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 Dis-

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

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

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

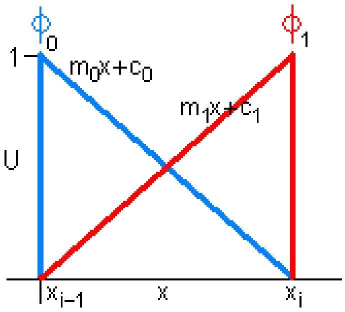
$$\pi_i(x) = \begin{cases} 1, & x_i \leq x \leq x_{i+1} \\ 0, & \text{otherwise} \end{cases}. \tag{5}$$

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,

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

which we can manipulate to investigate an expression for U_0,

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



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

Our appoximation is given as

$$U = U_0\varphi_0 + U_1\varphi_1. \tag{9}$$

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

$$\int_a^b (f(x) - U_0\varphi_0 - U_1\varphi_1)\varphi_i \, dx = 0 \quad i = 0, 1, \tag{10}$$

which can be re-arranged to obtain

$$\int_a^b f(x)\varphi_i - \int_a^b U_0\varphi_0\varphi_i - \int_a^b U_1\varphi_1\varphi_i = 0 \quad i = 0, 1. \tag{11}$$

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 $\varphi_0 = 1 - x$ and $\varphi_1 = x$ in $[0, 1]$.
Thus, (11) becomes

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

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

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,

$$Mu = f \tag{22}$$

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

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

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

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

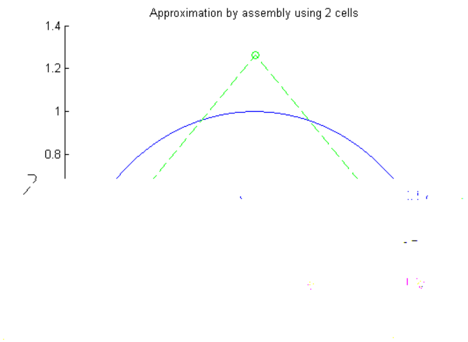
$$\int_0^N \left(f(x) - (mx + c) \right)^2 \tag{24}$$

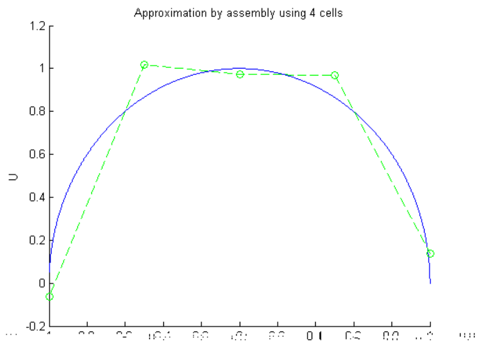
The gradient m and the y-intercept c is calculated by basic geometry.

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 $f = (1 - x^2)^{\frac{1}{m}}$.





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.

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

$$\varphi_1 = -x \quad \text{and} \quad \varphi_2 = x + 1. \tag{29}$$

Thus, we have $f_1 = \int_{-1}^{0} -x f(x)\, dx$ 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

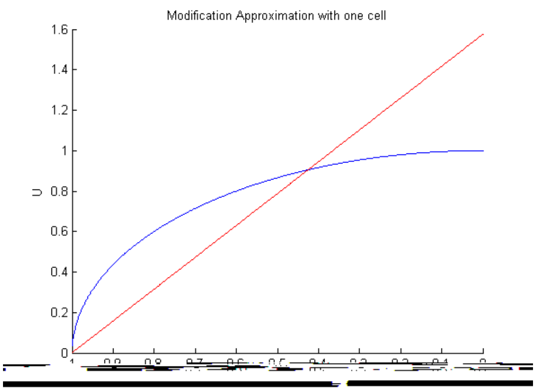


Figure 3 - Graph showing best fit approximation using modification.

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.

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.

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

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.

<u>$N = 2$</u>

We have,

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

Applying the set of rules,

$$\begin{bmatrix} d_0 & b_0 & 0 \\ l_0 + d_0 & b_0 + d_1 + l_1 & b_1 \\ 0 & l_1 & d_2 \end{bmatrix} \begin{bmatrix} U_0 \\ U_1 \\ U_2 \end{bmatrix} =$$

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

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

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.

