THE UNIVERSITY OF READING

School of Mathematics, Meteorology & Physics

A Moving Mesh Method for the Discontinuous Galerkin Finite Element Technique

Alison Brass

August 2007

Funded by the Natural Environment Research Council (NERC)

This dissertation is a joint MSc in the Departments of Mathematics & Meteorology and is submitted in partial fulfilment of the requirements for the degree of Master of Science.

Acknowledgements

I would like to thank my supervisors Pete Sweby and Mike Baines for their many thoughts and ideas, and also, my friends and family for their support and encouragement when everything seemed to be going wrong. Thank you :)

Declaration

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

Alison Brass

Abstract

In this dissertation, velocity-based moving mesh methods for the discontinuous galerkin finite element technique are investigated and applied to solving linear and nonlinear conservations laws with periodic boundary conditions. Two main approaches for the method are considered. The first approach is cell-based and uses a conservation principle on each cell to derive the boundary speeds. The second approach is boundary-based, finding boundary speeds dependent on the local discontinuity in the numerical solution at each boundary. In both cases, the numerical solution to the conservation equation is then found on the updated mesh, using Lax Friedrichs numerical fluxes at the discontinuities when required. The latter method is then extended to the 1D system of shallow water equations and applications to simple tidal bore and dam-break problems are considered.

Contents

List of Figures

Chapter 1

Introduction

Many equations used in atmosphere and ocean modelling, including the Euler equations of gas dynamics and the shallow water equations, are conservation laws derived assuming the conservation of a particular quantity. Increasingly, Finite Element Methods (FEM) are being employed to solve such equations due to their ability to handle complex geometries. Research is ongoing into ways to improve the accuracy of the numerical solution without significantly increasing the computational cost and generally follows one of two routes.

Conservation laws often have discontinuous numerical solutions even if the initial data is smooth and continuous. Using standard FEM which work in the continuous domain, there is a limitation on how well sharp gradients and shocks can be captured, so it would seem natural to model the solution in a discontinuous manner. The Discontinuous Galerkin (DG) method developed by Reed and Hill [15] is an example of such a technique.

The other approach commonly used is to apply grid adaptation techniques to the standard FEM. Such techniques may include mesh refinments or the use of higher order polynomial approximations in the region of the shock, and Arbitrary Lagrangian-Eulerian (ALE) methods [16], also known as moving meshes, which cluster nodes around the feature and follow the feature as it moves over time.

In more recent years, research has looked at combining these two approaches to provide even better results, incorporating grid adaptation techniques into the discontinuous FEM. The use of a moving mesh algorithm with the DG technique was investigated by Li and Tang [14] who looked at mapping-based methods. We shall also consider the use of moving mesh algorithms but choose to focus on velocity-based methods instead.

In this dissertation we firstly look at the stationary DG method and in Chapter 2 we consider the Runge-Kutta Discontinuous Galerkin (RKDG) method developed by Cockburn and Shu [11]. In Chapter 3 we discuss various grid adaptation techniques before progressing to include some velocity-based moving mesh algorithms into the DG method in Chapters 4 and 5.

In Chapter 4, we focus on cell-based methods and derive the boundary velocities through imposing a conservation principle on each cell. We derive a cell-based method, and some variations, using the local Lax Friedrichs numerical flux at cell boundaries. Additionally, we derive a cell-based method where no flux calculations are required. Through solving simple linear and nonlinear test cases, we evaluate the success of these cell-based methods.

In Chapter 5, we derive a moving DG method without the use of the conservation principle seen in Chapter 4. In this method, the velocities may be obtained from an external source, and we consider a boundary-based method where the boundary speeds may be taken as the notional shock speed associated with the discontinuity in the numerical solution. The results of numerical tests are given in Chatper 6 where we also consider the case of zero boundary speeds and compare with the stationary RKDG method from Chapter 2.

Application of the boundary-based moving mesh method to a 1D system of nonlinear equations is considered in Chapter 7, where the method is derived for the shallow water equations. Some results of preliminary tests for the tidal bore problem and dam-break problem are given in Chapter 8.

Finally, in Chapter 9, we make some general conclusions and consider possible future work.

Chapter 2

The Stationary RKDG Method

2.1 History

The Discontinuous Galerkin (DG) method falls within the category of finite element techniques, using local basis functions to approximate the exact solution on each element. Developed by Reed and Hill [15] for solving the linear neutron transport equation $\sigma u + \nabla(au) = f$ where σ is real and a is linear, the DG method is noteably different from continuous methods in that it allows the numerical solution to be discontinuous across element boundaries.

Having been used for linear problems, the DG method was then extended to solve nonlinear problems including hyperbolic conservation laws which required the introduction of time-stepping algorithms. Early research by Chavent and Salzano [3] aimed to retain the elementwise calculations that were possible for the linear system, but in doing so, they encountered very restrictive stability criteria making the method impractical. Further research into the method included the introduction of slope limiters and TVD time-stepping algorithms to improve on the early results.

In particular, Cockburn and Shu [9] developed a high-order accurate Runge-Kutta

Discontinuous Galerkin (RKDG) scheme and it is this which we will be using as the basis of our investigations into moving mesh algorithms. The scheme generates a block diagonal matrix, allowing the system to be solved element-by-element with little data transfer required between neighbouring elements, making it ideal for use on parallel processing computer systems.

2.2 Stationary RKDG Derivation in 1D

We follow the derivation of the RKDG method as given by Cockburn [5] and consider solving the conservation law

$$u_t + f(u)_x = 0 \qquad \text{on } [0,1] \times [0,T] \tag{2.1}$$

$$u(x,0) = u_0(x) \qquad \text{on } [0,1] \tag{2.2}$$

with periodic boundary conditions.

2.2.1 Spatial Discretisation

The spatial domain is partitioned into N cells, denoting the j^{th} cell interval as $I_j = (x_{j-1/2}, x_{j+1/2})$ and the corresponding cell width as $\Delta_j = x_{j+1/2} - x_{j-1/2}$. The cell node x_j is centered within the corresponding interval.

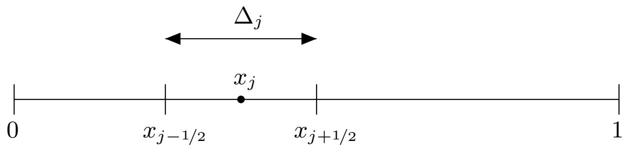


Figure 2.1: Spatial discretisation into N cells.

To solve for the approximate solution u_h, we must first find the weak formation for the problem on each cell by multiplying the equation (2.1) by an arbitrary, smooth function $v(x)$ and integrating over the cell interval:

$$\int_{I_j} \left(\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} \right) v \, \mathrm{d}x = 0.$$

Using integration by parts we obtain

$$\int_{I_j} \frac{\partial u}{\partial t} v \, \mathrm{d}x - \int_{I_j} f \frac{\partial v}{\partial x} \, \mathrm{d}x + f v \bigg|_{x_{j-1/2}}^{x_{j+1/2}} = 0. \tag{2.3}$$

Defining the finite dimensional subspace V_h to be

$$V_h = \left\{ v \in L^1(0,1) : v|_{I_j} \in P^k(I_j) \ , \ j = 1, \ldots, N \right\},$$

where P^k is the set of all polynomials up to degree k, we now replace the smooth $v(x)$ with a test function $v_h \in V_h$ and the exact solution u is replaced by an approximate solution u_h.

The Discontinuous Galerkin method does not require u_h to be continuous across a boundary. Instead, u_h may have two values; u_h^- derived from the cell to the left of the boundary and u_h^+ derived from the cell to the right of the boundary. This discontinuity and duplicity of values means that the analytical flux $f(u_h)$ is not uniquely defined at cell boundaries and so must be replaced by a numerical flux $h(u_h^-, u_h^+)$ which is derived from both values of u_h at that point.

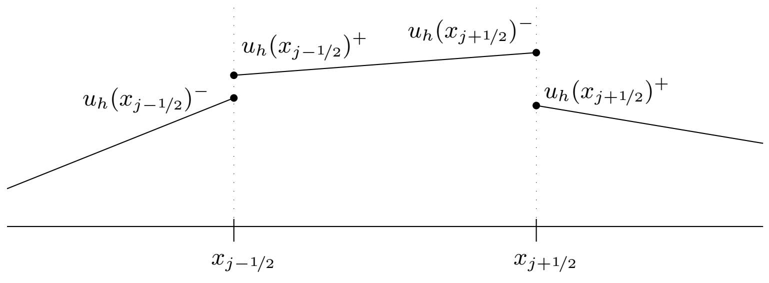


Figure 2.2: Discontinuity in the numerical solution u_h at cell boundaries.

The numerical flux may be calulated in many ways, with the Lax-Friedrichs and Godunov schemes providing typical forumlae. Although the accuracy of the schemes may vary, Cockburn [5] suggest that as the degree of the approximate solution increases, the significance of the choice of numerical flux diminishes, allowing the choice of flux to be based on ease of computation.

For the implementation of the stationary RKDG method, we shall take the local Lax-Friedrichs flux given in [5] by

$$\begin{aligned} h \quad (a,b) &= \frac{1}{2}\left[f(a)+f(b)-c(b-a)\right], \\ c \quad &= \max_{\min(a,b)\le s\le\max(a,b)} |f'(s)|. \end{aligned}$$

For simplicity, we will denote the numerical flux $h(u_h(x_{j+1/2})^-, u_h(x_{j+1/2})^+)$ by $h(x_{j+1/2})$, and similarly for $x_{j-1/2}$.

The weak formulation of the problem may now be written as

$\forall j = 1 \dots N$,

$$\int_{I_j} \frac{\partial u_h}{\partial t} v_h \,\mathrm{d}x - \int_{I_j} f\frac{\partial v_h}{\partial x}\,\mathrm{d}x + v_h(x_{j+1/2})h(x_{j+1/2}) - v_h(x_{j-1/2})h(x_{j-1/2}) = 0$$

$$\int_{I_j} u_h(x,0)v_h \,\mathrm{d}x = \int_{I_j} u_0(x)v_h\,\mathrm{d}x.$$

Introducing Legendre polynomials P_l as basis functions, we can represent the numerical solution u_h as a summation

$$u_h(x,t) = \sum_{l=0}^{k} w_j^l(t)\phi_l(x)$$

where $\phi_l(x) = P_l\left(\frac{2(x-x_j)}{\Delta_j}\right)$ and w_j^l are coefficients to be found.

The orthogonality property of the Legendre polynomials allows the weak form to be rewritten as

$\forall j = 1 \dots N,$

$$\left(\frac{\Delta_j}{2l+1}\right)\frac{\partial w_j^l}{\partial t} - \int_{I_j} f \frac{\partial \phi_l}{\partial x}\,\mathrm{d}x + \left\{h(x_{j+1/2}) - (-1)^l h(x_{j-1/2})\right\} = 0 \tag{2.4}$$

$$\frac{\Delta_j}{2l+1} w_j^l(0) = \int_{I_j} u_0 \phi_l \,\mathrm{d}x. \tag{2.5}$$

If we consider the first Legendre polynomial $P_0 = 1$, in each cell (2.4, 2.5) become

$$\Delta_j \frac{\partial}{\partial t} w_j^0 = -\left\{h(x_{j+1/2}) - h(x_{j-1/2})\right\}$$

$$\Delta_j w_j^0(0) = \int_{I_j} u_0 \,\mathrm{d}x.$$

Similarly, for $P_1 = x$ we obtain

$$\frac{\Delta_j}{3}\frac{\partial}{\partial t} w_j^1 = -\left\{h(x_{j+1/2}) + h(x_{j-1/2})\right\} + \int_{I_j} \frac{2}{\Delta_j} f \,\mathrm{d}x$$

$$\frac{\Delta_j}{3} w_j^1(0) = \int_{I_j} u_0 \left(\frac{2(x-x_j)}{\Delta_j}\right) \mathrm{d}x.$$

Using 2-point Gaussin Quadrature to approximate the integrals, and writing in vector form, the weak form (2.4, 2.5) becomes

$\forall j = 0 \dots N,$

$$\begin{bmatrix} \Delta_j & 0 \\ 0 & \frac{\Delta_j}{3} \end{bmatrix} \frac{\mathrm{d}}{\mathrm{d}t} \begin{bmatrix} w_j^0(t) \\ w_j^1(t) \end{bmatrix} = -\begin{bmatrix} h(x_{j+1/2}) - h(x_{j-1/2}) \\ h(x_{j+1/2}) + h(x_{j-1/2}) \end{bmatrix} + \begin{bmatrix} 0 \\ f(u_h(x_{j+1/2\sqrt{3}})) + f(u_h(x_{j-1/2\sqrt{3}})) \end{bmatrix} \tag{2.6}$$

with

$$\begin{bmatrix} w_j^0(0) \\ w_j^1(0) \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\left\{u_0\left(x_{j+1/2\sqrt{3}}\right) + u_0\left(x_{j-1/2\sqrt{3}}\right)\right\} \\ \frac{\sqrt{3}}{2}\left\{u_0\left(x_{j+1/2\sqrt{3}}\right) - u_0\left(x_{j-1/2\sqrt{3}}\right)\right\} \end{bmatrix}. \tag{2.7}$$

We may solve (2.7) on each cell to obtain the initial coefficients $w(0)$, and hence the initial numerical approximation $u_h(t=0)$.

2.2.2 Time Integration

We may rewrite (2.7, 2.7) as

$$\begin{aligned} \frac{\mathrm{d}u_h}{\mathrm{d}t} &= L_h(u_h) \qquad \text{in } [0,T] \\ u_h(0) &= u_{h0}, \end{aligned}$$

and partition $[0,T]$ into M equal intervals of size Δt.

To step through time and find $u_h(t=T)$, we will use the total variation diminishing (TVD) Runge-Kutta scheme given in [5].

For $m = 0, \dots, M-1$ compute u_h^{m+1} from u_h^m as follows:

- set $u_h^0 = u_h(t=0)$
- For $m = 0, \dots, M-1$, compute u_h^{m+1} from u_h^m as follows:
  - set $u_h^{(0)} = u_h^m$;
  - for $i = 1, \dots, k+1$ compute
  $$u_h^{(i)} = \left\{ \sum_{l=0}^{i-1} \alpha_{il} u_h^{(l)} + \beta_{il} \Delta t L_h(u_h^{(l)}) \right\};$$
  - set $u_h^{m+1} = u_h^{(k+1)}$

where the paramters of α and β may be taken from Table 2.1.

α_{ij}		β_{ij}	
1		1	
$\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{1}{2}$

Table 2.1: Second order TVD RK coefficients(taken from Cockburn [5])

2.3 RKDG Results on a Stationary Mesh

2.3.1 Linear Advection

We consider the RKDG method applied to the simple linear advection problem

$$\begin{aligned} u_t + (3u)_x &= 0 \qquad \text{on } [0,1] \times [0, 0.335] \\ u(x,0) &= 3\sin(2\pi x) + 1 \qquad \text{on } [0,1] \end{aligned}$$

with periodic boundary conditions.

At $t = 0.335$ s, we would expect the intial data to have completed slightly more than a single revolution as the wave speed $= 3$, and as the data should be simply advected, we expect no change in the amplitude of the sine wave. Figure 2.3 shows that the stationary RKDG method has been able to acurately capture the wave speed, advecting the intial data with no significant loss of amplitude.

