the discrete impedance function given by

\[f_k(\lambda) = \sum_{i=1}^{k} \frac{y_i}{\lambda - \theta_i}\]  

(3–11)

Once again, I outline a quick proof of this below:

**Proof.** As before, let us assume that $W_j(y)$ is a solution to the semidiscretized problem (3–9) and write it by separation of variables as:

\[W_j(y) = G_j \cdot h_j(y)\]  

(3–12)

Then, it follows from (3–9) that

\[\frac{\hat{d}d G_j}{G_j} = -\frac{h''_j}{h_j} = \omega_j^2 \text{ (say)}\]  

(3–13)
So, we can write

\[ h_j(y) = C_j e^{i\omega_j y} + D_j e^{-i\omega_j y} \quad (3-14) \]

Since \( W_j \) is \( \tilde{L} \)-periodic in \( y \), it follows that \( \omega_j = 2j\pi/\tilde{L} \), for \( j = 0, 1, \cdots \). Hence, we have

\[ W(y) = \sum_{j=0}^{\infty} G_j \cdot (C_j e^{i\omega_j y} + D_j e^{-i\omega_j y}) \quad (3-15) \]

Using the other boundary condition, namely, \(-\partial W / \partial y \big|_{0} = \phi(y)\) combined with Equation (2.8) in [4], we can solve for \( G_j \cdot C_j \) and \( G_j \cdot D_j \) whence we get (3-10).

The two proofs outlined above easily extend to the case of unbounded spectrum for the data \( \phi(y) \left( \phi(y) = \sum_{j=-\infty}^{\infty} a_j e^{i\omega_j y} \right) \), that is, we have the following two results:

**Lemma 3.2.2.1.** Consider the boundary value problem (3-1) for Laplace’s equation on the rectangle \((0, L) \times (0, \tilde{L})\). Suppose that \( \{a_j\}_{j=-\infty}^{\infty} \) is a sequence in \( l^2 \). Assume that the data is given by:

\[ \phi(y) = \sum_{j=-\infty}^{\infty} a_j e^{i\omega_j y}, \quad (3-16) \]

where \( \omega_j = 2j\pi/\tilde{L} \). Then we can obtain the Dirichlet data exactly as follows:

\[ w(0, y) = \sum_{j=-\infty}^{\infty} f(\omega_j^2) a_j e^{i\omega_j y} \quad (3-17) \]

where \( f \) is the impedance function given by (3-4) and the convergence is in the sense of \( L^2(0, \tilde{L}) \).

**Lemma 3.2.2.2.** Consider the semidiscretized version of (3-1) given by (3-9). Suppose that \( \{a_j\}_{j=-\infty}^{\infty} \) is a sequence in \( l^2 \). Assume that the data is given by:

\[ \phi(y) = \sum_{j=-\infty}^{\infty} a_j e^{i\omega_j y}, \quad (3-18) \]

where \( \omega_j = 2j\pi/\tilde{L} \). Then we can obtain the Dirichlet data exactly as follows:
\[ W_1(y) = \sum_{j=-\infty}^{\infty} f_k(\omega_j^2) a_j e^{i\omega_j y} \]  

(3–19)

where \( f_k \) is the discrete impedance function given by (3–11) and the convergence is in the sense of \( L^2(0, \tilde{L}) \).

### 3.2.3 Convergence Analysis

Our goal next is to do a detailed convergence analysis for the error of the Dirichlet data. Let us first begin with the bounded data spectrum case. For this case, we can bound the error of the Dirichlet data as follows [4]:

\[
\left\| w(0, y) - W_1(y) \right\|_{L^2[0, \tilde{L}]} \leq \max_{\lambda \in [\omega_1^2, \omega_m^2]} |f(\lambda) - f_k(\lambda)|. 
\]  

(3–20)

Furthermore, for elliptic problems on a finite spectral interval of interest \([\lambda_1, \lambda_2]\) which is to the right of the origin (so that the poles are outside the spectral interval), the Padé-Chebyshev near-best rational approximation has exponential convergence in \( k \) given by [6]:

\[
|f_k(\lambda) - f(\lambda)| \leq C e^{-4k/\sqrt{\kappa}}, \quad \forall \ \lambda \in [\lambda_1, \lambda_2]
\]  

(3–21)
where $\kappa = \lambda_2/\lambda_1$. For our boundary value problem (3–1), we see that $\kappa = \lambda_2/\lambda_1 = \omega_m^2/\omega_1^2 = m^2$ since $\omega_j = 2j\pi/L$. Thus, combining (3–20) and (3–21), we get the following exponential error bound for the Dirichlet data when the data spectrum is bounded:

$$\|w(0, y) - W_1(y)\|_{L^2[0, \tilde{L}]} \leq Ce^{-4k/\sqrt{m}}.$$  

(3–22)

The obvious question that arises then, is what do we do in the case of a semi-infinite spectral interval of interest $[\lambda_1, \infty)$? It is crucial to answer this question for the case of unbounded data spectrum, since in this case our estimate for the error bound in (3–21) is no longer valid. In order to analyze the convergence rate of our Dirichlet data error for the unbounded data spectrum case, we need an error estimate on the semi-infinite spectral interval. So far, we are unaware of an error estimate and hence, we present the following proposition which describes the convergence analysis of the Dirichlet error in more generality.

**Theorem 3.2.3.1.** Assume that the maximum error in approximating a function $f$ by some corresponding rational approximation of degree $k$, $f_k$, over a semi-infinite spectral interval $[\lambda_1, \infty)$ that is to the right of the origin is given by the function $E(k, \lambda_1)$, that is, suppose that

$$|f_k(\lambda) - f(\lambda)| \leq E(k, \lambda_1), \quad \forall \ \lambda \in [\lambda_1, \infty).$$  

(3–23)

Then, for the boundary value problem (3–1) where the data has unbounded spectrum and is given by (3–16), the Dirichlet data error is bounded by $E(k, \omega_1^2)$, that is,

$$\|w(0, y) - W_1(y)\|_{L^2[0, \tilde{L}]} \leq E(k, \omega_1^2).$$  

(3–24)

**Proof.** Let us try to estimate the Dirichlet data error in a manner similar to (3–20). In this case, using Lemmas 3.2.2.1 and 3.2.2.2, we have the following estimate:
\[
\frac{\|w(0, y) - W_1(y)\|_{L^2[0, \tilde{L}]} }{\|\phi\|_{L^2[0, \tilde{L}]}} = \frac{\left\| \sum_{j=-\infty}^{\infty} a_j e^{i\omega_j y} (f - f_k)(\omega_j^2) \right\|_{L^2[0, \tilde{L}]}}{\left\| \sum_{j=-\infty}^{\infty} a_j e^{i\omega_j y} \right\|_{L^2[0, \tilde{L}]}} = \sqrt{\sum_{j=-\infty}^{\infty} a_j^2 \left[ (f - f_k)(\omega_j^2) \right]^2} \leq \max_{\lambda \in [\omega_1^2, \infty)} |f(\lambda) - f_k(\lambda)| \leq E(k, \omega_1^2). \tag{3–25}
\]

The second equality in the above proof follows from Parseval’s equality in the unbounded spectrum case. Further, using the fact that the error function, \(f(\lambda) - f_k(\lambda)\), is \(L^1\) on \([\omega_1^2, \infty)\), we can estimate the error bound as in the third inequality shown above. Finally, using the maximum error estimate in (3–23), we arrive at the conclusion.

\[\square\]

### 3.3 Remes Grids

In this section, we introduce a new set of grids for use over semi-infinite spectral intervals which we will call Remes grids. We will begin with a theoretical overview of Remes grids illustrating how they are computed followed by an in-depth error convergence analysis when Remes grids are used to compute the numerical solution [1, 13].

#### 3.3.1 The Remes Algorithm

The Remes algorithm is used to produce an optimal approximating polynomial \(P(x)\) which approximates a given function \(f(x)\) over a given interval. It is an extremely simple iterative algorithm that converges to an \(N\)th degree polynomial that has an error function with \(N + 2\) level extrema which oscillate in sign. The Chebyshev Equioscillation Theorem [5] guarantees that this approximation is uniquely optimal. Although we will use a version
which finds an optimal rational function approximation we will describe this algorithm for polynomials for simplicity.

Suppose

\[ P(x) = p_0 + p_1 x + p_2 x^2 + \cdots + p_N x^N \]  

(3–26)

is an \( N^{th} \) degree polynomial that leads to an error function with \( N + 2 \) level extrema with values \( \pm \epsilon \) at \( N + 2 \) given test points \( x_1, x_2, \cdots, x_{N+2} \) (where usually \( x_1, x_{N+2} \) are the endpoints of the interval of interest). Then, we need to solve the following set of \( N + 2 \) linear equations:

\[
\begin{align*}
    p_0 + p_1 x_1 + p_2 x_1^2 + \cdots + p_N x_1^N - f(x_1) &= +\epsilon \\
    p_0 + p_1 x_2 + p_2 x_2^2 + \cdots + p_N x_2^N - f(x_2) &= -\epsilon \\
    \vdots \\
    p_0 + p_1 x_{N+2} + p_2 x_{N+2}^2 + \cdots + p_N x_{N+2}^N - f(x_{N+2}) &= \pm\epsilon
\end{align*}
\]

(3–27)

There are \( N + 2 \) unknowns, \( p_0, p_1, \cdots, p_N \) and \( \epsilon \). Given the test points one can easily solve for the polynomial \( P \) and the number \( \epsilon \). However, even though the resultant polynomial \( P \) will have error values \( \pm \epsilon \) at the \( N + 2 \) test points, they may not be the extrema of the error function. So, in the second step of the Remes algorithm, the test points are moved to the approximate locations where the error function has its actual extrema. This is achieved using Newton’s method. After moving the test points, the solving of the linear equations is repeated yielding a new polynomial. This process is continued until the result converges to the desired accuracy.

Remes algorithm is typically started by choosing the maxima of the \( N^{th} \) degree Chebyshev polynomial as the initial set of test points. The resultant polynomial approximation is called the “Chebyshev approximation” or the “minimax approximation”.
3.3.2 The Remes Grids

As we have already seen in the previous chapter, our method of constructing the grids involves computing a rational approximation to the impedance function. Previously, we used a near-optimal Padé-Chebyshev rational approximation over the desired spectral interval. If, on the other hand, we use an optimal minimax approximation employing Remes algorithm as described above, then the resulting set of grids we will call Remes grids.