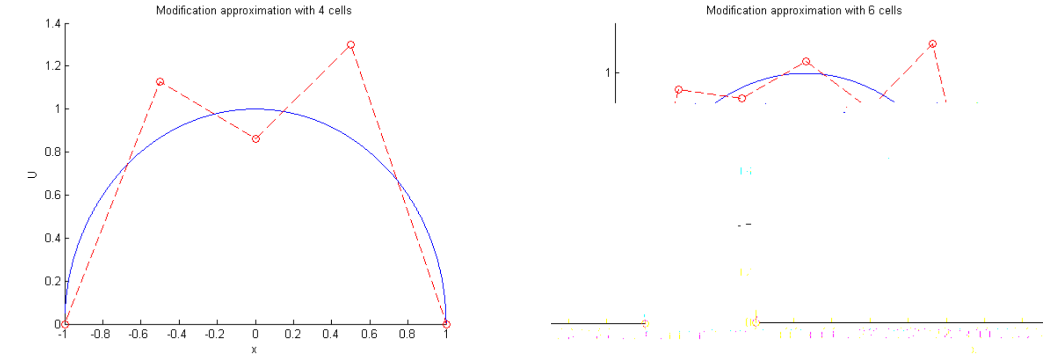


Figure 5 - Left: Modified approximation with four cells,
Right: Modified approximation with six cells.

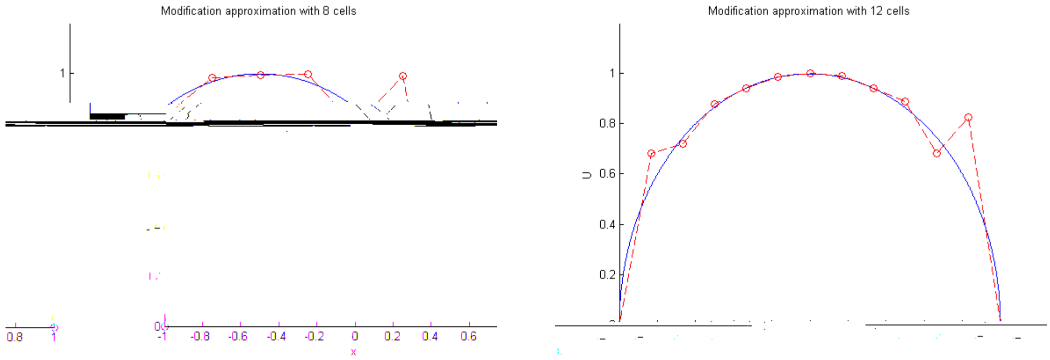


Figure 6 - Left: Modified approximation with eight cells,
Right: Modified approximation with twelve cells.

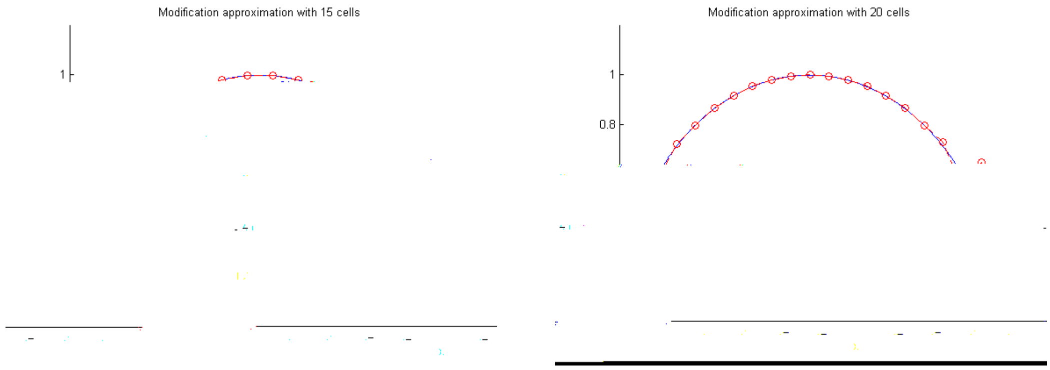


Figure 7 - Left: Modified approximation with fifteen cells,
Right: Modified approximation with twenty cells.