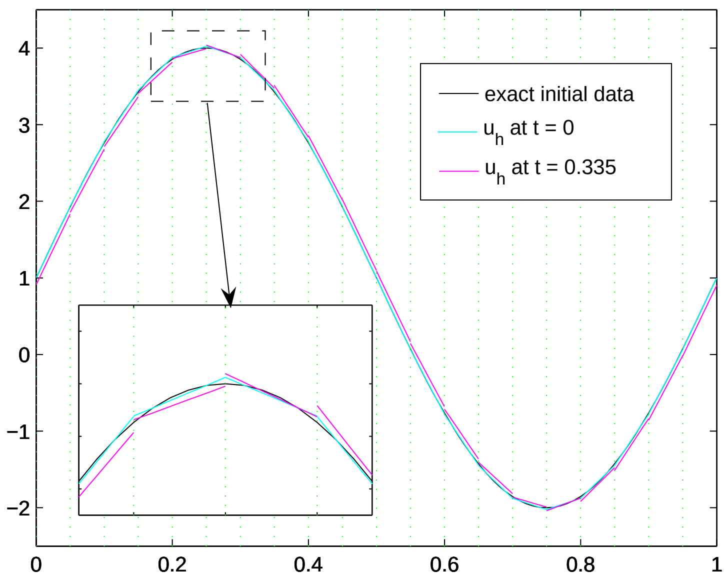


Figure 2.3: The solution of linear advection problem at $t = 0.335$ using RKDG on an equi-distributed stationary mesh with $dx = 0.05$ and $dt = 0.005$.

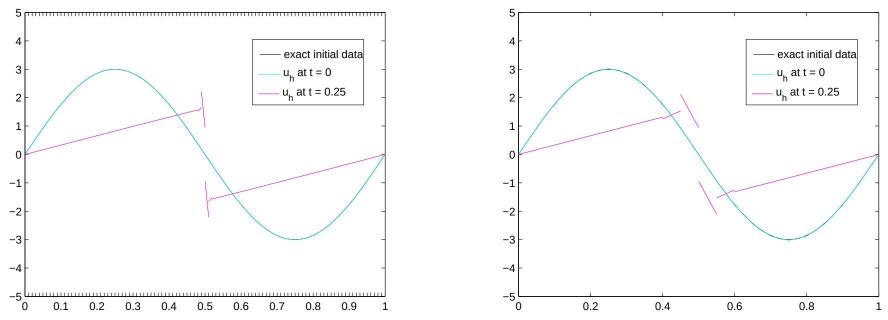
2.3.2 Inviscid Burgers

As a second test case, we consider the stationary RKDG method applied to the nonlinear advection problem

$$\begin{aligned} u_t + (\frac{1}{2}u^2)_x &= 0 \qquad \text{on } [0,1]\times[0,0.25] \\ u(x,0) &= 3\sin(2\pi x) \qquad \text{on } [0,1] \end{aligned}$$

with periodic boundary conditions.

Algebraic solution of this problem shows that a moving shock will form which we wish to capture accuratly. As demontrated in Figure 2.4, when dx is small and the nodes are densly packed, the DG method captures the vertical shock capture well, but the accuracy decreases as cell widths increase.



(a) The solution with $dx = 0.01$.

(b) The solution with $dx = 0.05$

Figure 2.4: The solution of inviscid burgers problem at $t = 0.25$ using RKDG on equi-distributed stationary meshs of different resolutions, taking $dt = 0.0005$ in both cases.

The stationary DG method does not require nodes to be equi-distibuted so it is possible to use small cells near the shock, and larger cells in other regions to reduce the computational cost. However, typically a shock will not be stationary, and so will eventually move out of the region of densly packed nodes into a region covered by larger cells and the accuracy of the shock capture will decrease. This provides the motivation for our investigations into a moving mesh method which would alow the mesh to move with the shock over time.

2.3.3 Stability

For the linear advection problem where $f(u) = cu$, using linear polynomial approximations and the 2nd order RKDG method, Chavent and Cockburn [4] showed the stability condition to be given by

$$c\frac{\Delta t}{\Delta x} \leq \frac{1}{3}. \tag{2.8}$$

The results of numerical investigations for our linear advection test case showed the method was stable for $\frac{\Delta t}{\Delta x} \leq 0.1$, which, as $c = 3$, is consistent with (2.8).

For our nonlinear inviscid burger test case, numerical investigation also showed stable results for $\frac{\Delta t}{\Delta x} \leq 0.1$. For our initial data, $|f'(u)| = |u| \leq 3$ so the general stability condition would be approximately given by

$$|f'(u)|\frac{\Delta t}{\Delta x} \leq \frac{1}{3}.$$

Chapter 3

Moving Mesh Methods

Grid adaptation is becoming increasing popular in numerical modelling as it allows local level mesh refinements to caputre features of interest without excessive increases to the overall computational cost. Adaptation techniques can be split into three main categories known as h-refinement, p-refinement and r-refinement, each of which takes a different approach to improving the accuracy of the numerical solution.

By changing the node-connectivities and introducing new nodes, h-refinement increases the resolution of the grid in a localised area. A common technique is to subdivide a large 'parent' cell into smaller 'child' cells, which may be done isotropically to increase the resolution equally in both the x and y directions, or anistropically to allow a finer resolution in one spatial direction.

Methods falling under the p-refinement classification keep the number of nodes and connectivity unchanged and instead increase the order of the polynomial approximations used in regions of interest. For example, a method may use 4th order polynomials near the feature and only 2nd order polynomials away from the feature.

In application to the stationary DG method, we will be looking at methods falling under the category of r-refinement, which are also commonly known as moving meshes. In these methods, the number of nodes is kept constant but they are redistbuted so that they are clustered around features of interest. There are two main approaches to moving mesh methods; one is based on mappings, the other on velocities.

3.1 Mapping-based moving mesh techniques

Mapping-based moving mesh methods have three main features. Firstly, there is a 1:1 mapping between nodes in the logical or computational domain, which are equally spaced, to the nodes in the physical domain, where they may be clusted in areas of interest. Li and Tang [14] give this mapping as

$$\xi : x \mapsto \xi, \qquad \Omega \mapsto \Omega_c,$$

where Ω denotes the physical domain, Ω_c denotes the logical domain, and where ξ may be found by solving the elliptic system

$$\nabla_x(m\nabla_x\xi) = 0.$$

Here, m is a monitor function, the second key feature of the method, which is used to guide the cell redistiubution. The third feature is interpolation of the numerical solution on the old mesh to obtain values at the nodes in the new mesh.

3.2 Velocity-based moving mesh techniques

Velocity-based methods, also known as Arbitrary Lagrangian-Eulerian (ALE) methods physically relocate the nodes as the numerical solution develops, allowing the mesh to ‘follow’ the feature of interest. In is common to use a cell-based technique and derive the boundary speeds through use of a conservation principle. We will investigate both this and a boundary-based method, which does not assume a conservation principle.

3.2.1 Cell-based techniques

In Chapter 4, we look at some cell-based techniques, where the boundary speeds may be derived directly from the conservation law by imposing a conservation principle on each cell such as

$$\int_{I_j} M dx = \text{ constant in time } ,$$

where M is chosen by the user. For example, Wells et al. [16] set $M = \rho$, the density of the fluid.

3.2.2 Boundary-based techniques

In Chapter 5, we consider a boundary-based approach, which does not rely on the cell-based conservation principle. The boundary speeds may be taken from an external source e.g. velocity of the fluid or the shock speed associated with the discontiunity in u_h at each boundary, and the numerical solution is updated accordingly.

This method, in its basic form, does not provide a monitor funciton to control cell distriubution, so this must be added seperately.

Chapter 4

Cell-based Moving Mesh Methods

We now combine a cell-based moving mesh grid adaptation technique with a DG method similar to the RKDG method seen in Chapter 2.

The conservation law problem

$$u_t + f(u)_x = 0 \qquad \text{on } [0,1] \times [0,T] \tag{4.1}$$

$$u(x,0) = u_0(x) \qquad \text{on } [0,1] \tag{4.2}$$

is now solved with periodic boundary conditions on a moving mesh.

For a cell-based moving mesh method, we make use of a conservation principle on each cell and, following the example of Baines et al. [2], seek to move the cell boundaries such that

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{x_{j-1/2}}^{x_{j+1/2}} vu \, \mathrm{d}x = 0 \tag{4.3}$$

holds for all time.

In the stationary DG method, we derived and solved a weak form of our conservation law problem for u_h. Now, we will instead derive a weak form of the problem in terms of boundary speed $\dot{x}$ which we will solve and then use with the conservation principle (4.3) to determine u_h.

4.1 Method A: a partial-DG technique

We derive a moving mesh method which will make use of the numerical flux calculations at cell boundaries, as seen in the stationary DG method.

4.1.1 The inclusion of boundary speeds

To derive the problem for $\dot{x}$, we firstly follow the stationary DG derivation from Chapter 2, partitioning the spatial domain into N cells and multiplying the problem (4.1) by an arbitrary, smooth function $v(x)$. Integrating over the cell and using integration by parts, we obtain

$$\int_{x_{j-1/2}}^{x_{j+1/2}} \frac{\partial u}{\partial t} v \,\mathrm{d}x = -fv|_{x_{j-1/2}}^{x_{j+1/2}} + \int_{x_{j-1/2}}^{x_{j+1/2}} f \frac{\partial v}{\partial x} \,\mathrm{d}x. \tag{4.4}$$

As the method is now for a moving mesh, we allow the cell boundaries $x_{j-1/2}$ and $x_{j+1/2}$ to vary in time.

Secondly, we use Leibnitz rule

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{x_{j-1/2}}^{x_{j+1/2}} m dx = m\dot{x}|_{x_{j+1/2}} - m\dot{x}|_{x_{j-1/2}} + \int_{x_{j-1/2}}^{x_{j+1/2}} \frac{\partial m}{\partial t} \,\mathrm{d}x$$

to derive an expression for our conservation quantity (4.3).

Taking $m = vu$ where $v(x)$ moves with $\frac{\mathrm{d}x}{\mathrm{d}t}$, we have

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t} \int_{x_{j-1/2}}^{x_{j+1/2}} vu \,\mathrm{d}x &= vu\dot{x}|_{x_{j+1/2}} - vu\dot{x}|_{x_{j-1/2}} + \int_{x_{j-1/2}}^{x_{j+1/2}} \frac{\partial}{\partial t} vu \,\mathrm{d}x \\ &= \int_{x_{j-1/2}}^{x_{j+1/2}} \frac{\partial}{\partial x}(vu\dot{x}) \,\mathrm{d}x + \int_{x_{j-1/2}}^{x_{j+1/2}} \frac{\partial}{\partial t}(vu) \,\mathrm{d}x \\ &= \int_{x_{j-1/2}}^{x_{j+1/2}} \left[v\frac{\partial}{\partial x}(u\dot{x}) + \frac{\partial v}{\partial x} u\dot{x} + v\frac{\partial u}{\partial t} + \frac{\partial v}{\partial t} u \right] \mathrm{d}x \\ &= \int_{x_{j-1/2}}^{x_{j+1/2}} v \left[\frac{\partial}{\partial x}(u\dot{x}) + \frac{\partial u}{\partial t} \right] \mathrm{d}x + \int_{x_{j-1/2}}^{x_{j+1/2}} u \left[\frac{\partial v}{\partial t} + \dot{x}\frac{\partial v}{\partial x} \right] \mathrm{d}x. \end{aligned}$$

Due to the conservation principle (4.3) and the fact that $v(x)$ moves with $\frac{\mathrm{d}x}{\mathrm{d}t}$, this simplifies to give

$$0 = \int_{x_{j-1/2}}^{x_{j+1/2}} v \left[\frac{\partial}{\partial x}(u\dot{x}) + \frac{\partial u}{\partial t} \right] \mathrm{d}x. \tag{4.5}$$

Combining (4.5) with (4.4), we obtain

$$-\int_{x_{j-1/2}}^{x_{j+1/2}} v \frac{\partial}{\partial x}(u\dot{x})\, \mathrm{d}x = -fv|_{x_{j-1/2}}^{x_{j+1/2}} + \int_{x_{j-1/2}}^{x_{j+1/2}} f \frac{\partial v}{\partial x}\, \mathrm{d}x.$$

Our problem now consists of three equations on each cell:

$\forall j = 0 \ldots N$,

$$-\int_{x_{j-1/2}}^{x_{j+1/2}} v \frac{\partial}{\partial x}(u\dot{x})\, \mathrm{d}x = -fv|_{x_{j-1/2}}^{x_{j+1/2}} + \int_{x_{j-1/2}}^{x_{j+1/2}} f \frac{\partial v}{\partial x}\, \mathrm{d}x. \tag{4.6}$$

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{x_{j-1/2}}^{x_{j+1/2}} vu\, \mathrm{d}x = 0 \tag{4.7}$$

$$\int_{x_{j-1/2}}^{x_{j+1/2}} u(x,0)v\, \mathrm{d}x = \int_{x_{j-1/2}}^{x_{j+1/2}} u_0(x)v\, \mathrm{d}x. \tag{4.8}$$

4.1.2 The weak formulation

Defining the finite dimensional subspace V_h to be

$$V_h = \left\{ v \in L^1(0,1) : v|_{I_j} \in P^k(I_j)\ ,\ j = 1, \ldots, N \right\},$$

where P^k is the set of all polynomials up to degree k, we now replace the smooth $v(x)$ with a test function $v_h \in V_h$ and the exact solution u is replaced by a numerical approximation u_h.

The weak formulation of our new problem (4.6, 4.7, 4.8) is then given by

$\forall j = 1 \ldots N$,

$$-\int_{x_{j-1/2}}^{x_{j+1/2}} v_h \frac{\partial}{\partial x}(u_h\dot{x})\, \mathrm{d}x = -fv_h|_{x_{j-1/2}}^{x_{j+1/2}} + \int_{x_{j-1/2}}^{x_{j+1/2}} f \frac{\partial v_h}{\partial x}\, \mathrm{d}x, \tag{4.9}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_{j-1/2}}^{x_{j+1/2}} v_h u_h \,\mathrm{d}x = 0 \tag{4.10}$$

$$\int_{x_{j-1/2}}^{x_{j+1/2}} u_h(x,0) v_h \,\mathrm{d}x = \int_{x_{j-1/2}}^{x_{j+1/2}} u_0(x) v_h \,\mathrm{d}x. \tag{4.11}$$

The analytic flux f is not defined at cell boundaries due to the discontinuity in u_h, so we introduce a numerical flux scheme $h(x) = h(u_h(x)^-, u_h(x)^+) \approx f(u_h(x))$. As for the stationary DG method, we choose to use the local Lax Friedrichs formula given in [5] as

$$\begin{aligned} h \quad (a,b) &= \frac{1}{2}\left[f(a) + f(b) - c(b-a)\right], \\ c \quad &= \max_{\min(a,b)\le s\le \max(a,b)} |f'(s)|. \end{aligned}$$

We take the test functions $v_h(x)$ to be Legendre polynomials and express the numerical solution u_h as a sum of Legendre polynomial basis functions:

$$u_h(x,t) = \sum_{l=0}^{k} w_j^l(t)\phi_l(x)$$

where $\phi_l(x) = P_l\left(\frac{2(x-x_j)}{\Delta_j}\right)$ and w_j^l are coefficients to be found.