This is achieved in Mathematica using the “GeneralMinimaxApproximation” command under the “NumericalMath“Approximations”” package. In the first step, a rational approximation is constructed using the “RationalInterpolation” command. This first approximation is then used to generate a better approximation using a scheme based on Remes’ algorithm. When we used the above command to generate this approximation, we observed that Mathematica forces two of the test points, $x_1, x_{N+2}$, to be the end-points of the interval of approximation. Further, it did not allow for a semi-infinite interval of approximation. This made it clear to us that in order to find an approximation that is optimal on the entire semi-infinite interval, we would need to increase the length of the interval of approximation by shifting the right end-point far enough so that the approximation error curve begins to turn back towards zero. Also, if we choose the right end-point to be too far to the right then the error curve overshoots and the right end-point is no longer an extremum. So, we needed to adjust the length of the approximation interval appropriately. We will refer to this optimal length interval as an “interval of just right length”. Examples of these approximations are illustrated in the next subsection.

3.3.3 Convergence of Remes Grids

We illustrate our convergence estimate for the finite spatial interval case by means of results from numerical experiments that were conducted. We used Mathematica 5.2 to compute a $[k - 1/k]$ minimax rational approximation to our impedance function
\( f(\lambda) = \tanh(L\sqrt{\lambda})/\sqrt{\lambda}. \) We used \( L = 1 \) for illustration purposes and measured the error between the minimax rational approximation and the true impedance function for various values of \( k \). The approximation was computed on a spectral interval that was long enough so that the maximum absolute error on the semi-infinite interval \([1, \infty)\) is actually achieved on the spectral interval of just right length. Figures (3-1) and (3-2) show the error plot for \( k = 7 \) and \( k = 8 \) respectively.

![Error plot for k=7 for a 'just right' approximation interval [1,5*10^8]](image)

**Figure 3-1.** A plot of the error between the true impedance function and numerically computed rational approximation for \( k = 7 \).

We then used the maximum error estimate from each such plot for various values of \( k \) to create a plot of the logarithm of the absolute value of the logarithm of the maximum error against the logarithm of the \( k \) values. This plot was almost a straight line with slope close to 0.5 indicating that the maximum absolute error decays as exponential of the square-root of the mesh size. Figure (3-3) shows this plot for \( k = 3, \ldots, 17 \).
Figure 3-2. A plot of the error between the true impedance function and numerically computed rational approximation for $k = 8$.

### 3.4 Source Problem on a Square

#### 3.4.1 Some Numerical Results

In this section, we present some numerical results for a source problem on the unit square $[0, 1] \times [0, 1]$ where we used a Remes grid along the $x$-direction and a very fine uniform grid along the $y$-direction. We also illustrate the convergence rate for these Remes grids and go on to compare these grids to the spectrally matched Padé-Chebyshev grids that were discussed in an earlier chapter.

Consider the following problem on the unit square $[0, 1] \times [0, 1]$

$$u_{xx} + u_{yy} - \lambda u = 0$$

$$-u_x(0, y) = \delta_{(0,0)}(y), \quad u(1, y) = 0,$$

$$u(x, 0) = 0, \quad u(x, 1) = 0.$$
Figure 3-3. A plot of \( \log(\text{abs}(\log \text{error})) \) vs. \( \log k \) for \( k = 3, \ldots, 17 \).

We studied this problem numerically in order to estimate the error convergence rate and then compare it with spectral techniques. Here \( \lambda = 1 \), and we used Remes grids for different values of grid sizes, \( k \), along the \( x \)-direction and a very fine uniform grid in the \( y \)-direction.

In addition to studying the convergence rate of the minimax rational approximation itself, we also studied the error convergence rate for the error in computing the numerical solution to the problem (3–28). We used the minimax rational approximation to our impedance function from Mathematica to compute Remes grids for all values of grid step sizes \( k = 3, \ldots, 17 \). We then used these grids along the \( x \)-direction and very fine uniform grid along the \( y \)-direction to compute a numerical finite difference solution to our problem. Figure (3-4) shows a plot of the logarithm of the \( L^2 \) error in computing the numerical solution for various values of both Remes and uniform grid sizes. Since we do not have a true analytic solution at hand, we used the numerical solution obtained for Remes grid
steps \( k = 17 \) and uniform grid steps \( M = 1,000,000 \) as our benchmark solution. Using this benchmark, we computed the \( L^2 \) error in computing the numerical solution for all other values of the Remes grid size.

Figure 3-4. A plot of logarithm of the \( L^2 \) error vs. \( k \) for \( k = 3, \ldots, 16 \) using the solution for \( k = 17 \) as a benchmark.

Figure (3-4) indicates a convergence rate of exponential in the square-root of the Remes mesh size for the first few values of \( k \) from \( k = 3, \ldots, 12 \). Thereafter, the error curve begins to move concave downward indicating that the number of uniform grid steps, \( M \), are not enough to capture the spectrum of the delta function boundary data for larger values of the Remes grid size. We are seeing the same exponential convergence one expects with a finite spectral interval [12]. These are some of the computational resources limitations that we faced in computing the numerical solution to the problem (3–28).
3.4.2 Comparison to Padé-Chebyshev Grids

In this section, we compare our solution using Remes grids to the solution using spectrally matched Padé-Chebyshev grids. Since Remes grids are computed using the best rational approximation to our impedance function on a semi-infinite spectral interval as opposed to the Padé-Chebyshev rational approximation which is only close to optimal and for a finite spectral interval, we would expect that the numerical solution to our problem would exhibit not only a lower overall absolute error but also a quicker error convergence rate when we use the Remes grids. We did a numerical study aimed at comparing the numerical solution from these two different sets of grids and the results from the study are shown in Figures (3-5) and (3-6).

Figure (3-6) clearly shows that the overall $L^2$ error in computing the numerical solution using Remes grids is much lower than the corresponding error using the Padé-Chebyshev grids. Also, note that the plot is a straight line for the Remes grids up to about $k = 12$ while the one for the Padé-Chebyshev grids is slightly concave up indicating a slower convergence rate for the Padé-Chebyshev grids than the Remes grids. Also, it indicates a convergence rate of exponential in the square-root of the mesh size for the Remes grids. Once again, the plot curves concave down after $k = 12$ because the number of uniform grid steps are not enough to capture the spectrum of the delta function boundary data which has an infinite spectrum.
Figure 3-5. A comparison plot of logarithm of the $L^2$ error vs. $k$ for $k = 3, \ldots, 16$ using the solution using Remes grids for $k = 17$ as a benchmark and $M = 100,000$ uniform steps along the $y$-direction.
Figure 3-6. A plot of logarithm of the $L^2$ error vs. $\sqrt{k}$ for $k = 3, \ldots, 16$ using the solution using Remes grids for $k = 17$ as a benchmark and $M = 100,000$ uniform steps along the $y$-direction.
4.1 Introduction

In the earlier chapters we have seen how the spectrally matched grids and in particular the Remes grids are constructed and have looked at their applications towards computing numerical solutions to isotropic equations. These techniques yield exponential convergence at the end-point receiver locations over a given spectral interval. However, when dealing with geophysical applications we have to frequently deal with piecewise smooth media that are inherently anisotropic or layered media dipped at various angles. These lead to anisotropic equations where a mixed derivative term might be present. An example of such an anisotropic equation is the 2-D wave equation in such media. When using staggered grids for such equations, we have to sum up terms living on different grids (primary and dual) in order to approximate the mixed derivative terms. A simple interpolation process will not work as it will destroy the optimal properties of these grids. In this chapter, we will begin with a theoretical background of handling anisotropic equations. We will consider a simple 1-D anisotropic Helmholtz equation to explain the idea of odd-even splitting of the solution to deal with anisotropy. We will present several plots from our numerical study on the two-sided 1-D anisotropic problem.

4.2 The 1-D Anisotropic Problem

In this section we wish to present a model 1-D anisotropic equation that arises from a constant coefficient anisotropic elliptic equation. We will first motivate the origin of this 1-D problem and then present the challenges involved in using our spectrally matched grids directly to solve this problem numerically.

4.2.1 Motivation

Consider the following 2-D anisotropic elliptic equation:

\[-v_{xx} + 2av_{xy} - v_{yy} = 0, \quad |a| < 1, \quad a \in \mathbb{R}\]  \hspace{1cm} (4-1)
on the rectangle \((0, L) \times (0, \tilde{L})\) with periodic boundary conditions in the \(y\)-direction and Neumann conditions on the left and right sides. One way to visualize the above equation is to think of it as the Laplace’s equation on a parallelogram with angle \(\phi = \cos^{-1} a\) between its bottom and left sides. With a suitable Fourier transform along the \(y\)-direction, the above equation can be rewritten as the following 1-D anisotropic problem [4]:

\[
\lambda u - i2a\sqrt{\lambda} u_x - u_{xx} = 0
\]  

(4.2)

Note that if \(a = 0\) (or \(\phi = \pi/2\)), we get back to our model problem (2–1). It is the presence of the first order derivative term, \(u_x\), that makes this problem anisotropic. For the isotropic case \((a = 0)\), it is obvious that the above equation has the linearly independent solutions

\[
u = e^\pm \sqrt{\lambda}x
\]  

(4.3)

For the anisotropic case \((a \neq 0)\), the solutions are of the form,

\[
u = e^{b\sqrt{\lambda}x}, \quad b \in \mathbb{C}
\]  

(4.4)

where \(b\) satisfies

\[
b^2 + i2ab - 1 = 0
\]  

(4.5)

So, \(b = -ia \pm \sqrt{1 - a^2} = -ie^{\pm i\phi}\).

In the next subsection, we consider a two-sided anisotropic problem of the form (4–2) with some boundary conditions on a finite interval. Towards the end of the subsection, we present numerical results on the corresponding problem.

4.2.2 The Two-Sided Anisotropic Problem on a Finite Interval

Consider the two-sided 1-D anisotropic problem
\[
\lambda u - i 2 a \sqrt{\lambda} u_x - u_{xx} = 0
\]

\[(-u_x(0) = \alpha, \quad u_x(1) = \beta)
\]

on the finite interval \((0, 1)\) with Neumann conditions at both end-points. We wish to solve this problem numerically using our spectrally matched grids so that convergence is achieved at both end-points. Note that if we use our sets of primary \(\{x_i\}\) and dual \(\{\hat{x}_i\}\) grids, then the terms \(u, u_{xx}\) lie on the primary grid while the first order derivative, \(u_x\), lies on the dual grid. So, we need a clever way of summing these terms since interpolation would destroy the optimal properties of these grids.

