Adaptive Finite Difference Solutions for Convection and Convection-Diffusion Problems

C.J.Smith

September 1992

Submitted to the

Department of Mathematics,

University of Reading,

in partial fulfillment of the requirements for the

Degree of Master of Science

Abstract

In the numerical solution of partial differential equations, by finite methods, greater efficiency can be obtained if the geometry of the grid is determined by the solution. Many techniques for adapting the grid to the numerical solution it supports, have been proposed. One such scheme for convection equations in one dimension is considered here, and its application to convection-diffusion problems is investigated.

Contents

List of Figures

List of Tables

Notation

$I\!R_+$	positive real numbers
u_j^n	finite difference approximation of $u(x_j, t_n)$
$(\triangle x_j)^n$	$x_j^n - x_{j-1}^n$
$(\triangle x^{n+1})_j$	$x_j^{n+1} - x_j^n$

Introduction

The subject of this dissertation is adaptive finite difference schemes, for the numerical solution of partial differential equations involving convection. A scheme is said to be adaptive if the underlying discretisation mesh (grid) undergoes changes in geometry, in response to the numerical solution it supports.

An existing adaptive finite difference scheme [1] for convection problems is investigated, with a view to its application to convection-diffusion problems. This scheme is based on the first order upwind method. Before the adaptive scheme is discussed in Chapter 3, the problems associated with first order schemes are investigated in Chapter 1.

Chapter 2 reviews two broad categories of adaptive grid methods. A particular adaptive scheme [7] is examined in some detail. This scheme highlights important considerations for all adaptive methods.

The adaptive scheme [1] discussed in Chapter 3, is presented together with two alternative discretisations of the differential equation. A comparison is made of the numerical results produced by the two discretisations. The test problem used for this purpose is one which consists of the time development of a steep front, confined to a fixed region.

The model problem is changed in Chapter 4 to one involving diffusion in addition to convection. An attempt is made to modify the scheme presented in Chapter 3, to provide solutions to the new problem. In addition, the solution of this test problem contains a steep, moving front which the grid is required to track.

Conclusions drawn from the numerical results contained in Chapter 4 lead to further modifications to the scheme, which are considered in Chapter 5.

Chapter 1

Background

The modelling of partial differential equations (PDEs) which involve dominant convection terms, has long been recognised to pose significant problems for numerical solution methods. For instance, first order upwind schemes provide some desirable properties : they preserve monotonicity of the solution and give reasonably accurate phase speed. However, for these schemes, numerical diffusion is severe. This is demonstrated in *figure* 1.1, which shows how the first order upwind scheme produces unwanted smearing in the solution. The initial data is shown in the first graph. Subsequent graphs show the numerical solution at intervals of 20 time steps, with the dotted lines representing the corresponding exact solutions. Numerical diffusion is a particular drawback when modelling convection-diffusion processes. Key features of the solution of such problems often involve steep fronts, which propagate through the solution domain. Numerical diffusion tends to cause smoothing of sharp features, which is often highly pronounced.

In an attempt to find improvements to simple first order schemes, a natural starting point would seem to be an investigation into numerical diffusion.

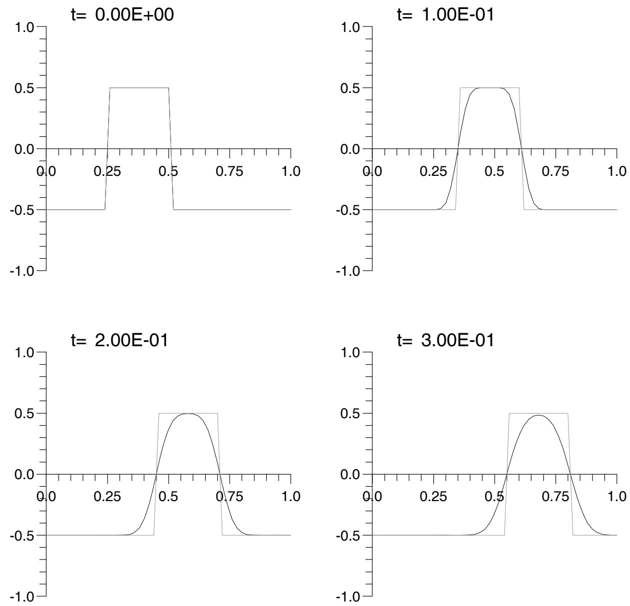

Figure 1.1: Numerical Diffusion in Upwind Solution for $u_t + u_x = 0$

1.1 Numerical Diffusion and the Modified Equation

One approach to investigating the phenomenon of numerical diffusion, is to consider the *Modified Equation* of the scheme which is to be used. This is the differential equation which the exact, smoothed solution to the difference scheme satisfies. It is obtained by expanding each term of the scheme in a Taylor series. Mixed space and time derivatives are then eliminated, along with all time derivatives higher than the first. This results in a differential equation with an infinite number of purely space derivative terms, which are equated to the first time derivative.

The first few terms of this modified equation are just those of the original PDE. Subsequent terms are scaled by the space and time discretisation lengths, and represent a form of truncation error for the scheme. It is the largest of these terms which will be of interest here.

There is a slight subtlety involved in obtaining the modified equation [9]. This occurs when the high order time and mixed derivatives are to be eliminated. It is not valid to use the original differential equation for this purpose, which is the familiar technique used in deriving, for example, the Lax-Wendroff scheme, the reason being that solutions of the modified equation are not also solutions of the original equation, but rather of the difference scheme. The modified equation itself must be used in the elimination process. This is best illustrated with an example.

Example Consider the numerical solution of the linear advection equation,

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = 0 \quad \text{where} \quad a > 0.$$

The first order upwind scheme for this equation is obtained by making the following one-sided discretisation,

$$\frac{u_j^{n+1} - u_j^n}{\triangle t} + a\left(\frac{u_j^n - u_{j-1}^n}{\triangle x}\right) = 0$$

which is well known to preserve the monotonicity of the solution, but also to smear it. (A scheme is said to be monotonicity preserving, if the solution data remains monotonic, given monotonic initial data.) The first step in obtaining the modified equation is, to replace each of the discrete terms with its Taylor expansion.

$$u_t + \frac{\triangle t}{2}u_{tt} + \cdots + \frac{a}{\triangle x}\left\{u - \left[u - u_x\triangle x + u_{xx}\frac{(\triangle x)^2}{2} - \cdots\right]\right\} = 0$$

where u is evaluated at the point $(j\triangle x, n\triangle t)$.

Collecting terms, this reduces to

$$u_t + au_x + \frac{\triangle t}{2}u_{tt} - au_{xx}\frac{\triangle x}{2} + \mathrm{O}(\triangle t^2, \triangle x^2) = 0 \tag{1.1}$$

By applying $-\frac{\triangle t}{2}\frac{\partial}{\partial t}$ to (1.1), the following equation is obtained,

$$-\frac{\triangle t}{2}u_{tt} - \frac{a}{2}\triangle t u_{xt} + \mathrm{O}(\triangle t^2, \triangle t\triangle x, \triangle x^2) = 0 \tag{1.2}$$

Adding (1.2) to (1.1) would remove the second time derivative but would leave a mixed time and space derivative. A further equation involving this unwanted derivative may be obtained by applying $a\frac{\triangle t}{2}\frac{\partial}{\partial x}$ to (1.1). This gives,

$$a\frac{\triangle t}{2}u_{tx} + a^2\frac{\triangle t}{2}u_{xx} + \mathrm{O}(\triangle t^2, \triangle t\triangle x, \triangle x^2) = 0 \tag{1.3}$$

Finally, adding (1.1),(1.2) and (1.3) we have the modified equation for the first order upwind scheme :

$$u_t + au_x + a^2\frac{\triangle t}{2}u_{xx} - a\frac{\triangle x}{2}u_{xx} + \mathrm{O}(\triangle t^2, \triangle t\triangle x, \triangle x^2) = 0$$

This may be written in a more revealing form by observing that,

$$\begin{aligned} a^2\frac{\triangle t}{2}u_{xx} - a\frac{\triangle x}{2}u_{xx} &= -a\frac{\triangle x}{2}(u_{xx} - \nu u_{xx}) \\ &= -\frac{(\triangle x)^2}{2\triangle t}\nu(1-\nu)u_{xx} \end{aligned}$$

where $\nu = a\frac{\triangle t}{\triangle x}$ is the Courant number.

For this particular problem, the Courant-Friedrichs-Lewy condition imposes

$$0 \le \nu \le 1$$

as a necessary stability condition. Hence, the modified equation may now be seen in its final form, which to first order in $\triangle t$ and $\triangle x$ is,

$$u_t + au_x = \frac{(\triangle x)^2}{2\triangle t}\nu(1-\nu)u_{xx} + \mathrm{O}(\triangle t^2, \triangle t\triangle x, \triangle x^2) \tag{1.4}$$

The upwind scheme models this equation more accurately than it does the original differential equation. The principal source of error is the numerical diffusion term on the right-hand side of the modified equation.

1.2 Improvements to First Order Schemes

As seen above, first order schemes offer desirable properties, such as monotonicity preservation and computational simplicity. But this is much to the detriment of accuracy, which suffers principally from the mechanism of numerical diffusion.

One obvious way in which to improve matters is to use a higher order scheme, such as Lax-Wendroff. This certainly removes the troublesome diffusion term from the modified equation. However the monotonicity preserving property is lost, allowing spurious oscillations to enter the solution in regions of rapid spatial change. This important property may only be recovered by the use of non-linear schemes. In particular, flux limiter schemes have enjoyed remarkable success in this respect, offering solutions free of unwanted oscillations and preserving, in general, the accuracy of a second order scheme.