The right-hand side of (4.9) can be taken directly from the standard DG derivation (2.7) to be approximated by

$$\begin{aligned} -f v_h|_{x_{j-1/2}}^{x_{j+1/2}} + \int_{x_{j-1/2}}^{x_{j+1/2}} f\frac{\partial v_h}{\partial x}\,\mathrm{d}x &= \\ -\begin{bmatrix} h(x_{j+1/2}) - h(x_{j-1/2}) \\ h(x_{j+1/2}) + h(x_{j-1/2}) \end{bmatrix} &+ \begin{bmatrix} 0 \\ f(u_h(x_{j+1/2\sqrt{3}})) + f(u_h(x_{j-1/2\sqrt{3}})) \end{bmatrix}. \end{aligned}$$

To derive formulae for the left-hand side of (4.9), we use integration by parts to obtain

$$-\int_{x_{j-1/2}}^{x_{j+1/2}} v_h \frac{\partial}{\partial x}(u_h\dot{x})\,\mathrm{d}x = -v_h u_h \dot{x}|_{x_{j-1/2}}^{x_{j+1/2}} + \int_{x_{j-1/2}}^{x_{j+1/2}} \frac{\partial v_h}{\partial x} u_h \dot{x}\,\mathrm{d}x.$$

For the simple case when $v_h = 1$, the first Legendre polynomial, this becomes

$$-\int_{x_{j-1/2}}^{x_{j+1/2}} v_h \frac{\partial}{\partial x}(u_h \dot{x}) \, \mathrm{d}x = -\left[u_h(x_{j+1/2})\dot{x}(x_{j+1/2}) - u_h(x_{j-1/2})\dot{x}(x_{j-1/2})\right].$$

For $v_h = \frac{2(x-x_j)}{\Delta_j}$, the second Legendre polynomial, we use 2-point Gaussian quadrature to evaluate the integral term and get

$$\begin{aligned}-\int_{x_{j-1/2}}^{x_{j+1/2}} v_h \frac{\partial}{\partial x}(u_h \dot{x}) \, \mathrm{d}x = &-\left[u_h(x_{j+1/2})\dot{x}(x_{j+1/2}) + u_h(x_{j-1/2})\dot{x}(x_{j-1/2})\right] \\ &+ \left[u_h(x_{j+1/2\sqrt{3}})\dot{x}(x_{j+1/2\sqrt{3}}) + u_h(x_{j-1/2\sqrt{3}})\dot{x}(x_{j-1/2\sqrt{3}})\right].\end{aligned}$$

By assuming $\dot{x}$ to be linear over each cell, we may obtain values for $\dot{x}$ at $x_{j-1/2\sqrt{3}}$ and $x_{j+1/2\sqrt{3}}$ using linear interpolation between the values at cell boundaries.

Similarly we apply the properties of Legendre polynomials to (4.10, 4.11), the weak forms of the conservation principle and the initial conditions respectively.

The weak formulation of problem (4.9, 4.10, 4.11) is then given in matrix form as

$\forall j = 1 \ldots N$,

$$\begin{bmatrix} u_h(x_{j-1/2}) & -u_h(x_{j+1/2}) \\ \left\{\begin{matrix} -u_h(x_{j-1/2})+ \\ (1+\frac{1}{2\sqrt{3}}-\frac{1}{2})u_h(x_{j-1/2\sqrt{3}}) \\ +(\frac{1}{2}-\frac{1}{2\sqrt{3}})u_(x_{j+1/2\sqrt{3}}) \end{matrix}\right\} & \left\{\begin{matrix} -u_(x_{j-1/2})+ \\ (\frac{1}{2}-\frac{1}{2\sqrt{3}})u_(x_{j-1/2\sqrt{3}}) \\ +(1+\frac{1}{2\sqrt{3}}-\frac{1}{2})u_(x_{j+1/2\sqrt{3}}) \end{matrix}\right\} \end{bmatrix} \begin{bmatrix} \dot{x}(x_{j-1/2}) \\ \dot{x}(x_{j+1/2}) \end{bmatrix}$$

$$= -\begin{bmatrix} h(x_{j+1/2}) - h(x_{j-1/2}) \\ h(x_{j+1/2}) + h(x_{j-1/2}) \end{bmatrix} + \begin{bmatrix} 0 \\ f(u_h(x_{j+1/2\sqrt{3}})) + f(u_h(x_{j-1/2\sqrt{3}})) \end{bmatrix} \tag{4.12}$$

$$\begin{bmatrix} \Delta_j(t) & 0 \\ 0 & \frac{1}{3}\Delta_j(t) \end{bmatrix} \begin{bmatrix} w_j^0(t) \\ w_j^1(t) \end{bmatrix} = \begin{bmatrix} \Delta_j(t=0)w_j^0(t=0) \\ \frac{1}{3}\Delta_j(t=0)w_j^l(t=0) \end{bmatrix} \tag{4.13}$$

$$\begin{bmatrix} w_j^0(0) \\ w_j^1(0) \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\left\{u_0\left(x_{j+1/2\sqrt{3}}\right) + u_0\left(x_{j-1/2\sqrt{3}}\right)\right\} \\ \frac{\sqrt{3}}{2}\left\{u_0\left(x_{j+1/2\sqrt{3}}\right) - u_0\left(x_{j-1/2\sqrt{3}}\right)\right\} \end{bmatrix}. \tag{4.14}$$

The weak formulation provides cell-by-cell matrix systems for determining the boundary speeds $\dot{x}$, which we may then use to find the updated numerical approximation u_h. We consider two ways of solving for the boundary speeds:

The first approach solves each cell-by cell matrix system seperately to find the boundary speeds for each cell. We then average the two values obtained for each cell boundary to obtain a single speed for that boundary.

The second approach is to combine the information for all cells to obtain a global matrix system, allowing all cell-by-cell matrix systems to be solved simultaneously to find the boundary speeds.

4.1.3 Time integration

We partition $[0, T]$ into M intervals of size Δt and use an Euler timestepping algorithm to step through time and find $u_h(t = T)$:

- Solve (4.14) to obtain $w_j^0(0)$ and $w_j^1(0)$ and hence find $u_h(t = 0)$;
- For $m = 0, \ldots, M - 1$,
  - Solve (4.12), either as a global matrix system over all cells, or on each cell individually followed by averaging, and obtain $\dot{x}$ for each boundary;
  - Compute the new position x of each boundary using

    $$x(t = m + 1) = x(t = m) + \Delta t \times \dot{x}$$

  - Find the new interval sizes and node positions based on the new boundary positions;

- Solve (4.13) to obtain $w_j^0(m+1)$ and $w_j^1(m+1)$ and hence find $u_h(t = m+1)$;

4.1.4 Results

If we choose to use only constant basis vectors for our numerical approximation u_h, each cell matrix system reduces to a single equation giving insufficient information to solve on each cell individually. The global matrix formed by combining each of the cell matrices has the form

$$\begin{bmatrix} u_1 & -u_1 & & & & \\ & u_2 & -u_2 & & & \\ & & \ddots & \ddots & & \\ & & & u_{N-1} & -u_{N-1} \\ -u_N & & & & u_N \end{bmatrix}$$

which is singular and cannot be inverted to solve the global matrix system. We therefore conclude that this moving mesh method is not suitable for the constant basis vector case.

Using linear basis vectors, it is still possible for the numerical approximation u_h to be constant or nearly constant on a cell. If this occurs, we are unable to solve the cell matrix system for that cell but in general, the global matrix is not singular, so we may use this approach.

We use the the global matrix approach to solve the linear advection test problem

$$\begin{aligned} u_t + (3u)_x &= 0 \qquad \text{on } [0,1] \times [0,T] \\ u(x,0) &= 3\sin(2\pi x) \qquad \text{on } [0,1] \end{aligned}$$

with periodic boundary conditions.

For this linear advection problem, we would expect the initial data to be moved, unchanged, with a wavespeed of 3. We can see from Figure 4.1, that whilst a solution to $u_h(t)$ may be obtained using the global matrix approach, it provides a poor representation of the exact solution after the first time step and quickly breaking down completly.

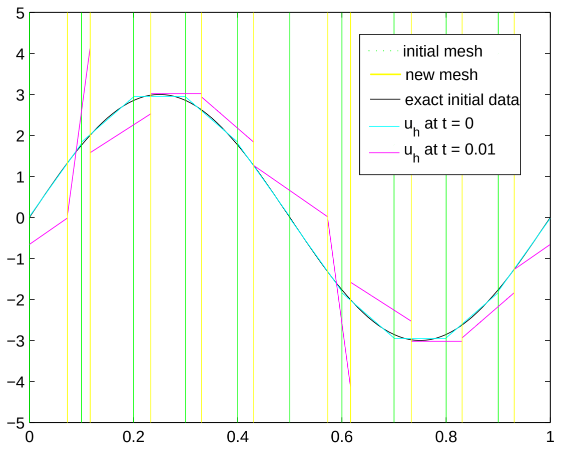


Figure 4.1: The solution for the linear advection problem $f(u) = 3u$, $u(x,0) = 3\sin(2\pi x)$ using the partial-DG method with global matrix approach at $t = 0.01$ taking $\Delta x = 0.1$ and $\Delta t = 0.01$.

On closer examination of the calculated boundary speeds, it is clear that the boundary speeds significantly different to the majority occur where $u_h^+ = 0$ or $u_h^- = 0$.

We therefore consider the linear advection test case with new initial data $u(x,0) = 3\sin(2\pi x) + 5$, so that such values may be avoided. Figure 4.2 shows that for this new problem, the initial data is advected at approximately the right wave speed for early timesteps, but some growth can be seen and the accuracy of the results decreases over time. After less than the time required for a single revolution, the solution breaks down.

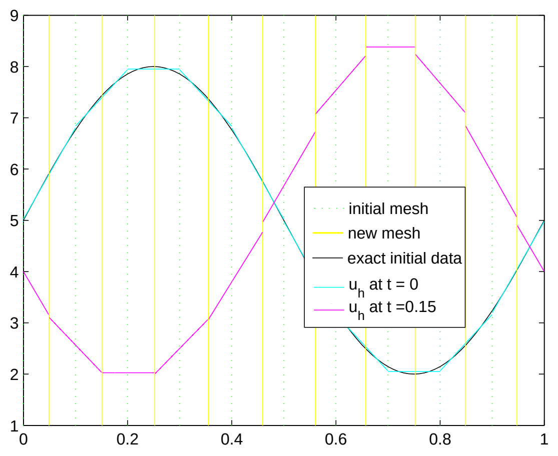


Figure 4.2: The solution for the linear advection problem $f(u) = 3u$, $u(x, 0) = 3\sin(2\pi x)+5$ using the partial-DG method with global matrix approach at $t = 0.15$ taking $\Delta x = 0.1$ and $\Delta t = 0.01$.

4.2 Method B: a non-DG technique

As an alternative to Method A, we derive another moving mesh algorithm that is able to find boundary speeds in regions with constant numerical approximations. As this method does not make use of a numerical flux at the cell boundaries it is seen to be missing a major component of the DG structure and so is refered to as a non-DG method.

4.2.1 The weak formulation

As seen in Method A, we may apply Leibniz rule to the conservation principle (4.3), and taking $v(x)$ to move with $\frac{\mathrm{d}x}{\mathrm{d}t}$, we may obtain (4.5):

$$0 = \int_{x_{j-1/2}}^{x_{j+1/2}} v \left[\frac{\partial}{\partial x}(u\dot{x}) + \frac{\partial u}{\partial t} \right] \mathrm{d}x.$$

We may define the finite dimensional subspace V_h to be

$$V_h = \left\{ v \in L^1(0,1) : v|_{I_j} \in P^k(I_j)\ ,\ j = 1, \dots, N \right\},$$

where P^k is the set of all polynomials up to degree k, and we now replace the smooth $v(x)$ with a test function $v_h \in V_h$ and the exact solution u by a numerical approximation u_h.

From the original conservation law (4.1), we may replace $\frac{\partial u_h}{\partial t}$ and obtain

$$0 = \int_{x_{j-1/2}}^{x_{j+1/2}} v_h \left[\frac{\partial}{\partial x}(u_h \dot{x}) - \frac{\partial f(u_h)}{\partial x} \right] \, \mathrm{d}x.$$

Alternatively, this may be written as

$$0 = \int_{x_{j-1/2}}^{x_{j+1/2}} v_h \left[(u_h \dot{x}) - f(u_h) \right]_x \, \mathrm{d}x. \tag{4.15}$$

One possible solution for equation (4.15) is to take

$$\dot{x} = \frac{f(u_h)}{u_h}.$$

We may use this formula twice at each boundary, once using the values from the cell to the left, and once using the values from the cell to the right. Then, in a similar manner as the cell-by-cell approach in Method A, we average to obtain a single speed for that boundary.

For the occasions when $u_h \approx 0$, we use L'Hopital's rule and take the boundary speed to be

$$\dot{x} = \lim_{u_h \to 0} \frac{f(u_h)}{u_h} = \frac{f'(u_h)}{u_h'} = f'(u_h)$$

The full weak formulation of the problem, including initial conditions is given by

$\forall j = 1 \dots N,$

$$\begin{cases} \dot{x}(x_{j+1/2}) = \frac{f(u_h(x_{j+1/2}))}{u_h(x_{j+1/2})} \\ \dot{x}(x_{j-1/2}) = \frac{f(u_h(x_{j-1/2}))}{u_h(x_{j-1/2})} \end{cases} \tag{4.16}$$

$$\begin{bmatrix} \Delta_j(t) & 0 \\ 0 & \frac{1}{3}\Delta_j(t) \end{bmatrix} \begin{bmatrix} w_j^0(t) \\ w_j^1(t) \end{bmatrix} = \begin{bmatrix} \Delta_j(t=0)w_j^0(t=0) \\ \frac{1}{3}\Delta_j(t=0)w_j^l(t=0) \end{bmatrix} \tag{4.17}$$

$$\begin{bmatrix} w_j^0(0) \\ w_j^1(0) \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\left\{u_0\left(x_{j+1/2\sqrt{3}}\right) + u_0\left(x_{j-1/2\sqrt{3}}\right)\right\} \\ \frac{\sqrt{3}}{2}\left\{u_0\left(x_{j+1/2\sqrt{3}}\right) - u_0\left(x_{j-1/2\sqrt{3}}\right)\right\} \end{bmatrix}. \tag{4.18}$$

4.2.2 Time integration

The time integration is essentially unchanged from Method A, requiring only a different method for calculating the boundary speeds $\dot{x}$. The method is included here for completeness.

We partition $[0, T]$ into M intervals of size Δt and use an Euler timestepping algorithm to step through time and find $u_h(t = T)$:

- Solve (4.18) to obtain $w_j^0(0)$ and $w_j^1(0)$ and hence find $u_h(t = 0)$;
- For $m = 0, \ldots, M - 1$,
  - Compute (4.16) for each cell and averege to obtian the boundary speeds;
  - Compute the new position x of each boundary using

    $$x(t = m + 1) = x(t = m) + \Delta t \times \dot{x};$$

  - Find the new interval sizes and node positions based on the new boundary positions;
  - Solve (4.17) to obtain $w_j^0(m + 1)$ and $w_j^1(m + 1)$ and hence find $u_h(t = m + 1)$;

4.2.3 Results

The non-DG method is first applied to a linear advection test problem where we seek to solve

$$u_t + (3u)_x = 0 \qquad \text{on } [0,1] \times [0,T] \tag{4.19}$$

$$u(x,0) = 3\sin(2\pi x) \qquad \text{on } [0,1] \tag{4.20}$$

with periodic boundary conditions.

The results obtained, as shown in Figure 4.3, are very good with the initial data being advected at approximately the right wave speed and with no significant growth or decay.

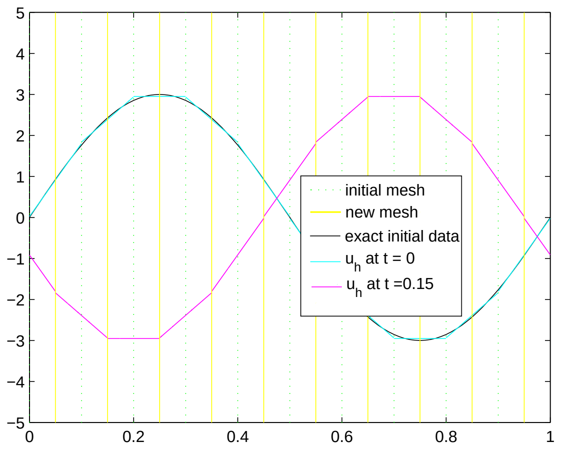


Figure 4.3: The solution for the linear advection problem $f(u) = 3u$, $u(x,0) = 3\sin(2\pi x)$ using the non-DG method at $t = 0.15$ taking $\Delta x = 0.1$ and $\Delta t = 0.01$.

We therefore consider this method applied to a nonlinear test case and solve

$$u_t + (\frac{1}{2}u^2)_x = 0 \qquad \text{on } [0,1] \times [0,T]$$

$$u(x,0) = 3\sin(2\pi x) \qquad \text{on } [0,1]$$

with periodic boundary conditions.

