Course AE4133

CFD II Part 1: Discretisations for Compressible Flows

April 2015 (Version 2015.2)

Faculty of Aerospace Engineering

S.J. Hulshoff

HSL 0.36, Aerodynamics Group
S.J.Hulshoff@TUDelft.NL

TU Delft Delft University of Technology
Artikelnummer 06917730032

Contents

Chapter 1

Introduction

1.1 A brief overview

What's different about compressible flows?

Compressible flows contain two crucial aspects which makes their numerical treatment different from that of incompressible flows. Firstly, in their inviscid limit compressible flows are hyperbolic, and are therefore governed by wave propagation phenomena. As a result, local flow features are subject to limited zones of influence and dependence. Recognition of this fact is essential in the formulation of boundary conditions which ensure well-posed computations as the inviscid limit is approached. It also provides the motivation for the design of interior discretisation schemes which inherently respect directions of information propagation.

The second crucial aspect of compressible flows is that they contain features with extremely strong gradients, such as shock waves and contact surfaces. In the inviscid limit, these features are discontinuous, while for viscous cases they are usually far too thin to resolve on a practical mesh. Consequently their computed representations are strongly affected by the discretisation errors of the scheme. The computation of flows which contain (near) discontinuities has has provided a major challenge for the design of CFD techniques.

In the following, a brief overview of the various methods which have been used to discretise the compressible flow equations is given. More details can be found in the cited references and in [30, 18, 19, 56].

Application of Boundary conditions

At boundaries, the need to respect inherent directions of information propagation is essential. This is most clear when considering artificial boundaries, i.e. those which are introduced to limit the size of the considered domain. If a process within the domain generates acoustic, entropic or vortical waves, these should be allowed to exit the boundaries with minimum numerical reflection.

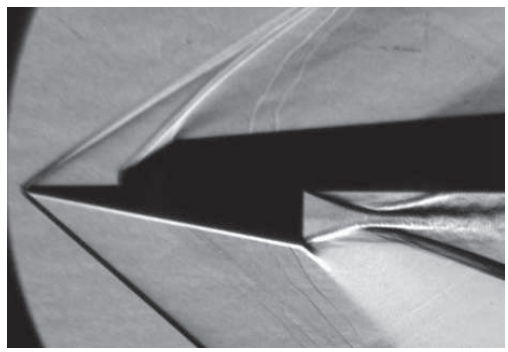


Figure 1.1: Schlieren image showing shocks and expansions in compressible flow (T.U. Delft , Aerodynamics group)

The development of boundary conditions which permit this has been the subject of intense study, particularly in the context of acoustics using the linearised equations, but also for the Euler and Navier Stokes equations themselves. For cases where the information travelling across the boundary is in the form of (nearly) planar waves, robust procedures have been developed based on locally one-dimensional inviscid (LODI) approximations [41, 37]. Where more complicated structures are involved, treatment becomes difficult due to the lack of information about how such structures influence the flow after leaving the domain. For such cases a range of techniques has been developed, including absorbing layers and methods including simplified models of the flow outside the domain [10, 42].

Upwind interior schemes

From the point of view of numerical consistency, it is not strictly necessary that interior discretisations for compressible flow explicitly consider inherent directions of information propagation. In the pursuit of robust and accurate numerical methods, however, such a requirement has often been used. Methods which locally respect the directions of information propagation are known collectively as upwind methods. Their design typically involves a trade off between the expense and fidelity of the local solution representation.

An early approach to upwinding was flux-vector splitting, for which flux terms in the governing equations are divided into components assumed to be travelling with or against a specified direction. Preliminary versions of flux-vector splitting suffered from problems near sonic points, but these were later overcome. The splitting concept received renewed attention during the 1990's in the definition of the Convective Upwind Split Pressure (CUSP) [31] and Advection Upstream Splitting (AUSM) [36] schemes.

An alternate method introduced by Godunov [12] during the 1950's later also became a popular upwinding approach. In Godunov's method, the flux

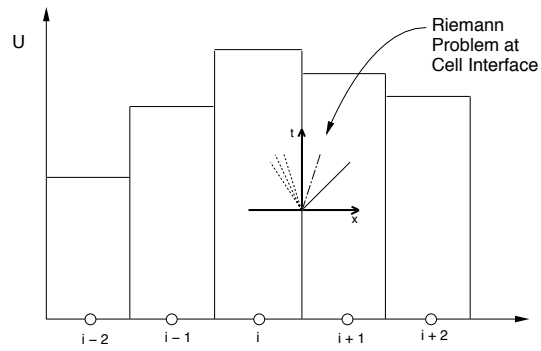


Figure 1.2: Godunov method - Assuming a piecewise constant solution leads to local Riemann problems at the cell interfaces, which can be solved for the local flux values

at the interface of a computational cell is found by assuming the state in each cell to be locally constant, and then solving the problem of two discontinuous states at the cell interfaces (figure 1.2). This problem is known as the Riemann problem, and has an exact solution for the one-dimensional Euler equations. Godunov's approach has since been used by a range of discretisation techniques. Discretisations which evaluate their fluxes based on the exact 1D Euler solution are said to employ "exact Riemann solvers". Repeated computation of the local non-linear Euler problem is relatively expensive, however, so a number of alternatives have also been developed. The most famous of these are the approximate Riemann solvers of Roe [43] and Osher [40].

The Godunov approach to flux evaluation is naturally combined with finite-volume methods, as these are based on state averages which change discontinuously from cell to cell. It is also naturally combined with discontinuous-Galerkin finite-element methods, which represent their solutions using a family of functions which are discontinuous between elements. Discontinuous Galerkin methods are particularly well suited to *hp* adaptation, as they allow local changes in mesh (h) or order of interpolation (p) while retaining a compact stencil (see [2, 17, 34]).

Flux-vector splitting and the Godunov method rely on one-dimensional concepts, in the sense that the upwinding is applied in a single chosen direction (usually the direction normal to the interface between computational cells). These methods tend to loose accuracy when convection is oblique to the mesh. Multi-dimensional schemes, which apply upwinding in arbitrary directions, have therfore also been the subject of research. The Streamwise-Upwind Petrov-Galerkin (SUPG) finite-element method pioneered by Brooks and Hughes [9] is an early approach which remains popular. Later multidimensional methods include the residual-distribution schemes pursued by Deconnick, Roe, Abgrall and

others, which make use of both finite-volume and finite-element concepts [1].

Discontinuities and discrete conservation

Two approaches have been used to represent flow discontinuities computationally. In the first of these, referred to as "shock-fitting", the discontinuity is treated as an internal boundary across which the Rankine-Hugoniot relations are applied. This eliminates the problem of treating high gradients numerically, but becomes inconvenient if the position of the discontinuity is unknown or changing in time. In the second approach, known as "shock capturing", the discontinuity is computed within the domain as a sharp layer with a thickness on the order of the grid size. This concept can be traced back to John von Neumann, who during World War II introduced the concept of artificial viscosity to give computed discontinuities a finite thickness. As discontinuities are weak solutions of the governing equations, they are in fact only admitted by the integral form of the equations, which directly express the conservation of mass, momentum and energy. It turned out that in order to achieve reasonable accuracy in the representation of discontinuities, discretisations which exactly reproduced the conservation of mass, momentum and energy were required. As a result, most of the early successful shock-capturing methods were based on the finite-volume approach, which is inherently discretely conservative. Due to the close relationship between finite-volume and finite-difference methods, it consequently became possible to design discretely-conservative finite-difference schemes. Discrete conservation is also automatically ensured by both continuous and discontinuous Galerkin finite-element methods which use conservative variables in their variational forms [48].