Another approach is that of the so-called *adaptive* schemes. These schemes attempt to adapt the grid, by moving the nodes, to provide finer resolution in regions of rapid spatial variation. By providing a greater density of grid nodes in such regions, computational effort can be concentrated where it is most needed. As may be seen from equation (1.4), the degree of numerical diffusion present in a particular region depends strongly on the grid spacing there. A finer grid, in regions where the gradient of the solution changes rapidly, will provide a marked reduction in numerical diffusion. It is this particular approach of adaptivity, which will be followed in the remainder of this dissertation.

Finally, there is an approach to the problem of numerical diffusion, which forms a natural complement to that of adaptive grids. This is the *anti-diffusive* velocitities introduced by Smolarkiewicz [8] and Margolin [6]. Here the diffusion term of the modified equation is re-written as an extra convective term. The scheme involves multiple passes (often just two) of a first order upwind scheme. The result of the first pass is used to gain a first estimate of the numerical diffusion present in the solution. This is manipulated into the form of a wave velocity. When this is added back to the original equation, and a second pass of the upwind scheme is made, a less diffused solution is obtained. Since corrections are made to compensate for the first order error only, further iterations improve the accuracy but not the order of the solution.

The connection with moving grids is most striking in the case of problems where there are steep features which develop in situ. In such cases any grid movement can be attributed solely to numerical diffusion considerations, since there are no translating features to be tracked. So whereas the Smolarkiewicz and Margolin scheme removes diffusion by imposing an extra velocity into the discretised equation, the adaptive scheme produces extra velocities in the grid.

Chapter 2

Adaptive Grid Methods

By imposing velocities onto grid nodes, which hold solution values, we are now attempting to solve the differential equation within a moving frame of reference. (The term *moving* is here to be read as *translating and deforming*.) It is important to consider how the form of the equation is changed, when its independent variables are transformed to those of the moving frame.

2.1 Transformation to the Moving (Lagrangian) Frame

Let (ξ, τ) be the space and time coordinates in the moving frame. Since the moving frame is free to distort, as well as translate, we must consider a general space transformation,

$$x = x(\xi, \tau)$$

It is commonly the case that the discretisation of the PDE proceeds in two stages. First a space discretisation is carried out, which results in a system of ordinary differential equations in the time variable. Then this system is discretised and solved using some time stepping procedure. This is the approach of the *Method of Lines*. It is in the first of these stages that the transformation to the moving frame occurs. Hence the time transformation equation plays a rôle of little importance, and we may make the convenient choice,

$$t = \tau.$$

To obtain the transformed differential equation, we require the partial derivatives of the above coordinate transformation. These are most readily calculated by means of the Jacobian.

The Jacobian For the coordinate transformation,

$$\begin{aligned} x &= x(\xi, \tau) \\ t &= \tau \end{aligned}$$

the Jacobian is readily seen to be

$$\frac{\partial(x,t)}{\partial(\xi,\tau)} \equiv \begin{pmatrix} x_\xi & x_\tau \\ t_\xi & t_\tau \end{pmatrix} = \begin{pmatrix} x_\xi & x_\tau \\ 0 & 1 \end{pmatrix}$$

and the inverse of the Jacobian is found to be

$$\frac{\partial(\xi,\tau)}{\partial(x,t)} \equiv \begin{pmatrix} \xi_x & \xi_t \\ \tau_x & \tau_t \end{pmatrix} = \begin{pmatrix} 1/x_\xi & -x_\tau/x_\xi \\ 0 & 1 \end{pmatrix}$$

Hence the required partial derivatives are

$$\begin{aligned} \xi_x &= \frac{1}{x_\xi} \qquad & \xi_t &= -\frac{x_\tau}{x_\xi} \\ \tau_x &= 0 \qquad & \tau_t &= 1 \end{aligned} \tag{2.1}$$

Now consider the differential equation

$$u_t = \mathcal{L}u,$$

where $\mathcal{L}$ is some operator containing space derivatives only. By defining,

$$\hat{u}(\xi,\tau) = u(x,t)$$

we obtain, by use of the partial derivatives found above,

$$\frac{\partial u}{\partial t} = \frac{\partial \hat{u}}{\partial \xi}\frac{\partial \xi}{\partial t} + \frac{\partial \hat{u}}{\partial \tau}\frac{\partial \tau}{\partial t} = \frac{\partial \hat{u}}{\partial \tau} + \frac{\partial \xi}{\partial t}\frac{\partial \hat{u}}{\partial \xi}$$

$$\Rightarrow \frac{\partial u}{\partial t} = \frac{\partial \hat{u}}{\partial \tau} - \frac{x_\tau}{x_\xi}\frac{\partial \hat{u}}{\partial \xi} \tag{2.2}$$

And the differential equation in the moving frame is,

$$\frac{\partial \hat{u}}{\partial \tau} = \frac{x_\tau}{x_\xi}\frac{\partial \hat{u}}{\partial \xi} + \hat{\mathcal{L}}\hat{u} \tag{2.3}$$

where $\hat{\mathcal{L}}$ is obtained by a suitable transformation of $\mathcal{L}$.

This equation often appears in a slightly different form. Since x_τ is the velocity of the moving frame, we may write

$$\dot{x} = x_\tau$$

and refer to $\dot{x}$ as the frame velocity. A further notational shorthand is also in use, where the symbol $\frac{\partial u}{\partial x}$ is used to denote $\frac{1}{x_\xi}\frac{\partial \hat{u}}{\partial \xi}$. The equation in the moving frame now reads,

$$\frac{\partial \hat{u}}{\partial \tau} - \frac{x_\tau}{x_\xi}\frac{\partial u}{\partial x} = \mathcal{L}u \tag{2.4}$$

Equations (2.3) and (2.4) will play an important part in the remaining chapters.

2.2 Adaptive Grids

Two broad classes of adaptive grid methods may be identified [5] :

I. Dynamic Rezone Methods

II. Static Rezone Methods

2.2.1 Dynamic Rezone Methods

In grid adaption algorithms of this class, grid node movement is bound to that of the solution. Individual grid nodes attempt to keep pace with moving features of the solution.

Example Petzold [7] has proposed the following two stage finite difference scheme.

The first stage produces grid movement which serves to minimise the time rate of change of the solution at individual grid nodes. This allows for a larger time step to be used, without loss of accuracy.

The second stage belongs properly to the static class of methods. It consists in applying refinements to the grid produced by the first stage, in order to obtain better resolution in regions of rapid spatial change.

Since this scheme contains elements of the two main classes of adaptive methods, it is worth considering it in a little more detail.

Petzold's Scheme The first stage is based on a transformation to a Lagrangian frame, due to Hyman [4]. This transformation is obtained by finding the frame velocity which minimises the quantity

$$Q = \left(\frac{\partial \hat{u}}{\partial \tau}\right)^2 + \alpha \left(\frac{\partial x}{\partial \tau}\right)^2, \quad \text{where} \quad \alpha \in I\!R_+$$

The notation proceeds a little more smoothly if the following definitions are made,

$$\begin{aligned} \dot{u} &= \frac{\partial \hat{u}}{\partial \tau} \\ \dot{x} &= \frac{dx}{d\tau} \\ u_x &= \frac{1}{x_\xi}\frac{\partial \hat{u}}{\partial \xi} \end{aligned}$$

Then (2.4) reads

$$\dot{u} = u_t + \dot{x}u_x$$

The quantity to be minimised now takes on the more revealing appearance

$$\begin{aligned} Q &= \dot{u}^2 + \alpha\dot{x}^2 \\ &= (u_t + \dot{x}u_x)^2 + \alpha\dot{x}^2 \end{aligned}$$

Minimising this with respect to the frame velocity,

$$\frac{dQ}{d\dot{x}} = 0 \quad \Rightarrow \quad \dot{x} = -\frac{u_t u_x}{\alpha + (u_x)^2}$$

This equation defines trajectories along which the time rate of change of the solution is

$$\dot{u} = \frac{\alpha u_t}{\alpha + (u_x)^2}$$

The free parameter, α, provides some control over the properties of the transformation.

Even though this transformation produces grid movement which follows the solution in some fashion, it is far from ideal. Extra parameters must be added to the minimisation to prevent nodes coalescing. The values which the parameters take to achieve their purpose are highly problem dependent, and must be found by inspection. This reduces the possibility of the method being incorporated into a fully automatic package, a standard aim of many methods of this kind. This is a problem common to many adaptive techniques.

In addition, the grid so far does not necessarily adapt to sharp features of the solution. A second, grid enhancement, stage is required. Petzold uses a grid redistribution technique, where nodes are added or deleted from the grid, to achieve

$$\triangle x\,|u_x| + (\triangle x)^2\,|u_{xx}| \leq \text{preset tolerance},$$

a process which also involves a number of interpolations.

2.2.2 Static Rezone Methods

The principal technique of this class of adaptive methods is that of equidistribution. The second stage of Petzold's scheme (discussed above) introduced the idea of redistributing the grid nodes, according to some predetermined criterion. Such a criterion may also be used as the starting point for grid adaption.

In the equidistribution method, a function is defined which we require to be equidistributed across the mesh. This function also serves as the space coordinate in the moving (Lagrangian) frame of reference. By *equidistributed* we mean that the function increments by a constant amount, between neighbouring nodes, throughout the grid. For instance, the arc-length along the solution may be used as the equidistributing function, in which case the nodes will be placed at equal intervals along the curve of the solution.