For the nonlinear problem, the non-DG method initially copes well as seen in Figure 4.4, but as the shock begins to form, boundaries begin overtaking and the solution breaks down. The problem of cell distibution may not be simple to solve here as any direct modification to the boundary speeds to restrict cell widths, would enforce a restriction on the numerical solution by equation (4.17).

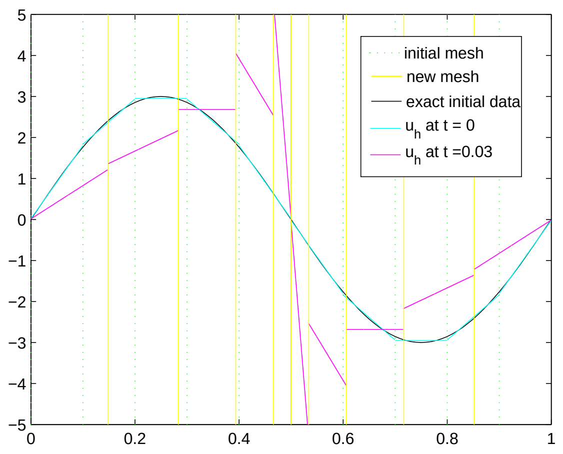


Figure 4.4: The solution for the nonlinear problem $f(u) = \frac{1}{2}u^2$, $u(x,0) = 3\sin(2\pi x)$ using the non-DG method at $t = 0.03$ taking $\Delta x = 0.1$ and $\Delta t = 0.01$.

4.3 Combined Methods

We have seen that Method A cannot be used cell-by-cell if any numerical approximation is constant or nearly constant on a cell. For these cells, we now consider obtaining the boundary speeds through alternative methods including using the non-DG method B and interpolation between known boundary velocities.

When applied to the linear advection test case (4.19, 4.20), we see in Figures 4.5 and 4.6 that neither method generates an accurate representation of the exact solution, with the values being significantly higher or lower depending on the cell widths.

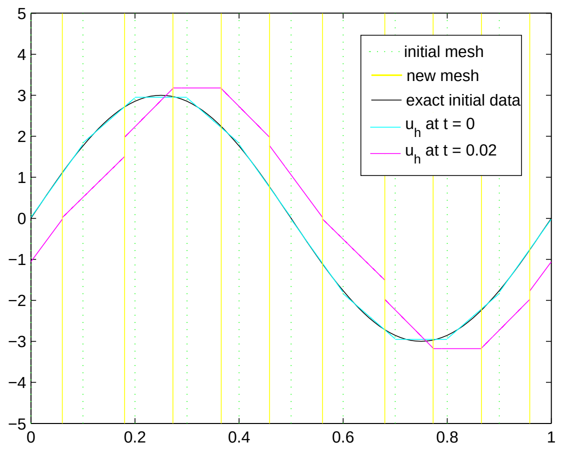


Figure 4.5: The solution for the linear advection problem $f(u) = 3u$, $u(x, 0) = 3\sin(2\pi x)$ using the partial-DG method with boundary speed interpolations, at $t = 0.02$ taking $\Delta x = 0.1$ and $\Delta t = 0.01$.

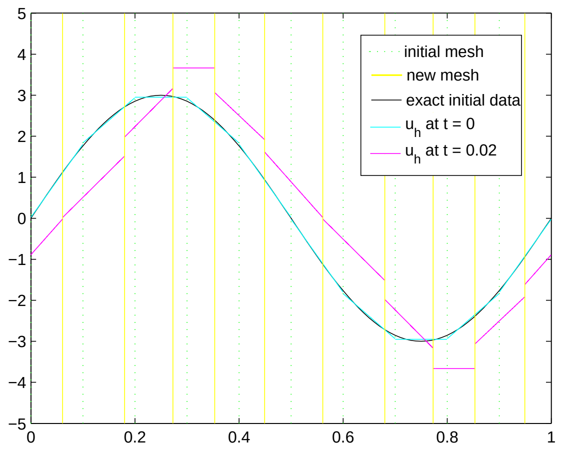


Figure 4.6: The solution for the linear advection problem $f(u) = 3u$, $u(x, 0) = 3\sin(2\pi x)$ using the partial-DG method with additional speeds using the non-DG method, at $t = 0.02$ taking $\Delta x = 0.1$ and $\Delta t = 0.01$.

4.4 Conclusions

We have seen from the results, that Method A used cell-by-cell, and it's modifications, have all failed to model linear advection to a suitable level of accuracy. This may be due to an inconsitency in the use of the conservation principle.

The conservation principle is used to obtain boundary speed values on each cell, which we then proceed to average so that we have only a single boundary speed for each boundary. In calculating the new coefficients w_j^0 and w_j^1, we use the new boundary positions, as found using the averaged boundary speeds, but then apply the conservation principle which would only hold on each cell if the exact, rather than the averaged, boundary speeds had been used.

The global matrix approach for Method A, solves for the boundary speeds of all cells simultaneously, so there is no averaging to introduce the inconsistency seen in the cell-by-cell approach. However, the boundary speeds obtained for linear advection are still not uniform across all cells, as would be required for smooth advection of the initial data by this method. This may be due to the inclusion of the numerical flux in boundary speed calculations.

Method B, the non-DG method, works well for linear advection, but experiences cell distribution issues for the non-linear motion. In desiging a method to control cell distribution, it is the conservation principle appied through equation (4.17) that creates difficulties by directly linking boundary speeds, which may need to be modified, with the numerical solution.

The difficulties experienced with these cell-based methods which use the conservation principle (4.3) provide the motivation to use a different approach to the moving mesh method. We therefore consider a boundary-based technique which does not rely on the conservation principle (4.3).

Chapter 5

A Boundary-based Moving Mesh Method

In the previous moving mesh methods investigated in Chapter 4, the boundary speeds have been derived based on the conservation principle (4.3) and this has directly linked cell width to the value of the numerical solution on that cell by (4.13). If we wish to overwrite boundary speeds with alternative values e.g. to prevent boundary overtaking, we must therefore derive a new moving mesh algorithm which does not depend on the conservation principle (4.3).

5.1 Derivation of a full-DG Method

The conservation law problem

$$\begin{aligned} u_t + f(u)_x &= 0 \qquad \text{on } [0,1]\times[0,T] \\ u(x,0) &= u_0(x) \qquad \text{on } [0,1] \end{aligned}$$

is again solved with periodic boundary conditions on a moving mesh.

5.1.1 The inclusion of boundary speeds

To include boundary speeds $\dot{x}$ we use Leibniz rule

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_{j-1/2}}^{x_{j+1/2}} m dx = m\dot{x}|_{x_{j+1/2}} - m\dot{x}|_{x_{j-1/2}} + \int_{x_{j-1/2}}^{x_{j+1/2}} \frac{\partial m}{\partial t}\,\mathrm{d}x$$

to expand

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_{j-1/2}}^{x_{j+1/2}} vu\,\mathrm{d}x$$

which is no longer assumed to be zero for all time.

Taking $m = vu$ where $v(x)$ moves with $\frac{\mathrm{d}x}{\mathrm{d}t}$, we have

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_{j-1/2}}^{x_{j+1/2}} vu\,\mathrm{d}x &= vu\dot{x}|_{x_{j+1/2}} - vu\dot{x}|_{x_{j-1/2}} + \int_{x_{j-1/2}}^{x_{j+1/2}} \frac{\partial}{\partial t} vu\,\mathrm{d}x \\
&= \int_{x_{j-1/2}}^{x_{j+1/2}} \frac{\partial}{\partial x}(vu\dot{x})\,\mathrm{d}x + \int_{x_{j-1/2}}^{x_{j+1/2}} \frac{\partial}{\partial t}(vu)\,\mathrm{d}x \\
&= \int_{x_{j-1/2}}^{x_{j+1/2}} \left[v\frac{\partial}{\partial x}(u\dot{x}) + \frac{\partial v}{\partial x}u\dot{x} + v\frac{\partial u}{\partial t} + \frac{\partial v}{\partial t}u \right] \mathrm{d}x \\
&= \int_{x_{j-1/2}}^{x_{j+1/2}} v\left[\frac{\partial}{\partial x}(u\dot{x}) + \frac{\partial u}{\partial t} \right] \mathrm{d}x + \int_{x_{j-1/2}}^{x_{j+1/2}} u\left[\frac{\partial v}{\partial t} + \dot{x}\frac{\partial v}{\partial x} \right] \mathrm{d}x.
\end{aligned}$$

As $v(x)$ moves with $\frac{\mathrm{d}x}{\mathrm{d}t}$, the last integral term is zero and substituting in for $\frac{\partial u}{\partial t}$ from our orginal conservation law, we obtain

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_{j-1/2}}^{x_{j+1/2}} vu\,\mathrm{d}x = \int_{x_{j-1/2}}^{x_{j+1/2}} v\left(\dot{x}u - f\right)_x\,\mathrm{d}x.$$

It may be possible to solve this equation by using quadrature to evaluate the integral on the right-hand side directly. However, we choose to follow the ideas of the stationary RKDG derivation and use integration by parts to obtain

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_{j-1/2}}^{x_{j+1/2}} vu\,\mathrm{d}x = -\, v\left(f - \dot{x}u\right)|_{x_{j-1/2}}^{x_{j+1/2}} + \int_{x_{j-1/2}}^{x_{j+1/2}} \left(f - \dot{x}u\right)\frac{\partial v}{\partial x}\,\mathrm{d}x. \qquad (5.1)$$

We now have a problem for u which includes $\dot{x}$ as required.

5.1.2 The weak formulation

Defining the finite dimensional subspace V_h to be

$$V_h = \left\{ v \in L^1(0,1) : v|_{I_j} \in P^k(I_j) \ , \ j = 1, \ldots, N \right\},$$

where P^k is the set of all polynomials up to degree k, we now replace the smooth $v(x)$ with a test function $v_h \in V_h$ and the exact solution u is replaced by u_h.

The full problem, including initial conditions then becomes

$\forall j = 1, \ldots, N$

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{x_{j-1/2}}^{x_{j+1/2}} v_h u_h \, \mathrm{d}x = - \, v_h F|_{x_{j-1/2}}^{x_{j+1/2}} + \int_{x_{j-1/2}}^{x_{j+1/2}} F \frac{\partial v_h}{\partial x} \, \mathrm{d}x \tag{5.2}$$

$$\int_{x_{j-1/2}}^{x_{j+1/2}} u_h(x,0) v_h \, \mathrm{d}x = \int_{x_{j-1/2}}^{x_{j+1/2}} u_0(x) v_h \, \mathrm{d}x. \tag{5.3}$$

where $F = f - \dot{x}u$.

As u_h is discontinuous at cell boundaries, $F(u_h)$ is undefined at these points. We introduce a numerical flux h such that $h(x) = h(u_h(x)^+, u_h(x)^-) \approx F(u_h(x))$, where $u_h(x)^+$ and $u_h(x)^-$ denoute the values of $u_h(x)$ from above and below respectively.

The numerical flux may be calculated using the local Lax Friedrichs formula

$$\begin{aligned} h \quad (a,b) &= \frac{1}{2} \left[F(a) + F(b) - c(b-a) \right], \\ c \quad &= \max_{\min(a,b) \le s \le \max(a,b)} |F'(s)| \end{aligned}$$

where $F' = f' - \dot{x}$

We take the test functions $v_h(x)$ to be Legendre polynomials and express the numerical solution u_h as a sum of Legendre polynomial basis functions:

$$u_h(x,t) = \sum_{l=0}^{k} w_j^l(t) \phi_l(x)$$

where $\phi_l(x) = P_l\left(\frac{2(x-x_j)}{\Delta_j}\right)$ and w_j^l are coefficients to be found.

Through the orthgonality properties of the Legendre polynomials we are able to express our problem (5.2, 5.3) as a matrix system

$$\forall j = 1, \dots, N$$

$$\begin{bmatrix} 1 & 0 \\ 0 & 1/3 \end{bmatrix} \begin{bmatrix} \dot{\Delta}_j w_j^0 + \Delta_j \dot{w}_j^0 \\ \dot{\Delta}_j w_j^1 + \Delta_j \dot{w}_j^1 \end{bmatrix} = - \begin{bmatrix} h(x_{j+1/2}) - h(x_{j-1/2}) \\ h(x_{j+1/2}) + h(x_{j-1/2}) \end{bmatrix} + \begin{bmatrix} 0 \\ F(x_{j+1/2\sqrt{3}}) + F(x_{j-1/2\sqrt{3}}) \end{bmatrix} \tag{5.4}$$

$$\begin{bmatrix} w_j^0(0) \\ w_j^1(0) \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\left\{u_0\left(x_{j+1/2\sqrt{3}}\right) + u_0\left(x_{j-1/2\sqrt{3}}\right)\right\} \\ \frac{\sqrt{3}}{2}\left\{u_0\left(x_{j+1/2\sqrt{3}}\right) - u_0\left(x_{j-1/2\sqrt{3}}\right)\right\} \end{bmatrix} \tag{5.5}$$

where $\dot{x}(x_{j-1/2\sqrt{3}})$ and $\dot{x}(x_{j+1/2\sqrt{3}})$ are found assuming $\dot{x}$ is linear on each cell.

5.1.3 Time integration

We partition the time interval $[0, T]$ into M intervals of size Δt.

To step through time and find $u_h(t = T)$, we will use an Euler timestepping method:

- Solve (5.5) to obtain $w_j^0(0)$ and $w_j^1(0)$ and hence find $u_h(t = 0)$;
- For $m = 0, \dots, M - 1$,
  - Obtain the boundary speeds (see Section 5.2)
  - Solve (5.4) to obtain $\dot{w}_j^0(t = m)$ and $\dot{w}_j^1(t = m)$;
  - Compute
    $$\begin{bmatrix} w_j^0(t = m+1) \\ w_j^1(t = m+1) \end{bmatrix} = \begin{bmatrix} w_j^0(t = m) \\ w_j^1(t = m) \end{bmatrix} + \Delta t \begin{bmatrix} \dot{w}_j^0(t = m) \\ \dot{w}_j^1(t = m) \end{bmatrix}$$
  - and hence find $u_h(t = m + 1)$

5.2 Boundary Speed Selection

Unlike previous moving mesh algorithms, this method does not generate the boundary speeds. Instead they may be input from an external source at each timestep, and then the changes to the numerical solution u_h are found accordingly.

In particular, we note that if we take $\dot{x} = 0$ for all boundaries and for all time, the method reverts back to the stationary DG method with Euler timestepping and should yield similar results to the RKDG method from Chapter 2, allowing for the difference in accuracy and stability between the Euler and Runge-Kutta time-stepping algorithms.

5.2.1 Selecting non-zero boundary speeds

At each boundary, the numerical solution u_h is discontinous and a jump in the solution occurs (see Figure 2.2). A natural choice for the boundary speed would be the notional shock speed associated with this discontunity in u_h. In the case when the jump in u_h is negligible, we can instead use the overall wave speed.

We therefore select the boundary speeds to be

$$\dot{x} = \begin{cases} f'(u_h^+) & \text{if } u_h^+ - u_h^- \approx 0 \\ \frac{[f(u_h)]}{[u_h]} & \text{otherwise} \end{cases} \tag{5.6}$$

where $[u_h]$ and $[f(u_h)]$ denote the jumps in u_h and $f(u_h)$ respectively.

5.2.2 Controlling cell distribution

Over time, the choice of boundary speeds may result in boundaries overtaking one another or cell widths becoming negligbly small. We need to overcome these issues.

To avoid cells of negligble width, it would may seem natural to remove a boundary and merge the small cell with a larger cell as shown in Figure 5.1. However, it is computationally difficult to work with a variable number of nodes and over time, as the number of nodes falls, the accuracy of the model would decrease. We will therefore fix the number of nodes and consider another approach.