Controlling oscillations near discontinuities

The initial application of conservative discretisations for shock capturing produced solutions containing substantial oscillations near the discontinuous parts of the solution (figure 1.3). This was true for both central schemes, which violate the zones of dependence of the original PDE (e.g. by looking downwind), *and* higher-order upwind methods. This observation was formalised by Godunov [12], who after introducing the concept of monotonicity as a property which rules out oscillations, proved that *linear* schemes for advection which preserve monotonicity can be at most first-order accurate. Eventually it was realised that Godunov's result did not preclude the success of non-linear schemes, which began to be developed in the early 1970's. In 1971 Boris introduced the concept of non-linear limiting to prevent oscillations, ultimately leading to the Flux-corrected transport (FCT) method [5, 7, 6, 57]. A similar concept was introduced by van Leer of the University of Leiden, eventually leading to a method known as the Monotone Upwind Scheme for Conservation Laws (MUSCL) [49, 50, 51, 52, 54]. Both of these were early examples of high-resolution schemes, a term now associated with methods which provide higher than first-order accuracy while remaining oscillation-free. It is also generally accepted that such schemes must satisfy an "entropy condition". For the Euler equations, this implies that the

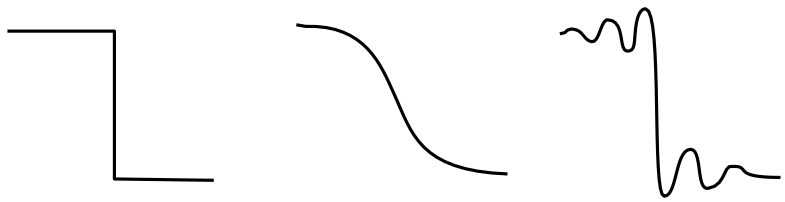


Figure 1.3: Discontinuous solution (left) with dissipative (middle) and oscillatory (right) approximations

second law must be satisfied, so that expansion shocks cannot be produced.

The second generation of non-oscillatory schemes

In 1970 Glimm and Lax [11] showed that the total variation of the solution a scalar 1D conservation law can only decrease. Harten [14] showed that numerical schemes designed using a discrete variant of this condition would be monotonicity preserving. This means that when starting from an initial non-oscillatory solution, these schemes do not introduce new oscillations. Methods satisfying the discrete condition are referred to as Total-Variation Diminishing (TVD) methods. The TVD conditions are a convenient mathematical tool for designing non-oscillatory schemes, and have therefore been widely applied.

The TVD conditions are based on one-dimensional concepts, and their application did not lead to schemes which are TVD when when direction of convection is oblique to the mesh lines. Eventually it was shown by Goodman and LeVeque [13] that a truly multidimensional TVD scheme can be no more than first-order accurate. This led Harten to the definition of Essentially Non-Oscillatory (ENO) schemes [16, 15]. The latter are instead total Variation Bounded (TVB) and can be extended to multiple dimensions. Some of the most sophisticated shock-capturing schemes available employ weighted ENO (WENO) conditions on unstructured meshes. These methods produce exceptionally good results (figure 1.4), but have been criticised for their programming complexity and large stencils which inhibit parallelisation.

Direction-independent schemes follow up

Upwind methods can require significant computational effort, due to the necessity of examining local directions of propagation. They can also be complex to program. As a result, there remains much interest in direction-independent discretisations.

A common approach to direction-independent discretisation is to combine traditional central methods with additional artificial dissipation. Such methods can be competitive since their low cost allows for greater refinement in regions where upwind methods are potentially more accurate. In 1981 Jameson, Schmidt and Turkel introduced a classic finite-volume central discretisation for the Euler equations [32]. This has become known as the JST scheme, and

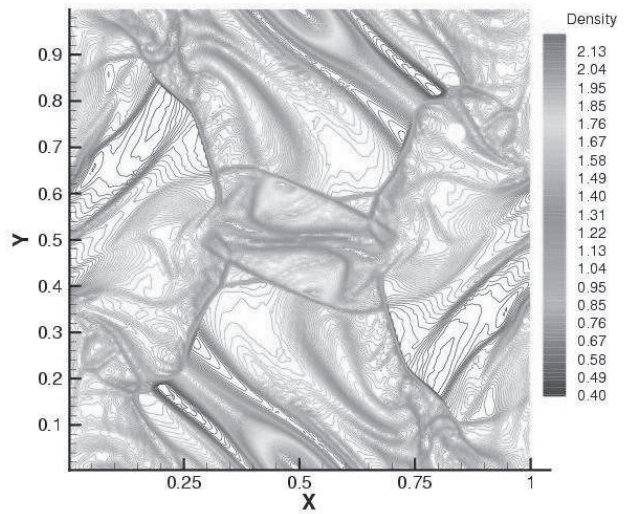


Figure 1.4: WENO computation of Orszag - Tang magnetohydrodynamic vortex (Chatterjee and Ghosh, ITT Bombay) showing multiple fine structures captured without oscillations

has been widely used. The JST scheme also incorporates a non-linear switch, but it was designed without discrete expressions associated with oscillation control. Later Jameson introduced local extrema diminishing (LED) conditions, which imply the TVD property, and was able to derive conditions under which the JST scheme is LED. He also derived a family of schemes known as symmetrically limited positive (SLIP) which use limiters similar to those used in upwind high-resolution schemes [31].

A completely separate area of research concerns least-squares finite-element methods [33]. These methods generally do not require additional artificial dissipation, and have the advantage of producing symmetric positive-definite matrices when applied to first-order systems. The latter can be efficiently stored and solved. h refinement has been employed by some researchers to improve shock representations [47], but it appears that p refinement is also very effective near discontinuities [3].

Methods of the future?

Although finite-difference methods were important in the development of methods for compressible flow, they are presently less popular due to the difficulty of applying them to unstructured meshes. In contrast, Finite-volume-based methods will be likely used for some time, primarily due to their relative speed and

generality.

Finite-element methods, on the other hand, are seeing increasing use. Their advantages include robustness and accuracy, albeit at the expense of higher computational cost. This trade-off is appealing in environments where the labour required for setting up computations is the main concern, rather than computer hardware limitations. Finite-element methods are also particularly well-suited for adaptive computations, as changes in the order of approximation remain local and are more efficiently handled in parallel computing environments.

1.2 An outline of these notes

As described above, there are many alternatives for discretising the equations of compressible flow. In the remainder of these notes, we will focus only on a few examples which illustrate the main concepts influencing method design. These include:

- The treatment of boundary conditions using a LODI approximation
- Upwinding via flux-vector splitting, the Godunov approach and SUPG
- The design of non-oscillatory methods using TVD conditions

We will restrict our consideration the Euler equations, as these are simpler to interpret. Normally schemes for the Euler equations are applied to the Navier-Stokes equations for high-Reynolds-number flows with only minor modification. Since it is advantages to make use specific properties of the Euler equations, the reader should be familiar with different ways of expressing them. Those without this background should consult the additional notes in the appendix. Furthermore, we will normally describe discretisations based on either finite-volume or finite-element methods. Basic descriptions of these methods are also provided in the appendix.