In general, the equidistributing function is defined in terms of a monitor function, $m(u, x)$,

$$\xi(x,t) = \frac{1}{M}\int_0^x m(u(x,t),x)\,dx$$

$$\text{where } M = \int_0^1 m(u(x,t),x)\,dx$$

The solution domain is assumed to be the unit interval, $0 \leq x \leq 1$. The monitor function is chosen to be some quantity related to the spatial variation of the solution.

Summary To conclude, the Petzold scheme highlights many aspects which are of importance in designing adaptive methods :

- The choice of solution properties to which the grid is to adapt
- The manner in which the grid is to be altered
- Can it be guaranteed that the grid will remain monotonic, that nodes will not overtake one another
- Can the technique be made fully portable, without the need for externally set parameters

Chapter 3

An Adaptive Finite Difference Scheme

Two desirable properties of any adaptive scheme are that it should be monotonic in both function values and node positions. In particular we do not want the solution to develop spurious oscillations near steep fronts, nor the grid to fold over on itself. One scheme which satisfies both these criteria has been outlined in [1].

This scheme (which is referred to as the *Masterful*[1] scheme) consists of a simple first order upwind scheme, in conjunction with a best-fit node displacing algorithm. The latter is taken from [2], and is in the form of an iteration. For the purposes of the scheme being considered here, consistent accuracy is obtained by making just one iteration of this algorithm.

[1] Monotonic Adaptive Solutions of Transient Equations using Recovery, Fitting, Upwinding and Limiters

3.1 Test Equations and Boundary Conditions

The *Masterful* scheme may be readily implemented to provide numerical solutions to the following differential equations,

$$u_t + au_x = 0 \qquad a = const \qquad \text{Linear Advection} \tag{3.1}$$

$$u_t + uu_x = 0 \qquad \text{Inviscid Burgers' Equation} \tag{3.2}$$

$$u_t + uu_x = \varepsilon u_{xx} \qquad \varepsilon = const > 0 \qquad \text{Viscous Burgers' Equation} \tag{3.3}$$

Solutions to these equations will be sought on the unit interval, $0 \leq x \leq 1$. This raises the need to find suitable boundary conditions to impose. For the type of solution sought here, in which the major interest is away from the boundaries, Dirichlet boundary conditions will be used, obtained from the initial data. For the test problems to be considered there exist concise analytic solutions, against which the accuracy of the numerical solutions may be tested.

3.2 Outline of the *Masterful* Scheme

The scheme is implemented on a grid which subdivides the unit interval into $J+1$ cells, or subintervals. Such a grid has J internal nodes, whose positions are free to be varied as the scheme proceeds. The nodes at either end of the interval remain fixed throughout.

The *Masterful* scheme is constructed from a first order upwind scheme and the node adjusting algorithm [2], combined in the following steps :

Scheme

a) Apply the upwind scheme to current solution, producing a first approximation to the solution at the next time level

b) Apply the best-fit algorithm to piecewise linear recovered function (see section 3.3.2.) This produces new nodal positions, which are better suited to represent the solution at the new time level

c) Use the nodal displacements to obtain a new frame of reference

d) Apply the upwind scheme to the current solution a second time, but this time the differential equation is solved within the new frame

3.3 Computational Details

Consider the differential equation

$$\frac{\partial u}{\partial t} + a(u,x,t)\frac{\partial u}{\partial x} = 0. \tag{3.4}$$

The scheme advances the numerical solution of this equation, through one time interval $\triangle t$, according to the following steps :

3.3.1 Predictor Step

Let $\{u_j^* : j = 0, 1, \cdots, J+1\}$ be the finite difference solution of (3.4) obtained by an upwind step on the data at time level t_n,

$$u_k^* = u_k^* - a_{j-\frac{1}{2}}^n \frac{\triangle t}{(\triangle x_j)^n}(u_j^n - u_{j-1}^n) \quad j = 1, \cdots, J \tag{3.5}$$

$$\text{where} \qquad k \;=\; \begin{cases} j & \text{if} \quad a_{j-\frac{1}{2}}^n > 0 \\ j-1 & \text{if} \quad a_{j-\frac{1}{2}}^n < 0 \end{cases}$$

$$\text{and} \qquad a_{j-\frac{1}{2}}^n \;=\; \frac{a_j^n + a_{j-1}^n}{2}$$

3.3.2 Grid Adaption Step

The grid adaption routine is based on a technique for obtaining piecewise constant best L_2 fits, to a piecewise linear continuous function [2]. The motivation for using this procedure, is the *Donor Cell* interpretation of upwind schemes. In this formalism the region is considered to be divided into cells. Each cell contains one node, where the cell value of the solution is stored, with cell interfaces midway between nodes, *figure* 3.1. In (3.5) the spatial difference term is considered to represent a flux difference across cell walls. Upwinding is achieved by selecting

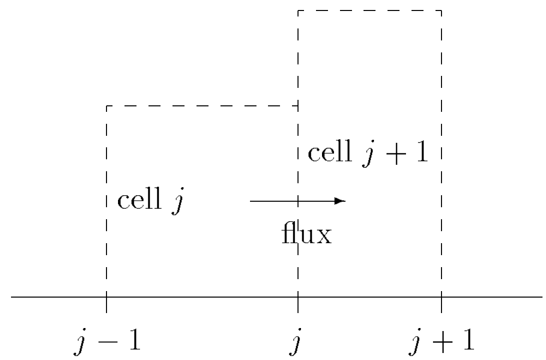

Figure 3.1: Donor Cell

which of the neighbouring cells receives this flux contribution, according to the direction of flow.

Hence the data produced by the predictor step constitutes a piecewise constant function. In order to apply the best fit algorithm, to obtain nodal positions better suited to represent this function, a piecewise continuous linear function must first be recovered. There are many ways to make such a recovered function, but only two will be considered here.

(a) linear interpolation between nodes

(b) linear interpolation between cell centres

The first option is the simplest to implement. The second provides a recovered function which is less prone to rapid variations of gradient than is the first. This latter smoothing property is perhaps counter to the desired aim of reducing the numerical diffusion in the solution, and will not be considered further here.

Given the recovered linear function obtained by option (a) above, new nodal positions are calculated for a best L_2 piecewise constant fit. In fact only one iteration of the algorithm which achieves this fit, is performed. Extra computational effort expended in this stage of the scheme, would not be matched by the accuracy attainable in the upwind solution stages.

If the nodal positions at time t_{n+1} are x_j^{n+1}, then the nodal displacements, $(\triangle x^{n+1})_j = x_j^{n+1} - x_j^n$, are given [1] by

$$(\triangle x^{n+1})_j = \begin{cases} \dfrac{\delta^2 u_j^*}{4\triangle u_j^*}(\triangle x_j)^n & \text{if } \left|\triangle u_j^*\right| > \left|\triangle u_{j+1}^*\right| \\ \dfrac{\delta^2 u_j^*}{4\triangle u_{j+1}^*}(\triangle x_{j+1})^n & \text{if } \left|\triangle u_j^*\right| < \left|\triangle u_{j+1}^*\right| \end{cases}$$

where $j = 1, \cdots, J$. This algorithm has the property that nodes can not overtake one another. The nodes at either end of the region remain fixed.

3.3.3 Transformation to the Moving Frame

Since the donor cell scheme employs velocities evaluated at cell walls, the frame velocity must also be calculated there for computational stability. Hence the following quantities, which describe the velocity field of the moving frame, are required

$$\dot{x}_{j-\frac{1}{2}}^n = \frac{1}{2}\left[\frac{x_j^{n+1} - x_j^n}{\triangle t} + \frac{x_{j-1}^{n+1} - x_{j-1}^n}{\triangle t}\right]$$

where $j = 1, \cdots, J+1$.

3.3.4 Corrector Step

The upwind scheme is applied to the data $\{u_j^n : j = 0, 1, \cdots, J+1\}$, on the grid $\{x_j^n : j = 0, 1, \cdots, J+1\}$, for a second time. This time allowance is made for the movement of the nodes during the time step. In [1] a discretisation of the moving frame equation (2.4) is used for this purpose

$$u_k^{n+1} = u_k^n - (a_{j-\frac{1}{2}}^n - \dot{x}_{j-\frac{1}{2}}^n)\frac{\triangle t}{(\triangle x_j)^n}(u_j^n - u_{j-1}^n) \quad j = 1, \cdots, J$$

$$\text{where} \quad k = \begin{cases} j & \text{if} \quad a^n_{j-\frac{1}{2}} - \dot{x}^n_{j-\frac{1}{2}} > 0 \\ j-1 & \text{if} \quad a^n_{j-\frac{1}{2}} - \dot{x}^n_{j-\frac{1}{2}} < 0 \end{cases}$$

Since both upwind passes preserve monotonicity , and the grid adaption method maintains the ordering of the nodes, the scheme as a whole is also monotonicity preserving.

3.4 Discretisation in the Moving Frame

The first two stages of the scheme are straight forward : first obtain a prediction for the solution at the next time level ; then adjust the nodal positions to give a better fit to this new solution. The nodal displacements per time step describe the required velocity of the frame of reference. In the last stage the upwind method is to be applied in this moving frame.

In [1], the differential equation in the Lagrangian frame is discretised in the form (2.4). However this was only notational shorthand for (2.3).