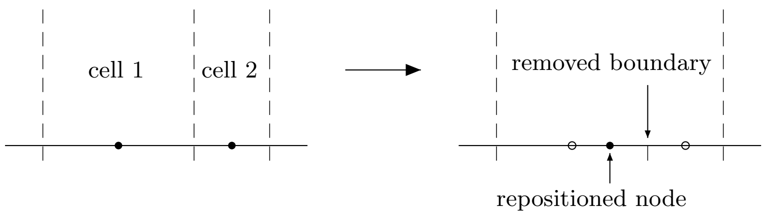


Figure 5.1: The removal of small cells through merging with larger cells.

We could use variable Δt, chosing the timestep so that no cell width falls below the minimum for given boundary speeds. The minimum allowable timestep on each cell can be found using

$$\Delta t_j = \frac{\Delta_{j\,\mathrm{min}} - \Delta_j}{\dot{x}(x_{j+1/2}) - \dot{x}(x_{j-1/2})} \tag{5.7}$$

where $\Delta_{j\,\mathrm{min}}$ is the minimum allowed cell width, and then we can take $\Delta t = \min\{\Delta t_j\}$ for the next timestep. However, in practise, this usually gives $\Delta t \to 0$ meaning that the model will not progress beyond a certain time.

As a third option, we consider changing the boundary speeds such that the timestep Δt and the cell widths Δ_j stay within the stability criteria but above given minimum values.

Firstly, we define our maximum allowed timestep, Δt_{max} based on stability restrictions, and the minimum allowed timestep Δt_{min} and minimum allowed cell width $\Delta_{j\,\mathrm{min}}$ based on user preferences. We then use (5.7) to find the suggested value of Δt for the given boundary speeds.

Provided that $\Delta t > \Delta t_{\mathrm{min}}$, boundaries will not overtake or get too close, so we may continue with the time stepping algorithm using $\min\{\Delta t, \Delta t_{\mathrm{max}}\}$ to ensure stability. However, if $\Delta t < \Delta t_{\min}$, we use $\Delta t_{\min}$ and must amend the boundary speeds.

Ideally, we only wish to amend the boundary speeds in problematic regions where the cell widths become too small otherwise. One way is to identify the effected cells and replace the speeds at the associated boundaries with an average speed as shown in Figure 5.2. However, this may then effect neighbouring cells, so the checks must be repeated until no problem cells are present.

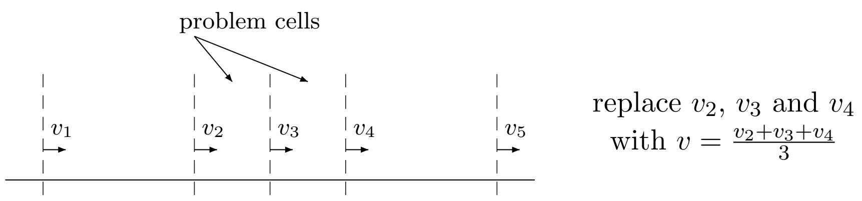

problem cells
v_1
v_2
v_3
v_4
v_5
replace v_2, v_3 and v_4 with $v = \frac{v_2+v_3+v_4}{3}$

Figure 5.2: Averaging boundary speeds for cells below the minimum width.

Another approach would be to revert to a constant speed across all boundaries when problem cells begin to develop. As problems usually occur near the shock formation, and we wish for the densly packed region of nodes to remain aligned with this feature, the shock speed is the ideal choice of speed for the boundaries. This is closely approximated by the boundary speed already calculated for the boundary most closely alligned with the shock, so we identify this boundary and set all boundaries to move at that speed.

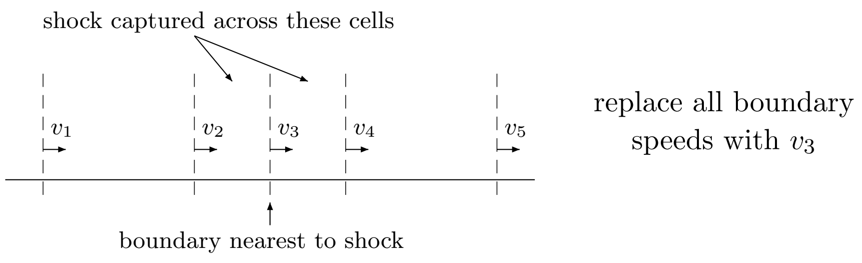

shock captured across these cells
v_1
v_2
v_3
v_4
v_5
replace all boundary speeds with v_3
boundary nearest to shock

Figure 5.3: Fixed boundary speeds taken as the approximate shock speed.

Chapter 6

Numerical Results for the full-DG Method

We now apply the full-DG method from Chapter 5 to solve the conservation law

$$u_t + (f(u))_x = 0 \qquad \text{on } [0,1] \times [0,T] \tag{6.1}$$

$$u(x,0) = 3\sin(2\pi x) \qquad \text{on } [0,1] \tag{6.2}$$

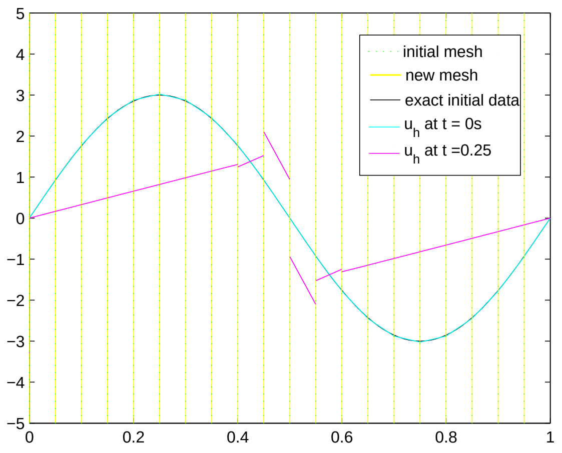
with periodic boundary conditions for some linear and nonlinear test problems.

Firstly, we consider the zero boundary speed case and compare the solutions to results obtained using the stationary 2nd order RKDG method from Chapter 2. We then look at the results for boundary speeds derived form the notional shock speeds at each boundary, and the average speed and fixed speed methods for controlling the cell distribution.

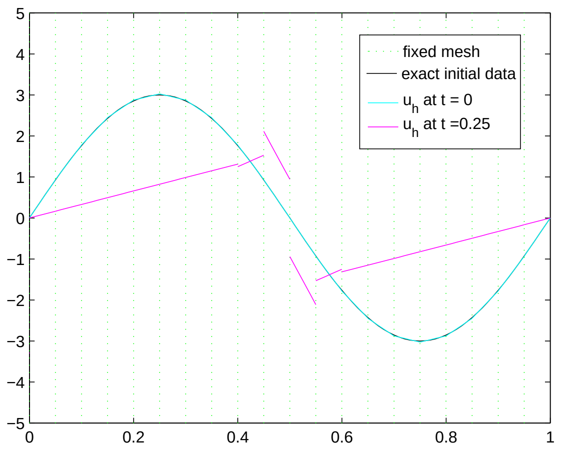
6.1 Reversion to the Stationary DG Method

Setting all boundary speeds to be zero for all time, equation (5.1) simplifies to equation (2.3), indicating that the full-DG method should revert to a stationary DG method under these conditions.

We solve the conservation law problem given by (6.1, 6.2) taking $f(u) = \frac{1}{2}u^2$ and $\dot{x} = 0$.



(a) The solution for the full-DG method with $\dot{x} = 0$.



(b) The solution for the stationary 2nd order RKDG method.

Figure 6.1: The solution of inviscid burgers problem $f(u) = \frac{1}{2}u^2$ at $t = 0.25$ taking $\Delta x = 0.05$ and $\Delta t = 0.0005$.

At $T = 0.25$, we can see, from Figure 6.1, that the two methods give visually identical results. For the moving mesh method, taking $\dot{x} = 0$ has meant the boundaries have remained fixed at their initial locations and it is only through considering the absolute difference between the solutions for u_h at the nodes, as in Figure 6.2, that we are able to see any difference in the numerical approximations found by the RKDG and full DG methods.

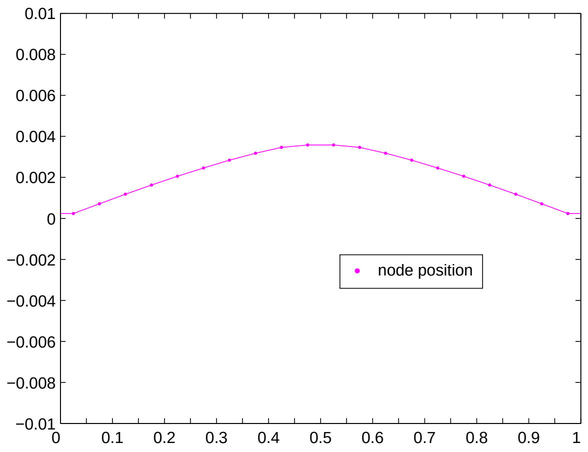


Figure 6.2: The absolute difference in solution between the 2nd order RKDG method and the full-DG method with $\dot{x} = 0$, for the inviscid burgers problem $f(u) = \frac{1}{2}u^2$ at $t = 0.25$ taking $\Delta x = 0.05$ and $\Delta t = 0.0005$.

Numerical investigations into stability indicate that for a linear $f(u) = cu$, the stability of the full-DG method with $\dot{x} = 0$ is comparable with that of the 2nd order RKDG method which is given in [4] as

$$c\frac{\Delta t}{\Delta x} \leq \frac{1}{3}.$$

In the nonlinar case, again the two methods exhibit similar stability. For the inviscid burger problem $f(u) = \frac{1}{2}u^2$, the numerical results were stable when $\frac{\Delta t}{\Delta x} \leq \frac{1}{10}$. Taking into consideration the value of $f'(u)$ for our intial data, the numerical results indicate the stability condition to be approximately

$$f'(u)\frac{\Delta t}{\Delta x} \leq \frac{1}{3}.$$

6.2 Moving mesh for linear advection

We now solve the conservation law problem (6.1, 6.2) for linear advection, taking $f(u) = 3u$ and finding the boundary speeds through the local discontinuities at each cell boundary.

At each boundary, equation (5.6) simplifies to give the wave speed, meaning all boundaries move with uniform speed and no provision need be made for boundary overtaking. The initial data is moved with the wave speed as in Figure 6.3.

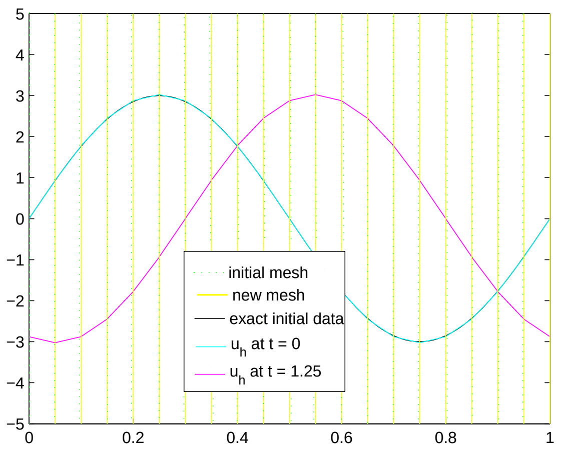


Figure 6.3: The solution for the linear advection problem $f(u) = 3u$ at $t = 0.1$ using the full-DG method taking the boundary speeds to be the local shock speeds, and with $\Delta x = 0.05$ and $\Delta t = 0.0005$.

Note, the new mesh has moved from its original position, so although the two meshes may appear aligned, for any given cell, boundary $x_{j+1/2}(t = 0)$ will not be aligned with boundary $x_{j+1/2}(t = 0.1)$ etc.

Numerical investigations indicate that the stability for the moving mesh algorithm with constant, uniform boundary speeds determined by the wave speed, is much better than for the stationary methods. This is most likely because, for this particular speed, the approximation u_h on each cell does not change over time;

the cell is simply moved. Figure 6.4 shows good results, even though $c\frac{\Delta t}{\Delta x} = 15$.

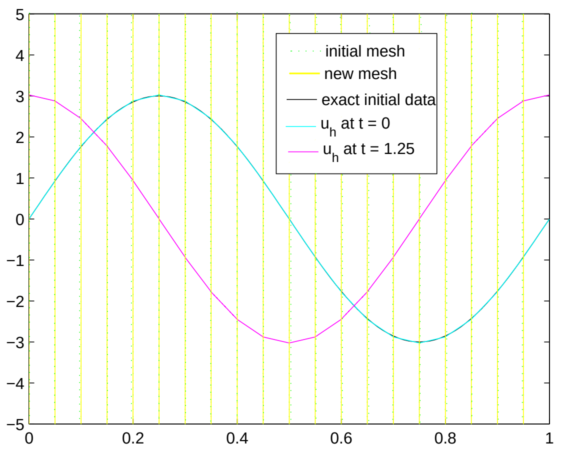


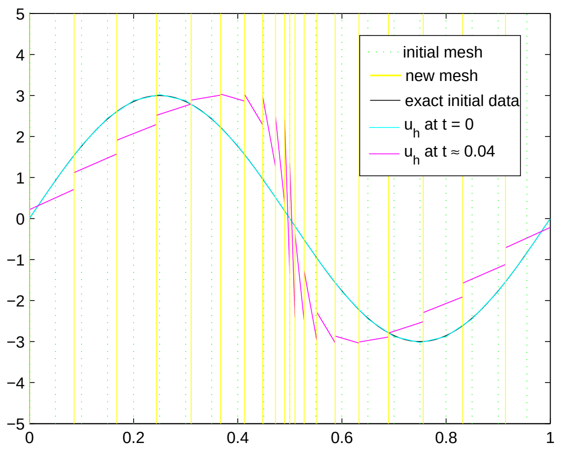
Figure 6.4: The solution for the linear advection problem $f(u) = 3u$ at $t = 1.25$ using the full-DG method taking the boundary speeds to be the local shock speeds, and with $\Delta x = 0.05$ and $\Delta t = 0.25$.

Again, we should note that although the two meshes may appear aligned, the individual boundaries are no longer in the position they started at.

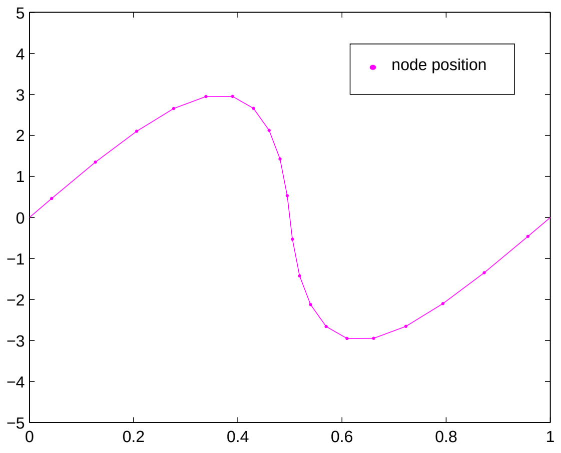
6.3 Moving mesh for nonlinear motion with controlled cell distribution

We now consider the nonlinear inviscid burgers problem, solving (6.1, 6.2) with $f(u) = \frac{1}{2}u^2$. Taking the initial mesh to be equally distributed with $\Delta x = 0.05$, and allowing timesteps of $0.001 \geq \Delta t \geq 0.0001$, we run the full-DG method, using local shock speeds for the boundary speeds, until cells are about to fall below the minimum width of $\Delta_j = 0.01$ and cell distribution must be further controlled.

For the test case, no intervention is required until the 40th timestep ($t \approx 0.04$). The numerical solution and mesh at the last time before intervention is required are shown in figure 6.5. In the nodal plot, we see that the gradient of the solution is becoming very steep, although the vertical shock has not yet formed.



(a) The numerical solution on each cell.



