University of Reading

School of Mathematics, Meteorology and Physics

Best fits with adjustable nodes and Scale invariance

Stefan L. P. King

August 2006

This dissertation is submitted to the Department of Mathematics in partial fulfilment of the requirements for the degree of Master of Science

Abstract

In this dissertation we look at how to improve the approximation to a function by minimising the L_2 error. After considering a single fixed cell the technique of *assembly* from finite elements is used to produce a connected approximation.

The method is modified to allow boundary constraints *and* conserve the area under the graph. We then examine best fits with adjustable nodes and show how the mesh points move. The graphs produced illustrate that the method works efficiently. The equidistribution predictions of Carey and Dinh are checked, showing that there is a link between optimal point locations and equidistribution.

The final chapter concerns self-similar solutions of the porous medium equation. In particular it is shown that for these solutions the best fit approximation is preserved over time and hence so also is Carey and Dinh equidistribution.

Declaration

I confirm that this work is my own and the use of all other material from other sources has been properly and fully acknowledged.

Stefan L. P King

Contents

1 Introduction

In applied mathematics we want to model real world systems. We do this by developing mathematical models, which are often systems of differential equations. In order to use computers we must represent solutions to these equations in a discrete manner. In this Dissertation we will examine how we can approximate functions in an optimal way.

To construct an approximation to a function in one dimension we must first discretise the region of interest and produce a *mesh*. This is just a vector of points that divide the region into a predetermined number of partitions. The approximation to the solution is constructed by obtaining appropriate values at each mesh point.

In this way we are considering a finite problem. The question is how accurate is the representation.

There are two main ways of representing functions in this way, finite differences and finite elements. The former just uses a set of values but the latter builds the approximation (which holds everywhere) by a linear combination of test functions (a Ritz expansion). We will use this idea here.

The Dissertation starts by considering how to approximate a function on a fixed mesh. To get the best solution we need to measure the error and in this work we shall use the L_2-norm. We shall seek an approximation by minimising this norm, thus finding the L_2 *best fit*. Hence, an examination of the approximation accuracy can take place.

It is possible to improve the approximation accuracy in various ways. Apart from increasing the number of points, we can use piecewise-polynomial interpolation [5] with polynomials of different orders. This investigation only looks at *piecewise-constants* and *piecewise-linears*. The first represents the function as a series of horizontal lines, while the latter has each partition represented by an inclined line.

It is widely noted that if the mesh point locations are allowed to move, the L_2-error will decrease. In 1985 Carey and Dinh [5] published a paper showing that there is an asymptotic relationship between the optimal locations of nodes (characterised by equidistribution) and various measures of error. In previous papers cited in [5], various authors had looked at how the mesh can be redistributed. From this motivation they developed the idea of a grading function which was then used in the mesh redistribution scheme. Having successfully found a Grading function which depends on the polynomial order being used, Carey and Dinh managed to obtain a link between error measures and equidistribution.

In 1990 P Loach and A Wathen examined the best least squares approximation with movable points, *free knots*. After examining the theory of older algorithms and comparing numerical convergence they propose a new algorithm which is *robust* and relativly *efficient* for finding a best approximation. It is noted in [7] that if the function that we are attempting to approximate is continuous then we will ensure that the approximation is also continuous.

In 1994 Baines published a paper investigating the best L_2 fits to a function with adjustable nodes. This introduced an algorithm for relocating mesh points. The central feature was cell by cell discontinuous approximation. There were two procedures that were used to adjust the node positions, *averaging* and *intersection*, as explained in chapter 3 below.

We shall follow this methodology here.

The project concludes by examining self-similar solutions of a *time dependent* partial differential equation (the porous medium equation ([6], [8])) to see whether the best fit to the self-similar solution is preserved in time. In recent years there has been a lot of interest in maintaining scale invariance while seeking a numerical solution to a PDE. In 1999 Budd [4] examined using self-similar problems to solve the porous medium equation. Many authors have introduced the idea of scale invariance to understand the solutions. We investigate the relationship between best fits with adjustable nodes and similarity solutions of the porous medium equation ([6], [8]) with interesting results.

The layout of the Dissertation is as follows.

In the next chapter we investigate how we can obtain a best fit for the function on a fixed mesh. As mentioned before, the region is assigned a mesh and we call each partition a cell. We first examine the single cell problem by looking at piecewise constant and piecewise linear approximation separately. Here we obtain a systematic way to approximate the function that minimises the L_2-error. We then use this to solve multi-cell meshes. From finite elements we can use the idea of assembly to enforce continuity at cell boundaries. The area under the curve (mass) is conserved.

If we wish to enforce a constraint this can be done but we lose mass. To ensure mass conservation we can, following Baines, Hubbard and Jimack (private communication), modify our equation to get round this difficulty. The last section of this chapter will illustrate how we modify the equations. Results are shown.

In chapter 3 we will focus on how we can move mesh points to improve the approximation. We first extend the single cell to the general problem of N-cells with discontinuous approximation by defining the cellwise procedure. This is just the production of the best fit on each mesh cell stuck together. We will then introduce the best fit procedure which is used when using adjustable nodes (which incorporates the cellwise approximations). We will look at constant and linear approximation separately, outlining the different procedures. Finally we will examine the statements about equidistribution from Carey and Dinh, illustrating the results.

Chapter 4 will introduce the porous medium equations and the idea of scale invariance. We derive the ODE for self-similar solutions and verify that a particular self-similar solution satisfies it. Finally, we will test whether the best fit is preserved over time.

The Dissertation will end by presenting conclusions and highlight possible further work.

2 Best fits on fixed meshes

We begin our investigation by examining how we can obtain a best fit approximation U to a function f on a fixed mesh. For simplicity we have decided that our mesh will be uniform. It has been stated that the approximation accuracy can be increased by either increasing the number of cells or by using higher order polynomials. In every approximation problem we strive to minimise the error between f and U. Let us first consider an arbitrary interval $[a, b]$. In this project we have decided to use the L_2-error given by

$$||f - U||_{L_2}^2 = \int_a^b \Big(f(x) - U(x)\Big)^2 \, dx. \tag{1}$$

to measure the difference. This norm is referred to as the Euclidean Norm because it can be considered to be a measure of the distance between two vectors. In fact, our approximation U is determined by a vector of points because we are seeking a piecewise representation of f. Thus, the overall objective is to find the vector U such that (1) is at a minimum.

We check whether U is a good approximation to f by minimising (1).

This can be achieved by differentiating the norm with respect to the ith coefficient U_i, giving

$$\frac{d}{dU_i} \int_a^b (f - U)^2 \, dx = \int_a^b 2(f - U)\chi_i(x) \, dx, \tag{2}$$

which is a minimum when

$$\int_a^b (f - U)\chi_i(x) \, dx = 0, \tag{3}$$

where $\chi_i(x)$ is the ith test function.

Thus, our minimisation problem can be solved by building the vector U such that it is a solution of (3). A property of best fits is that mass is conserved, meaning we get

$$\int_a^b f(x) \, dx = \int_a^b U \, dx. \tag{4}$$

To obtain a good understanding for the general problem (i.e. N-cells), the single cell problem is addressed. Let us assume that we are interested in approximating f over the interval $[0, 1]$ using only one cell. Thus, our vector will contain two mesh point locations, namely $x_0 = 0$ and $x_1 = 1$, and two unknowns U_0 and U_1.

2.1 Single Cell Examination

It has already been established that the approximation type is dependent on the polynomial that is used. Here we will examine the two simple cases presenting a precise way for reaching the best fit in each case.

2.1.1 Piecewise Constant Case

In each cell the function is approximated by a constant, a straight line parallel with the x-axis. Thus, the trial function that is used is the so-called characteristic function

$$\pi_i(x) = \begin{cases} 1, & x_i \leq x \leq x_{i+1} \\ 0, & \text{otherwise} \end{cases} . \tag{5}$$

The structure $u = U_0\pi_0(x)$ tells us that we only require one unknown U_0. To investigate how the best fit is achieved we look at the L_2-error for this approximation. We get,

$$||f - U||_{L_2}^2 = \int_a^b (f(x) - U(x))^2 \, dx = \int_a^b (f(x) - U_0\pi_0(x))^2 \, dx \tag{6}$$

which we can manipulate to investigate an expression for U_0,

$$\begin{aligned} ||f - u||_{L_2}^2 &= \int_a^b (f - U_0)^2 \, dx \\ &= \int_a^b f(x)^2 dx - 2U_0 \int_a^b f(x) dx + U_0^2 \int_a^b dx \\ &= \int_a^b f(x)^2 dx - 2U_0 \int_a^b f(x) dx + U_0^2. \end{aligned}$$

We are interested in the U_0 when the error is minimised. We notice that the equation above is a quadratic in U_0 and it is straightforward to solve. We differentiate this expression with respect to U_0.

$$\frac{\partial}{\partial U_0} ||f - u||_{L_2}^2 = -2 \int_a^b f(x) dx + 2U_0 = 0.$$

Re-arranging this equation we are able to show

$$U_0 = \int_a^b f(x) dx. \tag{7}$$

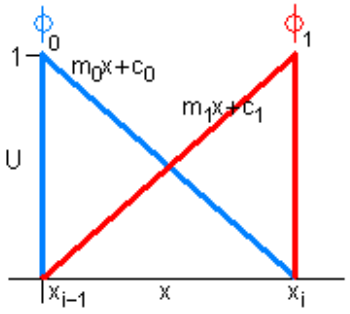
This is logical because the cell is being approximated by a constant and therefore it should be equal to the area under the function f.

2.1.2 Piecewise Linear Case

Each cell is now approximated by a straight line of the form $U = mx + c$ and therefore will have two U-values, U_0 and U_1, representing it. The basis functions that are used in this approximation are defined as:

$$\phi_i = \begin{cases} \dfrac{x - x_{i-1}}{x_i - x_{i-1}} & x_{i-1} < x < x_i \\ \dfrac{x_{i+1} - x}{x_{i+1} - x_i} & x_i < x < x_{i+1} \end{cases} \tag{8}$$

From (8) we can see that each cell depends on two basis functions, we call them ϕ_0 and ϕ_1.



As one can see ϕ_0 has negative slope whereas ϕ_1 has positive slope.

Our appoximation is given as

$$U = U_0\phi_0 + U_1\phi_1. \tag{9}$$

We next substitute (9) into (1) and minimising over U_0, U_1 to give

$$\int_a^b (f(x) - U_0\phi_0 - U_1\phi_1)\phi_i \, dx = 0 \quad i = 0, 1, \tag{10}$$

which can be re-arranged to obtain

$$\int_a^b f(x)\phi_i - \int_a^b U_0\phi_0\phi_i - \int_a^b U_1\phi_1\phi_i = 0 \quad i = 0, 1. \tag{11}$$

Using simple mathematics it is possible to simplify (11) by substituting the equations of the test functions that appear in (9). From the definition (8), we have $\phi_0 = 1 - x$ and $\phi_1 = x$ in $[0, 1]$.
Thus, (11) becomes

$$\int_0^1 f(x)\phi_i \, dx - U_0 \int_0^1 (1 - x)\phi_i \, dx - U_1 \int_0^1 x\phi_i \, dx = 0 \quad i = 0, 1. \tag{12}$$

We are now able to get two equations by examining the different cases separately.
Let $\underline{i = 0}$: (from (12))

$$\int_0^1 f(x)(1 - x) \, dx - U_0 \int_0^1 (1 - x)^2 \, dx - U_1 \int_0^1 x(1 - x) \, dx = 0 \tag{13}$$

Let $\underline{i = 1}$: (from (12))

$$\int_0^1 f(x)x \, dx - U_0 \int_0^1 x(1 - x) \, dx - U_1 \int_0^1 x^2 \, dx = 0 \tag{14}$$

With (13) and (14), we are able to obtain two equations with two unknowns.

Calculating the integrals, we get

$$\int_0^1 (1 - x)f(x) \, dx - \frac{1}{3}U_0 - \frac{1}{6}U_1 = 0 \tag{15}$$

and

$$\int_0^1 f(x)x\,dx - \frac{1}{6}U_0 - \frac{1}{3}U_1 = 0. \tag{16}$$

This can be written in matrix form as

$$\begin{pmatrix} \frac{1}{3} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{3} \end{pmatrix} \begin{pmatrix} U_0 \\ U_1 \end{pmatrix} = \begin{pmatrix} \int_0^1 (1-x)f(x)\,dx \\ \int_0^1 xf(x)\,dx \end{pmatrix} \tag{17}$$

It is very easy to get the inverse of the matrix in (17) and therefore we get

$$\begin{pmatrix} U_0 \\ U_1 \end{pmatrix} = \frac{1}{h_1} \begin{pmatrix} 4 & -2 \\ -2 & 4 \end{pmatrix} \begin{pmatrix} \int_0^1 (1-x)f(x)\,dx \\ \int_0^1 xf(x)\,dx \end{pmatrix} \tag{18}$$

We have shown how we can get a best fit approximation to the function f in one cell. When piecewise constants are used, this is just the integral of the function in the region and piecewise linears are obtained by solving the matrix system (18).