Consider the case where the differential equation is

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0$$

The equation in the moving frame, (2.3), is now

$$\frac{\partial u}{\partial t} + \frac{(u - \dot{x})}{x_\xi}\frac{\partial u}{\partial \xi} = 0 \tag{3.6}$$

where $u, t, \dot{x}$ are now used to denote $\hat{u}, \tau, x_\tau$ respectively.

A discretisation of (3.6) can be obtained by employing the identities between derivatives of the transformation, (2.1).

$$\xi_x = \frac{1}{x_\xi} \tag{3.7}$$

$$\xi_t = -\frac{\dot{x}}{x_\xi} \tag{3.8}$$

Using (3.7), (3.6) becomes

$$\frac{\partial u}{\partial t} + (u - \dot{x})\frac{\partial \xi}{\partial x}\frac{\partial u}{\partial \xi} = 0$$

This equation poses the problem of finding a discretisation for $\frac{\partial \xi}{\partial x}$.

In the moving frame coordinate system, the grid remains stationary with a constant spacing. There is freedom to choose any convenient spacing, say $\triangle\xi = 1$. The moving frame coordinate, ξ, now takes the same value as the node index at each node,

$$\xi(x_j^n, n\triangle t) = j \tag{3.9}$$

A differential equation for ξ is given by (3.8),

$$\frac{\partial \xi}{\partial t} + \dot{x}\frac{\partial \xi}{\partial x} = 0. \tag{3.10}$$

The coordinate function, ξ, must satisfy this equation together with the initial data,

$$\xi(x, 0) = \frac{x}{\triangle x}$$

where $\triangle x$ is the initial regular grid spacing.

It is not necessary to solve (3.10) completely, since it is only $\frac{\partial \xi}{\partial x}$ which is required. An approximation to this is found by making a time discretisation of (3.10),

$$\frac{\xi^{n+1} - \xi^n}{\triangle t} + \dot{x}^n\frac{\partial \xi^n}{\partial x} = 0$$

The scheme already employs a discretisation of $\dot{x}$, evaluated at $(x^n_{j-\frac{1}{2}}, n\triangle t)$. Using this, an approximation for $\frac{\partial \xi^n}{\partial x}(x^n_{j-\frac{1}{2}})$ is obtained,

$$\frac{\partial \xi^n}{\partial x}(x^n_{j-\frac{1}{2}}) = \frac{\xi^n(x^n_{j-\frac{1}{2}}) - \xi^{n+1}(x^n_{j-\frac{1}{2}})}{\dot{x}^n_{j-\frac{1}{2}}\triangle t} \tag{3.11}$$

Now ξ has been chosen such that $\xi(x_j^n, n\triangle t) = j, \quad j = 0, 1, \cdots, J+1 \quad \forall n$

By applying linear interpolations to this data, the two quantities in the numerator of (3.11) are found to have the following approximations,

$$\begin{aligned} \xi^n(x^n_{j-\frac{1}{2}}) &= j - \frac{1}{2} \\ \xi^{n+1}(x^n_{j-\frac{1}{2}}) &= j - 1 + \frac{x^n_{j-\frac{1}{2}} - x^{n+1}_{j-1}}{x^{n+1}_j - x^{n+1}_{j-1}}. \end{aligned}$$

Details of the calculations involved are given in the Appendix. Combining these with

$$\dot{x}^n_{j-\frac{1}{2}} = \frac{1}{2}\left(\frac{x^{n+1}_j - x^n_j}{\triangle t} + \frac{x^{n+1}_{j-1} - x^n_{j-1}}{\triangle t}\right)$$

equation (3.11) reduces to,

$$\frac{\partial \xi^n}{\partial x}(x^n_{j-\frac{1}{2}}) = \frac{1}{x^{n+1}_j - x^{n+1}_{j-1}}$$

When this is written into the upwind scheme, the resulting discretisation differs from that of (2.4) only in the calculation of the space interval. In this new scheme the grid differences are calculated at the new time level, instead of the current level. A comparison between the two discretisations is given in the next section.

3.5 Numerical Results

The schemes derived from equations (2.4) and (2.3) are here referred to as the first and second discretisations respectively.

A comparison is made between the two discretisations applied to the test problem,

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0$$

with initial data,

$$u(x,0) = \frac{1}{2} - x.$$

The measure of the error used here is

$$\text{error} = \frac{\left\{\sum_{j=0}^{J+1}[u(x_j^n, t_n) - u_j^n]^2\right\}^{\frac{1}{2}}}{J+1}$$

The results, in *table* 3.1, are typical of the two discretisations. The second form of discretisation provides a slight improvement in accuracy, in most cases.

Number of free nodes	Error at $t = 0.5$		Steps required	
	First discretisation	Second discretisation	First	Second
9	6.53×10^{-3}	7.77×10^{-3}	4	4
19	3.65×10^{-3}	3.03×10^{-3}	8	8
39	1.66×10^{-3}	1.30×10^{-3}	16	15
59	9.88×10^{-4}	7.72×10^{-4}	24	23
99	5.00×10^{-3}	3.89×10^{-4}	40	38

Table 3.1: Error comparison

Chapter 4

Convection-Diffusion Problems

So far, the scheme, presented in Chapter 3, has only been applied to problems which involve the development of *static*, steep fronts in the solution. The model equation for this purpose is the inviscid Burgers' Equation

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0 \tag{4.1}$$

The numerical results, given in *table* 3.1, exhibit quadratic convergence to the exact solution, as the number of nodes is increased. This demonstrates that the scheme significantly reduces the size of the numerical diffusion term, present in the modified equation for the upwind discretisation.

It was suggested that, as a next stage in the development of the scheme, problems involving physical diffusion processes should be investigated. The addition of a diffusion term to (4.1) provides a suitable model equation. Consequently, we shall now consider numerical solutions of the viscous Burgers' Equation,

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \varepsilon\frac{\partial^2 u}{\partial x^2} \quad \text{where} \quad \varepsilon \geq 0 \tag{4.2}$$

In the case of static fronts, equation (4.2) poses no new obstacles for the scheme. The static fronts for (4.2) merely differ from those for (4.1) in being smeared over a larger region, and less steep. Since the monotonicity preserving property of the scheme ensures sharp resolution of steep fronts for (4.1), the more rounded solutions of (4.2) should cause no extra difficulties. For this reason a test problem is chosen which possesses a moving front, in addition to the extra physical diffusion.

Before this problem can be tackled, it is first necessary to obtain a discretisation of the diffusion term in (4.2), and its associated stability requirements.

4.1 Discretisation of the Diffusion Term

In Section 2.1, two forms of the differential equation in the moving frame were given : the fully transformed equation (2.3), and a notationally shortened version (2.4). When a discretisation for the inviscid Burgers' Equation was required in Section 3.4, a comparison was made between discretisations based on these two equations. It was found that slightly better results may be obtained by using the full form, (2.3).

Again it is possible to identify two forms of (4.2), when it is transformed to a frame moving with velocity $\dot{x}$.

4.1.1 Full Discretisation

Consider the viscous Burgers' Equation, (4.2),

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \varepsilon\frac{\partial^2 u}{\partial x^2} \quad \text{where} \quad \varepsilon \geq 0$$

Each term may be written in (ξ, τ) coordinates by use of the elements of the Jacobian of the transformation,

$$\begin{aligned} \xi_x &= \frac{1}{x_\xi} \qquad \xi_t = -\frac{\dot{x}}{x_\xi} \\ \tau_x &= 0 \qquad \tau_t = 1 \end{aligned}$$

The time derivative transforms as follows,

$$\begin{aligned} \frac{\partial u}{\partial t} &= \hat{u}_\xi \xi_t + \hat{u}_\tau \tau_t \\ &= \hat{u}_\tau - \dot{x}\xi_x \hat{u}_\xi \end{aligned}$$

First space derivative,

$$\begin{aligned} \frac{\partial u}{\partial x} &= \hat{u}_\xi \xi_x + \hat{u}_\tau \tau_x \\ &= \xi_x \hat{u}_\xi \end{aligned}$$

And the second space derivative,

$$\begin{aligned} \frac{\partial^2 u}{\partial x^2} &= \frac{\partial}{\partial x}(\xi_x \hat{u}_\xi) \\ &= \xi_{xx}\hat{u}_\xi + \xi_x(\hat{u}_{\xi\xi}\xi_x + \hat{u}_{\xi\tau}\tau_x) \\ &= \xi_{xx}\hat{u}_\xi + (\xi_x)^2 \hat{u}_{\xi\xi} \end{aligned}$$

Hence the equation in the moving frame is

$$\frac{\partial \hat{u}}{\partial \tau} + [(u - \dot{x})\xi_x - \varepsilon \xi_{xx}]\frac{\partial \hat{u}}{\partial \xi} = \varepsilon(\xi_x)^2 \frac{\partial^2 \hat{u}}{\partial \xi^2} \qquad (4.3)$$

In order to discretise the ξ_{xx} term, consider the following,

$$\begin{aligned} \xi_x x_\xi = 1 \quad &\Rightarrow \quad \frac{\partial}{\partial x}(\xi_x x_\xi) = 0 \\ &\Rightarrow \quad \xi_{xx}x_\xi + \xi_x(x_{\xi\xi}\xi_x + x_{\xi\tau}\tau_x) = 0 \\ &\Rightarrow \quad \xi_{xx}x_\xi + (\xi_x)^2 x_{\xi\xi} = 0 \\ &\Rightarrow \quad \xi_{xx} = -(\xi_x)^3 x_{\xi\xi} \end{aligned}$$