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

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

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

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	

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

N	e_N	$\frac{e_{N+1}}{e_N}$
2		

N

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

N	e_N	$\frac{e_{N+1}}{e_N}$
2	0.	

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.

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

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

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

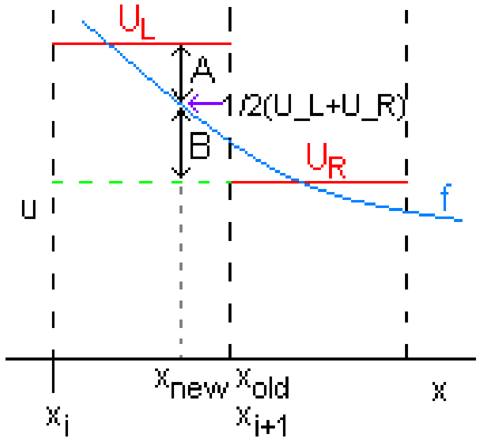
$f_L \neq f_R$ (Averaging).

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



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 interva4(mak)9(a)-3u-15110

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,

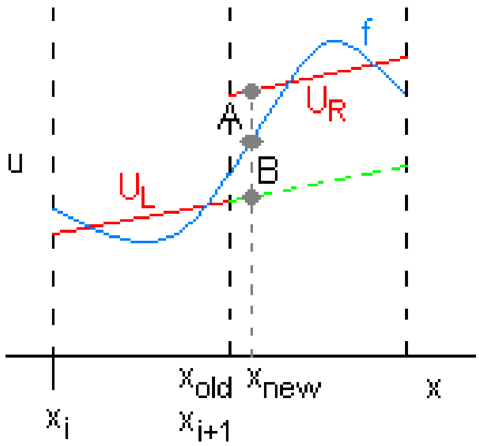
- Repeat algorithm until interval width is less than a tolerance.
    1. Calculate midpoint; $M = \frac{1}{2}$

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



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

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

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.

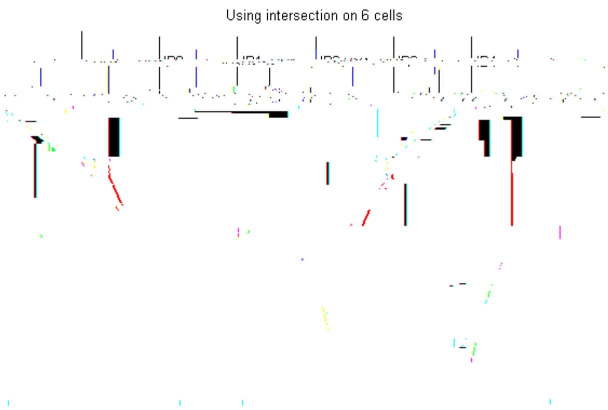


Figure 20: Construction of the new mesh using the intersection procedure with six cells.

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, illus-

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

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.

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

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

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 conser- γ s014.4738 0 4-32
vation of mass. ¿From (92) we have $\beta = -\gamma$, substitut; 5(soce)-327(from)-4f mass.ss014.47380451

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 $x \mapsto \lambda^{\beta}\overline{x}$ and $t \mapsto \lambda^{\alpha}\overline{t}$, we can write the new variables in terms of the scale invariant parameters, giving

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

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

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

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

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

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

$$
\begin{aligned}
u^m u_x &= \left(t^{\gamma} f\left(\frac{x}{t^{\beta}}\right)\right)^m t^{\gamma-\beta} f\left(\frac{x}{t^{\beta}}\right) \\
&= t^{(m+1)\gamma-\beta} f\left(\frac{x}{t^{\beta}}\right)^m f\left(\frac{x}{t^{\beta}}\right)
\end{aligned}
$$