Let us decompose our solution, \(u\), into its odd and even parts about \(x = 1/2\). Then we can write the solution as

\[u = u^o + u^e\]

where \(u^o, u^e\) are the odd and even parts respectively. For a 1-D isotropic problem it has been shown in [9] that if we interchange the primary and dual grids and then use them to compute the solution to the Neumann problem, then exponential convergence at the left endpoint \((x = 0)\) is still maintained. This can, therefore, be used to compute the solution to the two-sided isotropic problem by splitting the solution into its odd and even parts. Since the odd part of the solution satisfies the Dirichlet problem and the even part of the solution satisfies the Neumann problem, we can easily compute both of these parts using only one set of grids. This fact makes it easy to compute the solution to the anisotropic two-sided problem where we split the solution into its odd and even parts as above. In this case, plugging back into (4–6), we see that the original equation becomes a set of coupled equations as follows [4]:
\[ \lambda u^o - i2a\sqrt{\lambda}u^e_x - u^o_{xx} = 0 \quad (4-8) \]
\[ \lambda u^e - i2a\sqrt{\lambda}u^o_x - u^e_{xx} = 0 \]
\[ -u^o_x(0) = \frac{\alpha + \beta}{2} \]
\[ -u^e_x(0) = \frac{\alpha - \beta}{2} \]
\[ u^o(1/2) = u^e(1/2) = 0 \]

Now, observe that if we let the odd part of the solution, \( u^o \), live on the primary grid \( \{x_i\} \) and the even part of the solution, \( u^e \), live on the dual grid \( \{\hat{x}_i\} \), then their first order derivatives, \( u^o_x \) and \( u^e_x \), live on the dual and primary grids respectively. So, all of the terms in each of the above set of coupled equations lie on the same grid and hence, summing up these terms does make sense. Hence, we use the primary \( \{x_i\} \) and dual \( \{\hat{x}_i\} \) grids respectively for the odd and even parts \( U^o \) and \( U^e \) of the solution on \( (0, 1/2) \) and write the following numerical FD approximation to (4–8).

\[ \lambda U^o - i2a\sqrt{\lambda}\hat{d}U^e - \hat{d}dU^o = 0 \quad (4-9) \]
\[ \lambda U^e - i2a\sqrt{\lambda}\hat{d}U^o - \hat{d}dU^e = 0 \]
\[ dU^o(0) = \frac{\alpha + \beta}{2} \]
\[ \hat{d}U^e(0) = \frac{\alpha - \beta}{2} \]
\[ U^o(1/2) = \hat{d}U^e(1/2) = 0 \]

It has been shown that this finite difference solution will converge exponentially to the true solution at the boundary points \( x = 0 \) and \( x = 1 \).

Here, we present some results from numerical experiments that were conducted for the current problem. We computed the grid steps using a Padé-Chebyshev rational approximation for the spatial interval of length \( L = 1/2 \) for \( k = 2, \ldots, 25 \) and a spectral
interval of length $\lambda_1 = 1$ to $\lambda_2 = 100$. We then computed the solution to the two-sided 1-D anisotropic problem (4–6) using the numerical finite difference approximation (4–9) by splitting the true solution into its odd and even parts for several values of $k$. Figure (4-1) shows an overlay plot of the real parts of the true and numerically computed solutions. Figure (4-2) shows a similar plot for the imaginary parts of the true and numerically computed solutions, while Figure (4-3) shows the same for the magnitudes of the true and numerically computed solutions. Finally, Figure (4-4) shows a plot of the error in computing the numerical solution. All of these plots are drawn for $k = 6$ grid steps on $[0, 1/2]$ and $\lambda = 1$. Figures (4-5), (4-6), (4-7) and (4-8) show similar plots for $k = 13$. 

Figure 4-1. A comparison plot of the real parts of the true and numerically computed solutions for the two-sided 1-D anisotropic problem for $k = 6, \lambda = 1$. 

![Graph showing real part of computed and actual solution](image-url)
In addition to comparing the numerically computed solution to the true analytic solution, we also studied the spectral behavior of the relative error in computing the numerical solution. In particular, we computed the numerical solution by odd-even splitting for the two-sided 1-D anisotropic problem using $k = 6$ spectral grid steps for several values of $\lambda$ over the approximating spectral interval $\lambda = 1$ to $\lambda = 100$. The results from this study are indicated in Figure (4-9). From these plots, we see that the relative error at the endpoints, $x = 0$ and $x = 1$, gets exponentially worse as the spectral parameter value ranges over $\lambda = 1$ to $\lambda = 100$. Note here that the grids were computed by using a Padé-Chebyshev rational approximation to our impedance function over this spectral interval of approximation.

In chapter 3, we introduced a new set of grids which we called Remes grids. We have seen that they prove very useful for problems over semi-infinite spectral intervals. Earlier, we used these grids on a sample 2-d isotropic problem. We now wish to apply these
Figure 4-3. A comparison plot of the magnitudes of the true and numerically computed solutions for the two-sided 1-D anisotropic problem for $k = 6, \lambda = 1$.

grids to the two-sided 1-d anisotropic problem (4–6) and analyze the error convergence properties as we did when Padé-Chebyshev grids were used on the same problem. Since Remes grids are constructed by an optimal rational approximation of the impedance function, we expect to see them performing better than the traditional Padé-Chebyshev grids when applied to the 1-d anisotropic problem.

In order to perform our analysis, we first constructed a Remes approximation to the impedance function (2–3) over a finite spectral interval of $\lambda_1 = 1$ to $\lambda_2 = 100$. We used $L = 1/2$ in computing the approximation. We then used the approximation to construct Remes grids for several values of $k$ from $k = 3$ to $k = 17$. The computed grids were then used to numerically solve the two-sided 1-d anisotropic problem (4–6) using the finite difference approximation (4–9). We used our usual odd-even splitting of the solution to form a coupled system of finite difference equations. The goal was to perform a similar numerical analysis when these Remes grids are used to solve our problem. We
Figure 4-4. A comparison plot of the error between the true and numerically computed solutions for the two-sided 1-D anisotropic problem for $k = 6, \lambda = 1$.

used $\alpha = 1, \beta = 2$, and $\alpha = 0.5$ in our computations. Figure (4-10) shows an overlay plot of the real part of the true and numerically computed solutions. Figures (4-11) and (4-12) show similar plots for the imaginary part and the magnitude of the true and numerically computed solutions using the Remes grids. Finally, figure (4-13) shows the magnitude of the numerical error in computing the solution using Remes grids.
Figure 4-5. A comparison plot of the real parts of the true and numerically computed solutions for the two-sided 1-D anisotropic problem for $k = 13, \lambda = 1$.

In addition to studying the convergence properties of the real part, imaginary part, solution magnitude and error magnitude of the numerically computed solution, we also wished to study the spectral behavior of this solution over the spectral interval of interest. As such, we computed the solution to the two-sided 1-d anisotropic problem using Remes grids with $k = 6$, $a = 0.5$, $\alpha = 1$, $\beta = 2$, and $L = 1/2$ for several values of $\lambda$ in the spectral interval of interest $\lambda_1 = 1$ to $\lambda_2 = 100$. Figure (4-14) shows our resultant spectral behavior plot.

We wanted to see how our Remes grids performed in comparison to the traditional Padé-Chebyshev grids for the two-sided 1-d anisotropic problem. Since these grids were designed to give almost accurate solutions at the two end-points (receiver locations) $x = 0$ and $x = 1$, it made sense to compare the overall error magnitudes at the end-point locations. To this effect, we computed the magnitudes of the errors at $x = 0$ and $x = 1$ for
the numerically computed solutions using both the Padé-Chebyshev and the Remes grids. The following table summarizes our conclusion.

Table 4-1. Comparison of solution error magnitudes for the two-sided 1-d anisotropic problem using Remes grids over finite spectral interval for \( \lambda = 1 \)

<table>
<thead>
<tr>
<th></th>
<th>Using Padé-Chebyshev grids</th>
<th>Using Remes grids</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x = 0 )</td>
<td>( 7.6163 \times 10^{-14} )</td>
<td>( 7.7781 \times 10^{-15} )</td>
</tr>
<tr>
<td>( x = 1 )</td>
<td>( 4.0513 \times 10^{-14} )</td>
<td>( 1.0681 \times 10^{-14} )</td>
</tr>
</tbody>
</table>

The above table clearly reflects our expectation, that is, the overall error magnitudes are lower at each of the end-points when Remes grids are used to compute the numerical solution as opposed to the traditional Padé-Chebyshev grids. Since the Padé-Chebyshev grids are near optimal, we see that our Remes grids perform just marginally better than the corresponding Padé-Chebyshev grids as expected. Notice that in computing the above error estimates, we had all other things held the same such as we used the same
approximation interval $\lambda_1 = 1$ to $\lambda_2 = 100$, same values of $\lambda = 1$, $a = 0.5$, $\alpha = 1$, $\beta = 2$, and same spatial interval $L = 1/2$.

Recall that the Remes grids proved to be very useful in problems over semi-infinite spectral intervals. As such, it would be interesting to look at how these Remes grids which have been computed over semi-infinite spectral intervals perform in comparison to the traditional Padé-Chebyshev grids over a wider spectral interval of interest. To better understand this, we applied Remes grids with $k = 6$ grid steps computed over the just right interval $\lambda_1 = 1$ to $\lambda_2 = 2.95 \times 10^8$ to our sample two-sided 1-D anisotropic problem (4–6). We computed the overall relative error in computing our solution at the two end-points $x = 0$ and $x = 1$ and compared these to the relative error when Padé-Chebyshev grids were used. This comparison was made over a wider spectral interval $[5, 10^5]$. We picked equi-spaced spectral parameter values until $\lambda = 100$ and thereafter we used 50 equally spaced points in the logspace from $\lambda = 100$ to $\lambda = 10^5$. Figures (4-15)
and (4-16) show a comparison between the relative error plots for the solution error at $x = 0$ and $x = 1$ respectively. One easily sees that even though the Padé-Chebyshev grids perform better initially, the Remes grids perform much better over the entire wider spectral interval. In order to quantify our results, we computed the relative errors in computing the solution at the end-points for $\lambda = 10^5$. We observed that the relative error in computing this solution at $x = 0$ using Remes grids was only 5.9% whereas the corresponding relative error when Padé-Chebyshev grids were used was 137.61%. In a similar manner, the overall relative error in our solution at $x = 1$ using Remes grids was only 1.48% while that using the Padé-Chebyshev grids was 20.97%. We also did similar calculations for $k = 13$ grid steps and the corresponding relative errors along with the previous ones are described in Table (4-2). So, for problems involving $\delta$-function signal sources with infinite spectrum, using Remes grids will yield much better results than the Padé-Chebyshev grids.
Figure 4-9. Spectral behavior of the relative error in computing the numerical solution for the two-sided 1-D anisotropic problem.