In the previous Chapter, a discretisation of ξ_x was found,

$$\xi_x = \frac{1}{x_\xi} \sim \frac{1}{x_j^{n+1} - x_{j-1}^{n+1}} \quad \Rightarrow \quad (x_\xi)_{j-\frac{1}{2}}^n = x_j^{n+1} - x_{j-1}^{n+1}$$

By applying first differences to this, an approximation for $x_{\xi\xi}$ is obtained,

$$(x_{\xi\xi})_j^n = \frac{(x_\xi)_{j+\frac{1}{2}}^n - (x_\xi)_{j-\frac{1}{2}}^n}{\triangle\xi}$$

Using $\triangle\xi = 1$, this reduces to the simple relation,

$$(x_{\xi\xi})_j^n = x_{j+1}^{n+1} - 2x_j^{n+1} + x_{j-1}^{n+1}$$

These results produce the following upwind discretisation in the moving frame,

$$\frac{u_k^n - u_k^n}{\triangle t} + \left[\frac{\nu_{j-\frac{1}{2}}^n}{\triangle t} + \varepsilon\mu_k^n\right](u_j^n - u_{j-1}^n) = \varepsilon\frac{(u_{k+1}^n - 2u_k^n + u_{k-1}^n)}{(x_k^{n+1} - x_{k-1}^{n+1})^2} \tag{4.4}$$

$$\begin{aligned} \text{where} \quad \nu_{j-\frac{1}{2}}^n &= (u_{j-\frac{1}{2}}^n - \dot{x}_{j-\frac{1}{2}}^n)\left(\frac{\triangle t}{x_j^{n+1} - x_{j-1}^{n+1}}\right) \\ \mu_k^n &= \frac{x_{k+1}^{n+1} - 2x_k^{n+1} + x_{k-1}^{n+1}}{(x_k^{n+1} - x_{k-1}^{n+1})^3} \end{aligned}$$

$$\text{and} \qquad k = \begin{cases} j & \text{if} \quad \nu_{j-\frac{1}{2}}^n + \varepsilon\mu_k^n > 0 \\ j-1 & \text{if} \quad \nu_{j-\frac{1}{2}}^n + \varepsilon\mu_k^n < 0 \end{cases}$$

4.1.2 Abbreviated Equation

As an alternative to discretising equation (4.3), which exhibits the transformation explicitly, the diffusion term can be retained in the form $\varepsilon\dfrac{\partial^2 u}{\partial x^2}$. This is then discretised consistently using central differences on an irregular grid

$$\frac{\partial^2 u}{\partial x^2} \rightarrow \frac{2}{x_{j+1}^n - x_{j-1}^n}\left(\frac{u_{j+1}^n - u_j^n}{x_{j+1}^n - x_j^n} - \frac{u_j^n - u_{j-1}^n}{x_j^n - x_{j-1}^n}\right)$$

4.1.3 Stability of the Scheme

Fourier stability analysis for a scheme of the form,

$$\frac{u_j^n - u_j^n}{\triangle t} + a\frac{u_j^n - u_{j-1}^n}{\triangle x} = \varepsilon\frac{(u_{j+1}^n - 2u_j^n + u_{j-1}^n)}{(\triangle x)^2},$$

provides an indication of the quantities which are of importance to the stability of the explicit schemes to be considered here :

If $\nu = a\dfrac{\triangle t}{\triangle x}$ and $\mu = \varepsilon\dfrac{\triangle t}{(\triangle x)^2}$ then

- The necessary stability condition involves terms due to diffusion, as well as convection. For this particular example, $|\nu| + 2\mu < 1$.

- A measure of the relative importance of convection and diffusion processes, is given by the mesh Peclet number,

$$P_e = \frac{|\nu|}{2\mu}. \tag{4.5}$$

When the scheme, obtained by the method of Section 4.1.2, is implemented, some method must be found for selecting the time increment, $\triangle t$, to ensure stability. In general this value will change from step to step. It is a simple matter to select a value, of $\triangle t$, for which the condition above is satisfied by the data at the current time level. Furthermore, the adaptive scheme has the flexibility to allow $\triangle t$ to be changed, if need be, to satisfy the stability requirements of the second upwind pass. This is due to the fact that, the second pass only uses the nodal displacements, $\dot{x}\triangle t$, and not the nodal velocities.

However, for the discretisation in Section 4.1.1, the matter is not so simple. The second upwind pass now involves quantities from the Jacobian of the transformation. The transformation depends on the frame velocity, and not just on its displacement. Hence it is no longer valid to change $\triangle t$ between upwind passes. Some method must be found to select $\triangle t$, from the data at time t_n, which will guarantee the stability of both upwind passes. Such a method might be found from inequalities which provide bounds on the nodal movement. An alternative approach would be to make the scheme implicit, and so remove the stability constraint on $\triangle t$.

4.2 Results

The following results were obtained for the simple form of discretisation outlined in Section 4.1.2. Graphical output is given for the test problem derived from an analytical solution [5] of (4.2) :

$$u(x,t) = \mu - \alpha \tanh\left[\frac{\alpha}{2\varepsilon}(x - \beta - \mu t)\right]$$

The parameter values are chosen to be $\alpha = 0.4$, $\mu = 0.6$ and $\beta = 0.125$ This solution represents a steep front which progresses across the region from left to right. The value at the left hand boundary is held fixed at the initial value. The number of nodes used is 21.

There are two distinct regimes of behaviour possible for a convection-diffusion scheme. These are distinguished by the mesh Peclet number, P_e, which provides an comparison of how accurately the scheme represents the separate processes of diffusion and convection. A high value of P_e indicates that the convection process is the principal source of error. This may be seen by considering the non-dimensional quantity, μ, which appears in the denominator of (4.5).

$$\mu = \frac{\varepsilon}{\frac{(\triangle x)^2}{\triangle t}}.$$

If the discretisation measure, $\frac{(\triangle x)^2}{\triangle t}$, is small in comparison to the physical diffusion strength, ε, then the scheme models the diffusion process with great accuracy. This corresponds to a small value of, μ, and a consequently large value of P_e. It is when the value of P_e is large that convection-diffusion problems pose the greatest challenge to numerical methods.

For the current test problem, values of ε were chosen to provide high and low mesh Peclet numbers.

4.2.1 Low Peclet Number : $P_e \sim 0.01$, $\varepsilon = 0.01$

Figure 4.1 contains a graph showing the nodal positions, with time represented vertically. The thick line shows the position of the centre of the front, as it moves across the grid. Beneath this, is shown the numerical solution at times $t = 0.0$, 0.25, 0.5, 0.75, 1.0. The dotted lines show the corresponding exact solutions. This *figure* identifies two principal defects in the numerical solution.

First, the initial regular grid results in deterioration of the solution during the first few time steps. As the grid becomes better adapted to the solution, numerical diffusion decreases and the solution deteriorates much less rapidly.

Secondly, the grid fails to match the speed of the front, which quickly moves beyond most of the grid.

The first of these problems, is due to the fact that the time step is being chosen to suit stability requirements only. *Figure* 4.2 shows how the total time elapsed changes as the solution is stepped forward. As nodes converge on the steep front, the minimum of the grid spacings decreases, which forces $\triangle t$ to decrease to maintain stability. Once a distribution of nodes is achieved which fits