Chapter 2

Application of boundary conditions

We begin our examination of discretisation techniques for compressible flows by considering the application of boundary conditions for the Euler equations. In order to respect local directions of information propagation, the compatibility relations are used, along with upwinding concepts which will re-appear during the discussion of interior schemes in the following chapters. Those unfamiliar with the compatibility relations should review appendix A before continuing.

Boundaries may be divided into two basic types. The first type, which can be referred to as true boundaries, are those which have direct counterparts in the physical situation, such as the walls of a channel, or the surface of an airfoil (figure 2.1). The second type, known as artificial boundaries, are those which are introduced to limit the size of the numerical domain relative to the physical one. These include the inflow and outflow boundaries of a channel or the outer boundary of an airfoil problem.

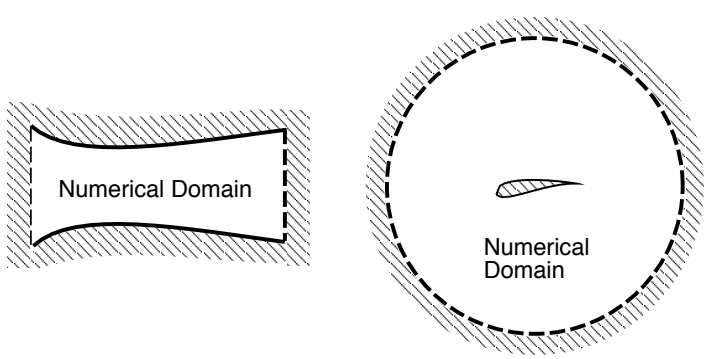


Figure 2.1: Numerical domains for channel flow (left) and an airfoil problem (right) with true boundaries (solid lines) and artificial boundaries (dashed lines)

A basic requirement for all boundary conditions is that they lead to a well-posed problem. For the Euler equations, this requires that the characteristic directions of information propagation to be maintained across the boundary surface. As a result, the number of physical quantities which may be specified at a boundary can be less than required to fix the local state. The remaining information must be determined from the evolving numerical solution, in a manner which preserves the character of waves which are transported towards the boundary from within the numerical domain. In general, if there are N_v variables are required to fix the state of a point on the boundary, ($N_v = 3$ for one-dimensional problems, 5 for three-dimensional problems), then the number of physical boundary conditions, N_p, which should be imposed is:

$$N_p = N_v - N_n \tag{2.1}$$

where N_n is the number of numerical conditions, equal to the number of characteristics carrying information from within the numerical domain (figure 2.2).

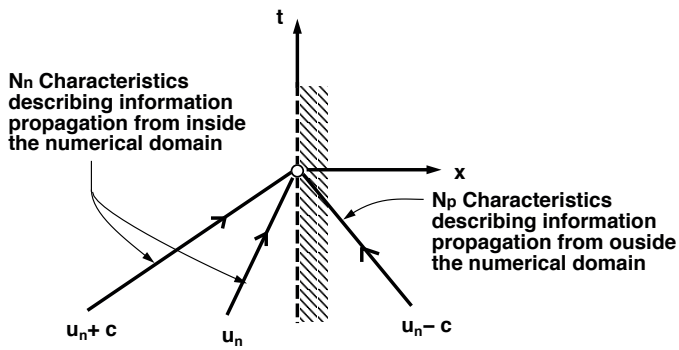


Figure 2.2: Characteristic lines for a subsonic outflow condition

The number and type of characteristics carrying information towards the boundary can be determined by considering the compatibility-relation form of the Euler equations, written in the direction of the boundary normal. As discussed in section A.1.5, the first three of these relations describe the convection of entropy and vorticity waves with a speed equal to the fluid velocity. The last two compatibility relations describe the propagation of acoustic waves with the sound speed added or subtracted from the local fluid velocity. The relative direction of propagation each type of wave is thus determined by the local normal velocity, u_n and local speed of sound, c.

The characteristics in figure 2.2, for example, appear with slopes u_n, $u_n + c$ and $u_n - c$ for conditions where the normal velocity is directed out of the numerical domain, with a magnitude less than the local speed of sound. In this case the strengths of the entropy and vorticity waves, along with that of the right-travelling acoustic wave, are determined within the numerical domain ($N_n = 4$ in three dimensions). A single physical condition must therefore be specified,

which directly or indirectly determines the strength of the left-travelling acoustic wave.

The remainder of this chapter considers the application of boundary conditions in more detail. First, factors influencing the application of of artificial boundary conditions are discussed, then an approach based on a locally one-dimensional inviscid (LODI) approximation is presented. The latter is explicitly expressed in terms of the propagation of the individual wave components of the Euler equations. This leads to effective boundary conditions, but also clarifies concepts employed in the description of upwind schemes later in the notes. After the discussion of artificial boundary conditions, two types of of true boundary conditions, the solid wall and Kutta conditions, are discussed.

It should be noted that the techniques described here for applying boundary conditions are by no means unique. There are in fact a large variety of boundary-treatment techniques in use, each of which respects the mathematical properties of the Euler equations to a different degree. The reader is referred to reference [19] for further examples.

2.1 Artificial Boundary Conditions

Artificial boundaries are typically introduced in order to truncate an unbounded physical domain, allowing computer resources to be focused on a particular region of interest. This implies that the conditions applied at the the artificial boundary should replace all the influences of the discarded outer region. Such influences may arise from both the outer region's internal behaviour, and from its interactions with the inner region. Two basic questions result:

- How do we model the behaviour of the outer region ?
- How should numerical and physical conditions be applied to preserve the character of the interactions between the inner and outer regions ?

These issues are discussed in the following sections.

2.1.1 Modelling of the Outer Region

The specification of physical conditions at an artificial boundary requires some model of the discarded outer region. The simplest approach is to choose the location of the artificial boundary to lie in an area where the incoming physical signals are known, and remain unaffected by the evolving numerical solution. For unsteady simulations, this requires that disturbances generated in the region of interest remain within the numerical domain for the duration of the simulation, or the assumption that the propagation of such disturbances into the outer region does not modify the outer region's behaviour. Although straight-forward, this approach can be inefficient in terms of the number of nodes or cells required to separate the region where signals are generated and the outer region.

A more sophisticated approach is to employ a simplified theoretical model which allows the outer region to respond to the numerical solution. In the

airfoil problem, for example, the inflow velocities at the outer boundary can be adjusted in time as using a point-vortex representation of the airfoil, with a value of circulation determined from the lift of the airfoil from the evolving numerical solution. For the the subsonic nozzle problem shown in figure 2.3, one could impose incoming acoustic waves which communicate information about the pressure rise in the exit tank.

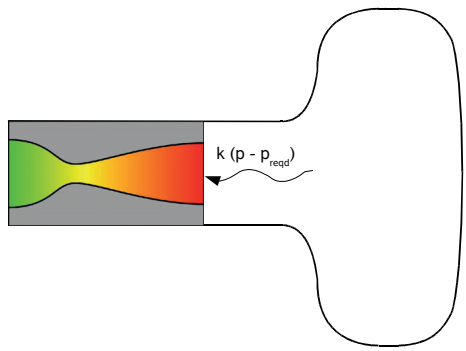


Figure 2.3: Model of incoming acoustic waves for a subsonic nozzle problem (numerical domain shown in grey).