Table 4-2. Comparison of solution error magnitudes for the two-sided 1-d anisotropic problem using Remes grids computed over semi-infinite spectral intervals for $\lambda = 10^5$

<table>
<thead>
<tr>
<th>$k$</th>
<th>Error at $x=0$</th>
<th>Using Padé-Chebyshev grids</th>
<th>Using Remes grids</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>137.61%</td>
<td>5.9%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>20.97%</td>
<td>1.48%</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>6.99%</td>
<td>0.16%</td>
<td>0.039914%</td>
</tr>
<tr>
<td></td>
<td>1.78%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


Figure 4-10. A comparison plot of the real parts of the true and numerically computed solutions for the two-sided 1-D anisotropic problem using Remes grids for $k = 6, \lambda = 1$.

4.3 The 2-D Anisotropic Problem

We now move on to a two-dimensional anisotropic problem where we wish to use semi-discretization with Remes grids in the $x$-direction and a very fine uniform grid in the $y$-direction. Our goal is to perform a numerical analysis for the error convergence rate as we did for the 2-d isotropic problem using Remes grids in chapter 3. Let us consider the following 2-d elliptic anisotropic problem on the unit square $[0,1] \times [0,1]$:

\[
\begin{align*}
    u_{xx} + 2au_{xy} + u_{yy} - \lambda u &= 0 \\
    u_x(0, y) &= -\delta_{(0,1/2)}(y), \quad u_x(1, y) = 0, \\
    u(x, 0) &= u(x, 1), \quad u_y(x, 0) = u_y(x, 1)
\end{align*}
\]
Figure 4-11. A comparison plot of the imaginary parts of the true and numerically computed solutions for the two-sided 1-D anisotropic problem using Remes grids for $k = 6, \lambda = 1$.

Here, there is a $\delta$-function signal source at the point $(0, 1/2)$. The solution is periodic in $y$ and this is reflected by the last two conditions in (4–10). We use odd-even splitting to compute the overall solution. This is again motivated by the fact that the odd part of the solution satisfies the Dirichlet problem while the even part of the solution satisfies the Neumann problem and computing the grids for the Dirichlet problem and then using them for the Neumann problem by simply interchanging the primary and dual grid steps maintains the exponential convergence at the receiver end-points. We employ a semi-discretization of Equation (4–10) using Remes grids in the $x$-direction and a very fine uniform grid in the $y$-direction. The Remes grids are computed from a Remes rational function approximation of the impedance function on a semi-infinite spectral interval. We used $L = 1/2, a = 0.5$ and computed the $L^2$ error in approximating the solution at the left edge, $x = 0$, for several values of $k$ from $k = 3$ to $k = 16$. Figures(4-17) and (4-18)
Figure 4-12. A comparison plot of the magnitudes of the true and numerically computed solutions for the two-sided 1-D anisotropic problem using Remes grids for $k = 6, \lambda = 1$.

show plots of the logarithm of the $L^2$ error against $k$ and $\sqrt{k}$ respectively. As can be seen in these plots, for lower values of $M$, the uniform grid step size, the $L^2$ error decays exponentially in the square-root of the mesh size $k$. As the uniform grid is refined, the $L^2$ errors get worse since the spectral interval gets wider and the Remes grids which had been constructed for the isotropic problem are now being applied to an anisotropic problem.

Figure (4-19) shows a surface plot of the computed solution to our 2-d anisotropic problem (4–10) with $k = 6$ Remes grid steps in the $x$-direction and $M = 100$ uniform grid steps in the $y$-direction.
Figure 4-13. A comparison plot of the error between the true and numerically computed solutions for the two-sided 1-D anisotropic problem using Remes grids for $k = 6, \lambda = 1$. 
Figure 4-14. Spectral behavior of the relative error in computing the numerical solution for the two-sided 1-D anisotropic problem using Remes grids.
Figure 4-15. Spectral behavior of the relative error in computing the numerical solution for the two-sided 1-D anisotropic problem using Remes grids and Padé-Chebyshev grids at $x = 0$ on a log-log scale.
Figure 4-16. Spectral behavior of the relative error in computing the numerical solution for the two-sided 1-D anisotropic problem using Remes grids and Padé-Chebyshev grids at \( x = 1 \) on a log-log scale.
Figure 4-17. A plot of logarithm of the $L^2$ error vs. $k$ for $k = 3, \ldots, 15$ using the solution for $k = 16$ as a benchmark.
Figure 4-18. A plot of logarithm of the $L^2$ error vs. $k$ for $k = 3, \ldots, 16$ using the solution for $k = 17$ as a benchmark.
Figure 4-19. A plot of the computed solution for the 2-d anisotropic problem with $k = 6$ Remes grid steps in the $x$-direction and $M = 100$ grid steps in the $y$-direction.
CHAPTER 5
CONCLUSIONS AND FUTURE WORK

In chapter 2, the idea of spectrally matched grids was introduced. We described how these grids were computed along with a pseudocode which details the computations involved. We also saw some examples of these grids where the staggeredness of the grids was illustrated even though it was not imposed a priori. Towards the end of the chapter we attempted to find an equivalent FEM which gives exactly the same system matrix for our FD approximation when the standard basis hat functions were computed over our system of primary and dual grids. We concluded that such a FEM doesn’t exist since the corresponding system matrix was not tridiagonal. However, we noted that perhaps a first-order formulation might be needed.

In chapter 3, we introduced a new set of grids which we called Remes grids. These were subsequently used in computing numerical finite difference approximation to the solution of Helmholtz equation on the unit square. We studied this problem numerically in great detail and conjectured that the error in computing our solution was converging exponentially in the square-root of the Remes mesh size. This was similar to the exponential convergence one sees with a finite spectral interval. A comparison to the Padé-Chebyshev grids was made and we illustrated numerically that the Remes grids outperformed the Padé-Chebyshev grids. We also remarked that so far we are unaware of an error estimate in the rational approximation of the impedance function over semi-infinite spectral intervals. As such, we presented a more general result which indicates that the overall relative Dirichlet data error in computing the solution to Equation (3–1) was bounded by this maximum error estimate in the rational function approximation.

In the last chapter, we applied our Padé-Chebyshev grids to a simple two-sided 1-D anisotropic problem where we use odd-even splitting to compute the overall solution. Our numerical studies exhibited convergence at the two endpoints. We also studied the spectral
behavior of the error at the end-points over the approximation interval. Earlier we had applied our Remes grids to a sample 2-D anisotropic problem. We now applied these grids to the same two-sided 1-D anisotropic problem in order to study the error convergence properties. We observed that the error in computing the solutions were slightly better than when Padé-Chebyshev grids were used for the same problem. This is because the traditional Padé-Chebyshev grids are near optimal while the Remes grids are optimal over the same finite spectral interval. We also studied the spectral behavior of our error over the approximation interval. Since Remes grids prove to be very useful in problems over semi-infinite spectral intervals, we made a comparison of the relative errors in computing the numerical solution to the 1-D anisotropic problem when both Padé-Chebyshev and Remes grids were used. In particular, it was interesting to observe that even though Padé-Chebyshev grids performed better at lower spectral parameter values, they were much worse than the Remes grids for higher values of the spectral parameter. Overall the Remes grids performed much better than the Padé-Chebyshev grids when working over semi-infinite spectral intervals. Finally, we applied our Remes grids to a sample 2-D anisotropic problem with a δ-function signal source at (0, 1/2). We found that the solution errors at the $x = 1$ edge were worse than the corresponding 2-D isotropic problem and we need more Remes grid steps in the $x$-direction in order to compute the solutions accurately at the right edge. This is not too surprising since the Remes grids which have been computed using the isotropic problem, that is, on a segment of the real line are now being applied to an anisotropic problem, that is, over a ray in the complex plane at an angle of $\theta = \cos^{-1} a$. However, due to lack of computational resources we were unable to perform the same calculations for larger values of $k$ but we expect that the solution error will converge no better than in the exponential of the square-root of the mesh size.

There are several questions that still need to be answered. We would like to work on finding the answers to these questions in the future. For instance, we would like to find out if an equivalent FEM exists. We would also like to be able to compute the Remes
grids for higher values of the grid steps and then subsequently use them in our 2-D anisotropic problem to better understand the error convergence rate. We would also like to analytically obtain an error estimate for the rational function approximation to the impedance function over semi-infinite spectral intervals. Lastly, we wish to extend these ideas to three dimensional problems and perform a detailed convergence analysis for such problems.
APPENDIX

All the relevant program codes are attached below.

1. MAPLE code that computes the \(y_i\) and \(\theta_i\) values in the rational function approximation of the impedance function.

```maple
restart;
with(orthopoly);
with(numapprox);
evala(Simplify(chebpade(1/sqrt(x),x = 3 .. 5,[2, 3])));
p := %;
q := solve(denom(p) = 0,x);
q1 := [q];
q2 := convert(q1,Vector);
pden := denom(p);
pdenprime := diff(pden,x);
pnum := numer(p);
r := pnum/pdenprime;
with(LinearAlgebra);
n := Dimension(q2);
for i to n
    do r2[i] := eval(r,x = q2[i])
end do;
y := r2;
y;
theta := q2;
theta;
```
2. Example of Mathematica code that computes the *optimal* Remes rational function approximation along with the maximum error in approximation over the interval of just right length. The code listed here does this for $L = 1/2, k = 13$, and the approximation interval of just right length is $[1, 1.55 \times 10^{13}]$.

```
<< NumericalMath'Approximations'
mmlist = GeneralMiniMaxApproximation[{t, Tanh[Sqrt[t]/2]/Sqrt[t], 1}, {t, {1, 1.55*10^13}, 12, 13}, x, WorkingPrecision -> 100, MaxIterations -> 500, PlotFlag -> True]
mmfunc = mmlist[[2, 1]]
err = Abs[Tanh[Sqrt[x]/2]/Sqrt[x] - mmfunc]
Maximize[{err, 1 < x < 1.55*10^13}, x]
```
3. MATLAB file runscript.m which computes the primary and dual grid steps given the
values of $y_i$, $\theta_i$ and $k$.