2.2 Extension to N cells

We wish now to extend this idea to a more realistic problem when we have N cells. Studying the constant case, there is only one straightforward way to do this. We just integrate each cell separately by traversing through the region, building the approximation by putting the cells in the right order. One must remember that to obtain the correct value we must divide each cell by the cell width.

When we approximate using continuous piecewise linears, we must respect continuity. This can be done using the Assembly procedure of finite elements, because it assembles the entries of the entire point vector together, creating a large mass matrix which is used to find all the U-values simultanously.

To demonstrate the idea of assembly let us consider solving the interval $[0, 1]$ with two cells, i.e. $[0, 0.5]$ and $[0.5, 1]$. Examining (17), we obtain two different values for the U value associated with $x = 0.5$. This is overcome by constructing a matrix from each sub-interval matrix. This is illustrated by:

$$\begin{pmatrix} \begin{pmatrix} \frac{1}{3} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{3} \end{pmatrix} & 0 \\ 0 & \begin{pmatrix} \frac{1}{3} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{3} \end{pmatrix} \end{pmatrix} \begin{pmatrix} U_{(x=0)} \\ U_{(x=0.5)} \\ U_{(x=0.5)} \\ U_{(x=1)} \end{pmatrix} \tag{19}$$

which leads to

$$\begin{pmatrix} \frac{1}{3} & \frac{1}{6} & 0 \\ \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \\ 0 & \frac{1}{6} & \frac{1}{3} \end{pmatrix} \begin{pmatrix} U_{(x=0)} \\ U_{(x=0.5)} \\ U_{(x=1)} \end{pmatrix} \tag{20}$$

We can see that the matrix in (20) can be extended to N cells without changing the values in the matrix. Hence, the diagonal entries are $\frac{2}{3}$, except for the first and last cells, which are at the boundaries and unchanged. Hence, our N problem will involve the following matrix,

$$\begin{pmatrix} \frac{1}{3} & \frac{1}{6} & 0 & 0 & \dots & 0 \\ \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 & \dots & 0 \\ 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \\ 0 & 0 & \dots & 0 & \frac{1}{6} & \frac{1}{3} \end{pmatrix}, \tag{21}$$

This matrix is tridiagonal and therefore the equation,

$$M\vec{u} = \vec{f} \tag{22}$$

where M is the above matrix, can be solved by the *Thomas Algorithm*.

The vector $\vec{f}$ in (22) contains the integral of the respective basis function $\phi_i(x)$ multiplied by the exact function $f(x)$. Hence, the j^{th} entry of the Load Vector is

$$\int_{x_{j-1}}^{x_{j+1}} \phi_j(x) f(x)\, dx \tag{23}$$

Using assembly, we will end up with a series of points that are connected at each mesh point. Hence, the approximation between two neighbouring points will be linear of the form $mx + c$. To investigate the error of this approximation we insert this general equation for U into equation (1). This is identical when we introduce piecewise linears. We have

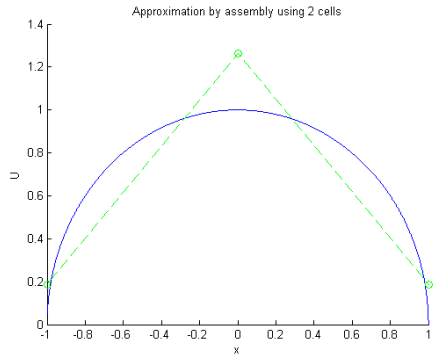
$$\sum_{0}^{N} \Big(f(x) - (mx + c) \Big)^2 \tag{24}$$

The gradient m and the y-intercept c is calculated by basic geometry.

All integrals are computed using 4-point Gaussian quadrature. Firstly the arbitary interval is mapped on $[0, 1]$ which then approximates the integral by a series of weights and strategically placed points along the interval. This gives satisfactory results without the need of extra computation.

2.2.1 Assembly Results

The following graphs show the approximation to the function $f = (1 - x^2)^{\frac{1}{m}}$.



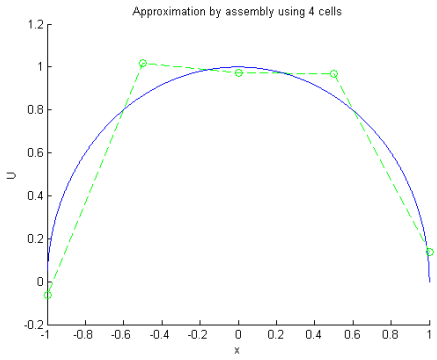
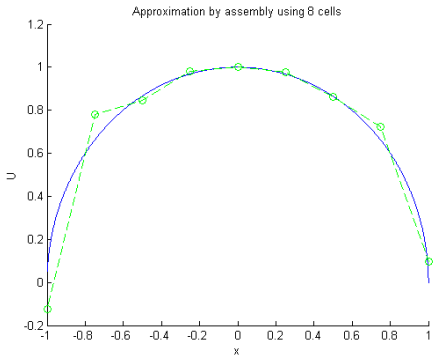


Figure 1 - Left: Assembly approximation with two cells,
Right: Assembly approximation with four cells.



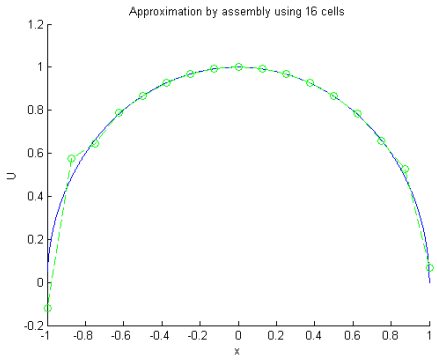


Figure 2 - Left: Assembly approximation with eight cells,
Right: Assembly approximation with sixteen cells.

Despite the program producing spurious results, one can see the approximation U becomes more accurate as more points are used. If the results were correct then it would be possible to calculate the L_2 error for each result and examine the relationship between the number of points and the error. This is done by looking at the ratio between the errors.

2.3 Ensuring Mass Conservation when an end point is fixed

2.3.1 Algorithm Modification

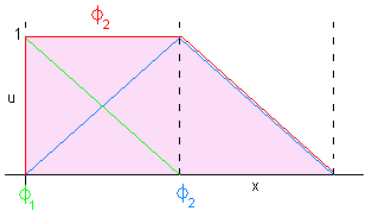
In our investigation so far we have disregarded any constraints on the values at the end-points. However, when assembly is being examined, we can incorporate this restriction, as in finite elements, by ignoring the first and last column in the mass matrix and assigning the boundary values.

A problem with this is that we will not conserve the area under the graph. If this is an issue we can repair this difficulty by adding the test functions for the first and second rows (equivalently adding the first two rows of the system together), as well as adding the last two rows together. The algorithm still removes the first and last column and row in the mass matrix, but performs these transformations prior to decreasing the matrix dimensions.

If the approximation is constrained to be zero at the endpoints, only these rows are affected. We apply this condition here because in our example we know that $(1-x^2)^{\frac{1}{2}}$ is zero at the end points of our region of interest.

We have the following two transformations,

$$\begin{aligned} \phi_2 &\mapsto \phi_2 + \phi_1 \\ \phi_{n-1} &\mapsto \phi_{n-1} + \phi_n \end{aligned}$$



As one can see the first test function that is used combines the area of the first two standard basis functions.

Let us investigate how this will affect the best fit algorithm. We first consider the local case. Recalling that we are solving a 2×2 matrix system, we can see the equations that involve the end points change.

2.3.2 Local Examination

Performing the algorithm outlined above on the 2×2 matrix system used to calculate the best fit of the first cell, we get

$$h_1 \begin{pmatrix} \frac{1}{3} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{3} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \end{pmatrix} \tag{25}$$

which becomes

$$h_1 \begin{pmatrix} \frac{1}{3} & \frac{1}{6} \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_1 + f_2 \end{pmatrix} \tag{26}$$

where h_1 is the step width between points x_1 and x_2.

Removing the first equation and putting $u_0 = 0$ gives

$$\frac{h_1}{2} u_2 = f_1 + f_2, \tag{27}$$

Similarily, we have for the last cell

$$\frac{h_{n-1}}{2}u_{n-1} = f_{n-1} + f_n \tag{28}$$

where h_{n-1} is the step width between points x_{n-1} and x_n.

To demonstrate this idea, let us look at $(1-x^2)^{\frac{1}{2}}$ on the interval $[-1, 0]$ and approximate it by one cell. Using (8) and the diagram we are able to find expressions for the two basis functions. They are

$$\phi_1 = -x \quad \text{and} \quad \phi_2 = x + 1. \tag{29}$$

Thus, we have $f_1 = \int_{-1}^{0} -xf(x)\,dx$ and $f_2 = \int_{-1}^{0} (x+1)f(x)\,dx$. We are then able to combine these integrals to obtain a value for u_2 in (27). Hence,

$$u_2 = 2\int_{-1}^{0} f(x)\,dx = 2 \times 0.787057 = 1.574114, \tag{30}$$

resulting in

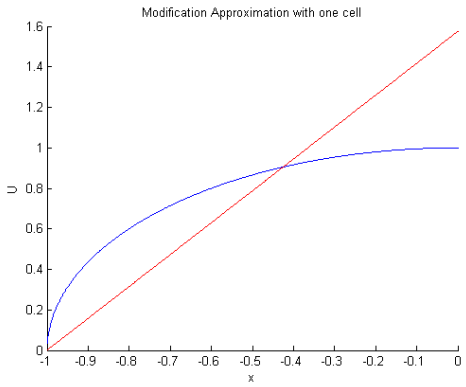


Figure 3 - Graph showing best fit approximation using modification.

2.3.3 Global Case

Let us now focus on the mass matrix so that we can see how the modification affects assembly. We need to remember that the mesh is no longer uniform and therefore each element in the tridiagonal system must be divisible by the specific width.

Before modifying the algorithm we must remember that the assembly results were obtained by using a uniform mesh. Obviously, we want to apply this method to a variable mesh, so that we can accurately approximate the function. Let us illustrate this by constructing the stiffness matrix for two elements, i.e. three points U_0, U_1, U_2. Remembering

that *assembly* is connecting the cells together, we just allow one value at each cell boundary. We do this by systematically adding common equations together. Hence,

$$\begin{pmatrix} h_1 \begin{pmatrix} \frac{1}{3} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{3} \end{pmatrix} & 0 \\ 0 & h_2 \begin{pmatrix} \frac{1}{3} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{3} \end{pmatrix} \end{pmatrix} \begin{pmatrix} U_0 \\ U_1 \\ U_1 \\ U_2 \end{pmatrix} \tag{31}$$

becomes

$$\begin{pmatrix} \frac{h_1}{3} & \frac{h_1}{6} & 0 \\ \frac{h_1}{6} & \frac{h_1}{3} + \frac{h_2}{3} & \frac{h_2}{6} \\ 0 & \frac{h_2}{6} & \frac{h_2}{3} \end{pmatrix} \begin{pmatrix} U_0 \\ U_1 \\ U_2 \end{pmatrix}. \tag{32}$$

Thus, the ith row of the mass matrix is

$$\begin{pmatrix} \dots & 0 & \frac{h_i}{6} & \frac{h_i}{3} + \frac{h_{i+1}}{3} & \frac{h_{i+1}}{6} & 0 & \dots \end{pmatrix}. \tag{33}$$

With this information we are able to construct the general mass matrix M for N points.

$$\begin{pmatrix} \frac{h_1}{3} & \frac{h_1}{6} & 0 & \dots & \dots & \dots & 0 \\ \frac{h_1}{6} & \frac{h_1}{3} + \frac{h_2}{3} & \frac{h_2}{6} & 0 & \dots & \dots & 0 \\ 0 & \frac{h_2}{6} & \frac{h_2}{3} + \frac{h_3}{3} & \frac{h_3}{6} & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \dots & 0 \\ \dots & 0 & \frac{h_i}{6} & \frac{h_i}{3} + \frac{h_{i+1}}{3} & \frac{h_{i+1}}{6} & 0 & \dots \\ \vdots & \dots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & \dots & 0 & \frac{h_{N-1}}{6} & \frac{h_N}{3} \end{pmatrix} \tag{34}$$