It is also possible to represent the outer region using an alternative numerical model, for which the boundary conditions are less complex, or with which the second numerical domain can be enlarged without significant cost. Examples include computations where the equivalent solution from a surface-singularity method for the linearised potential equation is used to correct the artificial boundary state, or where the Euler artificial boundary serves as an inner boundary of a finite-difference method for the full potential equation.

2.1.2 Application of Physical and Numerical Conditions

An important requirement for an artificial boundary procedure is that it does not produce spurious numerical reflections from waves originating within the numerical domain. The accuracy to which this requirement is met can have a significant effect on the quality of the numerical results. Spurious signals returned from artificial boundaries can, in turn, reflect from physical boundaries or other artificial boundaries, leading to standing numerical waves. These can significantly degrade accuracy, and prevent convergence.

If the waves traversing the boundary are nearly planar, one approach to developing boundary conditions with minimal spurious numerical reflections is to use locally one-dimensional inviscid (LODI) approximations, in which a single direction of propagation (normally aligned with a grid line or normal to a grid face) is considered. The approach described here is based on local discretisations of the compatibility relations presented in section A.1.5. Note that, in general,

the resulting conditions still allow for *physical* reflections. These may originate from the interaction of outward-propagating waves with the model of the outer region, and must remain part of the incoming signal.

The compatibility relations in three dimensions may be written as:

$$\left(\frac{\partial \rho}{\partial t} - \frac{1}{c^2}\frac{\partial p}{\partial t}\right) = G_1 \tag{2.2}$$

$$\left(-k_z\frac{\partial u}{\partial t} + k_x\frac{\partial w}{\partial t}\right) = G_2 \tag{2.3}$$

$$\left(k_y\frac{\partial u}{\partial t} - k_x\frac{\partial v}{\partial t}\right) = G_3 \tag{2.4}$$

$$\left(\frac{\partial u_k}{\partial t} + \frac{1}{\rho c}\frac{\partial p}{\partial t}\right) = G_4 \tag{2.5}$$

$$\left(\frac{\partial u_k}{\partial t} - \frac{1}{\rho c}\frac{\partial p}{\partial t}\right) = G_5 \tag{2.6}$$

where $u_k = \vec{u}\cdot\vec{k}$. $G_{1\rightarrow 5}$ represent the time rate of change of the entropy, vorticity and acoustic wave strengths. These can be expressed in terms of the spatial gradients of the primitive variables by comparison with expressions (A.28)-(A.32) in section A.1.5. For many cases, it is sufficient to consider only gradients normal to the boundary. If we choose $\vec{k}$ to be parallel to the normal vector, the expressions for $G_{1\rightarrow 5}$ simplify to:

$$G_1 = -u_k\left(\frac{\partial \rho}{\partial k} - \frac{1}{c^2}\frac{\partial p}{\partial k}\right) \tag{2.7}$$

$$G_2 = -u_k\left(-k_z\frac{\partial u}{\partial k} + k_x\frac{\partial w}{\partial k}\right) \tag{2.8}$$

$$G_3 = -u_k\left(k_y\frac{\partial u}{\partial k} - k_x\frac{\partial v}{\partial k}\right) \tag{2.9}$$

$$G_4 = -(u_k + c)\left(\frac{\partial u_k}{\partial k} + \frac{1}{\rho c}\frac{\partial p}{\partial k}\right) \tag{2.10}$$

$$G_5 = -(u_k - c)\left(\frac{\partial u_k}{\partial k} - \frac{1}{\rho c}\frac{\partial p}{\partial k}\right) \tag{2.11}$$

which leads to compatibility relations similar in form to the one-dimensional situation, with the addition of two expressions for the transport of vorticity waves. By analogy with the linear convection equation, the expressions show entropy and vorticity transported normal to the boundary with speed u_k, and the acoustic variables transported normal to the boundary with speeds $u_k + c$ and $u_k - c$. These expressions, or their counterparts incorporating tangential gradients, can be used to apply numerical conditions for general inflow and outflow conditions, as described below.

Supersonic Outflow

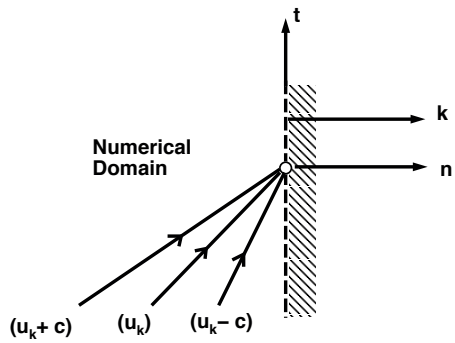


Figure 2.4: Characteristic lines for supersonic outflow conditions

At a supersonic outflow boundary (figure 2.4), all waves originate from within the domain. Therefore we may not specify any physical conditions, and we retain all five compatibility relations in order to track wave propagation within the numerical domain ($N_n = 5$). The compatibility relations (equations (2.2)-(2.6) + (2.7)-(2.11)) may then be rearranged to obtain expressions for the time rate of change of each of the primitive variables:

$$\frac{\partial \rho}{\partial t} = G_1 + \frac{\rho}{2c}(G_4 - G_5) \tag{2.12}$$

$$\frac{\partial u}{\partial t} = \frac{n_x}{2}(G_4 + G_5) + n_y G_3 - n_z G_2 \tag{2.13}$$

$$\frac{\partial v}{\partial t} = \frac{n_y}{2}(G_4 + G_5) - \frac{(1 - n_y^2)}{n_x} G_3 - \frac{n_y n_z}{n_x} G_2 \tag{2.14}$$

$$\frac{\partial w}{\partial t} = \frac{n_z}{2}(G_4 + G_5) + \frac{(1 - n_z^2)}{n_x} G_2 + \frac{n_y n_z}{n_x} G_3 \tag{2.15}$$

$$\frac{\partial p}{\partial t} = \frac{\rho c}{2}(G_4 - G_5) \tag{2.16}$$

These expressions can be evaluated numerically using one-sided expressions for $G_{1\to5}$, and a suitable time integration technique for the primitive state variables.

Supersonic Inflow

At supersonic inflow boundaries (figure 2.5), all waves originate from outside the numerical domain, so that the complete state may be specified with physical conditions. Since all the information is given, one can express the evolving boundary state in any set of variables. If primitive variables are used, for ex-

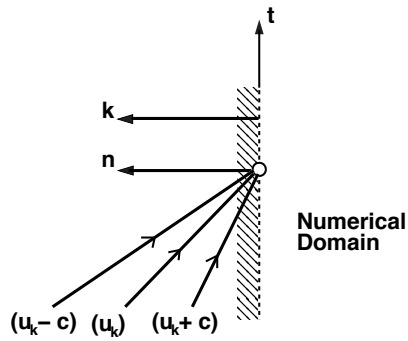


Figure 2.5: Characteristic lines for supersonic inflow conditions

ample, one could simply write:

$$\frac{\partial \rho}{\partial t} = \left(\frac{\partial \rho}{\partial t}\right)_o, \qquad \frac{\partial u}{\partial t} = \left(\frac{\partial u}{\partial t}\right)_o, \qquad ... \tag{2.17}$$

where the subscript o denotes the specified conditions in the outer region.