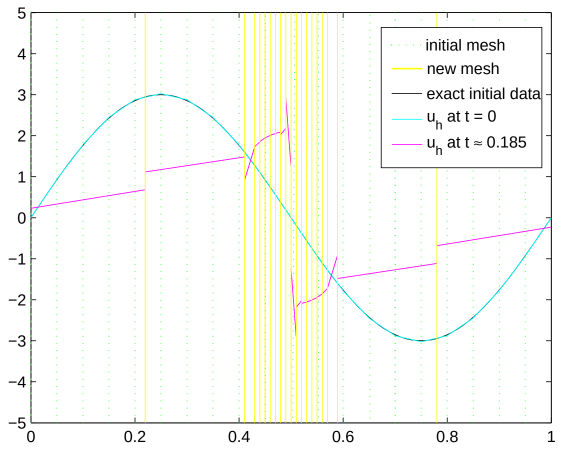
(b) The numerical solution at each node.

Figure 6.5: The solution for the problem $f(u) = \frac{1}{2}u^2$ at the last timestep before any intervention is required to control boundary movement.

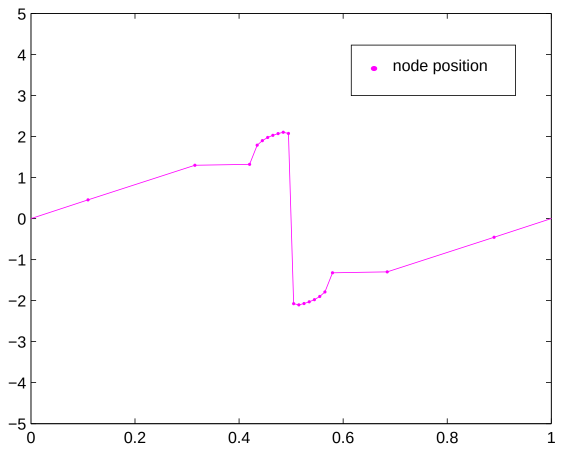
To progress further without cell widths falling below the minimum values, we need to control the cell distribution. We now investigate the two techniques discussed in Section 5.2.2, looking first at adjusting boundary speeds through averaging, and then at adopting a fixed boundary speed across all boundaries.

6.3.1 Adjustments through speed averaging

We first use the average speed technique to amend boundary speeds around problem cells which would otherwise become too small at the next timestep.



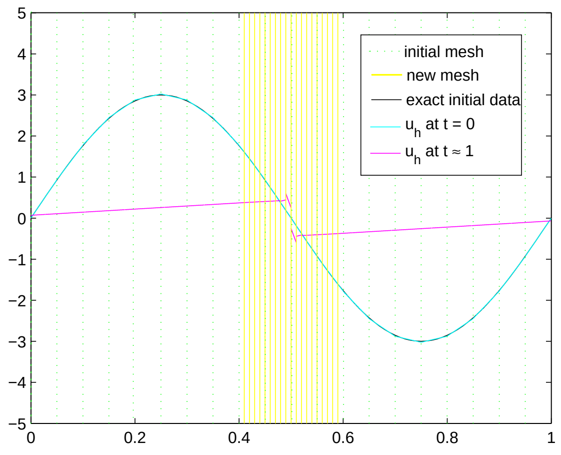
(a) The numerical solution on each cell.



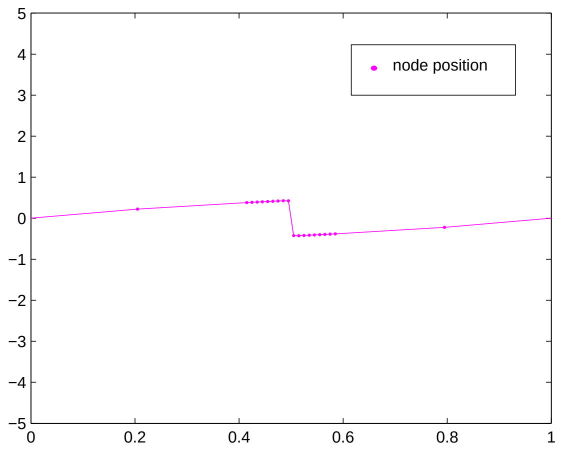
(b) The numerical solution at each node.

Figure 6.6: The solution at $t \approx 0.185$ for the average speed technique, showing difficulties in fully capturing the shock.

Initially we obtain good results as the shock begins to develop, but at later timesteps we begin to notice difficulties in fully capturing the shock, most noticable on the nodal plot as shown in Figure 6.6. The degree of this problem varies with each further timestep, but when no further node movement can occur the shock capture actually becomes much sharper as shown in Figure 6.7.



(a) The numerical solution on each cell.



(b) The numerical solution at each node.

Figure 6.7: The solution at $t \approx 1$ for the average speed technique shows sharp shock capture when it is no longer possible for any boundaries to move.

Although the shock capture is generlly not as good as the results seen for the stationary methods, even though the nodes are more densly concentrated, this method has the potential to allow different regions of the mesh to move at different average speeds, allowing multiple features of interest to be followed which may prove useful in some situations.

6.3.2 Adjustments through adopting a fixed speed

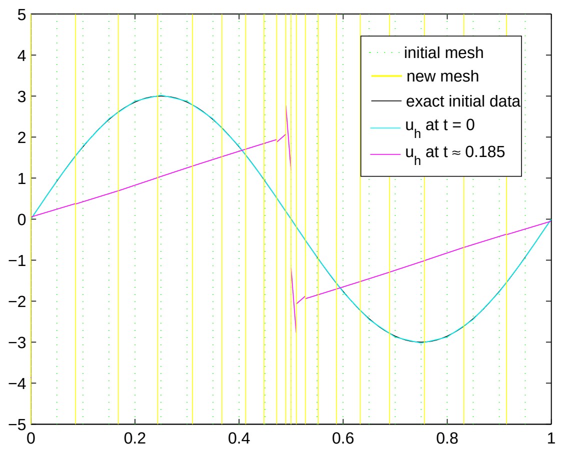
We now consider using the adoption of a fixed speed to control the cell distribution.

Using the fixed speed method, all boundaries are moved with a uniform velocity determined by an approximation to the shock speed. For the test case given by (6.1, 6.2) with $f(u) = \frac{1}{2}u^2$, we have a stationary shock, so the speed with which the cells move is approximately zero.

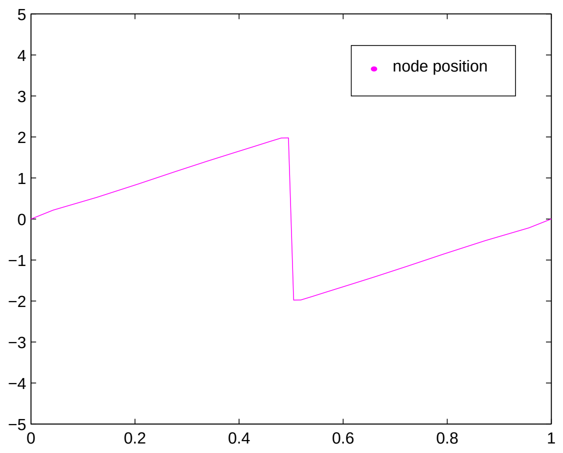
As shown in Figure 6.8, the method is able to capture the shock much more sharply than the average speed method of adjustments. The spread of the cells remains constant, and the dense region of cells remains around the shock, even in the case of a moving shock as shown in Figure 6.9.

A limitation of the method is that all cells move with the shock speed associated with the largest feature and so smaller features may be less well captured. This method is therefore not ideal for a system with multiple shocks, but could work well for single shock systems.

Of the two methods for controlling cell distribution, the fixed speed method has yielded the best results for the single shock problem and so will be the prefered method if such a technique is required when we apply the full-DG method to a 1D system of shallow water equations.

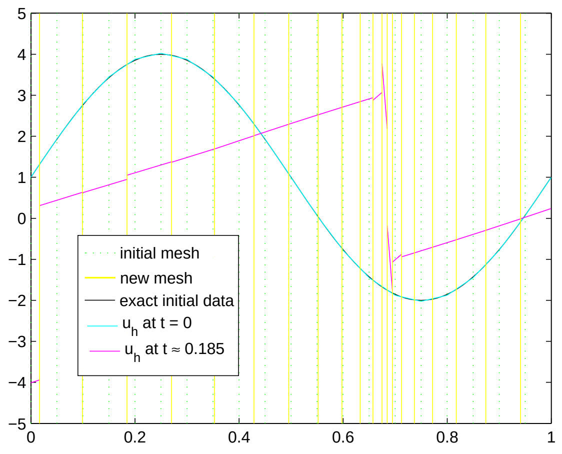


(a) The numerical solution on each cell.

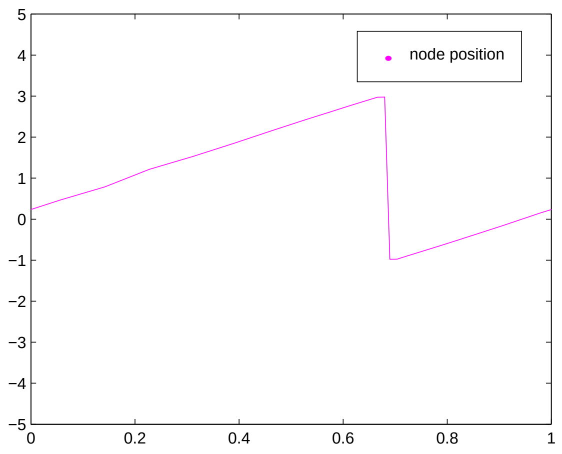


(b) The numerical solution at each node.

Figure 6.8: The solution for the stationary shock test problem using fixed speed adjustments at $t \approx 0.185$.



(a) The numerical solution on each cell.



(b) The numerical solution at each node.

Figure 6.9: The solution for the moving shock problem $(u(x,0) = 3\sin(2\pi x) + 1)$ using fixed speed adjustments at $t \approx 0.185$.

Chapter 7

Shallow Water Equations

The shallow water equations may be used for modelling fluid flow in situations where the vertical motion can be considered insiginificant in comparision to the horizontal motion. The equations desribe the flow of a fluid at a single pressure height and are not able to model factors which vary with height. For the use of the equations to be appropriate, the wavelength of the phenonmenon being modelled must be much larger than the depth of the fluid. This means that, in spite of the name, shallow water equations may be used in deep ocean basins if we are modelling tidal motion due to the large tidal wavelength.

The stationary DG method has been applied to many shallow water problems, with Yu and Kyozuka [17] investigating both tidal flows and the dam-break problem. For application of the moving mesh algorithm developed in Chapter 5, we shall look particularly at shallow water equations applied to fluid flow in a river, taking the surface of the river as our pressure surface.

In 1D, the shallow water equations for Figure 7.1 can be expressed as

$$h_t + (hu)_x = 0, \tag{7.1}$$

$$(hu)_t + (\frac{1}{2}gh^2 + u^2h)_x = -gh\frac{\partial B}{\partial x}. \tag{7.2}$$

where h is the height of water above the river bed B, and u is the horizontal velocity of the water [12]. We assume a stationary bed, making no allowances for sediment movement, and so take B to be independent of time.

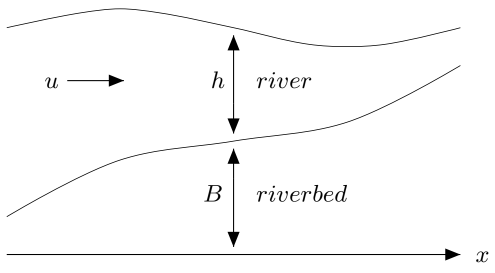


Figure 7.1: Shallow water variables: h is the height of water above the river bed B; u is the horizontal velocity of water.

Equations (7.1) and (7.2) respresent conservation of mass and conservation of momentum respectively. We may rewrite the later in terms of the conserved quantity $Q = hu$ as

$$Q_t + (\frac{1}{2}gh^2 + \frac{Q^2}{h})_x = -gh\frac{\partial B}{\partial x}. \tag{7.3}$$

7.1 Application of the full-DG Method to the Shallow Water System

Writing the shallow water equations in matrix form, we solve the conservation law problem

$$\frac{\partial}{\partial t}\begin{bmatrix} h \\ Q \end{bmatrix} + \frac{\partial}{\partial x}\begin{bmatrix} hu \\ \frac{1}{2}gh^2 + \frac{Q^2}{h} \end{bmatrix} = \begin{bmatrix} 0 \\ -gh\frac{\partial B}{\partial x} \end{bmatrix} \quad \text{on } [0,1]\times[0,T] \tag{7.4}$$

$$\begin{bmatrix} h(x,0) \\ Q(x,0) \end{bmatrix} = \begin{bmatrix} h_0(x) \\ h_0(x)u_0(x) \end{bmatrix} \quad \text{on } [0,1] \tag{7.5}$$

with periodic boundary conditions.

7.1.1 The inclusion of boundary speeds

The conservation law problem (7.4, 7.4) must be reworked to include boundary speeds $\dot{x}$.

To include the boundary speeds into the first shallow water equation, we follow the derivation in Chapter 5.

We use Leibniz rule

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_{j-1/2}}^{x_{j+1/2}} m dx = m\dot{x}|_{x_{j+1/2}} - m\dot{x}|_{x_{j-1/2}} + \int_{x_{j-1/2}}^{x_{j+1/2}} \frac{\partial m}{\partial t}\,\mathrm{d}x$$

to expand

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_{j-1/2}}^{x_{j+1/2}} vh\,\mathrm{d}x.$$

Taking $m = vh$ where $v(x)$ moves with $\frac{\mathrm{d}x}{\mathrm{d}t}$, we have

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_{j-1/2}}^{x_{j+1/2}} vh\,\mathrm{d}x &= vh\dot{x}|_{x_{j+1/2}} - vh\dot{x}|_{x_{j-1/2}} + \int_{x_{j-1/2}}^{x_{j+1/2}} \frac{\partial}{\partial t}vh\,\mathrm{d}x \\
&= \int_{x_{j-1/2}}^{x_{j+1/2}} \frac{\partial}{\partial x}(vh\dot{x})\,\mathrm{d}x + \int_{x_{j-1/2}}^{x_{j+1/2}} \frac{\partial}{\partial t}(vh)\,\mathrm{d}x \\
&= \int_{x_{j-1/2}}^{x_{j+1/2}} \left[v\frac{\partial}{\partial x}(h\dot{x}) + \frac{\partial v}{\partial x}h\dot{x} + v\frac{\partial h}{\partial t} + \frac{\partial v}{\partial t}h\right]\,\mathrm{d}x \\
&= \int_{x_{j-1/2}}^{x_{j+1/2}} v\left[\frac{\partial}{\partial x}(h\dot{x}) + \frac{\partial h}{\partial t}\right]\,\mathrm{d}x + \int_{x_{j-1/2}}^{x_{j+1/2}} h\left[\frac{\partial v}{\partial t} + \dot{x}\frac{\partial v}{\partial x}\right]\,\mathrm{d}x.
\end{aligned}$$

As $v(x)$ moves with $\frac{\mathrm{d}x}{\mathrm{d}t}$, the last integral term is zero and substituting in for $\frac{\partial h}{\partial t}$ from our first shallow water equation (7.1), we obtain

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_{j-1/2}}^{x_{j+1/2}} vh\,\mathrm{d}x = \int_{x_{j-1/2}}^{x_{j+1/2}} v\left(\dot{x}h - hu\right)_x\,\mathrm{d}x.$$

Using integration by parts this becomes

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_{j-1/2}}^{x_{j+1/2}} vh\,\mathrm{d}x = -\,v\left(hu - \dot{x}h\right)\Big|_{x_{j-1/2}}^{x_{j+1/2}} + \int_{x_{j-1/2}}^{x_{j+1/2}} \left(hu - \dot{x}h\right)\frac{\partial v}{\partial x}\,\mathrm{d}x$$

To include the boundary speeds into the second shallow water equation, we again follow the derivation in Chapter 5, and obtain

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_{j-1/2}}^{x_{j+1/2}} vQ\,\mathrm{d}x = \int_{x_{j-1/2}}^{x_{j+1/2}} v\left[\frac{\partial}{\partial x}(Q\dot{x}) + \frac{\partial Q}{\partial t}\right]\,\mathrm{d}x + \int_{x_{j-1/2}}^{x_{j+1/2}} Q\left[\frac{\partial v}{\partial t} + \dot{x}\frac{\partial v}{\partial x}\right]\,\mathrm{d}x.$$

Again, we take $v(x)$ to move with $\frac{\mathrm{d}x}{\mathrm{d}t}$ so the last integral on the right-hand side disappears. However, now when we substitute in from the second shallow water equation (7.3), we have an additional term as the right-hand side of (7.3) is non-zero.