Modifying the matrix by the rules given in 2.3.1 when $U_0 = U_N = 0$, we get

$$\begin{pmatrix} \cancel{\frac{h_1}{3}} & \cancel{\frac{h_1}{6}} & 0 & \dots & \dots & \dots & 0 \\ \cancel{\frac{h_1}{3}} + \cancel{\frac{h_1}{6}} & \frac{h_1}{6} + \frac{h_1}{3} + \frac{h_2}{3} & \frac{h_2}{3} & 0 & \dots & \dots & 0 \\ 0 & \frac{h_2}{6} & \frac{h_2}{3} + \frac{h_3}{3} & \frac{h_3}{6} & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \dots & 0 \\ \dots & 0 & \frac{h_i}{6} & \frac{h_i}{3} + \frac{h_{i+1}}{3} & \frac{h_{i+1}}{6} & 0 & \dots \\ \vdots & \dots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & 0 & \frac{h_{N-2}}{6} & \frac{h_{N-1}}{6} + \frac{h_{N-1}}{3} + \frac{h_N}{3} & \cancel{\frac{h_N}{6}} \\ 0 & \dots & \dots & \dots & 0 & \cancel{\frac{h_{N-1}}{6}} & \cancel{\frac{h_N}{3}} \end{pmatrix} \tag{35}$$

which reduces to

$$\begin{pmatrix} \frac{h_1}{2}+\frac{h_2}{3} & \frac{h_2}{6} & 0 & \dots & 0 \\ \frac{h_2}{6} & \frac{h_2}{3}+\frac{h_3}{3} & \frac{h_3}{6} & 0 & \dots \\ 0 & \ddots & \ddots & \ddots & \dots \\ 0 & \frac{h_i}{6} & \frac{h_i}{3}+\frac{h_{i+1}}{3} & \frac{h_{i+1}}{6} & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots \\ 0 & \dots & 0 & \frac{h_{N-2}}{6} & \frac{h_{N-1}}{2}+\frac{h_N}{3} \end{pmatrix} \tag{36}$$

i.e.

$$\frac{1}{6}\begin{pmatrix} 3h_1+2h_2 & h_2 & 0 & \dots & 0 \\ h_2 & 2h_2+2h_3 & h_3 & 0 & \dots \\ 0 & \ddots & \ddots & \ddots & \dots \\ 0 & h_i & 2h_i+2h_{i+1} & h_{i+1} & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots \\ 0 & \dots & 0 & h_{N-2} & 3h_{N-1}+2h_N \end{pmatrix} \tag{37}$$

The load vector is altered in a similar way, the first element is added to the second and the last is added to the penultimate entry, before removing them. Hence, the modified load vector is

$$\begin{pmatrix} F_0+F_1 \\ F_2 \\ F_3 \\ \vdots \\ F_{N-2} \\ F_{N-1}+F_N \end{pmatrix} \tag{38}$$

2.3.4 Mass Conservation Test

As mentioned before, the whole object of modifying the equation is that we aim to conserve area (or mass). We can check this by testing whether

$$\int_{X_0}^{X_N} \phi_i U(x)dx = \int_{X_0}^{X_N} \phi_i f(x)dx \tag{39}$$

is true, as predicted.

One way to perform this test is to see whether the sum of the equations in MU add up to the sum of the elements in (38) (remembering that the matrix (37) is multiplied by the U vector). Adding the rows together we get

$$\frac{1}{2}(h_1+h_2)U_1+\frac{1}{2}(h_2+h_3)U_2+\cdots+\frac{1}{2}(h_{N-1}+h_N)U_{N-1}=\sum_{i=0}^{N}F_i. \tag{40}$$

Thus, if (40) holds then we have mass conservation.

2.3.5 Program Simplification

To simplify the coding for the C++ *Mass Matrix Constructor* function, which creates the correct vectors for the matrix (36), we are able to find an explicit formula for the cases when 2 and 3 cells are used. For larger systems we use the Thomas algorithm to solve for the U-values.

For this reason we examine the $N = 2$ and $N = 3$ cases.

$\underline{N = 2}$
We have,

$$\begin{pmatrix} d_0 & b_0 & 0 \\ l_0 & d_1 & b_1 \\ 0 & l_1 & d_2 \end{pmatrix} \begin{pmatrix} U_0 \\ U_1 \\ U_2 \end{pmatrix} = \begin{pmatrix} f_0 \\ f_1 \\ f_2 \end{pmatrix} \tag{41}$$

Applying the set of rules,

$$\begin{pmatrix} \cancel{d_0} & \cancel{b_0} & \emptyset \\ \cancel{l_0 + d_0} & b_0 + d_1 + l_1 & \cancel{b_1} \\ \emptyset & \cancel{l_1} & \cancel{d_2} \end{pmatrix} \begin{pmatrix} \cancel{U_0} \\ U_1 \\ \cancel{U_2} \end{pmatrix} = \begin{pmatrix} \cancel{f_0} \\ f_0 + f_1 + f_2 \\ \cancel{f_2} \end{pmatrix} \tag{42}$$

Hence,

$$U_1 = \frac{f_0 + f_1 + f_2}{b_0 + d_1 + l_1} \tag{43}$$

From (32) we see that $b_0 = \frac{1}{6}h_0, d_1 = \frac{1}{3}h_0 + \frac{1}{3}h_1, l_1 = \frac{1}{6}h_1$
(Note: reverting back to $h_i = x_{i+1} - x_i$), therefore

$$U_1 = \frac{2(f_0 + f_1 + f_2)}{h_0 + h_1} \tag{44}$$

$\underline{N = 3}$

$$\begin{pmatrix} d_0 & b_0 & 0 & 0 \\ l_0 & d_1 & b_1 & 0 \\ 0 & l_1 & d_2 & b_2 \\ 0 & 0 & l_2 & b_3 \end{pmatrix} \begin{pmatrix} U_0 \\ U_1 \\ U_2 \\ U_3 \end{pmatrix} = \begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \end{pmatrix} \tag{45}$$

becomes,

$$\begin{pmatrix} \cancel{d_0} & \cancel{b_0} & \emptyset & \emptyset \\ \cancel{d_0 + l_0} & b_0 + d_1 & b_1 & \emptyset \\ \emptyset & l_1 & l_2 + d_2 & \cancel{b_2 + b_3} \\ \emptyset & \emptyset & \cancel{l_2} & \cancel{b_3} \end{pmatrix} \begin{pmatrix} \cancel{U_0} \\ U_1 \\ U_2 \\ \cancel{U_3} \end{pmatrix} = \begin{pmatrix} \cancel{f_0} \\ f_0 + f_1 \\ f_2 + f_3 \\ \cancel{f_3} \end{pmatrix} \tag{46}$$

Now we have a 2×2 matrix, therefore we can easily rewrite (46) as,

$$\begin{pmatrix} U_1 \\ U_2 \end{pmatrix} = \frac{1}{(l_2 + d_2)(b_0 + d_1) - b_1 l_1} \begin{pmatrix} l_2 + d_2 & -b_1 \\ -l_1 & b_0 + d_1 \end{pmatrix} \begin{pmatrix} f_0 + f_1 \\ f_2 + f_3 \end{pmatrix}. \tag{47}$$

We are now able to write an explicit formula for each of the unknowns,

$$U_1 = \frac{(l_2 + d_2)(f_0 + f_1) - b_1(f_2 + f_3)}{(l_2 + d_2)(b_0 + d_1) - b_1 l_1} \tag{48}$$

and

$$U_2 = \frac{(b_0 + d_1)(f_2 + f_3) - l_1(f_0 + f_1)}{(l_2 + d_2)(b_0 + d_1) - b_1 l_1} \tag{49}$$

where $b_0 = \frac{1}{6}h_0, b_1 = \frac{1}{6}h_1, d_1 = \frac{1}{3}h_0 + \frac{1}{3}h_1, d_2 = \frac{1}{3}h_1 + \frac{1}{3}h_2, l_1 = b_0, l_2 = b_1$. After simplifying and manipulating we get

$$U_1 = 6\left[\frac{(3h_1 + 2h_2)(f_0 + f_1) - h_1(f_2 + f_3)}{(3h_1 + 2h_2)(3h_0 + 2h_1) - h_0 h_1}\right] \tag{50}$$

and

$$U_2 = 6\left[\frac{(3h_0 + 2h_1)(f_2 + f_3) - h_0(f_0 + f_1)}{(3h_1 + 2h_2)(3h_0 + 2h_1) - h_0 h_1}\right]. \tag{51}$$

Remember that we are looking at three cells on a uniform mesh and therefore $h_i = \frac{2}{3}$. The function that we are considering is symmetric, causing the load vector to be also symmetric and therefore $(f_0 + f_1) = (f_2 + f_3)$. For this reason we only need to simplify (50), which becomes

$$U_1 = \frac{15(f_0 + f_1) - 3(f_2 + f_3)}{8} = \frac{3}{2}(f_0 + f_1). \tag{52}$$

We can thus construct the approximation once we have calculated the load vector.

Running the program we are able to get the load vector

$$(0.190793 \quad 0.595505 \quad 0.595505 \quad 0.190793), \tag{53}$$

which gives $U = 1.179447$.

This produces the graph

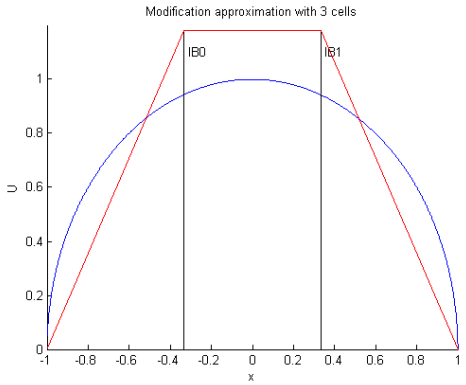


Figure 4: Approximation with three cells using (52).

2.3.6 Modification Results

The remainder of the graphs are calculated using the Thomas Algorithm. We will show the effect on the approximation when the end points are zero.

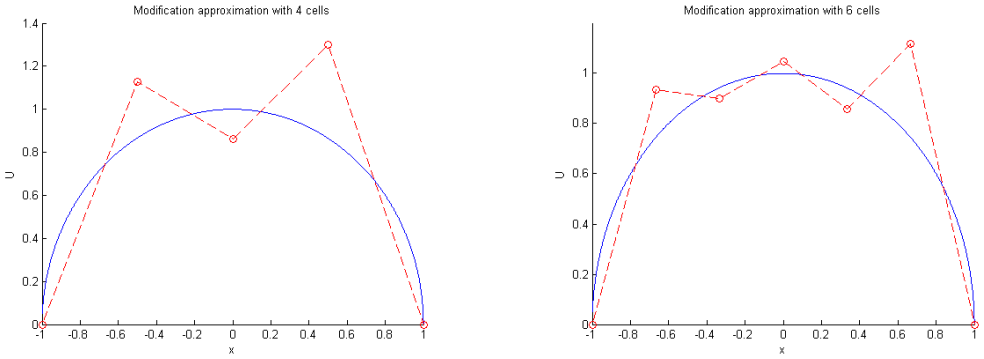


Figure 5 - Left: Modified approximation with four cells,
Right: Modified approximation with six cells.

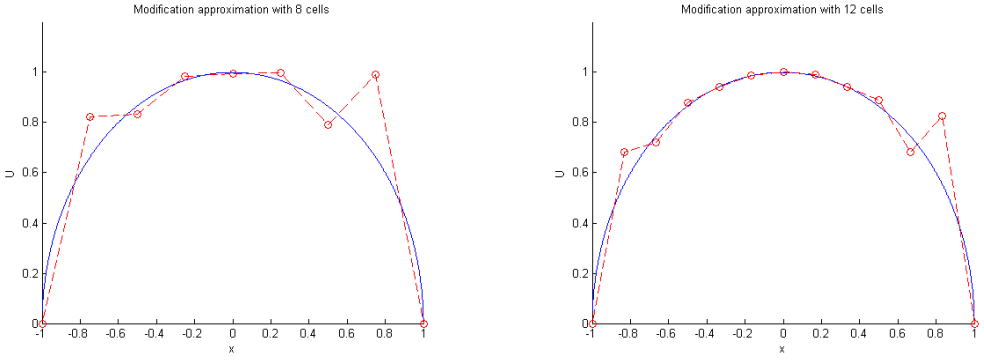


Figure 6 - Left: Modified approximation with eight cells,
Right: Modified approximation with twelve cells.

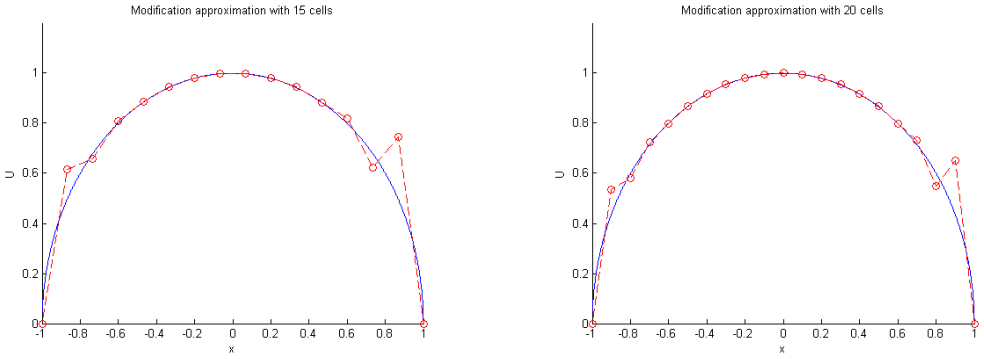


Figure 7 - Left: Modified approximation with fifteen cells,
Right: Modified approximation with twenty cells.

Again there was a difficulty with the program and hence we could not investigate the errors.