Subsonic Outflow

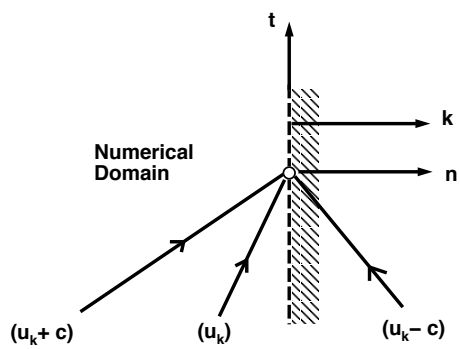


Figure 2.6: Characteristic lines for subsonic outflow conditions

Referring to figure 2.6, for subsonic outflow conditions, we may apply four numerical conditions by using the first four of the compatibility relations (equations (2.2)-(2.5) + (2.7)-(2.10)) to describe the propagation of waves with speed u_k (entropy and vorticity) and $u_k + c$ (right-running acoustic wave) from within the numerical domain. This leaves one degree of freedom associated with $u_k - c$ which must be fixed by specifying a physical condition. This is done using the fifth compatibility relation, and replacing G_5 with the known time-rate

of change of the second acoustic variable. For example, if the strength of the incoming (left-running) acoustic wave is known to be $f_w(t)$, we could write:

$$\left(\frac{\partial u_k}{\partial t} - \frac{1}{\rho c}\frac{\partial p}{\partial t}\right) = f_w(t) \tag{2.18}$$

Combining the above with the first four of the compatibility relations leads to expressions for the time rate of change of the primitive variables similar to those for the supersonic outflow case, with G_5 replaced by $f_w(t)$. An incoming harmonic acoustic wave, for example, could be specified with $f_w(t) = Asin(\omega t)$. If the outer domain is unbounded, and does not respond to outgoing signals, we may also specify $f_w(t) = 0$. This corresponds to the definition of a *physical* non-reflecting condition, in which no numerical or physical waves are permitted to enter from outside the numerical domain.

For many problems, the strength of the incoming wave is not known directly, but $u_o(t)$ or $p_o(t)$ are known at the outflow. If the problem is steady or relatively low-frequency, we may specify the strength of the incoming wave as proportional to the difference between the computed and specified quantity (figure 2.3):

$$f_w(t) = K(p - p_o) \tag{2.19}$$

At higher frequencies, however, this approach implies a non-stationary behaviour within the outer region which is not likely to be physical. For problems where a more sophisticated model of the exterior is available, we may use it to directly specify the physical condition in terms of $u_o(t)$ or $p_o(t)$, and use the compatibility relations valid in the numerical domain to determine the time rate of change of the primitive state variables. For example, if $\frac{\partial p_o}{\partial t} = f_p(t)$:

$$\frac{\partial \rho}{\partial t} = G_1 + \frac{f_p(t)}{c^2} \tag{2.20}$$

$$\frac{\partial u}{\partial t} = n_x(G_4 - \frac{f_p(t)}{c^2}) + n_y G_3 - n_z G_2 \tag{2.21}$$

$$\frac{\partial v}{\partial t} = n_y(G_4 - \frac{f_p(t)}{c^2}) - \frac{(1-n_y^2)}{n_x}G_3 - \frac{n_y n_z}{n_x}G_2 \tag{2.22}$$

$$\frac{\partial w}{\partial t} = n_z(G_4 - \frac{f_p(t)}{c^2}) + \frac{(1-n_z^2)}{n_x}G_2 + \frac{n_y n_z}{n_x}G_3 \tag{2.23}$$

$$\frac{\partial p}{\partial t} = f_p(t) \tag{2.24}$$

where the above equations are the first four of the compatibility relations, and the last specifying the imposed pressure condition.

Subsonic Inflow

Referring to figure 2.7, in this case u_k is negative. Thus four physical conditions must be specified:

$$\left(\frac{\partial \rho}{\partial t} - \frac{1}{c^2}\frac{\partial p}{\partial t}\right) = f_1(t) \tag{2.25}$$

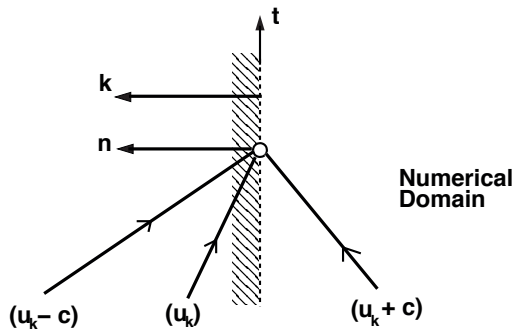


Figure 2.7: Characteristic lines for subsonic inflow conditions

$$\left(-k_z\frac{\partial u}{\partial t}+k_x\frac{\partial w}{\partial t}\right) = f_2(t) \tag{2.26}$$

$$\left(k_y\frac{\partial u}{\partial t}-k_x\frac{\partial v}{\partial t}\right) = f_3(t) \tag{2.27}$$

$$\left(\frac{\partial u_k}{\partial t}-\frac{1}{\rho c}\frac{\partial p}{\partial t}\right) = f_5(t) \tag{2.28}$$

and only a single numerical condition is required:

$$\left(\frac{\partial u_k}{\partial t}+\frac{1}{\rho c}\frac{\partial p}{\partial t}\right) = G_4 \tag{2.29}$$

For example, an incoming isentropic, irrotational acoustic wave can be specified with $f_1(t) = f_2(t) = f_3(t) = 0$ and $f_5(t) = Asin(\omega t)$. A non-reflecting physical condition is obtained with $f_{1,2,3,5}(t) = 0$. The resulting expressions for the primitive variable time derivatives are similar to those for the supersonic outflow case, with $f_{1,2,3,5}$ substituted for $G_{1,2,3,5}$. It is important to note that for the subsonic inflow case, not all combinations of flow variables can be specified. This becomes clear when examining the left hand sides of (2.28) and (2.29). Direct specification of $u(t)$ and $p(t)$ fixes the magnitude of both the incoming and outgoing acoustic waves, and thus violates the characteristic directions of information propagation.

The procedure presented above works well for planar waves, and is also rather effective for vortices and entropic spots provided that a sufficient level of refinement is used. It can be augmented by special edge treatments or the inclusion of tangential gradients, although the effect of the latter tends to be slight. Figure 2.8 compares the upstream pressure perturbation from a vortex exiting the domain obtained with the procedure to one obtained using a quasi-steady Riemann invariant approximation often used for outer boundaries. As the compatibility relations result in a system of ordinary differential equations in non-conservative form, however, the above approach does require distinct treatment when combined with a finite-volume or finite-element discretisation.

In the next chapter, we will examine upwind interior schemes, some of which can also be reformulated as boundary conditions. This lowers the combined complexity of implementation.