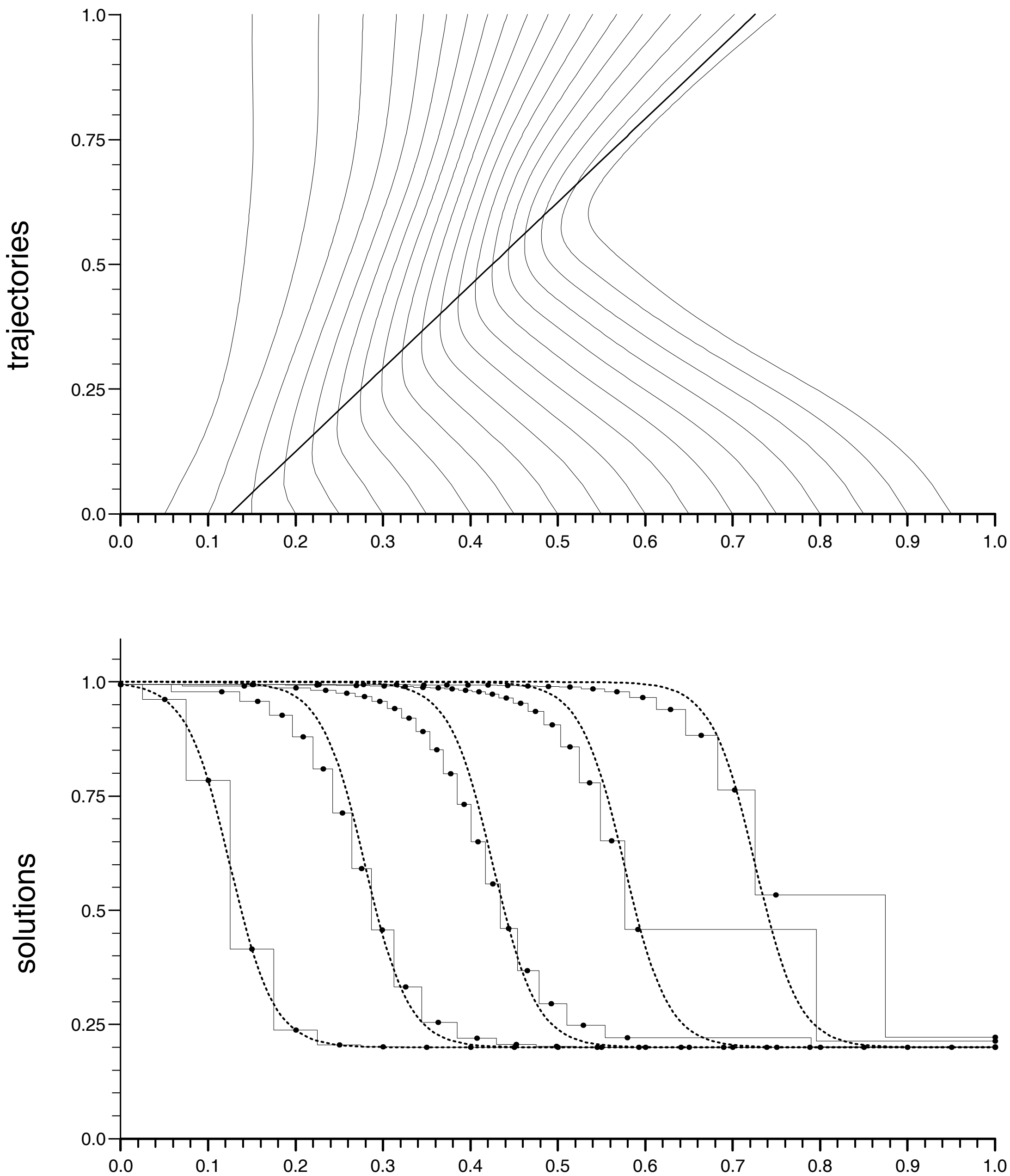

Figure 4.1: Moving Front with Low Peclet Number

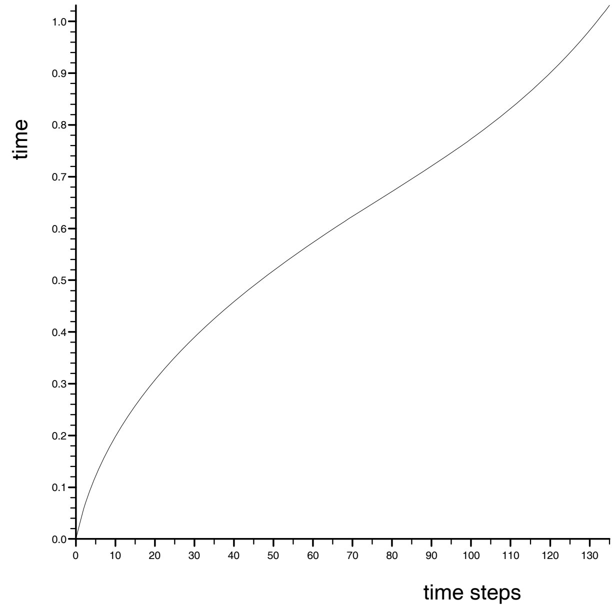

Figure 4.2: Time Elapsed v. Steps

the front well, the time step remains constant. After about 100 steps the front has almost left the grid completely, causing the nodes to separate again, to give a good representation of the constant valued solution which remains.

To remedy this problem, a measure of the goodness of fit to the solution is required. This will indicate if the node moving algorithm is failing to reduce numerical diffusion. In which case the time step should be reduced to maintain the required accuracy, until the nodes are better placed.

The second of the problems, outlined above, will be considered in the next Chapter.

4.2.2 High Peclet Number : $P_e \sim 0.1$, $\varepsilon = 0.001$

The difficulties which arose for the low Peclet number, are even more severe when the scheme attempts to solve the convection dominated problem. *Figure* 4.3

shows that the front effectively leaves the grid at about $t = 0.7$, and the solution becomes meaningless.

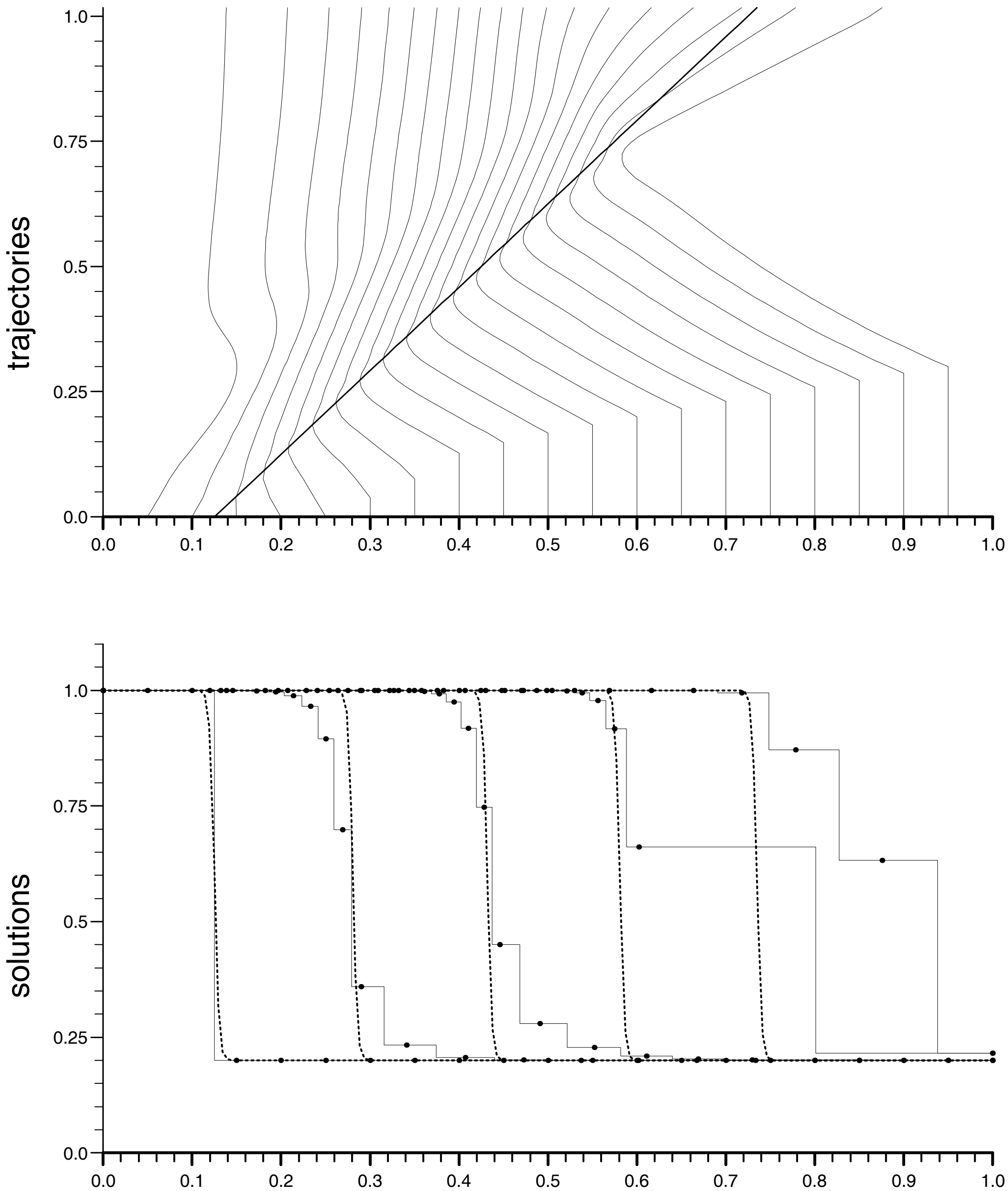

Figure 4.3: Moving Front with High Peclet Number

Chapter 5

Attempts to Solve the Problems Posed by Convection-Diffusion

5.1 Node Speed

The results given in Chapter 4 indicate that the adaptive scheme, so far developed, is not suited to the task of modelling a moving front. Grid adaption causes the separation between nodes to decrease near a steep front. In the Appendix it is shown that, at each time step, each node is limited to move, at most, one quarter of the distance towards either of its nearest neighbours. Hence, unless the time step is suitably reduced, there is a severe restriction on the translational speed attainable by the section of grid which contains the front.

The only control which the scheme allows over the speed of the nodes, is provided by the time step $\triangle t$. By reducing $\triangle t$, a small nodal displacement during one time step can represent a large nodal speed. In order to investigate the effect of different selections for $\triangle t$, it is convenient to remove restrictions due to stability of the upwind scheme. A fully implicit, unconditionally stable version of the scheme was applied to the problems presented in Chapter 4.

5.1.1 Results for Implicit Scheme

The fully implicit scheme requires a tridiagonal matrix to be solved at each time step. An algorithm employing LU decomposition, with forward and backward substitutions, is used for this purpose.

Figures 5.1, 5.2 show the result of applying the implicit scheme to the test problem of Chapter 4. In each case the time step has been selected, by trial and error, to obtain the most accurate solution.

The implicit scheme appears to offer little, if any, improvement on the explicit scheme, and will not be considered further.