3 Best fits on adjustable meshes

3.1 Cellwise Approximation

Having obtained a procedure to produce a best fit in one cell, it would be desirable if we could use the same technique for larger problems. As we saw in the piecewise constant case, one can just obtain the best approximation for a larger system simply by calculating the integration for every mesh partition. Caution must be made when doing this because in the single cell example we were examining the interval $[0, 1]$ which gave the interval width as one.

In the piecewise linear case the system that we need to solve in the interval $[x_i, x_{i+1}]$ is

$$\begin{pmatrix} U_i \\ U_{i+1} \end{pmatrix} = \frac{1}{h_{i+1}} \begin{pmatrix} 4 & -2 \\ -2 & 4 \end{pmatrix} \begin{pmatrix} F_i \\ F_{i+1} \end{pmatrix}, \tag{54}$$

where $h_{i+1} = x_{i+1} - x_i$.

The vector on the right hand side of (54) is the load vector. From (23) we know that each element is an integral of the product between the basis function and f. As mentioned before the test functions ϕ_i are discontinuous at the point x_i. However, now that we are only concerned with one cell at a time we only need to consider the appropriate section of ϕ_i (remembering ϕ_i is non zero in the interval $[x_{i-1}, x_{i+1}]$). Hence, in (54) the load vector reduces to

$$\begin{pmatrix} \int_{x_{i-1}}^{x_{i+1}} \phi_i f \, dx \\ \int_{x_{i-1}}^{x_{i+1}} \phi_{i+1} f \, dx \end{pmatrix} = \begin{pmatrix} \int_{x_i}^{x_{i+1}} \frac{x_{i+1} - x}{h_{i+1}} f \, dx \\ \int_{x_i}^{x_{i+1}} \frac{x - x_{i-1}}{h_{i+1}} f \, dx \end{pmatrix} = \frac{1}{h_{i+1}} \begin{pmatrix} \int_{x_i}^{x_{i+1}} (x_{i+1} - x) f \, dx \\ \int_{x_i}^{x_{i+1}} (x - x_{i-1}) f \, dx \end{pmatrix} \tag{55}$$

Using (54) we are able to obtain two equations for the U values,

$$U_i = \frac{2}{h_{i+1}^2} \left(2 \int_{x_i}^{x_{i+1}} (x_{i+1} - x) f \, dx - \int_{x_i}^{x_{i+1}} (x - x_{i-1}) f \, dx \right) \tag{56}$$

and

$$U_{i+1} = \frac{2}{h_{i+1}^2} \left(-2 \int_{x_i}^{x_{i+1}} (x_{i+1} - x) f \, dx + \int_{x_i}^{x_{i+1}} (x - x_{i-1}) f \, dx \right) \tag{57}$$

On close examination of (56) and (57) we see that there are two main integrations that need to be performed. They are, $\int f \, dx$ and $\int xf \, dx$. Once we create functions for these particular integrands we can use our quadrature procedure to obtain the correct values.

It will also be useful to construct the line equation for the best fit. We can do this by using basic geometry once we have the two end points. The Cellwise procedure solves equations (56) and (57) on each cell and then uses the line information for each approximation to display the best piecewise linear fit for the entire region.

Algorithm - This approximation method can be summarised by the following algorithm,

- Choose the number of cells N
- For each cell do
    1. Calculate best fit, obtain U values
    2. Use U values to construct line information
    3. Plot linears on their respective subinterval
- End For

3.1.1 Cellwise Results

The following graphs illustrate the Cellwise algorithm. These graphs were produced for the function $f(x) = (1 - x^2)^{\frac{1}{2}}$. To demonstrate the effect of increasing the cell number it was decided to run the program when $N = 2, 4, 8, 16$.

Piecewise Constants - We begin by looking at the Piecewise Constant case.

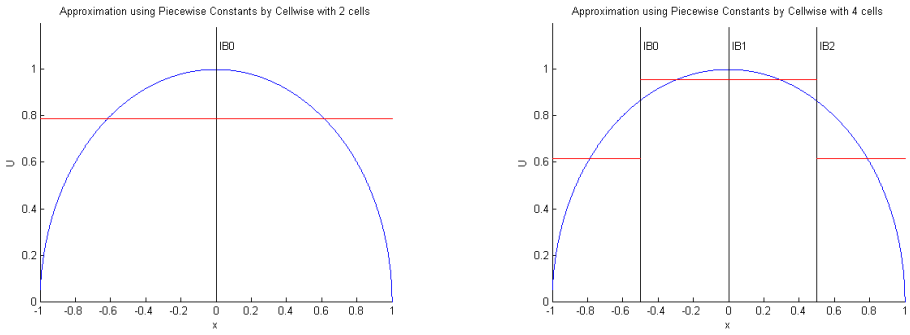


Figure 8 - Left: Constant Cellwise approximation with two cells,
Right: Constant Cellwise approximation with four cells.

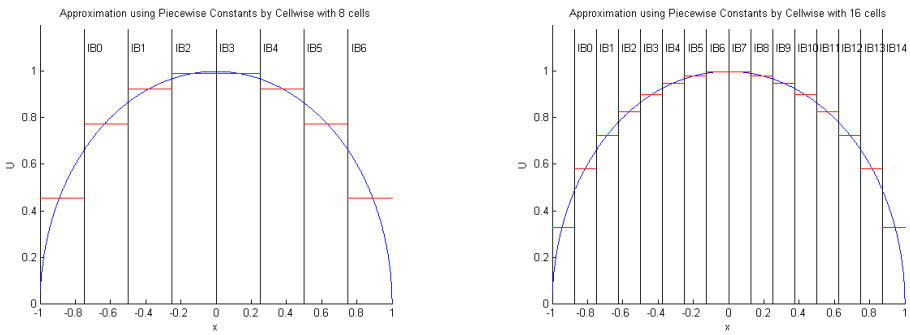


Figure 9 - Left: Constant Cellwise approximation with eight cells,
Right: Constant Cellwise approximation with sixteen cells.

As well as producing the graphs, the program also displayed the L_2 error for each case. We are able to investigate the rate of convergence by examining the ratio of two successive errors, this is illustrated in the table below.

N	e_N	$\frac{e_{N+1}}{e_N}$
2	0.30727	0.6474
4	0.198921	0.5878
8	0.118921	0.5634
16	0.068874	

Table 1 - Errors for Piecewise Constants using Cellwise.

It can be readily seen that the approximation is improved by increasing the number of cells. However, even with sixteen cells the error calculated using the L_2-norm is still relatively big.

Piecewise Linears

Let us now focus on the Piecewise Linear case and compare the errors with those above to see the effect of increasing the order of the approximating polynomial. These graphs were produced with the same function f.

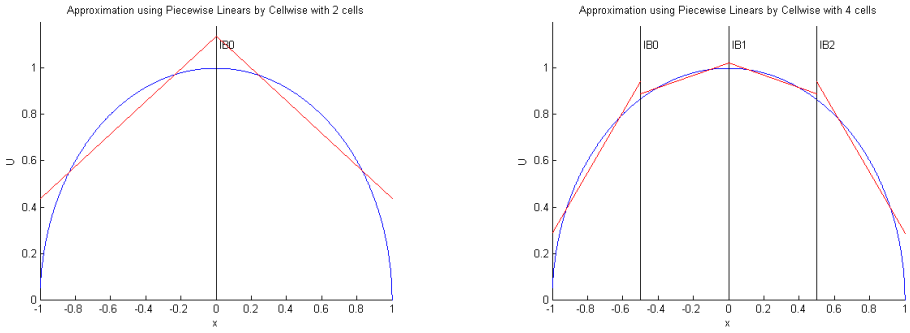


Figure 10 - Left: Linear Cellwise approximation with two cells,
Right: Linear Cellwise approximation with four cells.

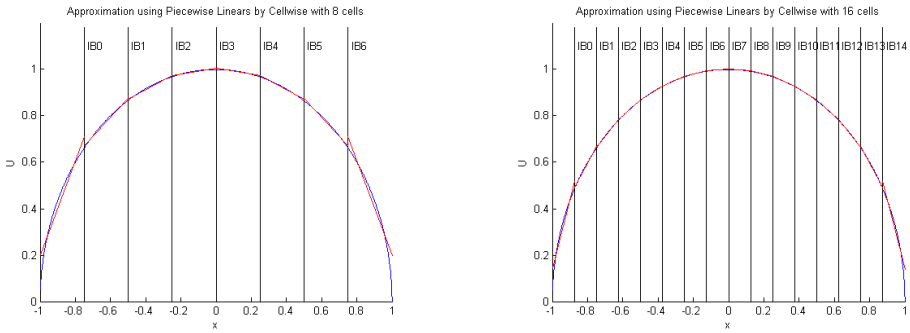


Figure 11 - Left: Linear Cellwise approximation with eight cells,
Right: Linear Cellwise approximation with sixteen cells.

We examine the piecewise linear error rate using the same technique as before.

N	e_N	$\frac{e_{N+1}}{e_N}$
2	0.111272	0.4329
4	0.048175	0.4634
8	0.022324	0.4812
16	0.010743	

Table 2 - Errors for Piecewise Linears using Cellwise.

Comparing Figures 9 and 11 we are able to see that it is a better approximation and gives a smaller error. For example, we see that taking only two cells using Linears gives a similar magnitude of error to if we use eight cells using piecewise constants.

3.1.2 Modified Cellwise Results

We can also apply the Cellwise procedure when we fix an endpoint. We use the equations found in 2.3.2 on the first and last cell. We have the following graphs.

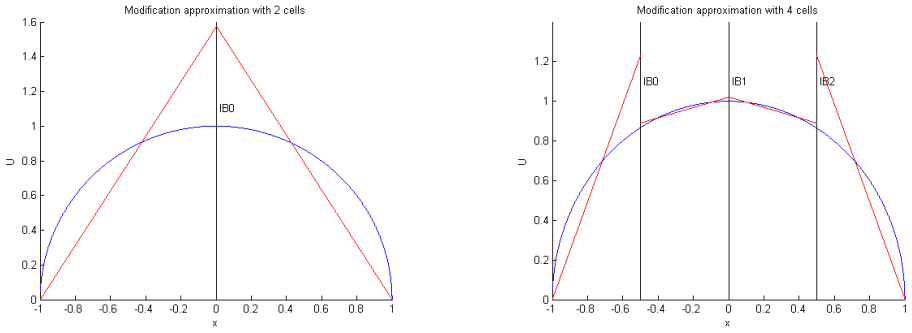


Figure 12 - Left: Modified Cellwise approximation with two cells,
Right: Modified Cellwise approximation with four cells.

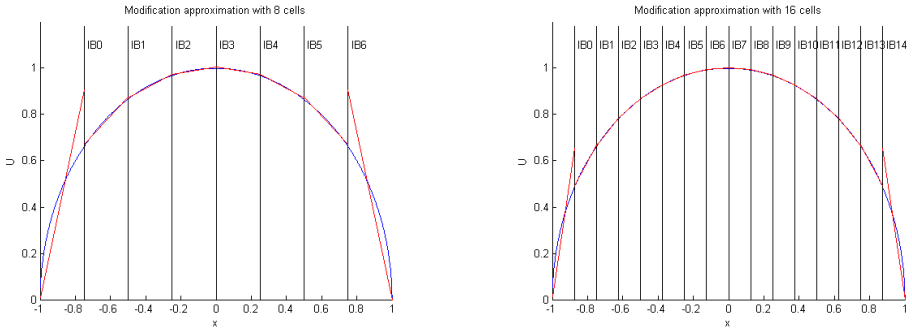


Figure 13 - Left: Modified Cellwise approximation with eight cells,
Right: Modified Cellwise approximation with sixteen cells.

We will now present the errors to compare with Table 2.

N	e_N	$\frac{e_{N+1}}{e_N}$
2	0.37319	0.4634
4	0.172918	0.4836
8	0.0836227	0.4922
16	0.0411605	

Table 3 - Errors for Piecewise Linears using Cellwise with fixed endpoints.

From Figure 12 and 13 we can see a large discontinuity at the first and last internal nodes. This causes the error to be increased.

3.2 Derivation of best fit equations with adjustable nodes

To attempt to improve a continuous piecewise linear approximation we can move the nodes or mesh points in order to minimise the L_2-error. In this section we will investigate how a variable mesh can be constructed so that (1) is minimised in this way.

To do this we look at the variation of equation (1). Following Baines, we examine this when the approximation is discontinuous at the nodes and internal points X_i vary.

It is shown in [1] that the first variation of (1) is then

$$\delta \int_a^b (f(x) - U)^2 \, dx = \int_a^b 2(f(x) - U)\delta U \, dx + \sum_i [(f(x) - U)^2]_{X_i} \delta X_i \tag{58}$$

where the square brackets denote the jump in the argument.
U is now expanded as $\sum_i \sum_{ik=1}^{2} U_{ik}\psi_{ik}$ (where $\psi_i = \phi_i$ in the piecewise linear case or $\psi_i = \pi_i$ in the piecewise constant case, so that $\delta U = \sum_i \sum_{ik=1}^{2} \delta U_{ik}\psi_{ik}$), giving

$$\delta \int_a^b (f(x) - U)^2 \, dx = \sum_i \sum_{ik=1}^{2} \int_a^b 2(f(x) - U)\delta U_j \psi_{ik} \, dx + \sum_i [(f(x) - U)^2]_{X_i} \delta X_i \tag{59}$$

The error is a minimum when the first variation vanishes. We can get closer to the mnimum by setting any term to zero. The minimisation can therefore be sought iteratively in two stages until the iteration converges.