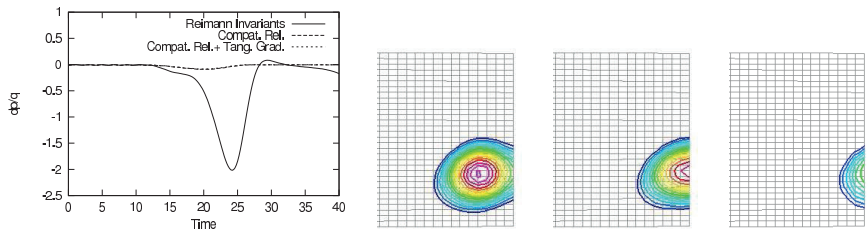


Figure 2.8: Reflected pressure waves for a vortex leaving the domain (left) and contours of vorticity magnitude for the compatibility relation B.C. (right)

2.2 True Boundary Conditions

2.2.1 Solid walls

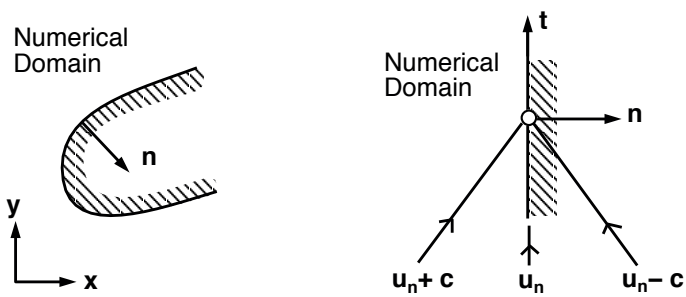


Figure 2.9: Solid wall in the $x-y$ and $n-t$ planes

Solid-wall boundaries are defined by a limiting case of the the $x-t$ diagram previously discussed, for which the velocity normal to the boundary is zero. This implies that only one left-running characteristic carries information from the boundary back into the numerical domain. Thus, the only physical condition which may be specified is $\vec{u}\cdot\vec{n}=0$, leaving the remaining quantities required to fix the state to be determined by the numerical solution procedure.

One way of determining the remaining quantities is to use a one-sided discretisation of the compatibility relations which describe the behaviour of waves propagating towards the boundary from within the numerical domain. Some solid-wall boundary procedures, however, make use of the fact that only the

wall surface pressure contributes to the time rate-of-change of the conservative variables, as shown by considering the flux vector of the integral form:

$$\vec{F} \cdot \vec{n} = \begin{bmatrix} \rho(\vec{u} \cdot \vec{n}) \\ \vec{m}(\vec{u} \cdot \vec{n}) + p\vec{n} \\ \rho e(\vec{u} \cdot \vec{n}) + p\vec{u} \cdot \vec{n} \end{bmatrix} = \begin{bmatrix} 0 \\ p\vec{n} \\ 0 \end{bmatrix} \tag{2.30}$$

and apply the boundary condition by applying an appropriate flux. The wall pressure is determined as part of the solution. For some discretisations (notably finite-difference and volume methods) improved values of wall pressure can be obtained using extrapolations based on the normal component of the momentum equation in the interior of the numerical domain. For structured meshes this is often done using a local curvilinear coordinate transform, for which the steady normal momentum equation in two dimensions can be expressed:

$$p_\eta = \frac{1}{x_\xi^2 + y_\xi^2} \left[\rho(y_\eta u - x_\eta v)(v x_{\xi\xi} + u y_{\xi\xi}) + (x_\xi x_\eta + y_\xi y_\eta) \right] \tag{2.31}$$

where it is assumed that the solid boundary lies on an an $\eta = const$ surface (figure 2.10).

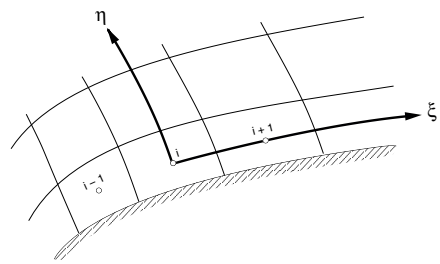


Figure 2.10: Local coordinate system for application of the normal momentum equation

2.2.2 Kutta Condition

Separation from sharp edges, as observed in physical flows, arises purely due to viscous phenomena. In the case of the airfoil shown in figure 2.11 (a), for example, the large acceleration and deceleration produced near the sharp trailing edge will result in a strong adverse pressure gradient, which a viscous boundary layer will be unable to negotiate. The result will be the generation of vorticity (b), until a flow is established where the singular pressure field has been eliminated (c).

In the absence of viscous forces, however, there exists no mechanism to force separation from a sharp edge. In fact, each of the conditions shown in figure 2.11 are possible solutions to the unsteady Euler equations, which admit

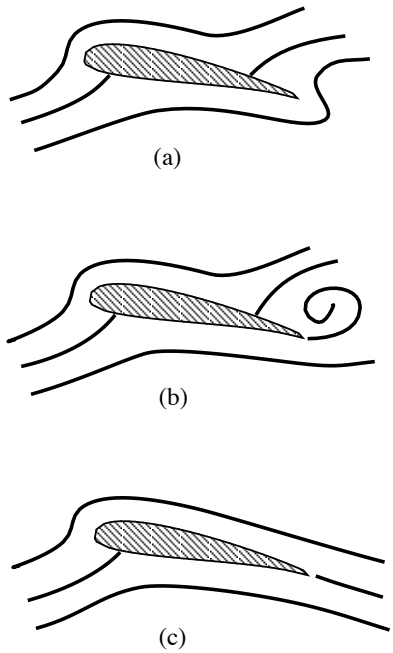


Figure 2.11: Valid solutions of the Euler equations

slip surfaces as weak solutions. In order to obtain a solution which corresponds to the situation observed in physical flows, an additional condition is required. This condition, known as the Kutta condition, is a familiar requirement for methods which use potential-flow approximations of the inviscid equations.

Formally, the Kutta condition is also required in order to chose a physically consistent solution of the Euler equations. Numerical solutions based on finite-difference and finite-volume techniques, however, are typically constructed to contain dissipative truncation errors. In the presence of the large cell-to-cell gradients induced near sharp edges, these will produce significant amounts of vorticity. Although the local time history of such numerical vorticity generation is not be physical, it will continue until the singular pressure field has been eliminated, and the physically consistent flow field has been established. As a result, explicit application of the Kutta condition is not usually required when dissipative numerical techniques are used.

Chapter 3

Upwind interior schemes

In this chapter we will examine four classical upwind discretisation techniques for the interior of the numerical domain. The first three of these are directly applicable to finite-volume and discontinuous finite-element methods, where one typically needs to evaluate the flux on the interface between two mesh cells based on the computed states within the two cells. The last upwind method is applicable to continuous finite-element methods, in which intra-element fluxes are not considered. Those unfamiliar with finite-volume or finite-element methods should review appendices B and C before continuing.