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_{j-1/2}}^{x_{j+1/2}} vQ\,\mathrm{d}x = \int_{x_{j-1/2}}^{x_{j+1/2}} v\left[\frac{\partial}{\partial x}(Q\dot{x}) - \frac{\partial}{\partial x}\left(\frac{1}{2}gh^2 + \frac{Q^2}{h}\right) - gh\frac{\partial B}{\partial x}\right]\,\mathrm{d}x.$$

Using integration by parts, this may be rewritten as

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_{j-1/2}}^{x_{j+1/2}} vQ\,\mathrm{d}x = &- v\left(\left(\frac{1}{2}gh^2 + \frac{Q^2}{h}\right) - \dot{x}Q\right)\Bigg|_{x_{j-1/2}}^{x_{j+1/2}} \\ &+ \int_{x_{j-1/2}}^{x_{j+1/2}}\left(\left(\frac{1}{2}gh^2 + \frac{Q^2}{h}\right) - \dot{x}Q\right)\frac{\partial v}{\partial x}\,\mathrm{d}x - \int_{x_{j-1/2}}^{x_{j+1/2}} vgh\frac{\partial B}{\partial x}\,\mathrm{d}x.\end{aligned}$$

7.1.2 The weak formulation

We define the finite dimensional subspace V_h to be

$$V_h = \left\{v \in L^1(0,1) : v|_{I_j} \in P^k(I_j)\ ,\ j = 1,\ldots,N\right\},$$

where P^k is the set of all polynomials up to degree k, and replace the smooth $v(x)$ with a test function $v_h \in V_h$. The exact solutions h, u, and hence Q are replaced by numerical approximations h_h, u_h and Q_h respectively.

The full problem, in matrix form, including initial conditions then becomes

$$\forall j = 1, \ldots, N$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_{j-1/2}}^{x_{j+1/2}} v_h \begin{bmatrix} h_h \\ Q_h \end{bmatrix} \mathrm{d}x = -v_h \begin{bmatrix} h_h u_h - \dot{x} h_h \\ \frac{1}{2} g h_h^2 + \frac{Q_h^2}{h_h} - \dot{x} Q_h \end{bmatrix} \Bigg|_{x_{j-1/2}}^{x_{j+1/2}}$$

$$+ \int_{x_{j-1/2}}^{x_{j+1/2}} \begin{bmatrix} h_h u_h - \dot{x} h_h \\ \frac{1}{2} g h_h^2 + \frac{Q_h^2}{h_h} - \dot{x} Q_h \end{bmatrix} \frac{\partial v_h}{\partial x} \,\mathrm{d}x - \int_{x_{j-1/2}}^{x_{j+1/2}} \begin{bmatrix} 0 \\ v_h g h_h \frac{\partial B}{\partial x} \end{bmatrix} \mathrm{d}x \quad (7.6)$$

$$\int_{x_{j-1/2}}^{x_{j+1/2}} \begin{bmatrix} h_h(x,0) \\ Q_h(x,0) \end{bmatrix} v_h \,\mathrm{d}x = \int_{x_{j-1/2}}^{x_{j+1/2}} \begin{bmatrix} h_0(x) \\ h_0(x) u_0(x) \end{bmatrix} v_h \,\mathrm{d}x \qquad (7.7)$$

We note that

$$F = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} h_h u_h - \dot{x} h_h \\ \frac{1}{2} g h_h^2 + \frac{Q_h^2}{h_h} - \dot{x} Q_h \end{bmatrix}$$

is undefined at cell boundaries due to this discontinuities in h_h, u_h and Q_h, and so we introduce a numerical flux H such that

$$H(x) = H(\{h_h(x)^+, u_h(x)^+\}, \{h_h(x)^-, u_h(x)^-\}) \approx F(h_h(x), u_h(x)).$$

As F is a vector, we use a slightly different version of the local Lax Friedrichs formula to that seen in previous chapters. Basing the value of c on the eigenvalues of the Jacobian matrix [12], we use

$$\begin{aligned} H \quad (a,b) &= \frac{1}{2}\left[F(a) + F(b) - c(b-a)\right], \\ c \quad &= \max\left\{ \begin{bmatrix} 1 \\ u_h^+ \pm \sqrt{g h_h^+} \end{bmatrix}, \begin{bmatrix} 1 \\ u_h^- \pm \sqrt{g h_h^-} \end{bmatrix} \right\}. \end{aligned}$$

We take the test functions $v_h(x)$ to be Legendre polynomials and express the numerical solutions h_h and Q_h as a sum of Legendre polynomial basis functions:

$$h_h(x,t) = \sum_{l=0}^{k} w_j^l(t) \phi_l(x)$$

$$Q_h(x,t) = \sum_{l=0}^{k} z_j^l(t) \phi_l(x)$$

where $\phi_l(x) = P_l\left(\frac{2(x-x_j)}{\Delta_j}\right)$ and w_j^l and z_j^l are coefficients to be found.

Now, through the orthogonality properties for the Legendre polynomials, the full problem may be given as

for $j = 1, \ldots, N$

Equations for h:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1/3 \end{bmatrix} \begin{bmatrix} \dot{\Delta}_j w_j^0 + \Delta_j \dot{w}_j^0 \\ \dot{\Delta}_j w_j^1 + \Delta_j \dot{w}_j^1 \end{bmatrix} = - \begin{bmatrix} H(x_{j+1/2}) - H(x_{j-1/2}) \\ H(x_{j+1/2}) + H(x_{j-1/2}) \end{bmatrix} + \begin{bmatrix} 0 \\ f_1(x_{j+1/2\sqrt{3}}) + f_1(x_{j-1/2\sqrt{3}}) \end{bmatrix} \tag{7.8}$$

$$\begin{bmatrix} w_j^0(0) \\ w_j^1(0) \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\left\{h_0\left(x_{j+1/2\sqrt{3}}\right) + h_0\left(x_{j-1/2\sqrt{3}}\right)\right\} \\ \frac{\sqrt{3}}{2}\left\{h_0\left(x_{j+1/2\sqrt{3}}\right) - h_0\left(x_{j-1/2\sqrt{3}}\right)\right\} \end{bmatrix} \tag{7.9}$$

Equations for Q:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1/3 \end{bmatrix} \begin{bmatrix} \dot{\Delta}_j z_j^0 + \Delta_j \dot{z}_j^0 \\ \dot{\Delta}_j z_j^1 + \Delta_j \dot{z}_j^1 \end{bmatrix} = - \begin{bmatrix} H(x_{j+1/2}) - H(x_{j-1/2}) \\ H(x_{j+1/2}) + H(x_{j-1/2}) \end{bmatrix} + \begin{bmatrix} 0 \\ f_2(x_{j+1/2\sqrt{3}}) + f_2(x_{j-1/2\sqrt{3}}) \end{bmatrix} - \frac{\Delta_j}{2\sqrt{3}} \begin{bmatrix} f_3(x_{j+1/2\sqrt{3}}) + f_3(x_{j-1/2\sqrt{3}}) \\ f_3(x_{j+1/2\sqrt{3}}) - f_3(x_{j-1/2\sqrt{3}}) \end{bmatrix} \tag{7.10}$$

$$\begin{bmatrix} z_j^0(0) \\ z_j^1(0) \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\left\{h_0\left(x_{j+1/2\sqrt{3}}\right) u_0\left(x_{j+1/2\sqrt{3}}\right) + h_0\left(x_{j-1/2\sqrt{3}}\right) u_0\left(x_{j-1/2\sqrt{3}}\right)\right\} \\ \frac{\sqrt{3}}{2}\left\{h_0\left(x_{j+1/2\sqrt{3}}\right) u_0\left(x_{j+1/2\sqrt{3}}\right) - h_0\left(x_{j-1/2\sqrt{3}}\right) u_0\left(x_{j-1/2\sqrt{3}}\right)\right\} \end{bmatrix} \tag{7.11}$$

where $f_1 = h_h u_h - \dot{x} h_h$, $f_2 = \frac{1}{2} g h_h^2 + \frac{Q_h^2}{h_h} - \dot{x} Q_h$ and $f_3 = g h_h \frac{\partial B}{\partial x}$. We assume that $\dot{x}$ is linear on each cell to obtain $\dot{x}(x_{j-1/2\sqrt{3}})$ and $\dot{x}(x_{j+1/2\sqrt{3}})$.

7.1.3 Choosing boundary speeds

In the full-DG method from Chapter 5, the boundary speeds are taken as the local shock speeds at each boundary (5.6). However, we now have two variables h and Q, both of which are discontinuous at boundaries and so both could provide a boundary speed.

We select to use the variable h and the first shallow water equation (7.1) to drive the mesh movement, so boundary speeds may be calulated by

$$\dot{x} = \begin{cases} u_h^+ & \text{if } u_h^+ - u_h^- \approx 0 \\ \frac{[h_h u_h]}{[h_h]} & \text{otherwise} \end{cases} \tag{7.12}$$

where $[h_h]$ and $[h_h u_h]$ denote the jumps in h_h and $h_h u_h$ respectively.

When it becomes necessary to control the cell distribution, again we will choose to favour the information provided by the h variable, and will move the speed at the speed of the boundary nearest the steepest gradient in h.

7.1.4 Time integration

The time integration method requires no further modification for working with a system of two equations as it is explicit and the values of h and Q at the new timestep are found from previous known values only. For completeness, the timestepping method, with cell distribution control, is included here.

We partition $[0, T]$ into M intervals of variable size Δt_m where m denotes the timestep.

To step through time and find $h_h(t = T)$, $Q_h(t = T)$, we will use an Euler timestepping method:

- Solve (7.9, 7.11) to obtain $w_j^0(0)$, $w_j^1(0)$, $z_j^0(0)$, and $z_j^1(0)$ and hence find $h_h(t=0)$, $Q_h(t=0)$;
- For $m = 0, \ldots, M-1$,
  - Find the boundary speeds $\dot{x}$ using (7.12) on h.
  - Check that the minimum cell width and timestep criteria are not broken, and modifiy boundary speeds as necessary.
  - Solve (7.8, 7.10) to obtain $\dot{w}_j^0(t=m)$, $\dot{w}_j^1(t=m)$, $\dot{z}_j^0(t=m)$, $\dot{z}_j^1(t=m)$;
  - Compute

    $$\begin{bmatrix} w_j^0(t=m+1) \\ w_j^1(t=m+1) \end{bmatrix} = \begin{bmatrix} w_j^0(t=m) \\ w_j^1(t=m) \end{bmatrix} + \Delta t \begin{bmatrix} \dot{w}_j^0(t=m) \\ \dot{w}_j^1(t=m) \end{bmatrix}$$

    and similarly for $z_j(t=m+1)$;
  - and hence find $h_h(t=m+1)$ and $Q_h(t=m+1)$

We now have a velocity-based moving mesh DG technique for a 1D system, including a mechanism to control cell distribution. In Chapter 8, we briefly consider the tidal bore and dam-break applications.

Chapter 8

Numerical Results for Shallow Water Equations

The full-DG moving mesh method with fixed speed adjustements to control cell distribution has been derived for a 1D system of shallow water equations in Chaptershallow. We now apply the method to two simple test problems, firstly considering the dam-break problem, as considered by Yu and Kyozuka [17], and then a tidal bore.

8.1 A Dam-Break

The dam-break problem, with a well-documented solution, has been frequently used as a preliminary test for modelling a system of shallow water equations. The set-up usually starts with the fluid being at rest and partioned into two heights by a dam which is then instantaneously removed at $t = 0$ and the fluid begins to flow.

Yu and Kyozuka [17] consider the conservation law problem

$$\frac{\partial}{\partial t}\begin{bmatrix} h \\ Q \end{bmatrix} + \frac{\partial}{\partial x}\begin{bmatrix} hu \\ \frac{1}{2}gh^2 + \frac{Q^2}{h} \end{bmatrix} = 0 \qquad \text{on } [0,1] \times [0,T]$$

taking the initial conditions to be

$$\begin{aligned} h(x,0) &= \begin{cases} 1\text{m} & \text{if } x \leq 0.5\text{m} \\ 0.5\text{m} & \text{if } x > 0.5\text{m} \end{cases} \\ u(x,0) &= 0. \end{aligned}$$

The solution they obtain, taken directly from [17] is given in Figure 8.1, and shows the formation of two shocks from the intial single shock.

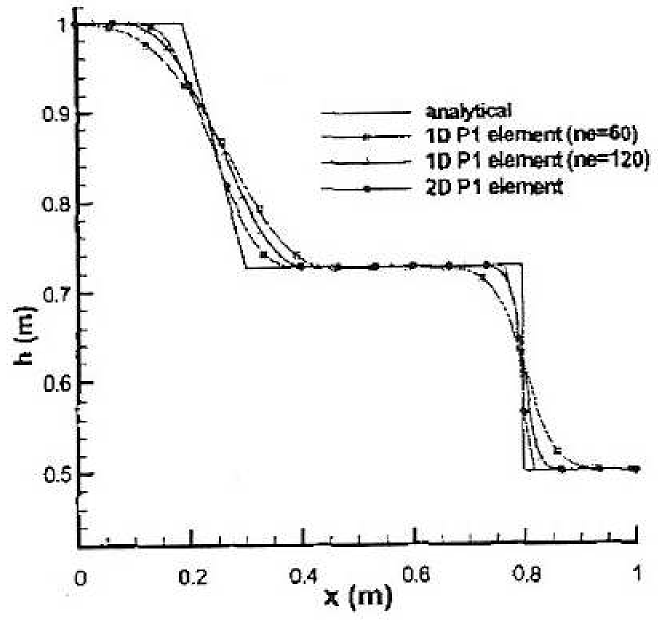


Figure 8.1: Taken from [17] Fig.2 p1528, Mean depth for the one-dimensional dam-break at $t = 0.1$ s of analytical solution and solutions obtained from various DG FEM methods.

8.1.1 Our problem

Due to the periodic boundary conditions used in the derivation and implementation of our moving mesh method, the dam-break problem must be slightly re-designed. We also note that we are working with a flat-bottomed bed, taking

$\frac{\partial B}{\partial x} = 0$, and h therefore represents the surface height, as well as the height of water above the bed.

The conservation law problem to be solved is given by

$$\frac{\partial}{\partial t}\begin{bmatrix} h \\ Q \end{bmatrix} + \frac{\partial}{\partial x}\begin{bmatrix} hu \\ \frac{1}{2}gh^2 + \frac{Q^2}{h} \end{bmatrix} = 0 \qquad \text{on } [0,1] \times [0,T]$$

taking the initial conditions to be

$$\begin{aligned} h(x,0) &= \begin{cases} 0.5 & \text{if } 0 \leq x \leq 0.1 \\ 1 & \text{if } 0.1 < x < 0.5 \\ 0.5 & \text{if } 0.5 \leq x \leq 1 \end{cases} \\ u(x,0) &= 0. \end{aligned}$$

8.1.2 Results

We firstly check the basic algorithm, without control of cell distribution, by setting all boundary speeds to zero, giving a stationary mesh. We set the maximum / minimum parameters to be given by

- Initial cell width: $\Delta_j = 0.005$
- Minimum allowed cell width: $\Delta_{j\ \text{min}} = 0.005$
- Minimum allowed timestep: $\Delta t_{\text{min}} = 0.00001$
- Maximum allowed timestep: $\Delta t_{\text{max}} = 0.00001$

After 200 steps, ($t = 0.002$), we see from Figure 8.2 that the right-hand shock has begun to form into two shocks, as seen in Figure 8.1. However, on most of the region, the values are registered as 'NaN', possibly due to complications with the left-hand shock and periodicity. At further timesteps, the solution breaks down.