1. Fix X_i and solve $\int_a^b (f - U)\psi_{ik} \, dx = 0 \ \ \forall i$
2. Fix U_i and solve $[(f - U)^2]_{X_i} = 0 \ \ \forall i$

The first of these is exactly the Cellwise best fit procedure of the previous Section.

To investigate the second, let U_L be the U value at the boundary coming from the left and U_R be the value from the right. We can then formulate the following equations for the second minimisation.

$$\begin{aligned}(U - f(x))_L^2 &= (U - f(x))_R^2 \\ (U - f(x))_L^2 - (U - f(x))_R^2 &= 0 \\ \big((U - f(x))_L - (U - f(x))_R\big)\big((U - f(x))_L + (U - f(x))_R\big) &= 0\end{aligned}$$

It follows that either

$$(U - f(x))_L - (U - f(x))_R = 0 \Rightarrow (U - f(x))_L = (U - f(x))_R \Rightarrow U_L = U_R \tag{60}$$

since $f_L = f_R$ (Intersection), or

$$(U - f(x))_L + (U - f(x))_R = 0 \Rightarrow U_L + U_R - 2f \tag{61}$$

$f_L \neq f_R$ (Averaging).

3.3 Piecewise constants

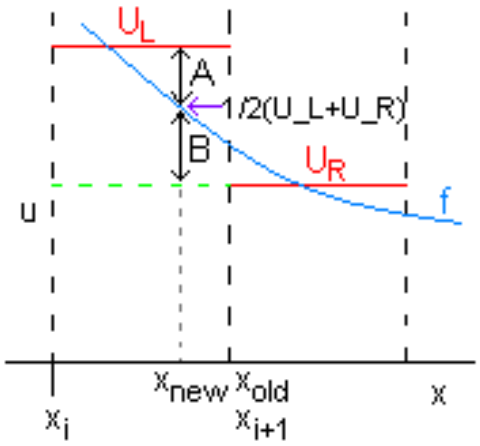
We now investigate how it is possible to minimise the norm in the piecewise constant case when the nodes are adjustable. This is achieved by changing the location of each mesh point so that (61) holds. The initial approximation is constructed using the Cellwise procedure on an initial mesh.

3.3.1 Averaging

Basically we improve the interior mesh point location by calculating the position on the horizontal axis where the distance between the function and the cell approximation coming from left or right is the same. Let us examine the case for the i^{th} boundary. Piecewise constants are being considered and therefore let us denote the U value for the cell approximation on the left of the boundary as U_L. Similarly, U_R corresponds to the right cell. This allows us to mathematically describe the aim of this procedure. We require to solve

$$|U_L - f| = |U_R - f| \tag{62}$$

Once the procedure has traversed through the region, a new vector of x coordinates is obtained. With these new points, the approximation is recalculated before traversing the region again. This iteration is terminated when the locations only differ a small amount.



Theory

Let U_L denote the magnitude of the constant approximation in the $(i-1)$th cell and U_R in the ith cell. In general one of the differences between the approximation and the function will be negative. If we closely inspect the diagram (a blow up of the boundary) we can see that the sign of the lower difference is always the opposite sign to the upper difference. Although this may not occur at the boundary, it will if the approximation is extrapolated. We have,

$$\begin{aligned} |U_L - f| &= |U_R - f| \\ U_L - f &= -(U_R - f) \\ U_L - f &= f - U_R \\ f &= \frac{1}{2}(U_L + U_R) \end{aligned}$$

The right hand side of the last equation is known and therefore this is just an inverse problem.

If we, for an instance, consider the line $f = \frac{1}{2}(U_L + U_R)$ to be the x -axis, then finding the point where the curve crosses this line is the same as finding a polynomial root. One simple way of executing this procedure is to use the *Bisection* method. The constant approximation is seldom zero and therefore the method requires a little modification. To make full use of the Bisection method, we require that the function must change sign when it crosses zero. This is easily achieved by defining the new function,

$$g(x) = f(x) - \frac{1}{2}(U_L + U_R). \tag{63}$$

On close inspection of (63), we can see that there is a change of sign at the intersection.

The Bisection method works by examining the signs of the two boundaries along with the interval midpoint. The sign of (63) evaluated at the midpoint determines which half of the interval the intersection is in.

This is an iterative method and thus the interval is divided into two each time until the intersection is found.

Let x_L and x_R denote the x position of the interval boundaries.

This method can be implemented by following the algorithm given below,

- Repeat algorithm until interval width is less than a tolerance.
    1. Calculate midpoint; $M = \frac{1}{2}(x_L + x_R)$.
    2. If $g(x_L).g(x_M) > 0$ then
       intersection lies in the *right* half.
       Thus, $x_L = x_M$ (location of midpoint).
    3. However, if $g(x_L).g(x_M) < 0$ then
       intersection lies between the left boundary and the midpoint.
       Hence, $x_R = x_M$.
- End Loop

The Bisection method enables us to locate the new mesh point in a relatively straightforward manner. Once all the new points have been calculated, we examine the displacement of each node. The Averaging procedure will continue if at least one displacement is greater than a tolerance. This ensures that we have found the optimal mesh to minimise the L_2-error.

3.3.2 Averaging results

Let us examine the function

$$f = (1 - x^2)^{\frac{1}{2}}, \tag{64}$$

over the interval $[-1, 1]$ and see how the point locations change.

For clarity reasons, the graph will only show the initial (green) and final (red) approximation. The graphs will also clearly indicate how each mesh point moves. To aid with the analysis of the results, the program will display the number of iterations that are executed along with the initial and final error. To give the reader a flavour of how the points move, we will only consider the cases when $N = 3, 4, 6, 7$. If we look at the case when $N = 2$, the piecewise approximation is always connected at $x = 0$ due to the symmetry of the curve $(1 - x^2)^{\frac{1}{2}}$. The mesh point displacement tolerance is 0.0001.

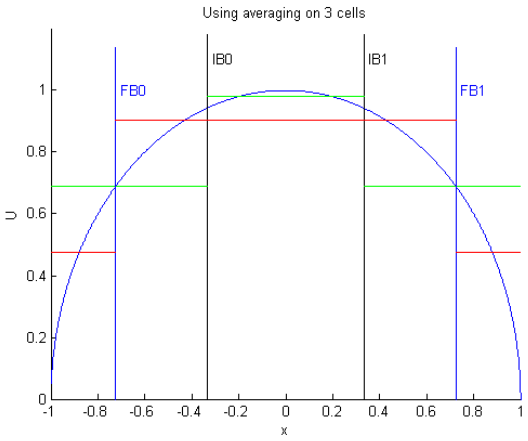


Figure 14: Construction of the new mesh using the averaging procedure with three cells.

The algorithm took 12 iterations to form the new mesh.
The initial error = 0.242554.
The final error = 0.158911.

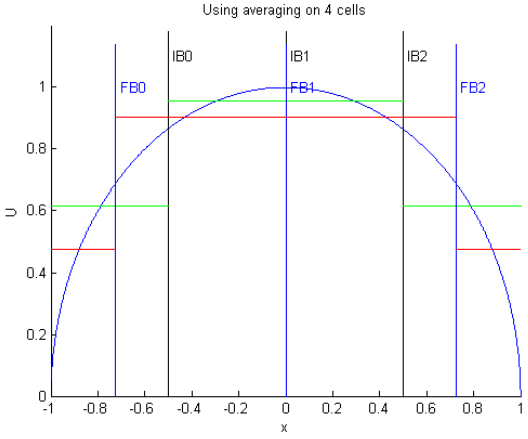


Figure 15: Construction of the new mesh using the averaging procedure with four cells.

The algorithm took 13 iterations to form the new mesh.
The initial error = 0.198921.
The final error = 0.159287.

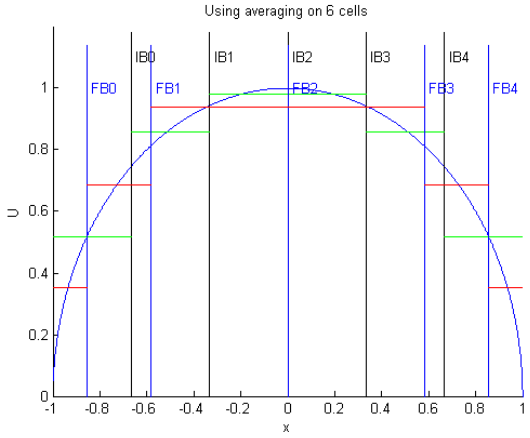


Figure 16: Construction of the new mesh using the averaging procedure with six cells.

The algorithm took 22 iterations to form the new mesh.
The initial error = 0.146777.
The final error = 0.107181.

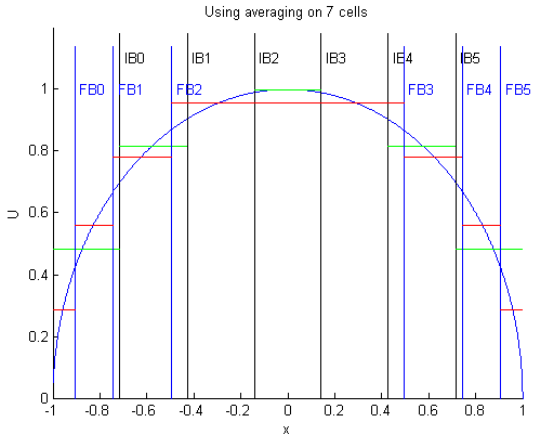


Figure 17: Construction of the new mesh using the averaging procedure with seven cells.

The algorithm took 40 iterations to form the new mesh.
The initial error = 0.130066.
The final error = 0.0807251.

Discussion

As one can see the point moves outwards faster where the curve has a higher derivative. In each case the approximating error was decreased showing that the procedure has the desired effect. The number of interations increases with the number of points, meaning that there will be a computational limit on how many points it is practical to use.

3.4 Piecewise Linears

The solution in each cell is represented by a straight line with varying slopes and we can produce a better mesh by one of two different techniques. Using Figure 2 in [1], we see that there are two possible cases that can arise when we use piecewise linears.

The first case is when the gradients in the cell approximations are significantly different from one another, providing an intersection close to the mesh point in question. We will refer to this method as 'intersection'. However if this gradient criterion fails, i.e. the two neighbouring best fits are virtually parallel, we use a method which resembles the averaging procedure outlined in section 3.2. The new mesh is formed by using one or other of these techniques.

It was decided to use the averaging procedure only when the gradient criterion fails. An alternative way of executing this would be to execute both methods and use the one that produces the smallest displacement (or more usefully the greatest norm reduction). Due to the time constraint, one could not fullly compare the difference between these two methods.

3.4.1 Intersection

Let us look at the ith cell boundary. After constructing the best discontinuous linear fit cellwise for the entire region, the approximation in each cell will be represented by a line of the form $mx + c$. Therefore we have

$$m_i x + c_i \quad \text{and} \quad m_{i+1} x + c_{i+1}. \tag{65}$$

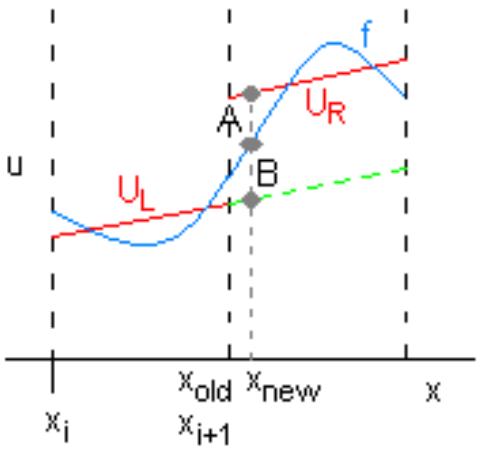
We can obtain the intersection of these two lines as

$$x = \frac{c_{i+1} - c_i}{m_i - m_{i+1}}. \tag{66}$$

This algorithm traverses through the region calculating the intersection at each internal node, forming the new mesh. The best fits are then recalculated using (56) and (57), to produce another linear approximation. The procedure is terminated when all points converge to their optimal location. We do this by comparing the previous mesh with the new constructed one to see whether each node displacement is less than a tolerance.

3.4.2 Linear Averaging

When it is not feasible to calculate the intersection (when the lines are virtually parallel) we shall use the averaging procedure, as in the case for piecewise constants.



It is readily seen that we have exactly the same problem as in Section 3.3.1 because we have two distinct values at the boundary and we are attempting to move the point where the function curve intersects the average of these values. This is also solved by using the Bisection Method. Hence we have the equation

$$f(x_{\text{new}}) = \frac{1}{2}\Big(u_L(x_{\text{new}}) + u_R(x_{\text{new}})\Big). \tag{67}$$

We assume that either the intersection or the averaging procedure will produce a point that is in the interval $[x_{i-1}, x_{i+1}]$.