3.1 Flux vector splitting

The Euler and Navier-Stokes equations in their conservation form express the time rate of change of the conservative variables in terms of fluxes. One approach to upwinding is to produce discrete expressions for the fluxes which respect the physical directions of information propagation. Or more precisely, to express terms such as $\vec{\nabla} \cdot \vec{F}$ (finite-difference discretisations) or $\vec{F} \cdot \vec{n}$ (finite-volume or element discretisations), as a function of the local state U, in a manner which takes limited zones of dependence into account. This is most easily done using the characteristic form of the equations. For one dimensional problems, these can be obtained from the quasi-linear form

$$\frac{\partial U}{\partial t} + A\frac{\partial U}{\partial x} = 0 \tag{3.1}$$

by pre-multiplying with P^{-1}, the inverse of the matrix of eigenvectors of the conservative flux Jacobian A, which results in (see appendix A.1):

$$\frac{\partial U^*}{\partial t} + \Lambda\frac{\partial U^*}{\partial x} = 0 \tag{3.2}$$

where Λ is the diagonal matrix of eigenvalues containing the wave speeds u, $u+c$ and $u-c$. To facilitate upwinding, Λ is split into Λ^+ and Λ^- defined by:

$$\Lambda^{\pm} = \begin{bmatrix} \frac{1}{2}(u \pm \mid u \mid) & . & . \\ . & \frac{1}{2}(u+c \pm \mid u+c \mid) & . \\ . & . & \frac{1}{2}(u-c \pm \mid u-c \mid) \end{bmatrix} \tag{3.3}$$

so that the characteristic form becomes.

$$\frac{\partial U^*}{\partial t} + \Lambda^+ \frac{\partial U^*}{\partial x} + \Lambda^- \frac{\partial U^*}{\partial x} = 0 \tag{3.4}$$

Note that Λ^+ contains the positive wave speeds and Λ^- contains the negative wave speeds. and defining

$$A^+ = P\Lambda^+ P^{-1}, \qquad A^- = P\Lambda^- P^{-1} \tag{3.5}$$

the split characteristic form (3.4) can be pre-multiplied by P to obtain a split quasi-linear form:

$$\frac{\partial U}{\partial t} + A^+ \frac{\partial U}{\partial x} A^- \frac{\partial U}{\partial x} = 0$$

The Euler equations have the unusual property where the flux, f, can be written as $f = AU$ (this is not true of the Burgers equation, for example). We can then write:

$$\frac{\partial U}{\partial t} + \frac{\partial f^+}{\partial x} + \frac{\partial f^-}{\partial x} = 0 \tag{3.6}$$

with split fluxes defined as

$$f^+ = A^+ U, \qquad f^- = A^- U \tag{3.7}$$

To use flux vector splitting within a finite-volume method, we apply the divergence theorem to (3.6) to retrieve the integral form:

$$\frac{\partial}{\partial t} \int_V U \, dV + \int_S (\vec{f^+} + \vec{f^-}) \cdot d\vec{S} = 0 \tag{3.8}$$

For the 1D domain in space-time shown in figure 3.1, we can obtain a finite-volume method (appendix B) by applying the integral form to individual cells with centres i and volumes $V_i = \Delta x_i$. In this case, the discrete faces are defined by $\vec{S}_{i-1/2} = -1, \vec{S}_{i+1/2} = +1$. To implement upwinding, f^+ is then evaluated using the flow state to the left of each face and f^- using the flow state to the right. This allows the time rate of change of the state variables to be expressed:

$$\frac{\partial U_i}{\partial t} = \frac{1}{V_i}([f^-_{i+1} + f^+_i] - [f^-_i + f^+_{i-1}]) \tag{3.9}$$

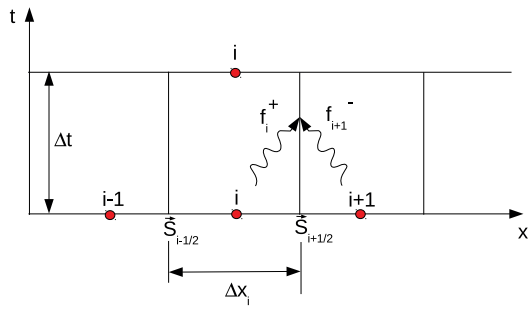


Figure 3.1: Flux at interface $i + \frac{1}{2}$ given by flux vector splitting in a one-dimensional finite-volume scheme

To complete the discretisation a time-marching method can be used. Applying an Euler explicit time march with time step Δt from $t = t^n$ to $t = t^{n+1}$ results in:

$$U_i^{n+1} = U_i^n - \frac{\Delta t}{\Delta x_i}([f_{i+1}^- + f_i^+] - [f_i^- + f_{i-1}^+])^n \tag{3.10}$$

where the subscripts indicate the indices of the cells bordering the flux interface, as shown in figure 3.1.

Flux vectors can be split in many other ways as well. The approach used for (3.3) is known as Steger-Warming splitting. It has been widely used, but suffers from discontinuous behaviour near sonic points. A more sophisticated approach derived by van Leer using Mach-number-dependent splitting avoids this problem (see [53] for more details). Further variants can be found in [19].

3.2 The Godunov method

In the Godunov method, the interface flux is determined by introducing a local exact solution to the Euler equations. This is done by first considering the states in each cell to be piecewise constant, and then solving the resulting set of Riemann problems which occur at the interfaces between cells (Figure 1.2). For the one-dimensional Euler equations, the solution of the Riemann problem is well known and can be represented on the the x-t plane as a family of waves centred at the cell interfaces (Figure 3.2). Godunov considered the states on either side of the interface to be defined by the volume average within each the cell:

$$\bar{U}_i^n = \frac{1}{\Delta x} \int_{i-\frac{1}{2}}^{i+\frac{1}{2}} U^n(x)dx \tag{3.11}$$

Assuming the cell geometry remains fixed in time, applying the integral form of the Euler equations (A.1) to cell i along with Euler explicit time integration

results in:

$$\bar{U}_i^{n+1} = \bar{U}_i^n - \frac{\Delta t}{\Delta x}\left[f^R_{i+\frac{1}{2}}(U_i, U_{i+1}) - f^R_{i-\frac{1}{2}}(U_i, U_{i-1})\right]^n \tag{3.12}$$

where $f^R_{i+\frac{1}{2}}$ and $f^R_{i-\frac{1}{2}}$ are the fluxes provided by the Riemann solutions at the right and left interfaces. Referring to figure 3.2, The flux on interface $i+\frac{1}{2}$ will remain constant during the time interval, provided that Δt is small enough to prevent waves from neighbouring interfaces from influencing the flux values at $i+\frac{1}{2}$. This leads to a CFL-like condition:

$$\Delta t < \frac{\Delta x}{a_{max}} \tag{3.13}$$

where a_{max} is the maximum wave speed computed in all of the interface Riemann solutions. An example where this is violated is shown in figure 3.3. Note that in supersonic conditions, the expansion, shock, and contact discontinuity are all one one side of the t axis, meaning that the interface flux will be determined entirely by the state in the upstream cell.

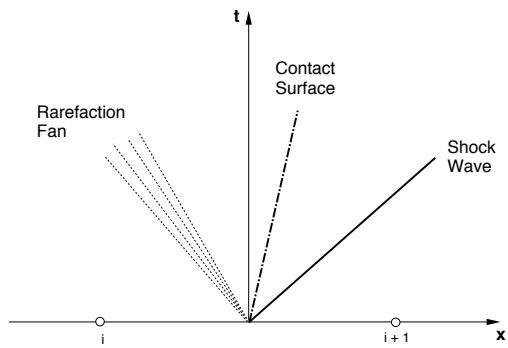


Figure 3.2: Subsonic solution to the Riemann problem for the 1D Euler equations, showing the expansion fan (left) contact discontinuity (right) and shock (far right) on the x-t plane.

Godunov's original method used the exact solution for the 1D Euler Riemann problem, which requires solving a local nonlinear problem in order to determine the flux. It is also only first-order accurate in space. One could argue that it is a little excessive to spend so much computational effort solving exact Riemann problems when the discretisation error is relatively large. However, the Godunov scheme contains some properties which are highly desired by numerical analysts. Firstly it is monotonic which means it does not produce non-physical oscillations. Secondly, it automatically satisfies the entropy condition, which means that it does not produce non-physical expansion shocks. We will reconsider these properties in chapter 4.