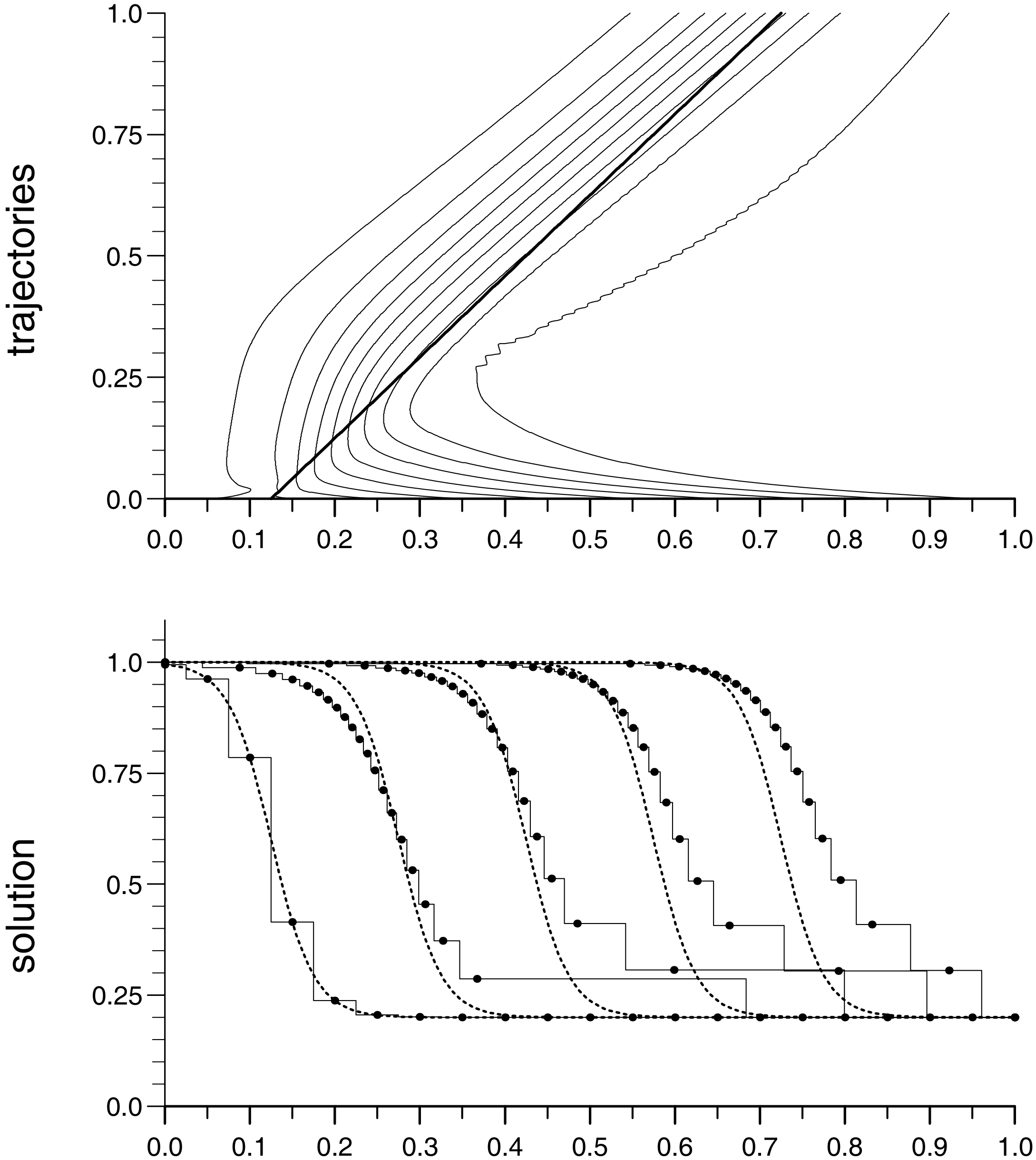

Figure 5.1: Implicit Scheme : Low Peclet Number

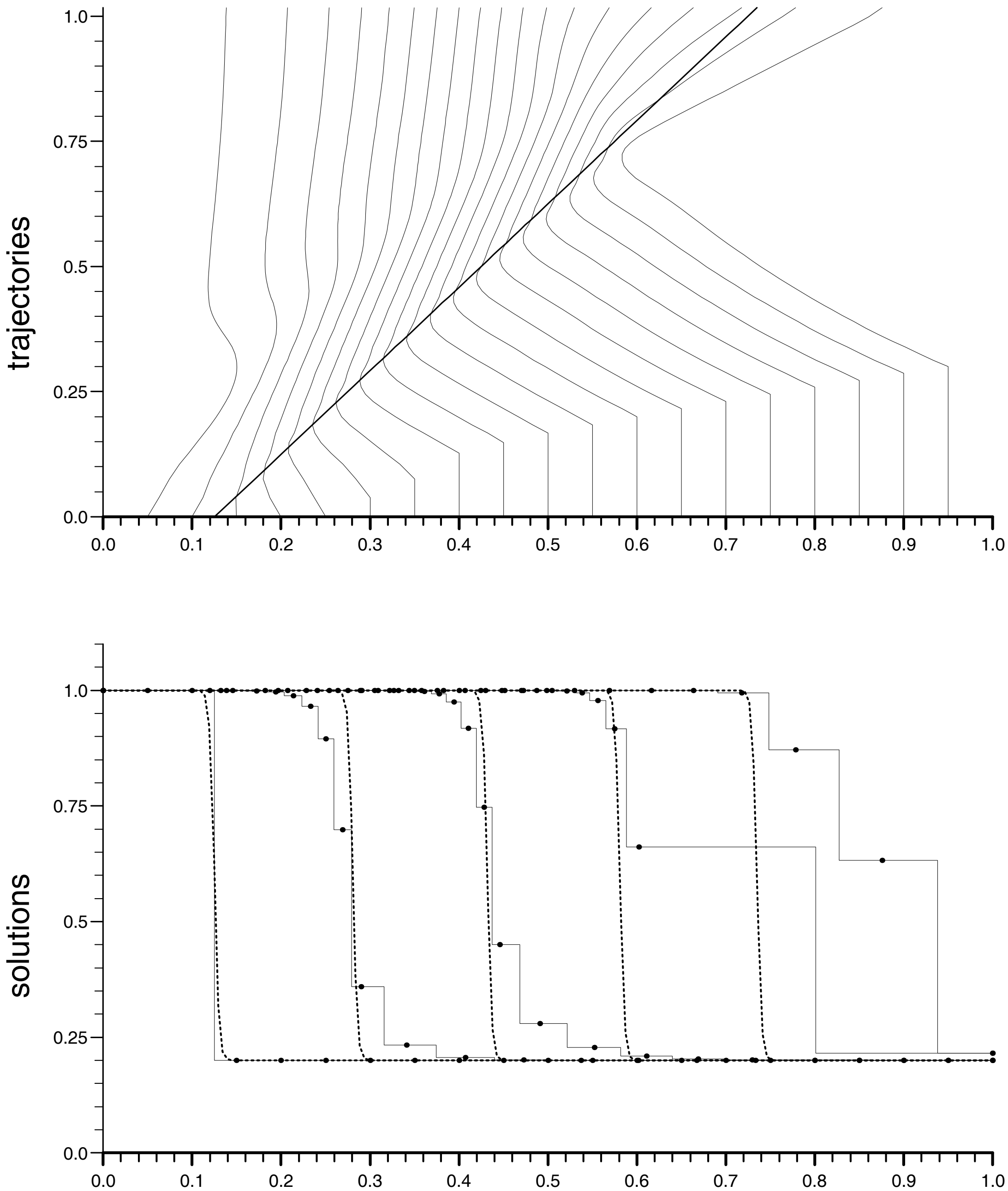

Figure 5.2: Implicit Scheme : High Peclet Number

5.2 Drift Velocity and Numerical Diffusion

It now seems to be necessary, to remove the requirement of the grid to follow the moving front. This is achieved by imposing the drift velocity (i.e. the velocity of displacement) of the front onto the grid. The grid is now co-moving with the front. Any grid adjustment will be solely a response to numerical diffusion and the changing geometry (if any) of the front.

In Chapter 3, it was demonstrated that the scheme is efficient at reducing the numerical diffusion present at a static front. However for the particular test problem considered, the initial regular grid provides a good representation of the initial data. Thus it is able to keep pace with the developing steep front, and so reduce the numerical diffusion present for each time step.

For the test problem in Chapter 4, the initial regular node distribution disregards the presence of a localised steep front in the initial data. It takes several time steps for the adaption algorithm to correct for this. When a better node distribution is achieved, numerical diffusion is drastically reduced but the early loss of accuracy has degraded the solution. The initial ineffectiveness of the grid adusting algorithm, must be compensated for by other means. A search for a suitable time step control, based on the limits on node movement given in the Appendix, was not successful.

5.2.1 Adaptive Scheme in a Moving Frame

The problem set out in Chapter 4 is now recast in a slightly different form. The initial function is the same tanh curve, but now given a central position in the region. The grid on which the adaptive scheme is to work is set in a frame which is moving with the speed of the front. *Figure* 5.3 shows that, under these conditions, the nodes eventually move to stable positions around the front. Once the grid has reached this stable configuration, the solution displays a dramatic reduction in numerical diffusion. The remaining error, which can be seen in the lower graph (solution at $t = 1.0$), is almost entirely a remnant of the early ill-placing of nodes.