D=diag(theta); % Assemble the diagonal matrix
h1hat=1/sum(y); % Compute hhat(1)
s2=h1hat.*y; % Assuming normalization, compute s, s^2
s=s2.^0.5;
D_prime(:,:,1)=D;
B(:,:,1)=zeros(length(D));

[a,b,P,nVec]=LanTri(D_prime,B,s); % Use Lanczos with reorthogonalization
% to tridiagonalize the matrix

[h,hhat]=buildgrid(a,b,h1hat); % Now build the grid

clear D_prime;
clear B;
4. MATLAB file LanTri.m that is needed for the runscript.m code. It computes the Lanczos tridiagonalization using reorthogonalization.

```matlab
function [a,b,P,nVec] = LanTri(M,B,r)

% [a,b,P,nVec] = LanTri(M,B,r)
%
% Lanczos tridiagonalization of a complex-symmetric and block tridiagonal matrix. Modified partial orthogonalization is applied if required
%
% Input
% M M(:,:,i) are the main diagonal blocks of the block tridiagonal
% B B(:,:,i) are the subdiagonal blocks of the block tridiagonal
% r starting vector
%
% Outputs
% a main diagonal of the tridiagonal
% b subdiagonal of the tridiagonal
% P unitary
% nVec number of vectors selected for reorthogonalization
%
% so that
% J = P*(diag(a) + diag(b,1) + diag(b,-1))*P.'
% where J is block tridiagonal whose main diagonal blocks are M(:,:,i)
% and the subdiagonal blocks are B(:,:,i).
%
% Dependency
% ./sbmvmul.m symmetric block tridiagonal matrix and vector multiplication

```
n = length(r); % get the dimension of the starting vector

% constants
IM = sqrt(-1);
SQRTEPS = sqrt(eps);
MIDEPS = sqrt(SQRTEPS)^3; % between sqrt(eps) and eps
%
% initialize two column vectors for diagonals
a = zeros(n,1);
b = zeros(n-1,1);
wOld = zeros(n,1); % orthogonality estimates
wCur = zeros(n,1);
wOld(1) = 1.0;
%
up = ones(n,1); % upper and lower bounds for orthogonalization intervals
low = ones(n,1);
interNum = 0; % orthogonalization interval number
doOrtho = 0; % if do orthogonalization
second = 0; % if this is the second partial orthog
nVec = 0; % number of vectors for reorthogonalization
%
P(:,1) = r/norm(r); % set the first column of P
%
for j=1:n
    tmp = sbmvmul(M,B,conj(P(:,j))); % J*conj(p(j)). Band multiplication
    a(j) = P(:,j)'*tmp; % a(j) = p(j)'*J*conj(p(j))
    % calculate r = J*conj(p(j)) - a(j)*p(j) - b(j-1)*p(j-1)
if j == 1
    r = tmp - a(j)*P(:,j);
else
    r = tmp - a(j)*P(:,j) - b(j-1)*P(:,j-1);
end

% if (j < n)
    b(j) = norm(r);
%

if (j > 2)  % compute orthogonality estimates
    wOld(1) = (b(1)*conj(wCur(2)) + a(1)*conj(wCur(1)) ... 
                 - a(j)*wCur(1) - b(j-1)*wOld(1)/b(j) ... 
                 + eps*(b(1)+b(j))*0.3*(randn + IM*randn);
    wOld(2:j-1) = (b(2:j-1).*conj(wCur(3:j)) ... 
                   + a(2:j-1).*conj(wCur(2:j-1)) ... 
                   - a(j)*wCur(2:j-1) ... 
                   + b(1:j-2).*conj(wCur(1:j-2)) ... 
                   - b(j-1)*wOld(2:j-1)/b(j) ... 
                   + eps*0.3*(b(2:j-1)+b(j)*ones(j-2,1)) ... 
                  .*(randn(j-2,1) + IM*randn(j-2,1));
%
    % swap wOld and wCur
    tmp = wOld(1:j-1);
    wOld(1:j-1) = wCur(1:j-1);
    wCur(1:j-1) = tmp;
    wOld(j) = 1.0;
end  % if j>2
wCur(j) = eps*n*(b(1)/b(j))*0.6*(randn + IM*randn);
wCur(j+1) = 1.0;

if (second == 0) % not the second time, determine intervals
    doOrtho = 0; % initialization
    interNum = 0;
    k = 1;
    while k <= j
        if (abs(wCur(k)) >= SQRTEPS) % lost orthogonality
            doOrtho = 1;
            interNum = interNum + 1;
            p = k + 1;
            while ((p < (j + 1)) & (abs(wCur(p)) >= MIDEPS))
                p = p + 1; % nearly lost orthogonality
            end % while
            up(interNum) = p - 1;
            % find the lower bound
            p = k - 1;
            while ((p > 0) & (abs(wCur(p)) >= MIDEPS))
                p = p - 1; % nearly lost orthogonality
            end % while
            low(interNum) = p + 1;
        end % if
    end % while
    k = up(interNum) + 1; % continue search
else
    k = k + 1;
end % else
end % if lost orthogonality
end % while k
end % if not second time

if ((doOrtho == 1) | (second == 1)) % now we have intervals,
% carry out orthogonalization
for (k = 1:interNum) % for each interval
    for (i = low(k):up(k))
        r = r - (P(:,i)'*r)*P(:,i); % do orthogonalization
        % reset ortho estimates
        wCur(i) = eps*1.5*(randn + IM*randn);
    end % for i

    nVec = nVec + up(k) - low(k) + 1;
    % count the number of vectors selected
    if (second == 1) % this is the second time
        second = 0; % reset
        low(k) = 0;
        up(k) = 0;
    else
        second = 1; % do second time
        doOrtho = 0; % reset

        % adjust orthogonalization intervals for the second time
        low(k) = max(1, low(k) - 1);
        up(k) = min(j + 1, up(k) + 1);
    end % if
end % for k
\begin{verbatim}
b(j) = norm(r); % recalculate b(j)
end % if

% if (abs(b(j)) < eps) % b(j)=0, quit
a = a(1:j);
b = b(1:j-1);
return
else
    P(:,j+1) = r/b(j);
end % if
end
end
\end{verbatim}
5. MATLAB file buildgrid.m that is needed for the runscript.m file. This piece of code builds the primary and dual grids given the vectors that form the tridiagonal matrix.

```matlab
function [h,hhat]=buildgrid(a,b,h1hat);

%This function computes the grid steps (both primary and dual) using the
%relations illustrated in the Druskin, Knizhnermann paper in Numer. Alg.

hhat(1)=h1hat;

h(1)=-1/(hhat(1)*a(1));

for i=2:length(a),
    hhat(i)=1/(b(i-1)^2*(h(i-1))^2*hhat(i-1));
    h(i)=-1/(a(i)*hhat(i)+1/h(i-1));
end
```
6. MATLAB file anisotropic1_modified.m. This file computes the solution to the two-sided 1-D anisotropic problem given the grid steps \( h_i, \hat{h}_i, \alpha, \beta, a, \) and \( \lambda. \)

```matlab
function [u_o_comb,u_e_comb,x,xhat,xtot,u_actual,u_error]=
anisotropic1_modified(h,hhat,alpha,beta,a,lambda);

%Description
%
%This program solves the 1-D anisotropic problem:
% \( u'' - 2*i*a*sqrt(\lambda)*u' - \lambda*u = 0; \)
% \( u'(0) = \alpha, u'(1) = \beta; \)
% on \([0,1]\).
%
%Input parameters: h=primary gridsteps,
% hhat=dual gridsteps,
% alpha=u'(0),
% beta=u'(1),
% a=anisotropic coefficient,
% lambda=spectral parameter.
%
%Output parameters: u_o_comb=u_o on \([0,1]\),
% u_e_comb=u_e on \([0,1]\),
% x=primary grid point coordinates
% xhat=dual grid point coordinates
% u_actual=analytical solution on \([0,1]\)
% u_error=u_actual - u_computed; where u_computed is the
% computed solution on \([0,1]\).
```
% Form the grid on the semi-interval
k=length(h);
x(1)=0; xhat(1)=0;
for m=2:k+1,
    x(m)=x(m-1)+h(m-1);
    xhat(m)=xhat(m-1)+hhat(m-1);
end

for m=k+2:2*k+1,
    x(m)=1-x(2*k+2-m);
    xhat(m+1)=1-xhat(2*k+3-m);
end;

xhat(k+2)=0.5;
xhat(2*k+3)=1;

%---------------------------------------------------------------

%Solve the finite difference scheme to get u_o and u_e
A=zeros(2*k,2*k);
b=zeros(2*k,1);
A(1,1)= lambda+1/(hhat(1)*h(1));
A(1,2)= -1/(hhat(1)*h(1));
A(k,k-1)= -1/(hhat(k)*h(k-1));
A(k,k)=lambda+1/(hhat(k)*h(k))+1/(hhat(k)*h(k-1));
A(k,2*k-1) = 2*a*sqrt(lambda)*i/hhat(k);
A(k,2*k) = -2*a*sqrt(lambda)*i/hhat(k);
A(k+1,1) = 2*a*sqrt(lambda)*i/h(1);
A(k+1,2) = -2*a*sqrt(lambda)*i/h(1);
A(k+1,k+1) = lambda + 1/(h(1)*hhat(2));
A(k+1,k+2) = -1/(h(1)*hhat(2));
A(2*k,k) = 2*a*sqrt(lambda)*i/h(k);
A(2*k,2*k-1) = -1/(h(k)*hhat(k));
A(2*k,2*k) = lambda+1/(h(k)*hhat(k));
b(1)=a*sqrt(lambda)*(alpha-beta)*i-(alpha+beta)/(2*hhat(1));
b(k+1)=-(alpha-beta)/(2*h(1));

for m=2:k-1,
    A(m,m-1)=-1/(hhat(m)*h(m-1));
    A(m,m)=lambda + 1/(hhat(m)*h(m))+1/(hhat(m)*h(m-1));
    A(m,m+1)=-1/(hhat(m)*h(m));
    A(m,k+m-1)=2*a*sqrt(lambda)*i/hhat(m);
    A(m,k+m)=-2*a*sqrt(lambda)*i/hhat(m);
    A(k+m,m)=2*a*sqrt(lambda)*i/h(m);
    A(k+m,m+1)=-2*a*sqrt(lambda)*i/h(m);
    A(k+m,k+m-1)=1/(h(m)*hhat(m));
    A(k+m,k+m)=lambda + 1/(h(m)*hhat(m+1))+1/(h(m)*hhat(m));
    A(k+m,k+m+1)=-1/(h(m)*hhat(m+1));
end

u=A\b;
u_o=u(1:k);
\[ u_e = u(k+1:end); \]

\%---------------------------------------------------------------

\%Now extend solution to the whole grid \([0,1]\]
\[ u_o = [u_o; 0]; \]
\[ u_e = [-\hat{h}(1)\cdot(\alpha-\beta)/2 + u_e(1); u_e]; \]

for \( m = 1:k, \)
    \[ u_o(k+1+m) = -u_o(k+1-m); \]
    \[ u_e(k+2+m) = u_e(k+2-m); \]
end;

\[ u_e(k+2) = u_e(k+1)\cdot(xhat(k+2) - xhat(k+3))/(xhat(k+1) - xhat(k+3)) \]
+ \[ u_e(k+3)\cdot(xhat(k+2) - xhat(k+1))/(xhat(k+3) - xhat(k+1)); \]
\[ u_e(2\cdot k+3) = u_e(1); \]