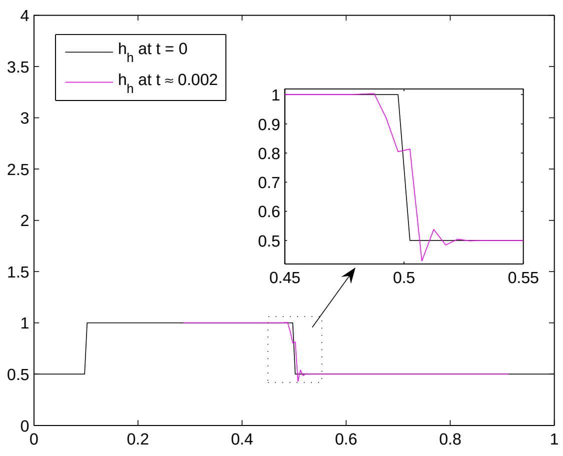


Figure 8.2: The height of the water in the dam-break problem at $t = 0.002$, found using the full-DG method with stationary boundaries.

We now compare the results for a moving mesh, allowing boundary speeds to be taken as the notional shock speed at each boundary, or fixed at the largest shock speed. To allow some movement, we set the maximum / minimum parameters to be given by

- Initial cell width: $\Delta_j = 0.005$
- Minimum allowed cell width: $\Delta_{j\ \text{min}} = 0.005$
- Minimum allowed timestep: $\Delta t_{\text{min}} = 0.00001$
- Maximum allowed timestep: $\Delta t_{\text{max}} = 0.0001$

Again, we view the solution after 200 steps ($t \approx 0.002$) and see in Figure 8.3 that the results are very similar to that of the stationary mesh. If we encourage further movement by reducing the minimum cell width to 0.001, without changing any other parameters, the system breaks down due to a problem with the implementation of the cell distribution algorithm. Due to time constraints, this was not investigated any further, but one may speculate that in addition to the periodicity issues faced by the stationary mesh, the problems may be also due to the cell distibution algorithm being poorly designed to cope with the multiple shocks that are present.

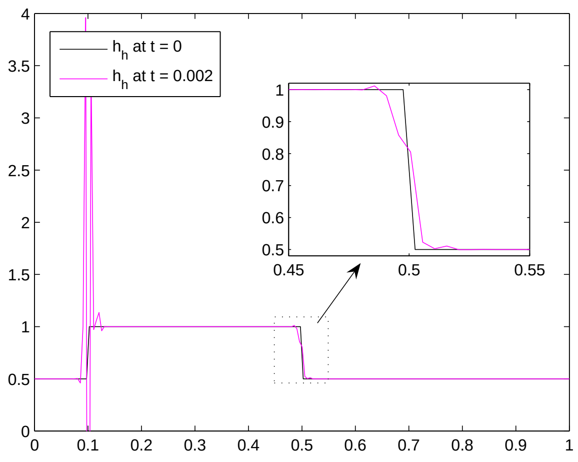


Figure 8.3: The height of the water in the dam-break problem at $t \approx 0.002$, found using the full-DG method with some boundary movement.

8.2 A Tidal Bore

A tidal bore is formed when the leading edge of an incoming tide forms a wave which travels up a river against the prevailing current. It may appear as a single breaking wavefront or as a smooth wave followed by a series of solitons. This application may provide a single-shock problem which the cell distrbution algorithm based on a fixed speed, may be better able to cope with.

Generally formed in rivers where both the depth and width decrease to create a funnelling effect, examples may be found around the world including the Severn Bore on the River Severn. The modelling of tidal bores has become important in recent years to help with river planning as they are known to be potentially dangerous for shipping, but also act as an attraction to tourists.

8.2.1 Our problem

The height and speed of a bore is effected by many factors including the height of freshwater in the river, offshore and opposing winds and pressure levels. However, we will assume a very simple model for our bore, and seek to solve the conservation law problem

$$\frac{\partial}{\partial t}\begin{bmatrix} h \\ Q \end{bmatrix} + \frac{\partial}{\partial x}\begin{bmatrix} hu \\ \frac{1}{2}gh^2 + \frac{Q^2}{h} \end{bmatrix} = 0 \qquad \text{on } [0,1]\times[0,T]$$

where

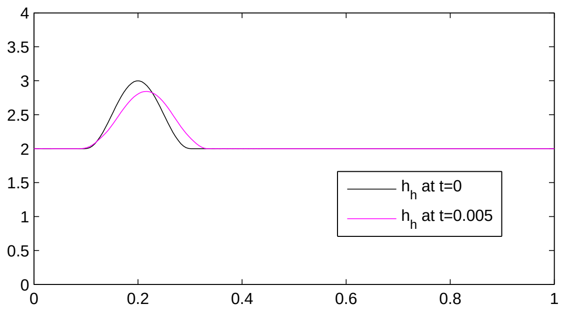
$$\begin{aligned} h(x,0) &= \begin{cases} 2 & \text{if } 0 \le x \le 0.1 \\ (sin(5*pi*(x-0.1))).^2+2 & \text{if } 0.1 < x < 0.3 \\ 2 & \text{if } 0.3 \le x \le 1 \end{cases} \\ u(x,0) &= 3. \end{aligned}$$

We shall use periodic boundary conditions, as this is how the full-DG method has been developed, although we note that they are unrealistic for a river.

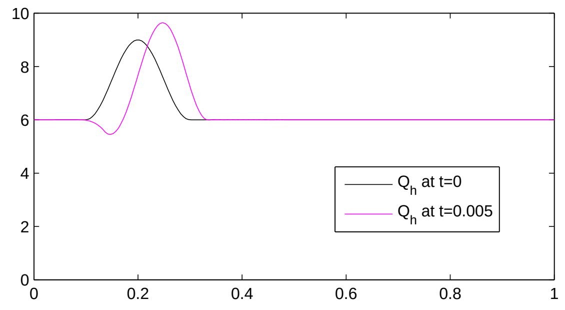
8.2.2 Results

The system was solved for stationary boundaries, setting $\dot{x} = 0$, and the results from an early timestep may be seen in Figure 8.4.

Without any results to compare this to, we cannot be sure that this is giving the correct solution, although the results are plausible, with the increase in Q, without the increase in h, indicating an increase in the velocity u of the water. For the initial conditions considered, no shock appears to be forming, but this is not an immediate worry as we have not verified that a shock should form. At further timesteps, and when considering a moving mesh, the solution breaks down, most likely due to the periodic boundary conditions and difficulties in the cell distribution algorithm.



(a) The numerical solution for h



(b) The numerical solution for Q

Figure 8.4: The height of the water in the tidal bore problem at $t = 0.002$, found using the full-DG method with zero boundary speeds, $dt = 0.00001$, and $dx = 0.005$.

8.3 Conclusions

While we may have obtained some promising initial results for the dam-break and tidal bore problems, there is clearly much more we could have investigated had there been sufficient time. We have used periodic boundary conditions which were unrealistic for the test problems and may have introduced errors. Ideally, the full-DG method would be rederived, allowing for non-periodic boundary conditions, before any further investigations into the moving mesh algorithm were carried out. The test problems may then be extended to investigate the effects of the shape of the river bed on the fluid flow.

Chapter 9

Summary and Further Work

9.1 Summary

This dissertation looked to find a moving mesh method for use with the Discontinuous Galerkin (DG) Finite Element Method , and this has been achieved for a single equation, although only preliminary results were avaliable for the extended algorithm for a 1D system.

We began by considering the stationary Runge-Kutta DG method developed by Cockburn and Shu [11], and commonly used grid adaptation techniques, including velocity-based moving mesh methods. From this, we persued two different routes for obtaining a moving DG method.

Firstly, we considered cell-based moving mesh methods, where the boundary speeds were derived assuming a conservation principle on each cell. Such methods had limited success, possibly due to the use of numerical fluxes, and inconsistencies in the use of the conservation principle which directly links cell widths to the value of the numerical solution. Method B, a non-DG technique, was partially successful, being able to accurately model linear advection but encountering some difficulties due to boundary overtaking for the nonlinear case. As the method did not involve numerical fluxes, which are a main component of the DG method, and because additional controls for cell distribution may have been difficult to incorporate due to the conservation principle, this method was not pursued.

Secondly, motivated by the results of the cell-based methods, we considered a boundary-based moving mesh method derived without the use of a cell-based conservation principle. The derived full-DG method does not directly calculate the boundary speeds which instead may taken from an external source. Setting the boundary speeds to be zero, the method accurately reverted to a stationary DG method and provided comparable results to the RKDG method considered initially. Using boundary speeds derived using the notional shock speeds associated with the discontinuities in the numerical solution at cell boundaries, the mesh sucessfully adapted to cluster around a developing shock, at which point two methods for controlling cell distribution were considered. The fixed speed method for controlling cell distribution worked well, preventing further issues and allowing the clustered cells to follow the shock as it moved. The shock capture was also much sharper than that of the average speed method at the same point in time. However, the fixed speed method has no mechanism for handling multiple shocks with different shock speeds and will always choose to favour the speed of the largest shock, whereas the average speed method could more easily adapt to a multiple shock case.

The full-DG method with fixed speed boundary adjustments was then extended to the 1D system of shallow water equations. Whilst it was possible to form a method, neither of the two simple test problems was ideally suited to the periodic nature of the boundary conditions applied. Due to time constraints, the method was tested only very briefly, and although some plausible results were obtained for the method when boundaries remained stationary, significant further work would be required to obtain a working moving DG method for the system.

9.2 Extensions

Due to time constraints, we were unable to fully develop and test a velocity-based moving mesh DG method for a 1D system, and there is the potential for much futher work in this area. The periodicity of the boundary conditions was not realistic for the dam-break and tidal bore test problems, so amending the full-DG method for non-periodic boundary conditions would be a natural first step. We could then re-investigate our test problems and make more direct comparisons with existing known results for the dam-break problem.

The preliminary numerical results for the shallow water system also indicated potential problems with the cell distribution algorithm. In particular, it would be advantageous to investigate alternative ways to adjust the boundary speeds so that whilst no cell falls below the minimum allowed width, the mesh is still free to adjust to follow local features, rather than being forced to follow the largest shock. If a good method for handling multiple shocks can be found, the dam-break problem is then an ideal first test case due to the mulitple shocks formed from the single initial discontinuity.

The 1D shallow water equations incoporate a term relating to the river bed, making it possible to see how the shape of the bed effects fluid flow. In both test cases, we only considered the flat-bed problem, but it should only be a small step to add this additional term to a working method and investigate the effects of a sloping or irregular bed. Another potential extension would be to include the motion of the river bed, as studied by Hudson [12], as sand and silt are moved by the fluid. Such a method could have practical uses in river planning and management, allowing the impact of changes in the river bed to be assessed for potential hazards to shipping before any real changes are made. For a good river model, it would also be advantageous to extend the full-DG method to a multidimensional case, allowing for changes in river width as well as river depth to be included. This could be particularly important for the tidal bore test case, as bores are known to develop in rivers that not only become shallower, but also significantly narrower, creating a funnelling effect on the water.

Finally, we note that this dissertation has only very briefly considered the accuracy and stability properties of the full-DG method, even for the working single conservation law case. For the effectiveness of the full-DG method to be really known, such properties would need to be investigated in more detail and comparisons with other Finite Element and Finite Difference Methods should be made.

Bibliography

[1] M.Baines, M.Hubbard, P.Jimack. A Moving Mesh Finite Element Algorithm for Fluid Flow Problems with Moving Boundaries, *International Journal for Numerical Methods in Fluids*, Vol 47, pp 1077-1083, (2005).

[2] M.Baines, M.Hubbard, P.Jimack. A Moving Mesh Finite Element Algorithm for the Adaptive Solution of Time-Depenedent Partial Differential Equations with Moving Boundaries, *Applied Numerical Mathematics*, Vol 54, pp 450-469, (2005).

[3] G.Chavent, G.Salzano. A Finite Element Method for the 1D Water Flooding Problem with Gravity, *Journal of Computational Physics*, Vol. 45, pp 307-344, (1982).

[4] G.Chavent, B.Cockburn. The Local Projection P^0 P^1- Discontinuous-Galerkin Finite Element Method for Scalar Conservation Laws, *RAIRO Modél. Math. Anal. Numér*, No.23, pp 565-592, (1989).

[5] B.Cockburn. Discontinuous Galerkin Methods for Convection Dominated Problems, *High-Order Methods for Computational Science and Engineering*, Vol.9, Springer-Verlag, Berlin (1999).

[6] B.Cockburn, S.Hou, C.Shu. The Runge-Kutta Local Projection Discontinuous Galerkin Finite Element Method for Conservation Laws IV: The Multidimensional Case, *Mathematics of Computation*, Vol.54, No. 190, pp 545-581, (1990).

[7] B.Cockburn, S.Lin, C.Shu. TVB Runge-Kutta Local Projection Discontinuous Galerkin Finite Element Method for Conservation Laws III: One-Dimensional Systems , *Journal of Computational Physics*, Vol.84, Issue 1, pp 90-113, (1989).

[8] B.Cockburn, C.Shu. The Runge-Kutta Local Projection P^1-Discontinuous Galerkin Finite Element Method for Scalar Conservation Laws, *American Institute of Aeronautics and Astronautics*, pp 636-643, (1988).

[9] B.Cockburn, C.Shu. TVB Runge-Kutta Local Projection Discontinuous Galerkin Finite Element Method for Conservation Laws II: General Framework, *Mathematics of Computation*, Vol.52, Issue 186, pp 411-435, (1989).

[10] B.Cockburn, C.Shu. The Runge-Kutta Discontinuous Galerkin Method for Conservation Laws V: Multidimensional Systems, *Journal of Computational Physics*, Vol.141, No.2, pp 119-224, (1998).

[11] B.Cockburn, C.Shu. Runge-Kutta Discontinuous Galerkin Methods for Convection-dominated Problems, *Journal of Scientific Computing*, Vol.16, No.3, (2001).

[12] J.Hudson. Numerical Techniques for Morphodynamic Modelling, *University of Reading : Thesis*, (2001)

[13] A.Kuo, L.Polvani. Time-Dependent Fully Nonlinear Geostrophic Adjustment, *Journal of Physical Oceanography*, Vol.27, pp 1614-1634, (1997).

[14] R.Li, T.Tang. Moving Mesh Discontinuous Galerkin Method for Hyperbolic Conservtion Laws, *Journal of Scientific Computing*, Vol.27, Nos. 1-3, (2006).

[15] W.H.Reed, T.R.Hill. Triangular Mesh Methods for the Neutron Transport Equation, *Los Alamos Scientific Laboratory Technical Report* LA-UR-73-479, (1973).

[16] B.Wells, M.Baines, P.Glaister. Generation of Arbitrary Lagrangian-Eulerian (ALE) Velocities, based on Monitor Functions, for the Solution of Compressible Fluid Equations, *International Journal for Numerical Methods in Fluids*, Vol.47, pp 1375-1381, (2005).

[17] Z.Yu, Y.Kyozuka. A Discontinous Galerkin Finite Element Shallow Water Model in Simulating Tidal Flows, *OCEANS '04. MTS/IEEE TECHNO-OCEAN '04*, Vol.3, pp 1526-1531, (2004).

[18] H.Anton, I.Bivens, S.Davis. Calculus, 7th Ed. *John Wiley & Sons, Inc*, (2002).

[19] K.Eriksson, D.Estep, P.Hansbo, C.Johnson. Computational Differential Equations, *Cambridge University Press*, (1997).

[20] www.severn-bore.co.uk

[21] www.tidalbore.info