3.4.3 Linear Results

We use the same cases as we used in the previous section and the same tolerance so that we can easily compare results.

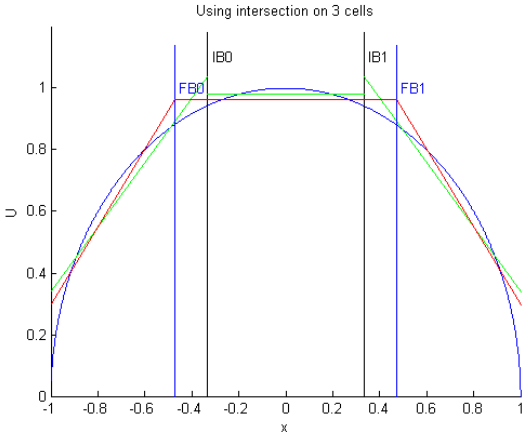


Figure 18: Construction of the new mesh using the intersection procedure with three cells.

The algorithm took 15 iterations to form the new mesh.
The initial error = 0.0674806.
The final error = 0.0605073.

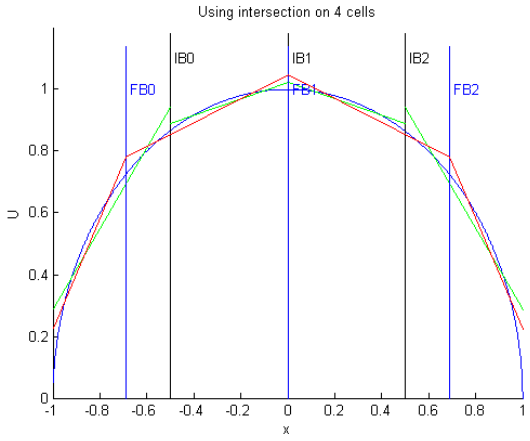


Figure 19: Construction of the new mesh using the intersection procedure with four cells.

The algorithm took 23 iterations to form the new mesh.
The initial error = 0.0481746.
The final error = 0.0383336.

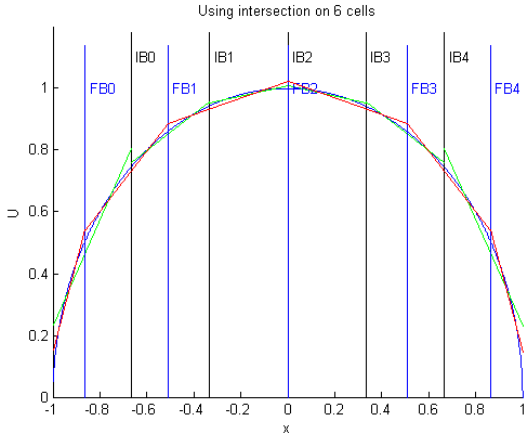


Figure 20: Construction of the new mesh using the intersection procedure with six cells.

The algorithm took 42 iterations to form the new mesh.
The initial error = 0.0305351.
The final error = 0.0194741.

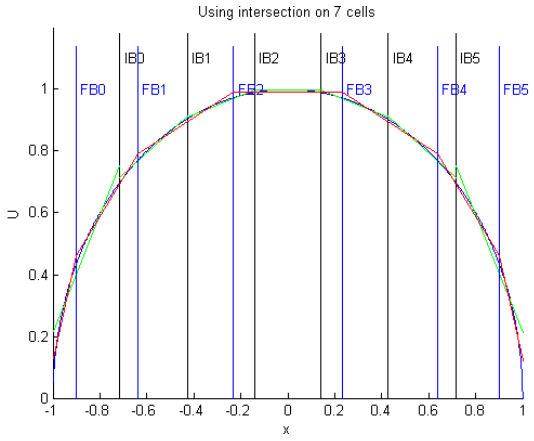


Figure 21: Construction of the new mesh using the intersection procedure with seven cells.

The algorithm took 53 iterations to form the new mesh.
The initial error = 0.0257943.
The final error = 0.0149098.

Discussion

In every example the error between the initial and final approximations decreases, illustrating that the method is indeed seeking to minimise the L_2-error. It can be seen that in the final approximation (red) for piecewise linears involving only the intersection construction, the discontinuous approximation is connected at the cell boundaries (since we have removed the jumps), yielding a continuous approximation.

With the piecewise constant approximations, the points are moved to where the derivative of the function is large, while with the piecewise linear approximations, the points are moved to where the second derivatives are large.

3.5 Carey and Dinh

As discussed in the introduction, Carey and Dinh introduced a Grading Function to help with mesh redistribution. They focused on two-point boundary value problems.

Carey and Dinh (1985) [5] consider the error norm (squared)

$$|e|_m^2 = \int_a^b (e^{(m)})^2 \, dx, \tag{68}$$

where $e = f - u$.

After several manipulations they arrive at the Grading Function

$$\xi = \frac{\int_a^x (u^{(k+1)})^{2/(2(k+1-m)+1)}\, dx}{\int_a^b (u^{(k+1)})^{2/(2(k+1-m)+1)}\, dx}, \tag{69}$$

where k denotes the degree of the approximating polynomial in each cell.

Remembering that we are looking at the L_2-norm, we have $m = 0$.

Thus, for piecewise constants ($k = 0$) (69) reduces to

$$\xi_0 = \frac{\int_a^x (u')^{\frac{2}{3}}\, dx}{\int_a^b (u')^{\frac{2}{3}}\, dx} \tag{70}$$

and similarly when $k = 1$ we look at piecewise linears we get

$$\xi_1 = \frac{\int_a^x (u'')^{\frac{2}{5}}\, dx}{\int_a^b (u'')^{\frac{2}{5}}\, dx}. \tag{71}$$

The subscripts in (70) and (71) allows us to distinguish between the cases.

Equivalently, following [1]

$$\xi_0(x) \propto (u')^{\frac{2}{3}} \tag{72}$$

$$\xi_1(x) \propto (u'')^{\frac{2}{5}}. \tag{73}$$

3.5.1 Practical Discussion

To test whether

$$\varepsilon_0'(x) \propto |f'(x)|^{\frac{2}{3}}, \tag{74}$$

and

$$\varepsilon_1'(x) \propto |f''(x)|^{\frac{2}{5}}, \tag{75}$$

hold for the meshes that we have constructed we integrate the right hand side of these equations over the moved mesh. If the statements are true then the magnitude of each integral should be the same (at least asymptotically).

We can simplify the problem by removing the modulus sign. For piecewise linears this is achieved by considering only *convex* functions (ensuring that the second derivative does not change sign). While when we are looking at the constant case, the convex function must also be *monotonic*. Hence, (72) is examined only for $(0, X_N)$.

3.5.2 Results

These are the vectors of the integral quantities over the moved mesh.

Constants - Looking at $[0, 1]$

Two Cells:

$$(0.388080.456362)$$

Four Cells:

$$(0.193322, 0.213026, 0.215243, 0.232069)$$

Eight Cells:

$$(0.0958847, 0.105681, 0.106335, 0.10685, 0.107509, 0.108335, 0.109231, 0.118168)$$

Twelve Cells:

$$(0.0630868, 0.0694965, 0.0699234, 0.0704597, 0.0708207, 0.071439,\\ 0.0718315, 0.0723914, 0.0727605, 0.073264, 0.0739493, 0.0799937)$$

Sixteen Cells:

$$(0.0463427, 0.051065, 0.0514491, 0.0517464, 0.0521589, 0.0525083, 0.0529767, 0.0534649,\\ 0.0537734, 0.0542632, 0.0546344, 0.055243, 0.0556285, 0.0562839, 0.0570835, 0.0614792)$$

Linears

Two Cells:

$$(1.57085, 1.57085)$$

Four Cells:

$$(0.89524, 0.775711, 0.775711, 0.89524)$$

Eight Cells:

$$(0.4901, 0.423745, 0.416545, 0.414365, 0.414365, 0.416545, 0.423745, 0.4901)$$

Twelve Cells:

$$(0.339009, 0.293147, 0.287836, 0.285603, 0.284342, 0.283732,\\ 0.283732, 0.284342, 0.285603, 0.287836, 0.293147, 0.339009)$$

Sixteen Cells:

$$(0.259924, 0.224775, 0.220624, 0.218722, 0.217408, 0.216415, 0.215725, 0.215368,\\ 0.215368, 0.215725, 0.216415, 0.217408, 0.218722, 0.220624, 0.224775, 0.259924)$$

Ignoring the endpoints, the vector elements hardly vary in magnitude. This illustrates an equidistribution feature of the solution.

4 Similarity solutions and best fits

Having investigated a successful way for computing the best fit approximation of a function with adjustable nodes, we turn our attention to using these techniques to investigate the best fit properties of the self-similar solution of a Partial Differential Equation (PDE) with a moving boundary. We will first introduce the idea of scale invariance and examine the effects on the approximation u when the solution evolves with time.

4.1 Porous Medium Equation (PME)

This PME is used in many different applications. One area is modelling flow of a liquid through a porous media, such as certain types of rocks (see [6]).

Let $u = u(x,t)$ be the function that we are interested in. We can write the Porous Medium Equation (PME) as

$$\frac{\partial u}{\partial t} = \nabla \cdot (u^m \nabla u). \tag{76}$$

In this project we are only interested in the $1D$ case and therefore we can reduce this equation to

$$u_t = (u^m u_x)_x. \tag{77}$$

We apply the boundary condition $u = 0$ which induces a moving boundary.

4.2 Scale Invariance

When a mathematical model is constructed it will represent physical quantities, such as length. These can be measured in different units, either metres or kilometres. To avoid any difficulties, one scales each quantity depending on the fundamental units. This idea is a very useful technique in mathematical modelling. Fundamentally each variable becomes a ratio of the quantity being considered and the basic unit of measurement. The result is dimensionless and therefore can be used in the model without being dependent on the measurement system.

In a similar way if we are investigating a solution to a PDE, we can express it independently of any units. We will examine this further.

A partial differential equation is referred to as being scale Invariant if after scaling variables the equation does not change ([3], [4]).

Defining an arbitrary variable λ to be a scaling parameter, we can mathematically describe this technique by defining the following transformations,

$$u = \lambda^\gamma \bar{u}, \quad x = \lambda^\beta \bar{x}, \quad t = \lambda^\alpha \bar{t} \tag{78}$$

where $\alpha, \beta, \gamma \in \mathbb{R}$.

We next rewrite (77) using these new variables, to get an expression enabling us to find conditions on α, β, γ so that the equation is scale Invariant, i.e.

$$\bar{u}_{\bar{t}} = (\bar{u}^m \bar{u}_{\bar{x}})_{\bar{x}}. \tag{79}$$

Examining the LHS of (77) and manipulating the variables we obtain

$$u_t = \frac{\partial u}{\partial t} = \frac{\partial(\lambda^\gamma \bar{u})}{\partial(\lambda^\alpha \bar{t})} = \lambda^{(\gamma-\alpha)} \frac{\partial \bar{u}}{\partial \bar{t}}. \tag{80}$$

Next we find an expression for the RHS of (77) using a similar procedure. Due to the complexity of the equation, it was decided to evaluate the bracket before differentiating it with respect to x. Thus, let $\eta = u^m u_x$.

$$\eta = u^m u_x = (\lambda^\gamma \bar{u})^m \frac{\partial(\lambda^\gamma \bar{u})}{\partial(\lambda^\beta \bar{x})} = \lambda^{m\gamma} \bar{u}^m \lambda^{\gamma-\beta} \frac{\partial \bar{u}}{\partial \bar{x}} = \lambda^{(m\gamma+\gamma-\beta)} \bar{u}^m \frac{\partial \bar{u}}{\partial \bar{x}}$$

It follows that

$$\frac{\partial \eta}{\partial x} = \frac{\partial \eta}{\partial(\lambda^\beta \bar{x})} = \lambda^{\left((m+1)\gamma-\beta-\beta\right)} \frac{\partial(\bar{u}^m \frac{\partial \bar{u}}{\partial \bar{x}})}{\partial \bar{x}}$$

meaning

$$(u^m u_x)_x = \lambda^{\left((m+1)\gamma-2\beta\right)} (\bar{u}^m \bar{u}_{\bar{x}})_{\bar{x}} \tag{81}$$

Equating (80) and (81) we obtain the full equation in terms of the scaled variables.

$$\lambda^{(\gamma-\alpha)} \frac{\partial \bar{u}}{\partial \bar{t}} = \lambda^{\left((m+1)\gamma-2\beta\right)} (\bar{u}^m \bar{u}_{\bar{x}})_{\bar{x}} \tag{82}$$

We now cancel the powers of λ by forcing them to equal one another,

$$\gamma - \alpha = (m+1)\gamma - 2\beta. \tag{83}$$

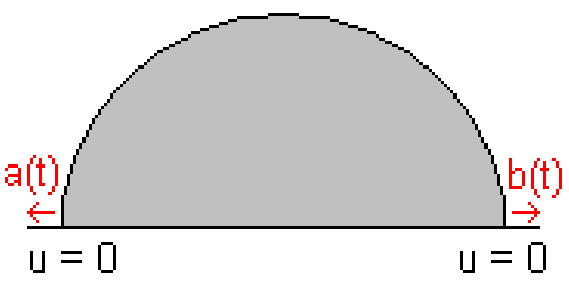
Without loss of generality we can choose $\alpha = 1$, and substituting this into (83) we can obtain an expression that only includes β and γ.

$$\gamma - 1 = (m+1)\gamma - 2\beta. \tag{84}$$

We now incorporate the boundary condition. A consequence of the boundary condition $u = 0$ is conservation of mass, as we now show. Let us examine the rate of change of the total mass integral, i.e.

$$\frac{d}{dt} \int_{a(t)}^{b(t)} u(t,x)\, dx, \tag{85}$$

where $a(t)$ and $b(t)$ are movable boundary points.