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

$$
\begin{aligned}
(u^m u_x)_x &= t^{(m+1)\gamma-\beta}\left(f\left(\frac{x}{t^{\beta}}\right)^m f\left(\frac{x}{t^{\beta}}\right)\right) \\
&= t^{(m+1)\gamma-\beta}\left(t^{-\beta}\left(f\left(\frac{x}{t^{\beta}}\right)\right)^m f\left(\frac{x}{t^{\beta}}\right) + t^{-\beta} m\left(f\left(\frac{x}{t^{\beta}}\right)\right)^{m-1}\left(f\left(\frac{x}{t^{\beta}}\right)\right)^2\right) \\
&= t^{(m+1)\gamma-2\beta}\left(f\left(\frac{x}{t^{\beta}}\right)\right)^m\left(f\left(\frac{x}{t^{\beta}}\right) + m\left(f\left(\frac{x}{t^{\beta}}\right)\right)^{-1}\left(f\left(\frac{x}{t^{\beta}}\right)\right)^2\right)
\end{aligned}
$$

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 $v = f(y)$),

$$
\gamma t^{\gamma-1} v - \beta y t^{\gamma-1} v = t^{(m+1)\gamma-2\beta} v^m \left(v + m v^{-1}(v)^2\right) \tag{102}
$$

W2e are able to simplify this equation by using the known valu449.33388awT5nc3-11.955Tf2.7T5nc(u

We now confirm that

$$v = c(1 - y$$

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,

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

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

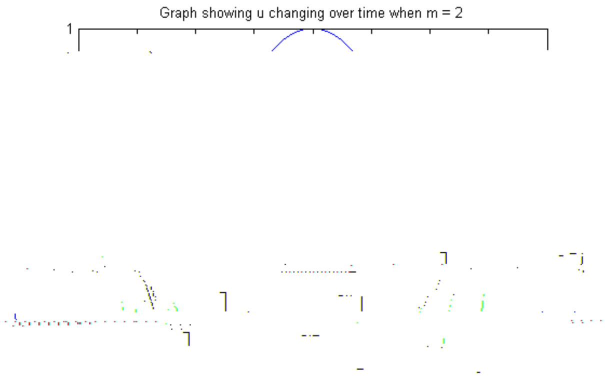


Figure 22: Graph showing how (110) evolves over time.

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

4.5 Preservation of Best Fit over time

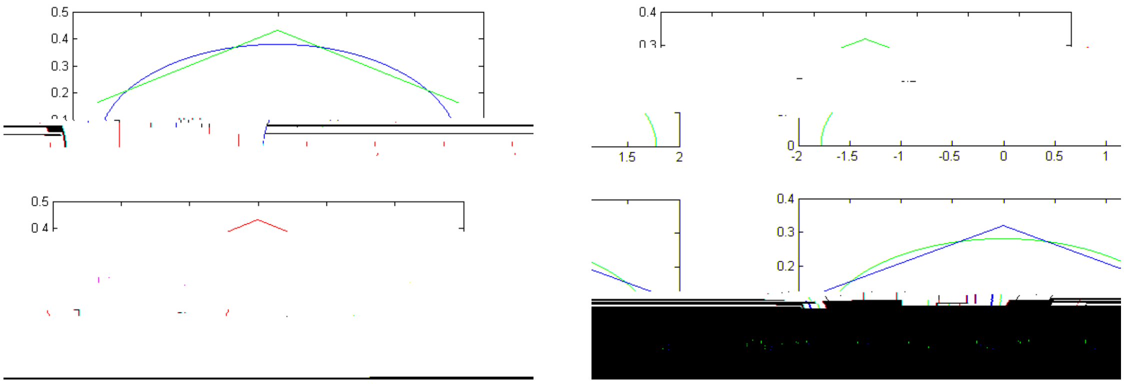
We wish now to investigate whether the best fit is preserved over time. Having a time dependent variable we can proceed in two different ways. We can evolve the solution and then apply the best fit procedure to it, or find the best fit at the initial time and transform the modified mesh along with the 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.

<u>Graphs</u>

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.



0.5
0.4
0.3
0.2
0.4
0.3
1.5
2
0
-2
-1.5
-1
-0.5
0
0.5
1
0.5
0.4
0.4
0.3
0.2

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

The final chapter studied the Porous Medium Equation with a time dependent solution.

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.