\%---------------------------------------------------------------

\%Interpolate between primary and dual grids

for \( m = 1:k, \)
    \[ u_o_temp(m) = u_o(m)\cdot(xhat(m+1) - x(m+1))/(x(m) - x(m+1)) \]
    + \[ u_o(m+1)\cdot(xhat(m+1) - x(m))/(x(m+1) - x(m)); \]
    \[ u_o_temp(k+m) = u_o(k+m)\cdot(xhat(k+m+2) - x(k+m+1))/(x(k+m) - x(k+m+1)) \]
    + \[ u_o(k+m+1)\cdot(xhat(k+m+2) - x(k+m))/(x(k+m+1) - x(k+m)); \]
end;

\[ u_o_temp = [u_o_temp.','; \]
for m=1:k-1,
    u_e_temp(m)=u_e(m+1)*(x(m+1)-xhat(m+2))/(xhat(m+1)-xhat(m+2))
    +u_e(m+2)*(x(m+1)-xhat(m+1))/(xhat(m+2)-xhat(m+1));
    u_e_temp(k-1+m)=u_e(k+m+2)*(x(k+m+1)-xhat(k+m+3))/
    (xhat(k+m+2)-xhat(k+m+3))+u_e(k+m+3)*(x(k+m+1)-xhat(k+m+2))/
    (xhat(k+m+3)-xhat(k+m+2));
end;

u_e_temp=u_e_temp.';

for m=1:2:4*k-1,
    u_o_comb(m)=u_o((m+1)/2);
    u_o_comb(m+1)=u_o_temp((m+1)/2);
end;

u_o_comb=u_o_comb.';

u_o_comb(4*k+1)=u_o(2*k+1);

u_e_comb(1)=u_e(1);
for m=2:2:2*k-2,
    u_e_comb(m)=u_e(m/2+1);
    u_e_comb(m+1)=u_e_temp(m/2);
end;

u_e_comb(2*k)=u_e(k+1);

u_e_comb(2*k+1)=u_e(k+2);
for m=2*k+2:2:4*k-2,
    u_e_comb(m)=u_e(m/2+2);
    u_e_comb(m+1)=u_e_temp(m/2-1);
end;

u_e_comb(4*k)=u_e(2*k+2);
u_e_comb(4*k+1)=u_e(2*k+3);
u_e_comb=u_e_comb.';

%----------------------------------------------------------------

%Construct total grid on [0,1]

x=x';
xhat=xhat';

for m=1:2:2*k-1;
    xtot(m)=x((m+1)/2);
    xtot(m+1)=xhat((m+1)/2+1);
end;

xtot(2*k+1)=x(k+1);
for m=2*k+2:2:4*k;
    xtot(m)=xhat(m/2+2);
    xtot(m+1)=x(m/2+1);
end;

xtot=xtot';
%----------------------------------------------------------------

%Compute numerical solution

u_computed=u_o_comb+u_e_comb;
%Compute analytical solution and error and plot the results

b1=-a*i-sqrt(1-a^2);
b2=-a*i+sqrt(1-a^2);
k1=(beta-alpha*exp(b2*sqrt(lambda)))/(b1*sqrt(lambda)*
(exp(b1*sqrt(lambda)/2)-exp((2*b2-b1)*sqrt(lambda)/2)));
k2=(beta-alpha*exp(b1*sqrt(lambda)))/(b2*sqrt(lambda)*
(exp(b2*sqrt(lambda)/2)-exp((2*b1-b2)*sqrt(lambda)/2)));
u_actual=k1*exp(b1*sqrt(lambda)*(xtot-0.5*ones(length(xtot),1)))
+k2*exp(b2*sqrt(lambda)*(xtot-0.5*ones(length(xtot),1)));
u_error=u_actual-u_computed;

% figure; plot(xtot,real(u_computed),'x',xtot,real(u_actual),'o');
% title('Real part of computed and actual solution');
% legend('Computed','Actual');
% figure; plot(xtot,imag(u_computed),'x',xtot,imag(u_actual),'o');
% title('Imaginary part of computed and actual solution');
% legend('Computed','Actual');
% figure; plot(xtot,abs(u_computed),'x',xtot,abs(u_actual),'o');
% title('Magnitude of computed and actual solution');
% legend('Computed','Actual');
% figure; plot(xtot,abs(u_error));
% title('Magnitude of the error');
% abs(u_error(1))
% abs(u_error(end))
7. MATLAB file minimaxonlyoptsolvec.m. This file computes the solution to the 2-D
anisotropic problem with a $\delta$-function signal source at the origin using Remes and/or
Padé-Chebyshev grids.

%This program solves the problem $u_{xx}+u_{yy}-\lambda u = 0$, on $[0,1] \times [0,1]$
%using the BC's $u(x,0)=u(x,1)=u(1,y)=0$, and $u_x(0,y)=-\delta_{(0,0)}(y)$
%We use the optimal minimax rational approximation to generate the grids
%and solve this problem using an optimal grid in the x-direction and fine
%uniform grid in the y-direction

%This program plots all the solutions at $x=0$ using $k=3$ to 17.

tic;
lambda=1;
M=100000;
hu=1/M;
solatzero=zeros(M+1,3);
% figure; hold on;

for k1=3:17,
    mk=k1;
k=30;
runc_script;
k=k1;

%Initialize the solution
Aopt=sparse((k+1)*(M+1));
bopt=sparse((k+1)*(M+1),1);

%Setup the matrix for the optimal solution
vec1 = 1/(hhat(1)*h(1)).*ones(1,M+1);
vec2 = [1/hu^2.*ones(1,M) 0];
vec3 = [0 1/hu^2.*ones(1,M)];
vec4 = zeros(1,M+1);

for i=2:k,
    vec1 = [vec1 0 1/(hhat(i)*h(i)).*ones(1,M-1) 0];
    vec2 = [vec2 0 1/hu^2.*ones(1,M-1) 0];
    vec3 = [vec3 0 1/hu^2.*ones(1,M-1) 0];
    vec4 = [vec4 0 1/(hhat(i)*h(i-1)).*ones(1,M-1) 0];
end

vec1 = [vec1 zeros(1,M+1)];
vec2 = [vec2 zeros(1,M+1)];
vec3 = [vec3 zeros(1,M+1)];
vec4 = [vec4 zeros(1,M+1)];

vec5 = -(vec1+vec2+vec3+vec4)-lambda.*ones(1,(k+1)*(M+1));

Aopt = spdiags([vec5' 0 vec2(1:end-1)]', [vec3(2:end) 0]',
             [zeros(1,M+1) vec1(1:end-M-1)]', [vec4(M+2:end) zeros(1,M+1)]',
             [0 1 -1 M+1 -M+1], (k+1) *(M+1), (k+1) *(M+1));

%full(Aopt(604:804,:))
bopt(1) = -1/(hhat(1)*hu);
clear vec1;
clear vec2;
clear vec3;
clear vec4;
clear vec5;

%Solve the system of equations for the solution
uopt=Aopt\bopt;
%uopt=pcgm(Aopt,bopt);

uopt=full(uopt);

for i=1:k+1,
  % if (i==1), xo(i) = 0; else xo(i)=xo(i-1)+h(i-1); end
  uuopt(:,i) = uopt((i-1)*(M+1)+1:i*(M+1));
end
% yo=0:1/M:1;

solatzero(:,k1-2)=uuopt(:,1);

disp('here');

clear uopt;
clear uuopt;
% clear xo;
% clear yo;
clear Aopt;
clear bopt;

% plot(yo,log(uuopt(:,1)),'r:');
% pause(3);
end

% for i=1:14,
% L2err(i)=sqrt(sum((solatzero(:,i)-solatzero(:,end)).^2)/(M+1));
% end

% figure; plot([3:16],log(L2err'),'o:');
% figure; plot(sqrt([3:16]),log(L2err'),'o:');
% figure; plot([3:16].*log([3:16]),log(L2err'),'o:');

toc;
8. MATLAB file anisotropic2d_minimaxonlyoptsolvec.m. This file computes the solution to the 2-D anisotropic problem with a δ-function signal source at (0,1/2).

%This program solves the problem \( u_{xx} + 2a u_{xy} + u_{yy} - \lambda u = 0 \), on \([0,1] \times [0,1]\) using
%the BC's \( u_x(0,y) = -\delta_{(0,0.5)}(y) \), \( u_x(1,y) = 0 \), \( u \) periodic in \( y \) =>
%\( u(x,0) = u(x,1) \) and \( u_y(x,0) = u_y(x,1) \)
%
%We use the optimal minimax rational approximation to generate the grids and solve this problem using an optimal grid in the x-direction and fine uniform grid in the y-direction

%We shall use the Remes grids for \( L=0.5 \) on semi-infinite spectral intervals.

%We use odd-even splitting about \( x=0.5 \)
%For \( u_0 \): \( i=1 \) => \( x=0 \), \( i=k+1 \) => \( x=1 \), \( j=1 \) => \( y=0 \), \( j=M+1 \) => \( y=1 \)
%For \( u_0 \): \( i=0 \) => \( x=0 \), \( i=k \) => \( x=1 \), \( j=1 \) => \( y=0 \), \( j=M+1 \) => \( y=1 \)
%Unknowns: \([u_0(1,1) \ u_0(1,2) \ldots u_0(k,M)]\) and \([u_1(1,1) \ u_1(1,2) \ldots u_1(k,M)]\)

%This program plots all the solutions at \( x=0 \) using \( k=3 \) to 17.

tic;
lambda=1;
M=10000;
hu=1/M;
solatzero=zeros(M+1,14);
solatzero=zeros(M+1,14);
% figure; hold on;
%Form the grid on the semi-interval
x(1)=0; xhat(1)=0;
for m=2:k+1,
    x(m)=x(m-1)+h(m-1);
    xhat(m)=xhat(m-1)+hhat(m-1);
end

for m=k+2:2*k+1,
    x(m)=1-x(2*k+2-m);
    xhat(m+1)=1-xhat(2*k+3-m);
end;

xhat(k+2)=0.5;
xhat(2*k+3)=1;

ygrid=[0:hu:1];
%---------------------------------------------------------------
%Initialize the solution
Aopt=sparse(2*k*M);
bopt=sparse(2*k*M,1);
Aopt_odd=sparse(k*M, 2*k*M);
Aopt_even=sparse(k*M, 2*k*M);