We have three variables in (85), namely $a(t), b(t), u(t)$. To obtain the derivative of this equation we in turn fix two and vary the third, obtaining

$$\frac{d}{db}\int_{a(t)}^{b(t)} u(t)\ dx.\frac{db}{dt} + \frac{d}{da}\int_{a(t)}^{b(t)} u(t)\ dx.\frac{da}{dt} + \frac{d}{dt}\int_{a(t)}^{b(t)} u(t)\ dx \tag{86}$$

Evaluating each integral we reach

$$u(b(t))\frac{db}{dt} - u(a(t))\frac{da}{dt} + \int_{a(t)}^{b(t)} \frac{\partial u}{\partial t}\ dx \tag{87}$$

Using the boundary conditions, as illustrated in the diagram, we see that the first two terms of (87) equate to zero. This leads us to

$$\int_{a(t)}^{b(t)} \frac{\partial u}{\partial t}\ dx = \int_{a(t)}^{b(t)} (u^m u_x)_x\ dx. \tag{88}$$

The RHS integral just becomes $u^m u_x$ between the limits. In this integral, therefore,

$$u^m u_x|_{a(t)}^{b(t)} = 0, \tag{89}$$

since $u(a) = u(b) = 0$.

We can deduce from (85) that

$$\int_{a(t)}^{b(t)} u(t,x)\ dx = \text{constant}, \tag{90}$$

meaning that the area under the graph remains constant.

Under scaling, (90) becomes

$$\lambda^{\gamma+\beta}\int \bar{u}\ d\bar{x} = \text{constant}. \tag{91}$$

This gives us another condition for the scaling variables α, β, γ, namely,

$$\gamma + \beta = 0. \tag{92}$$

Combining (84) and (92), it is possible to find a scaling which satisfies the PDE and conservation of mass. ¿From (92) we have $\beta = -\gamma$, substituting this into (84) we get,

$$-\beta - 1 = -(m+1)\beta - 2\beta \Rightarrow \beta - 1 = -m\beta - \beta \Rightarrow (2+m)\beta = 1$$

Hence,

$$\beta = \frac{1}{2+m} \tag{93}$$

It follows that the PME problem is scale invariant if

$$\alpha = 1, \quad \beta = \frac{1}{2+m}, \quad \gamma = -\frac{1}{2+m} \tag{94}$$

4.3 Similarity variables and self-similar solutions

We now look at similarity variables. We can define these as

$$y = \frac{x}{t^\beta}, \quad v = \frac{u}{t^\gamma}, \tag{95}$$

(notice from (78) that these variables are independent of λ).
Consider the scaling of the y and v variables. Since $x \mapsto \lambda^\beta \bar{x}$ and $t \mapsto \lambda^\alpha \bar{t}$, we can write the new variables in terms of the scale invariant parameters, giving

$$y = \frac{x}{t^\beta} = \frac{\lambda^\beta \bar{x}}{(\lambda^\alpha \bar{t})^\beta}. \tag{96}$$

Note that when $\alpha = 1$, equation (96) simplifies to

$$\bar{y} = \frac{\bar{x}}{\bar{t}^\beta}. \tag{97}$$

Following similar arguments and using (95), we obtain that $\bar{v} = \dfrac{\bar{u}}{\bar{t}^\gamma}$.
This means that under the scaling (95),

$$y \mapsto \bar{y} \quad \text{and} \quad v \mapsto \bar{v}. \tag{98}$$

With these variables we can seek a self-similar solution of the PME of the form $v = f(y)$.

To find an *ODE* for f, we write

$$v = f(y) \Rightarrow \frac{u}{t^\gamma} = f\left(\frac{x}{t^\beta}\right)$$

Hence

$$u = t^\gamma f\left(\frac{x}{t^\beta}\right). \tag{99}$$

Now we have an expression for u, we substitute it into (77). Let us look at each side of the PME separately.
LHS: (using product rule)

$$\begin{aligned} u_t &= \gamma t^{\gamma-1} f\left(\frac{x}{t^\beta}\right) + t^\gamma f'\left(\frac{x}{t^\beta}\right)\left(\frac{-\beta x}{t^{\beta+1}}\right) \\ &= \gamma t^{\gamma-1} f\left(\frac{x}{t^\beta}\right) - \frac{\beta x}{t^{\beta+1}} t^\gamma f'\left(\frac{x}{t^\beta}\right) \\ &= \gamma t^{\gamma-1} f\left(\frac{x}{t^\beta}\right) - \beta x t^{\gamma-\beta-1} f'\left(\frac{x}{t^\beta}\right) \end{aligned}$$

Next we look at the RHS of the PME, $(u^m u_x)_x$. First we examine u_x. It can be easily shown that

$$u_x = t^{\gamma-\beta} f'\left(\frac{x}{t^\beta}\right). \tag{100}$$

Multiplying (100) by u^m gives

$$\begin{aligned} u^m u_x &= \left(t^\gamma f\left(\frac{x}{t^\beta}\right)\right)^m t^{\gamma-\beta} f'\left(\frac{x}{t^\beta}\right) \\ &= t^{(m+1)\gamma-\beta}\Big(f\left(\frac{x}{t^\beta}\right)\Big)^m f'\left(\frac{x}{t^\beta}\right) \end{aligned}$$

In order to get an expression for the RHS, it is required that the above equation is differentiated with respect to x.

$$\begin{aligned} (u^m u_x)_x &= t^{(m+1)\gamma-\beta}\Big(\Big(f\left(\frac{x}{t^\beta}\right)\Big)^m f'\left(\frac{x}{t^\beta}\right)\Big)_x \\ &= t^{(m+1)\gamma-\beta}\left(t^{-\beta}\left(f\left(\frac{x}{t^\beta}\right)\right)^m f''\left(\frac{x}{t^\beta}\right) + t^{-\beta} m\left(f\left(\frac{x}{t^\beta}\right)^{m-1}\right)\left(f'\left(\frac{x}{t^\beta}\right)\right)^2\right) \\ &= t^{(m+1)\gamma-2\beta}\left(f\left(\frac{x}{t^\beta}\right)\right)^m\left(f''\left(\frac{x}{t^\beta}\right) + m\left(f\left(\frac{x}{t^\beta}\right)\right)^{-1}\left(f'\left(\frac{x}{t^\beta}\right)\right)^2\right) \end{aligned}$$

Thus, equating the left and right hand sides, we obtain the equation,

$$\gamma t^{\gamma-1} f\left(\frac{x}{t^\beta}\right) - \beta x t^{\gamma-\beta-1} f'\left(\frac{x}{t^\beta}\right) =$$

$$t^{(m+1)\gamma-2\beta}\left(f\left(\frac{x}{t^\beta}\right)\right)^m\left(f''\left(\frac{x}{t^\beta}\right) + m\left(f\left(\frac{x}{t^\beta}\right)\right)^{-1}\left(f'\left(\frac{x}{t^\beta}\right)\right)^2\right) \tag{101}$$

Next, rewrite (101) in terms of v and y (remembering that $v = f(y)$),

$$\gamma t^{\gamma-1} v - \beta y t^{\gamma-1} v' = t^{(m+1)\gamma-2\beta} v^m\Big(v'' + m v^{-1}(v')^2\Big) \tag{102}$$

We are able to simplify this equation by using the known values, $\beta = -\gamma = \frac{1}{2+m}$.

$$\begin{aligned} t^{\gamma-1}(\gamma v + \gamma y v') &= t^{((m+1)\gamma+2\gamma)} v^m\Big(v'' + m v^{-1}(v')^2\Big) \\ t^{\gamma-1}\gamma(v + y v') &= t^{(m+3)\gamma} v^m\Big(v'' + m v^{-1}(v')^2\Big) \end{aligned}$$

Hence we get the ODE

$$\gamma(v + y v') = v^m\Big(v'' + m v^{-1}(v')^2\Big). \tag{103}$$

4.3.1 A self-similar solution of the ODE

We now consider the initial data

$$u_1 = (1 - x^2)^{\frac{1}{m}} \qquad x^2 \le 1 \tag{104}$$

at t=1 for the PME. We would like to investigate how the solution changes over time and whether the best fit of the approximation is preserved.

We now confirm that

$$v = c(1-y^2)^{\frac{1}{m}} \quad \text{where } c = \left(\frac{m}{2(2+m)}\right)^{\frac{1}{m}} \tag{105}$$

is a solution to (102).
First let us calculate the necessary derivatives of (105).

$$v' = \frac{c}{m}(1-y^2)^{\frac{1-m}{m}}.(-2y) = -\frac{2cy}{m}(1-y^2)^{\frac{1-m}{m}}$$

and

$$\begin{aligned} v'' &= -\frac{2c}{m}(1-y^2)^{\frac{1-m}{m}} - \frac{2cy}{m}.(-2y)(1-y^2)^{\frac{1-2m}{m}}\left(\frac{1-m}{m}\right) \\ &= -\frac{2c}{m}(1-y^2)^{\frac{1-m}{m}} + \frac{4cy^2(1-m)}{m^2}(1-y^2)^{\frac{1-2m}{m}}. \end{aligned}$$

Substituting both of these expressions into LHS of (103).

$$\begin{aligned} \text{LHS} &= c\gamma\Big((1-y^2)^{\frac{1}{m}} - \frac{2y^2}{m}(1-y^2)^{\frac{1-m}{m}}\Big) \\ &= c\gamma(1-y^2)^{\frac{1}{m}}\Big(1 - \frac{2y^2}{m(1-y^2)}\Big) \\ &= \frac{c(1-y^2)^{\frac{1}{m}}}{2+m}\left(\frac{2y^2}{m(1-y^2)} - 1\right). \end{aligned}$$

Now we examine the RHS;

RHS =
$c^m(1-y^2)\Big(-\frac{2c}{m}(1-y^2)^{\frac{1-m}{m}} + \frac{4cy^2(1-m)}{m^2}(1-y^2)^{\frac{1-2m}{m}} + \frac{4cy^2}{m}(1-y^2)^{-(\frac{1}{m})}(1-y^2)^{\frac{2-2m}{m}}\Big)$.

By simplifying and factorising we have

$$c^{m+1}\Big(-\frac{2}{m}(1-y^2)^{\frac{1}{m}} + \frac{4y^2(1-m)}{m^2}(1-y^2)^{\frac{1-m}{m}} + \frac{4y^2}{m}(1-y^2)^{\frac{1-m}{m}}\Big),$$

then factorising again it reduces to

$$c^{m+1}(1-y^2)^{\frac{1}{m}}\left(-\frac{2}{m} + \frac{4y^2(1-m)}{m^2(1-y^2)} + \frac{4y^2}{m(1-y^2)}\right).$$

After this we group to give

$$c^{m+1}(1-y^2)^{\frac{1}{m}}\left(-\frac{2}{m} + \frac{4y^2}{m^2(1-y^2)}\right).$$

It only remains for us to check whether the RHS equals the LHS.

We have

$$\text{LHS} = \frac{c(1-y^2)^{\frac{1}{m}}}{2+m}\left(\frac{2y^2}{m(1-y^2)} - 1\right) \tag{106}$$

and

$$\text{RHS} = c^{m+1}(1-y^2)^{\frac{1}{m}}\left(-\frac{2}{m} + \frac{4y^2}{m^2(1-y^2)}\right). \tag{107}$$

Manipulating,

$$\begin{aligned} \frac{c(1-y^2)^{\frac{1}{m}}}{2+m}\left(\frac{2y^2}{m(1-y^2)} - 1\right) &= c^{m+1}(1-y^2)^{\frac{1}{m}}\left(\frac{4y^2}{m^2(1-y^2)} - \frac{2}{m}\right) \\ \cancel{\frac{c(1-y^2)^{\frac{1}{m}}}{2+m}}\cancel{\left(\frac{2y^2}{m(1-y^2)} - 1\right)} &= \frac{2}{m}c^{m+1}\cancel{(1-y^2)^{\frac{1}{m}}}\cancel{\left(\frac{2y^2}{m(1-y^2)} - 1\right)} \end{aligned}$$

which are identical since

$$c = \left(\frac{m}{2(2+m)}\right)^{\frac{1}{m}}. \tag{108}$$

This verifies that $v = c(1-y^2)^{\frac{1}{m}}$ is a solution of the PME.