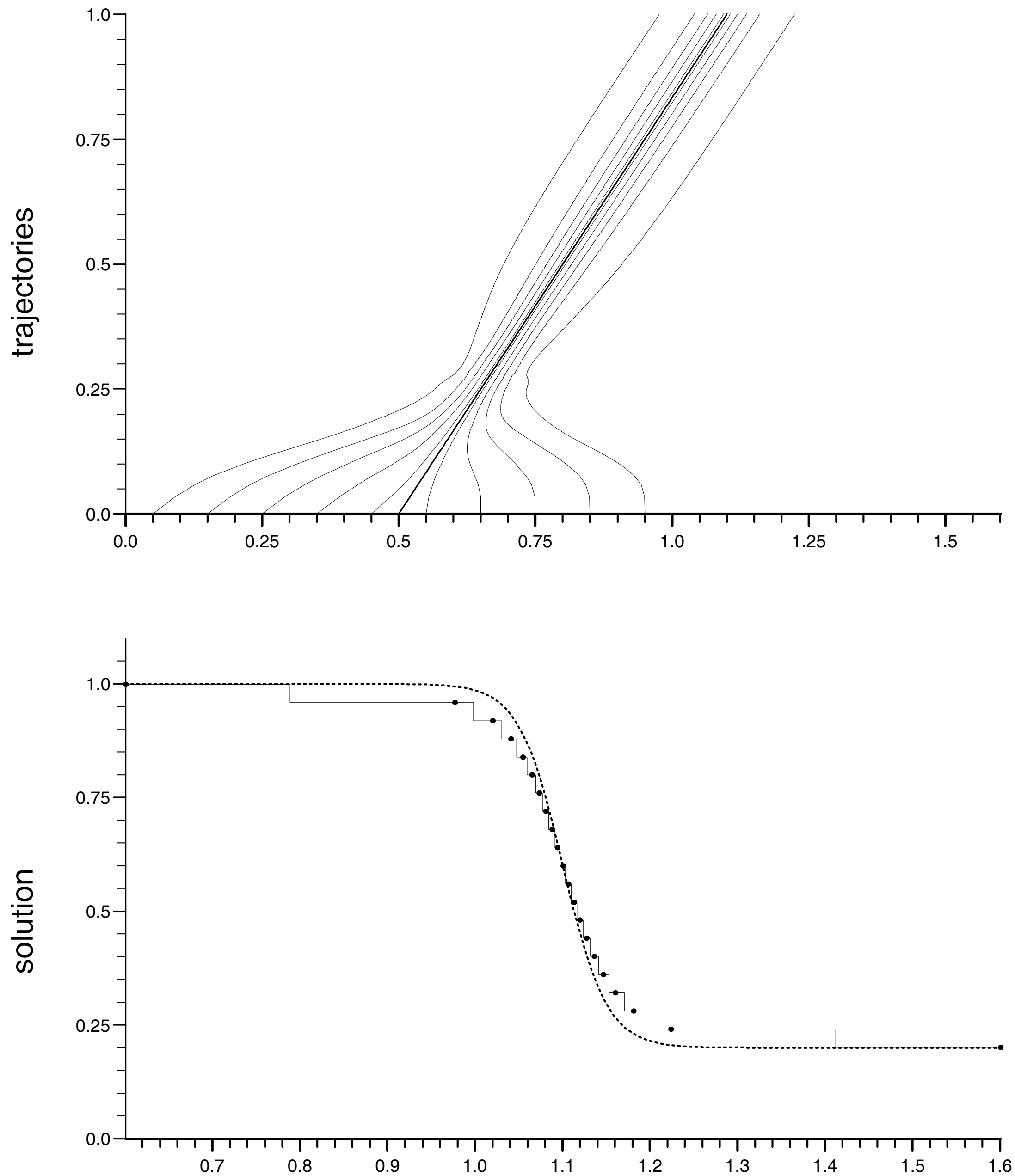

Figure 5.3: Grid with Drift Velocity

Conclusion

Chapter 1 introduced the problems, associated with first order schemes, for solving differential equations involving convection. The principal source of error was seen to be the phenomenon of numerical diffusion. The technique of the modified equation was used to analyse the numerical diffusion, associated with a simple upwind scheme for convection. The numerical diffusion term of the modified equation (1.4), demonstrates the strong dependence, of numerical diffusion, on the grid spacing in regions of high curvature.

This provided the motivation for examining adaptive grid schemes in Chapter 2. Two basic methods of grid adaption were outlined. Dynamic Rezone Methods involve grids which move with features of the solution, such as sharp fronts. In Static Rezone Methods individual grid nodes do not track the solution, but are distributed to give a good representation of the solution.

In Chapter 3, an existing adaptive finite difference scheme, [1], was discussed. A test was carried out, between the original form of this scheme and a modified version. This second form of the scheme was obtained from a discretisation of the equation, in a moving frame of reference.

The adaptive convection scheme, presented in Chapter 3 and extended to convection-diffusion in Chapter 4, shows the ability to reduce numerical diffusion.

However, the grid is seen to be unable to respond to moving features in the solution, Section 4.2. This indicates that the scheme is not of the Dynamic Rezone type.

In Chapter 5, the adaptive grid is given the translational speed of the front to be tracked. This drift velocity must be found by means independent of the adaptive scheme. The numerical solution, though much improved, still suffers deterioration due to numerical diffusion on the initial, ill-adapted grid. This problem may be remedied by monitoring the goodness of fit of the grid to the solution. If the grid is not well adapted, other means must be employed to reduce numerical diffusion and maintain accuracy.

Appendix

Let $\{u_j^* : j = 0, \cdots, J+1\}$ be the result of the upwind predictor step and also be monotonic. The grid adaption routine moves the nodes according to the following scheme [1],

$$x_j^{n+1} = x_j^n + (\triangle x^{n+1})_j \tag{i}$$

$$\text{where} \qquad (\triangle x^{n+1})_j = \frac{1}{4}\frac{\delta^2 u_j^*}{\triangle u_j^*}(\triangle x_j)^n \qquad \text{if } \left|\triangle u_j^*\right| > \left|\triangle u_{j+1}^*\right|$$

$$\text{and} \qquad (\triangle x^{n+1})_j = \frac{1}{4}\frac{\delta^2 u_{j+1}^*}{\triangle u_{j+1}^*}(\triangle x_{j+1})^n \quad \text{if } \left|\triangle u_j^*\right| < \left|\triangle u_{j+1}^*\right|$$

Consider the case $\left|\triangle u_j^*\right| > \left|\triangle u_{j+1}^*\right|$. Since $\{u_j^*\}$ is monotonic there can only be two possibilities,

a. $\triangle u_j^* > \triangle u_{j+1}^*$

b. $-\triangle u_j^* > -\triangle u_{j+1}^*$

Both (a) and (b) result in the same inequality for ratios of the function differences,

$$1 > \frac{\triangle u_{j+1}^*}{\triangle u_j^*} > 0$$

The node movement formula (i) for this particular case is,

$$(\triangle x^{n+1})_j = \frac{1}{4}\left(\frac{\triangle u_{j+1}^*}{\triangle u_j^*} - 1\right)(\triangle x_j)^n$$

A similar inequality is obtained for the case $\left|\triangle u_j^*\right| < \left|\triangle u_{j+1}^*\right|$. Combining these inequalities and using the definition,

$$(\triangle x_j)^n = x_j^n - x_{j-1}^n$$

the following limit on the movement of node j, in one time step, is obtained :

$$\frac{x_{j-1}^n + 3x_j^n}{4} < x_j^{n+1} < \frac{3x_j^n + x_{j+1}^n}{4} \tag{ii}$$

When this inequality is combined with, $x_{j-\frac{1}{2}}^n = \dfrac{x_{j-1}^n + x_j^n}{2}$, it is a simple matter to obtain,

$$x_{j-1}^n < x_{j-\frac{1}{2}}^n < x_j^{n+1} \tag{iii}$$

This inequality is the justification for the particular linear interpolation, which is chosen to obtain $\xi^{n+1}(x_{j-\frac{1}{2}}^n)$ in Section 3.4.

Bibliography

[1] Baines M.J. (1992) *Monotonic Adaptive Solutions of Transient Equations using Recovery, Fitting, Upwinding and Limiters*. Numerical Analysis Report 5/92, University of Reading.

[2] Baines M.J. (1992) *Algorithms for Best Piecewise Discontinuous Linear and Constant L_2 Fits to Continuous Functions with Adjustable Nodes in One and Two Dimensions*. To appear.

[3] Furzeland R.M., Verwer J.G. & Zegeling P.A. (1990) *A Numerical Study of Three Moving Grid Methods for One-dimensional Partial Differential Equations which are Based on the Method of Lines*. Journal of Computational Physics **89**, 349-388.

[4] Hyman J.M. (1982) *Adaptive Moving Mesh Methods for Partial Differential Equations*. Los Alamos National Laboratory Report LA-UR-82-3690.

[5] Johnson I.W. (1986) *Moving Finite Elements for Diffusion Problems*. PhD Thesis, University of Reading.

[6] Margolin L.G. & Smolarkiewicz P.K. (1989) *Antidiffusive Velocities for Multipass Donor Cell Advection*. Lawrence Livermore National Laboratory UCID-21866.

[7] Petzold L.R. (1987) *Observations on an Adaptive Moving Grid Method for One-dimensional Systems of Partial Differential Equations*. Applied Numerical Mathematics **3**, 347-360.

[8] Smolarkiewicz P.K. (1983) *A Simple Positive Definite Advection Scheme with Small Implicit Diffusion*. Monthly Weather Review **111**, 479-486.

[9] Warming R.F. & Hyett B.J. (1974) *The Modified Equation Approach to the Stability and Accuracy of Finite Difference Methods*. Journal of Computational Physics **14**, 159-179.

Acknowledgements

I would like to thank Dr. M.J.Baines of the Mathematics Department, University of Reading for suggesting this project and for his guidance throughout. Also Drs. P.K.Sweby and A.Priestley for their assistance.

I would also like to acknowledge the financial support of the SERC.