%Setup the matrix for the optimal solution

%First start with the Aopt_odd matrix (corresponding to odd equations)
vec1_odd = 1/(hhat(1)*h(1)).*ones(1,M);
vec2_odd = 1/(hhat(2)*h(1)).*ones(1,M);
vec3_odd = [1/hu^2 zeros(1,M-1)];
vec4_odd = [zeros(1,M-1) 1/hu^2];
vec5_odd = [1/hu^2.*ones(1,M-1) 0];
vec6_odd = [0 1/hu^2.*ones(1,M-1)];
vec1_even = zeros(1,M);
vec2_even = zeros(1,M);
vec3_even = zeros(1,M);
vec4_even = zeros(1,M);
vec5_even = zeros(1,M);
vec6_even = zeros(1,M);

for i=2:k-1,
    vec1_odd = [vec1_odd 1/(hhat(i)*h(i)).*ones(1,M)];
    vec2_odd = [vec2_odd 1/(hhat(i+1)*h(i)).*ones(1,M)];
    vec3_odd = [vec3_odd 1/hu^2 zeros(1,M-1)];
    vec4_odd = [vec4_odd zeros(1,M-1) 1/hu^2];
    vec5_odd = [vec5_odd 1/hu^2.*ones(1,M-1) 0];
for
vec6_odd = [vec6_odd 0 1/hu^2.*ones(1,M-1)];
vec1_even = [vec1_even 2*a/(hu*hhat(i)).*ones(1,M-1) 0];
vec2_even = [vec2_even 2*a/(hu*hhat(i)).*ones(1,M)];
vec3_even = [vec3_even -2*a/(hu*hhat(i)).*ones(1,M)];
vec4_even = [vec4_even -2*a/(hu*hhat(i)).*ones(1,M-1) 0];
vec5_even = [vec5_even zeros(1,M-1) 2*a/(hu*hhat(i))];
vec6_even = [vec6_even zeros(1,M-1) -2*a/(hu*hhat(i))];
end

vec1_odd = [vec1_odd 1/(hhat(k)*h(k)).*ones(1,M)];
vec2_odd = [zeros(1,M) vec2_odd];
vec3_odd = [vec3_odd 1/hu^2 zeros(1,M-1)];
vec4_odd = [vec4_odd zeros(1,M-1) 1/hu^2];
vec5_odd = [vec5_odd 1/hu^2.*ones(1,M-1) 0];
vec6_odd = [vec6_odd 0 1/hu^2.*ones(1,M-1)];
vec1_even = [vec1_even 2*a/(hu*hhat(k)).*ones(1,M-1) 0];
vec2_even = [vec2_even 2*a/(hu*hhat(k)).*ones(1,M)];
vec3_even = [vec3_even -2*a/(hu*hhat(k)).*ones(1,M)];
vec4_even = [vec4_even -2*a/(hu*hhat(k)).*ones(1,M-1) 0];
vec5_even = [vec5_even zeros(1,M-1) 2*a/(hu*hhat(k))];
vec6_even = [vec6_even zeros(1,M-1) -2*a/(hu*hhat(k))];

vec7_odd = -(vec1_odd+vec2_odd+vec3_odd+vec4_odd+vec5_odd+vec6_odd)
-lambda.*ones(1,k*M);

Aopt_odd = spdiags([vec7_odd' vec5_odd' vec6_odd' vec3_odd' vec4_odd'
vec1_odd' vec2_odd' vec1_even' vec2_even' vec3_even'])
clear vec1_odd;
clear vec2_odd;
clear vec3_odd;
clear vec4_odd;
clear vec5_odd;
clear vec6_odd;
clear vec7_odd;
clear vec1_even;
clear vec2_even;
clear vec3_even;
clear vec4_even;
clear vec5_even;
clear vec6_even;

% %----------------------------------------------------------------
%Now set up the Aopt_even matrix

vect1_even = 1/(h(1)*hhat(2)).*ones(1,M);
vect2_even = 1/(h(2)*hhat(2)).*ones(1,M);
vect3_even = [1/hu^2 zeros(1,M-1)];
vect4_even = [zeros(1,M-1) 1/hu^2];
vect5_even = [1/hu^2.*ones(1,M-1) 0];
vect6_even = [0 1/hu^2.*ones(1,M-1)];
vect1_odd = [2*a/(hu*h(1)).*ones(1,M-1) 0];
vect2_odd = [2*a/(hu*h(1)).*ones(1,M)];
vect3_odd = [-2*a/(hu*h(1)).*ones(1,M)];
vect4_odd = [-2*a/(hu*h(1)).*ones(1,M-1) 0];
vect5_odd = [zeros(1,M-1) 2*a/(hu*h(1))];
vect6_odd = [zeros(1,M-1) -2*a/(hu*h(1))];

for i=2:k-1,
    vect1_even = [vect1_even 1/(h(i)*hhat(i+1)).*ones(1,M)];
    vect2_even = [vect2_even 1/(h(i+1)*hhat(i+1)).*ones(1,M)];
    vect3_even = [vect3_even 1/hu^2 zeros(1,M-1)];
    vect4_even = [vect4_even zeros(1,M-1) 1/hu^2];
    vect5_even = [vect5_even 1/hu^2.*ones(1,M-1) 0];
    vect6_even = [vect6_even 0 1/hu^2.*ones(1,M-1)];
    vect1_odd = [vect1_odd 2*a/(hu*h(i)).*ones(1,M-1) 0];
    vect2_odd = [vect2_odd 2*a/(hu*h(i)).*ones(1,M)];
    vect3_odd = [vect3_odd -2*a/(hu*h(i)).*ones(1,M)];
    vect4_odd = [vect4_odd -2*a/(hu*h(i)).*ones(1,M-1) 0];
    vect5_odd = [vect5_odd zeros(1,M-1) 2*a/(hu*h(i))];
    vect6_odd = [vect6_odd zeros(1,M-1) -2*a/(hu*h(i))];
end

vect1_even = [vect1_even zeros(1,M)];
vect2_even = [zeros(1,M) vect2_even];
vect3_even = [vect3_even 1/hu^2 zeros(1,M-1)];
vect4_even = [vect4_even zeros(1,M-1) 1/hu^2];
vect5_even = [vect5_even 1/hu^2.*ones(1,M-1) 0];
vect6_even = [vect6_even 0 1/hu^2.*ones(1,M-1)];
vect1_odd = [vect1_odd zeros(1,M)];
vect2_odd = [vect2_odd 2*a/(hu*h(k)).*ones(1,M)];
vect3_odd = [vect3_odd -2*a/(hu*h(k)).*ones(1,M)];
vect4_odd = [vect4_odd -2*a/(hu*h(k)).*ones(1,M-1) 0];
vect5_odd = [vect5_odd zeros(1,M)];
vect6_odd = [vect6_odd zeros(1,M-1) -2*a/(hu*h(k))];

vect7_even = -(vect1_even+vect2_even+vect3_even+vect4_even+vect5_even
+vect6_even)-lambda.*ones(1,k*M);

Aopt_even = spdiags([vect7_even' vect5_even' vect6_even' vect3_even'
vect4_even' vect1_even' vect2_even' vect1_odd'
vect2_odd' vect3_odd' (vect4_odd+vect5_odd)' vect6_odd'
[k*M k*M+1 k*M-1 (k+1)*M-1 (k-1)*M+1 (k+1)*M (k-1)*M M+1
0 M 1 -(M-1)], k*M, 2*k*M);

clear vect1_odd;
clear vect2_odd;
clear vect3_odd;
clear vect4_odd;
clear vect5_odd;
clear vect6_odd;
clear vect1_even;
clear vect2_even;
clear vect3_even;
clear vect4_even;
clear vect5_even;
clear vect6_even;
clear vect7_even;

Aopt = [Aopt_odd; Aopt_even];
bopt(M/2) = a/hu^2;
bopt(M/2+1) = -a/hu^2-1/(2*hu*hhat(1));
bopt(k*M+M/2+1) = -1/(2*hu*h(1));

uopt=Aopt\bopt;
%
%---------------------------------------------------------------

% Aopt = spdiags([vec5' [0 vec2(1:end-1)]' [vec3(2:end) 0]',
% [zeros(1,M+1) vec1(1:end-M-1)]' [vec4(M+2:end) zeros(1,M+1)]',
% [0 1 -1 M+1 -(M+1)], (k+1)*(M+1), (k+1)*(M+1));
% %full(Aopt(604:804,:))
% bopt(1) = -1/(hhat(1)*hu);
%
% clear vec1;
% clear vec2;
% clear vec3;
% clear vec4;
% clear vec5;
solve the system of equations for the solution
\[ u_{\text{opt}} = A_{\text{opt}} b_{\text{opt}}; \]
\[ u_{\text{opt}} = \text{pcgm}(A_{\text{opt}}, b_{\text{opt}}); \]
\[ u_{\text{opt}} = \text{full}(u_{\text{opt}}); \]

for \( i = 1:k, \)
\[ \text{if (} i == 1, xo(i) = 0; \text{ else } xo(i) = xo(i-1) + h(i-1); \text{ end} \]
\[ u_{\text{uuopt odd}}(:,i) = u_{\text{opt}}((i-1)*M+1:i*M); \]
\[ u_{\text{uuopt even}}(:,i) = u_{\text{opt}}((i+k-1)*M+1:(i+k)*M); \]
end

\[ u_{\text{uuopt odd}}(M+1,:) = u_{\text{uuopt odd}}(1,:); \]
\[ u_{\text{uuopt even}}(M+1,:) = u_{\text{uuopt even}}(1,:); \]
\[ u_{\text{uuopt odd}}(:,k+1) = \text{zeros}(M+1,1); \]
\[ u_{\text{uuopt even}} = [[u_{\text{uuopt even}}(1:M/2,1); u_{\text{uuopt even}}(M/2+1,1)+h_{\text{hat}}(1)/(2*hu); \]
\[ \quad u_{\text{uuopt even}}(M/2+2:end,1)] u_{\text{uuopt even}}]; \]

%-----------------------------------------------------------------
%Now extend solution to the whole grid [0,1]

for \( m = 1:k, \)
\[ u_{\text{uuopt odd}}(:,k+1+m) = -u_{\text{uuopt odd}}(:,k+1-m); \]
\[ u_{\text{uuopt even}}(:,k+2+m) = u_{\text{uuopt even}}(:,k+2-m); \]
end;
uuopt_even(:,k+2)=uuopt_even(:,k+1).*(xhat(k+2)-xhat(k+3))
    /(xhat(k+1)-xhat(k+3))+uuopt_even(:,k+3)
    .*(xhat(k+2)-xhat(k+1))/(xhat(k+3)-xhat(k+1));
uuopt_even(:,2*k+3)=uuopt_even(:,1);

%-----------------------------------------------------------------
%Interpolate between primary and dual grids
%-----------------------------------------------------------------