Using the definitions of the similarity variables,

$$u = t^{\gamma}\left(1 - \left(\frac{x}{t^{\beta}}\right)^2\right)^{\frac{1}{m}} \qquad \left(\frac{x}{t^{\beta}}\right)^2 \leq 1 \tag{109}$$

is a solution of the PME problem with the given initial data.

4.4 A PME problem

Choose the initial data for the PME to be the function

$$u = (1-x^2)^{\frac{1}{m}}. \tag{110}$$

at $t = 1$.

The PME problem is an initial value problem and therefore the shape will change as time increases. This can be plotted using (109).

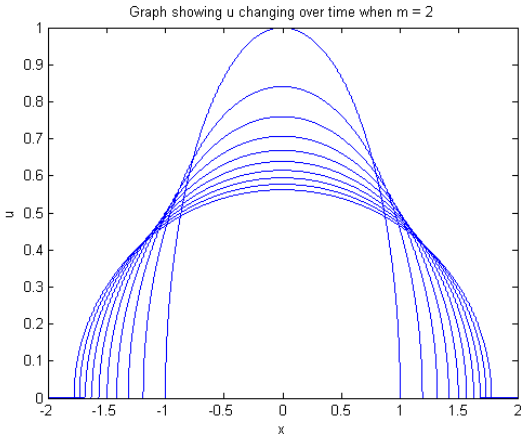


Figure 22: Graph showing how (110) evolves over time.

It is noted that the boundary points move outwards as the solution changes.

4.5 Preservation of Best Fit over time

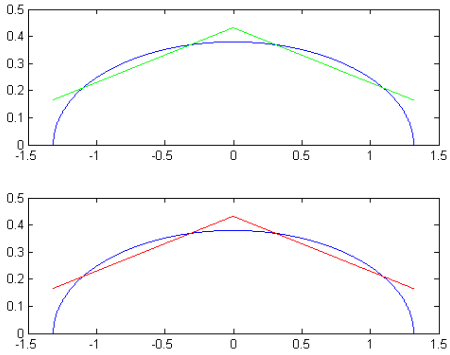
We wish now to investigate whether the best fit is preserved over time. Having a time dependent variable we can proceed in two different ways. We can evolve the solution and then apply the best fit procedure to it, or find the best fit at the initial time and transform the modified mesh along with the piecewise approximation using (78). The object of this section is to see whether these approximations are the same. If they are then the best fit is preserved over time.

This test is outlined in the following algorithm:

1. Perform Best Fit at $t = 1$.
2. Transform to a new mesh positions and U values at $t = T$ using (78).
3. Independently, evolve solution to $t = T$ using (109).
4. Compare graphs and errors of steps 2 and 3.

Graphs

The upper subplots (on the next page in green) show the solution evolved and then fitted. While the red shows the result when the initial best fit is transformed using the transformations.



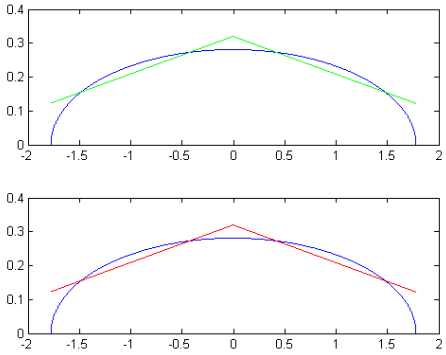
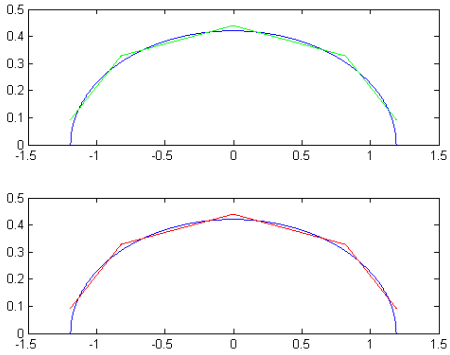


Figure 23 - Left: using two cells and $T = 3$, Right: using two cells and $T = 10$.



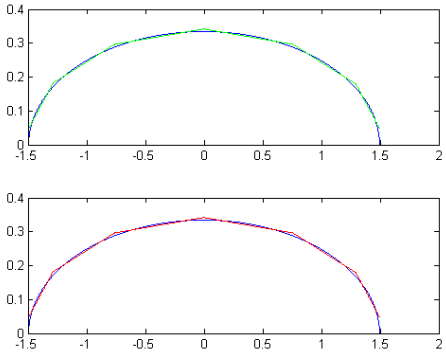
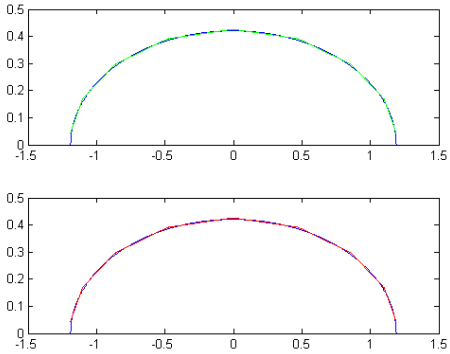


Figure 24 - Left: using three cell and $T = 5$, Right: using six cells and $T = 5$.



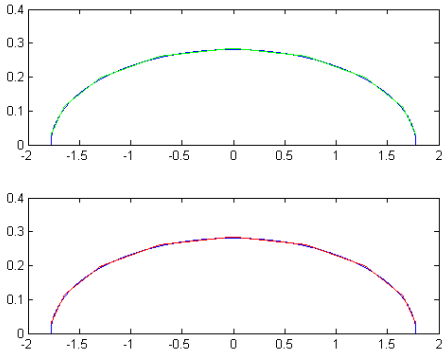


Figure 25 - Left: using eight cells and $T = 2$, Right: using eight cells and $T = 10$.

As one can see, both solutions are exactly the same showing preservation.

This is a very powerful result because it shows us that, providing we have a systematic way of transforming the variable to a later time, once we have obtained an approximation to a function we can always accurately evolve our approximation.

The result also shows us that equidistribution is preserved over time.

5 Conclusion

The aim of this Dissertation was to examine how we could find a best L_2 fit to a given function on a mesh where the nodes are allowed to move, and to check the validity of the Carey and Dinh equidistribution formula. In addition we wanted to investigate the conjecture that the best fit with adjustable nodes of a self-similar solution of a PDE was preserved.

The L_2 approximation U was constructed from a Ritz expansion. When the general problem was being considered we proceeded by either constructing a continuous or discontinuous approximation. The discontinuous case used the procedure Cellwise, which found the best fit in each cell independently and formulated the approximation by just placing the cells in the right order.

The continuous case involved solving all the equations simultaneously, which was achieved by using Assembly on the discontinuous approximation, creating a tridiagonal mass matrix, solved by the Thomas algorithm. We used the discontinuous approximation to construct the mesh adjustment, by setting the relevant part of the L_2 norm variation to zero.

In the first chapter we investigated how we could create an approximation on a fixed mesh. We began our study by first considering a single cell. We decided to restrict our approximation to piecewise constants and piecewise linears. We found that when we were using constants, we were only required to calculate the integral under the exact function in that cell. However, piecewise linears caused us to solve a 2×2 matrix system and hence it was relatively easy to obtain the approximation vector.

Armed with this matrix we were able to formulate the mass matrix that would be used in the Assembly procedure, obtaining a continuous function. We then turned our attention to restricting the values at the end of the region while conserving mass. We modified the algorithm to conserve mass by using a modified matrix. Only being concerned with the interior nodes we were able to explicitly find the approximation for two and three cells. If the cell number was ≥ 4 then we applied the same Thomas algorithm. At the end of Chapter 1 we presented the results for both the local and global cases.

As noted, the approximation can be improved by allowing the nodes to move, hence Chapter 3 dealt with adjustable meshes. The chapter began by recalling the cellwise procedure and displaying results for both constants and linears. Next, following [1], we were able to derive the best fit equations that are needed when we are dealing with these types of meshes. By means of an iteration we were able to find improved approximations. We noted two different procedures for relocating the mesh points. Results were given and conclusions drawn. Overall the program that produced these results ran very well.

After obtaining the optimal mesh in the L_2 sense, we tested Carey and Dinh's statement about equidistribution. The vectors presented in Section 3.5.1 give satisfactory results. If one looks at the interior values of these vectors one can see that the values do not differ a lot, showing an equidistributing feature of the approximation.

The final chapter studied the Porous Medium Equation with a time dependent solution. Introducing the idea of Scale Invariance and Similarity Variables we were able to generate an ODE which the self-similar solution would satisfy and verified a well-known solution of the equation.

We then tested whether the best fit with adjustable nodes to this solution at time $t = 1$ was preserved over time. The results shown in Section 4.5 illustrate that this best fit is indeed preserved over time in this case. When the initial best fit was transformed using the similarity transformations, the resulting approximation was exactly the same as if we had just done the best fit with adjustable nodes on the evolved solution. Unfortunately, time ran out and we were unable to produce a mathematical reason for this.

The obvious next step of this work would be closer examination of the similarity variables, along with the scale invariance transformations and the best fit equations, to investigate the mathematical reasoning behind this phenomena. One may hope to deduce that the equations for the evolution of the best fit approximation is not dependent on time.

This dissertation has been very enjoyable and has allowed an exploration of advanced mathematical research. Despite the short time period the investigation covered several interesting techniques to improve approximating a function. However, it was not possible to complete every test.

For example it would be beneficial to see the effect of the adjustable mesh techniques (averaging and intersection) when the modification case was being looked at. This could give very interesting results because we have shown the difficulty for obtaining our approximation even on a fixed mesh. Also, it would be intersting to see if the similarity result holds when the boundary condition is imposed on the PME best fit.

Another useful comparison would be to study how the adjustable mesh methods are affected if the tolerance of mesh point proximity was altered.

An interesting extension to the work would be to look at how we can improve the approximation by increasing the polynomial order in each cell. We can begin by considering quadratic polynomials for each cell. If this was the case we would have to develop new techniques for relocating points. One should expect the L_2 error to be decreased dramatically; however, the mathematics involved would be much more difficult.

Another interesting area one can look at could be if a combination of different approximations could be used at once. If this is a viable option then once we have established the solution regions where there is less activity we can use lower order polynomials in this region.

One could enhance the approximation by using a combination of different order polynomials. Once we have an idea of the solution we can apply lower order polynomials to places where the solution is smooth.

After examining the one dimensional problem extensively, one will want to move to two dimensions. Baines in [1] and Tourigny and Baines [9] have already devised techniques which can be used in improving the two dimensional approximation. One could follow the layout of this project and investigate whether the results obtained in Section 4.5 can be extended to higher dimensions.

Acknowledgements

I would like to express my sincere gratitude to Professor Mike Baines for his help and support over the MSc year, especially during the last three months. I have found it very enjoyable working with him.

I am very grateful to all my support workers who have helped with coding and writing up.

Thank you to all my University friends who have given me programming assistance.

My thanks too to all the staff in the Maths Department for their support.

A big thank you to my parents and friends, particularly Ashok Vaidya, who have given me so much encouragement.

Finally, I would like to acknowledge this work was supported by a CTA Studentship supplied by the Engineering and Physical Sciences Research Council.

References

[1] M.J. Baines, *Algorithms for optimal discontinuous piecewise linear and constant L_2 fits to continuous functions with adjustable nodes in one and two dimensions*, Mathematics of Computation 62 (1994), 645-669.

[2] M.J. Baines, M.E.Hubbard and P.K.Jimack, *Scale-invariant moving finite elements for nonlinear PDEs*, Applied Numerical Mathematics (2006), 230-252.

[3] G.I. Barenblatt, *Scaling*, Cambridge Texts in Applied Mathematics (2003).

[4] C.J. Budd, G.J. Collins, W.Z. Huang and R.D.Russell, *Self-Similar Numerical Solutions of the Porous-Medium Equation Using Moving Mesh Methods*, Philosophical Transactions: Mathematical, Physical and Engineering Sciences 357, Geometric Integration: Numerical Solution of Differential Equations on Manifolds (1999), 1047-1077.

[5] G.F. Carey and H.T. Dinh, *Grading Functions and Mesh Redistribution*, SIAM Journal on Numerical Analysis 22 (1985), 1028-1040.

[6] C. Farmer, *Geological Modelling and Reservoir Simulation* in A. Iske and T. Randen, Mathematical Methods and Modelling in Hydrocarbon Exploration and Production, Springer (2004).

[7] P.D. Loach and A. J. Wathen, *On the best least squares approximation of continuous functions using linear splines with free knots*, IMA Journal of Numerical Analysis 11 (1991), 393-409.

[8] J.Ockendon, S. Howison, A. Lacey and A. Movchan, *Applied Partial Differential Equations* (Revised Edition), Oxford University Press (2003).

[9] Y. Tourigny and M.J. Baines, *Analysis of an algorithm for generating locally optimal meshes for L_2 approximation by discontinuous piecewise polynomials*, Mathematics of Computation 66 (1997), 623-650.