%Interpolate between primary and dual grids
for m=1:k,
    uuopt_odd_temp(:,m)=uuopt_odd(:,m).*(xhat(m+1)-x(m+1))/(x(m)-x(m+1))
        +uuopt_odd(:,m+1).*(xhat(m+1)-x(m))/(x(m+1)-x(m));
    uuopt_odd_temp(:,k+m)=uuopt_odd(:,k+m).*(xhat(k+m+2)-x(k+m+1))
        /(x(k+m)-x(k+m+1))+uuopt_odd(:,k+m+1)
        .*(xhat(k+m+2)-x(k+m))/(x(k+m+1)-x(k+m));
end;
%uuopt_odd_temp=uuopt_odd_temp.‘;

for m=1:k-1,
    uuopt_even_temp(:,m)=uuopt_even(:,m+1).*(x(m+1)-xhat(m+2))
        /(xhat(m+1)-xhat(m+2))+uuopt_even(:,m+2)
        .*(x(m+1)-xhat(m+1))/(xhat(m+2)-xhat(m+1));
    uuopt_even_temp(:,k-1+m)=uuopt_even(:,k+m+2).*(x(k+m+1)-xhat(k+m+3))
        /(xhat(k+m+2)-xhat(k+m+3))+uuopt_even(:,k+m+3)
        .*(x(k+m+1)-xhat(k+m+2))/(xhat(k+m+3)-xhat(k+m+2));
end;
%uuopt_even_temp=uuopt_even_temp.‘;
for m=1:2:4*k-1,
    uuopt_odd_comb(:,m)=uuopt_odd(:,(m+1)/2);
    uuopt_odd_comb(:,m+1)=uuopt_odd_temp(:,(m+1)/2);
end;
%uuopt_odd_comb=uuopt_odd_comb.';
uuopt_odd_comb(:,4*k+1)=uuopt_odd(:,2*k+1);

uuopt_even_comb(:,1)=uuopt_even(:,1);
for m=2:2:2*k-2,
    uuopt_even_comb(:,m)=uuopt_even(:,m/2+1);
    uuopt_even_comb(:,m+1)=uuopt_even_temp(:,m/2);
end;
uuopt_even_comb(:,2*k)=uuopt_even(:,k+1);
uuopt_even_comb(:,2*k+1)=uuopt_even(:,k+2);
for m=2*k+2:2:4*k-2,
    uuopt_even_comb(:,m)=uuopt_even(:,m/2+2);
    uuopt_even_comb(:,m+1)=uuopt_even_temp(:,m/2-1);
end;
uuopt_even_comb(:,4*k)=uuopt_even(:,2*k+2);
uuopt_even_comb(:,4*k+1)=uuopt_even(:,2*k+3);
% uuopt_even_comb=uuopt_even_comb.';

%-----------------------------------------------------------------

%Construct total grid on [0,1]
x=x’;
xhat=xhat’;

for m=1:2:2*k-1;
    xtot(m)=x((m+1)/2);
    xtot(m+1)=xhat((m+1)/2+1);
end;
xtot(2*k+1)=x(k+1);
for m=2*k+2:2:4*k;
    xtot(m)=xhat(m/2+2);
    xtot(m+1)=x(m/2+1);
end;
xtot=xtot’;

%-------------------------------------------------------------------

% Compute numerical solution

uuopt_computed=uuopt_odd_comb+uuopt_even_comb;

%-------------------------------------------------------------------

% Plot the overall solution

% figure; surf(xtot,ygrid(4901:end),uuopt_computed(4901:end,:));
% title('Computed solution');
% xlabel('x (k=6)');
% ylabel('y (M=100)');
% zlabel('Solution value');
% legend('Computed');
% Extract the solutions at x=1 as a function of y

solatzero(:,k1-2)=uuopt_computed(:,1);

clear Aopt_odd;
clear Aopt_even;
clear Aopt;
clear bopt;
clear uopt;
clear uuopt_odd;
clear uuopt_even;
clear uuopt_odd_temp;
clear uuopt_odd_comb;
clear uuopt_even_temp;
clear uuopt_even_comb;
clear uuopt_computed;
clear x;
clear xhat;
clear xtot;

dead

% Now compute the L2 error in the solution at x=1 assuming the solution
%for k=17 to be the true solution

for i=1:13,
    L2err(i)=sqrt(sum((solatzero(:,i)-solatzero(:,end)).^2)/(M+1));
end

figure; plot([3:15],log(L2err'),'o:');
xlabel('k');
ylabel('log(L2err)');
title('log(L2err) vs. k');
figure; plot(sqrt([3:15]),log(L2err),'o:');
% figure; plot([3:16].*log([3:16]),log(L2err),'o:');

%--------------------------------------------------------------------
% yo=0:1/M:1;
% %
% solatzero(:,k1-14)=uuopt(:,1);
% %
% disp('here');
% %
% clear uopt;
% % clear uuopt;
% % clear xo;
% % clear yo;
% % clear Aopt;
% % clear bopt;
% %
%% plot(yo,log(uuopt(:,1)),'r:');
%% pause(3);
%% end

%% for i=1:14,
%% L2err(i)=sqrt(sum((solatzero(:,i)-solatzero(:,end)).^2)/(M+1));
%% end

%% figure; plot([3:16],log(L2err'),'o:');
%% figure; plot(sqrt([3:16]),log(L2err'),'o:');
%% figure; plot([3:16].*log([3:16]),log(L2err'),'o:');

%
toc;
9. MATLAB file schlum_pres.m. This file creates the spectral behavior plot for the relative errors at \( x = 0 \) and \( x = 1 \) for the finite spectral interval for both the Remes and Padé-Chebyshev grids.

\[
\text{for } \lambda = 5:5:95, \\
\quad [u_{o\_comb}, u_{e\_comb}, x, xhat, xtot, u_{\_actual}, u_{\_error}] = \\
\quad \text{anisotropic1\_modified}(h, hhat, 1, 2, 0.5, \lambda); \\
\quad \text{err1}(\lambda/5) = \text{abs}(u\_error(1))/\text{abs}(u\_actual(1)); \\
\quad \text{err2}(\lambda/5) = \text{abs}(u\_error(\text{end}))/\text{abs}(u\_actual(\text{end})); \\
\text{end}
\]

\[
\text{figure}; \\
\text{subplot}(2,1,1), \text{semilogx}([5:5:95], \text{err1}, 'rx-'); \\
\text{title}('Plot of relative error at \( x=0 \) vs. \( \lambda \) with \( k=6 \) and Remes grids'); \\
\text{xlabel}('\lambda'); \text{ylabel}('Magnitude of relative error'); \\
\text{subplot}(2,1,2), \text{semilogx}([5:5:95], \text{err2}, 'bo-'); \\
\text{title}('Plot of relative error at \( x=1 \) vs. \( \lambda \) with \( k=6 \) and Remes grids'); \\
\text{xlabel}('\lambda'); \text{ylabel}('Magnitude of relative error');
MATLAB file schlum_pres_Remes_PC.m. This file creates the spectral behavior plot for the relative errors at \( x = 0 \) and \( x = 1 \) for both types of grids and then plots them on a loglog scale.

```matlab
% Set grids

k=30;
mk=13; % Remes grids over semi-infinite spectral interval
run_script;

for lambda = 5:5:95,
    [u_o_comb,u_e_comb,x,xhat,xtot,u_actual,u_error] =
        anisotropic1_modified(h,hhat,1,2,0.5,lambda);
    Remes_err1(lambda/5)=abs(u_error(1))/abs(u_actual(1));
    Remes_err2(lambda/5)=abs(u_error(end))/abs(u_actual(end));
end

temp_lambda = logspace(2,5);

for i = 1:length(temp_lambda),
    lambda=temp_lambda(i);
    [u_o_comb,u_e_comb,x,xhat,xtot,u_actual,u_error] =
        anisotropic1_modified(h,hhat,1,2,0.5,lambda);
    Remes_err1(19+i)=abs(u_error(1))/abs(u_actual(1));
    Remes_err2(19+i)=abs(u_error(end))/abs(u_actual(end));
end

figure(1);
```
loglog([5:5:95,temp_lambda], Remes_err1,'ro--');
title('Comparison plot of relative error at x=0 vs. \lambda with k=6 using P-C and Remes grids');
xlabel('\lambda'); ylabel('Magnitude of relative error');

figure(2);
loglog([5:5:95,temp_lambda], Remes_err2,'bo--');
title('Comparison plot of relative error at x=1 vs. \lambda with k=6 using P-C and Remes grids');
xlabel('\lambda'); ylabel('Magnitude of relative error');

Remes_err1(end)
abs(u_actual(1))
Remes_err2(end)
abs(u_actual(end))
clear all;

%Set grids

k=13;
mk=30; %P-C grids over finite spectral interval
s='o';
run_script;

for lambda = 5:5:95,
    [u_o_comb,u_e_comb,x,xhat,xtot,u_actual,u_error]=
        anisotropic1_modified(h,hhat,1,2,0.5,lambda);
PC_err1(lambda/5)=abs(u_error(1))/abs(u_actual(1));
PC_err2(lambda/5)=abs(u_error(end))/abs(u_actual(end));
end

temp_lambda = logspace(2,5);

for i = 1:length(temp_lambda),
    lambda=temp_lambda(i);
    [u_o_comb,u_e_comb,x,xhat,xtot,u_actual,u_error]=
        anisotropic1_modified(h,hhat,1,2,0.5,lambda);
    PC_err1(19+i)=abs(u_error(1))/abs(u_actual(1));
    PC_err2(19+i)=abs(u_error(end))/abs(u_actual(end));
end

figure(1); hold on;
loglog([5:5:95,temp_lambda], PC_err1,'bx--');

figure(2); hold on;
loglog([5:5:95,temp_lambda], PC_err2,'rx--');

PC_err1(end)
abs(u_actual(1))
PC_err2(end)
abs(u_actual(end))
REFERENCES


BIOGRAPHICAL SKETCH

The author, Adnan H. Sabuwala, was born in Mumbai, India, on 18th November, 1978. He is the only child of Hatim A. Sabuwala and Fatema H. Sabuwala. He lived there for 22 years where he completed his B.Tech. in electrical engineering from Indian Institute of Technology, Bombay, in 2000. He then joined the Electrical and Computer Engineering Department at the University of Florida in August, 2000 as a master’s student. He then graduated with an M.S. in electrical and computer engineering from University of Florida in December, 2002. He worked with Dr. John G. Harris on a project sponsored by Motorola, Inc., for his degree. Thereafter, he joined the Mathematics Department at the University of Florida, where he was first enrolled as a master’s student. He graduated with an M.S. in mathematics in August, 2004. He has taught several classes as part of his teaching assistantship. Some notable mentions include Calculus II, Calculus III, Elementary Differential Equations. He has won the departmental teaching certificate of excellence in the academic year 2004-2005 and subsequently won the university-wide graduate student teaching award in the academic year 2005-2006. He was admitted to the doctoral program in August, 2004 and graduated with a Ph.D. in mathematics from University of Florida in May, 2008. He is now an assistant professor of mathematics at California State University, Fresno.