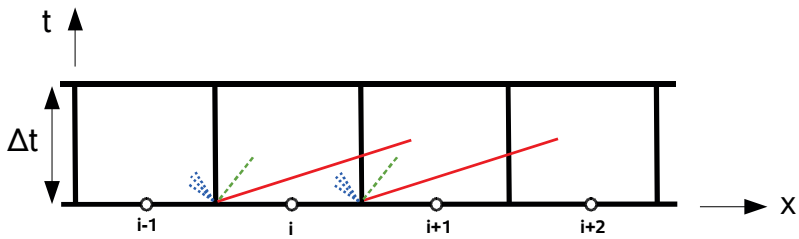


Figure 3.3: Wave interference for $\Delta t > \frac{\Delta x}{a_{max}}$

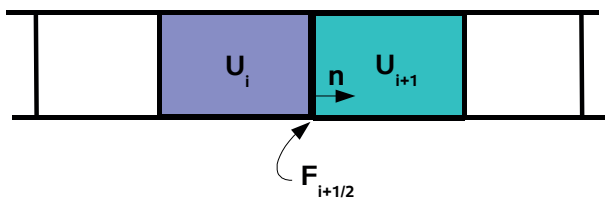


Figure 3.4: Nomenclature for the flux at the interface between two cells

3.3 Roe's approximate Riemann solver

Godunov's concept of evaluating local fluxes using a Riemann solver has become very popular. It can be made more efficient by using approximate Riemann solutions which avoid an iterative procedure. A popular example is the approximate Riemann solver of Roe. To formulate Roe's Riemann solver for multidimensional flows, we assume that the information between cells is exchanged by waves travelling normal to the interfaces (figure 3.4). The differential form of the Euler equations in the normal direction may be written:

$$\frac{\partial U}{\partial t} + \frac{\partial F_n}{\partial \hat{n}} = 0 \tag{3.14}$$

where $\hat{n}$ is a coordinate in the normal direction, $\mathbf{n}$ and:

$$F_n = \mathbf{F} \cdot \mathbf{n} = F_x \cdot n_x + F_y \cdot n_y + F_z \cdot n_z \tag{3.15}$$

Equation 3.14 may then be expressed in quasi-linear form as:

$$\frac{\partial U}{\partial t} + A\frac{\partial U}{\partial \hat{n}} = 0 \tag{3.16}$$

where A is the flux Jacobian matrix defined by:

$$A_{ij} = \frac{\partial (F_x)_i}{\partial U_j} n_x + \frac{\partial (F_y)_i}{\partial U_j} n_y + \frac{\partial (F_z)_i}{\partial U_j} n_z = \frac{\partial F_n}{\partial U} \tag{3.17}$$

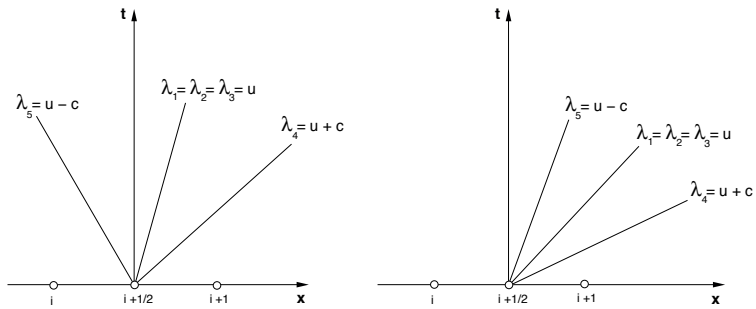


Figure 3.5: Positive and negative waves for Roe's approximate Riemann solver.

Roe's approximation is to replace A by a constant matrix $\tilde{A}$, evaluated at a state appropriate for the local conditions. This implies a local linearisation of the Euler equations, resulting in solutions to the Riemann problem such as those shown in figure 3.5. Note that the continuous transition through the expansion fan present in the exact solution (figure 3.2) is replaced here by a discontinuity. To determine a suitable $\tilde{A}$, Roe specified the following desirable properties:

- The eigenvectors of $\tilde{A}$ should be linearly independent
- As $U_{i+1} \to U_i$, $\tilde{A}(U_{i+1}, U_i) \to \dfrac{\partial F_n}{\partial U}(U_i)$
- $\tilde{A}(U_{i+1}, U_i) \cdot (U_{i+1} - U_i) = (F_n)_{i+1} - (F_n)_i$

The last of these conditions ensures that the Rankine-Hugoniot relations are satisfied if a discontinuity lies between the two states. An $\tilde{A}$ with the above properties can be obtained by evaluating the flux Jacobian A with "Roe-Averaged" variables (Reference [43]):

$$\tilde{\rho} = \sqrt{\rho_{i+1}\ \rho_i} \tag{3.18}$$

$$\tilde{u} = \frac{u_{i+1}\sqrt{\rho_{i+1}} + u_i\sqrt{\rho_i}}{\sqrt{\rho_{i+1}} + \sqrt{\rho_i}} \tag{3.19}$$

$$\tilde{H} = \frac{H_{i+1}\sqrt{\rho_{i+1}} + H_i\sqrt{\rho_i}}{\sqrt{\rho_{i+1}} + \sqrt{\rho_i}} \tag{3.20}$$

where

$$H = E + \frac{p}{\rho} \tag{3.21}$$

In order to evaluate the flux at the interface between two cells, we begin with:

$$\Delta F_n = \tilde{A}\Delta U \tag{3.22}$$

where the Δ indicates the change in values across the interface, e.g. $\Delta U = U_{i+1} - U_i$. Denoting the eigenvector matrix of $\tilde{A}$ by $\tilde{T}$, and its diagonal matrix of eigenvalues by $\tilde{\Lambda}$, we can write:

$$(F_n)_{i+1} - (F_n)_i = \tilde{T}(\tilde{T}^{-1}\tilde{A}\tilde{T})\tilde{T}^{-1}\Delta U = \tilde{T}\tilde{\Lambda}\Delta U^* \tag{3.23}$$

$$\text{where} \qquad \Delta U^* = \tilde{T}^{-1}\Delta U \tag{3.24}$$

The elements of $\tilde{\Lambda}$, denoted by λ_p, represent the wave speeds of the decoupled system. Their sign determines the direction of propagation of changes in the decoupled variables, U^*. If we split $\tilde{\Lambda}$ into two matrices which only contain the positive and negative diagonal entries, e.g for subsonic conditions:

$$\tilde{\Lambda}^+ = \begin{bmatrix} \lambda_1 & 0 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & 0 & 0 \\ 0 & 0 & \lambda_3 & 0 & 0 \\ 0 & 0 & 0 & \lambda_4 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}; \qquad \tilde{\Lambda}^- = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \lambda_5 \end{bmatrix} \tag{3.25}$$

we can write the interface flux in a number of different ways:

$$(F_{n\ Roe})_{i+1/2} = (F_n)_i + \tilde{T}\tilde{\Lambda}^-\Delta U^* \tag{3.26}$$

$$= (F_n)_{i+1} - \tilde{T}\tilde{\Lambda}^+\Delta U^* \tag{3.27}$$

$$= \frac{(F_n)_{i+1} + (F_n)_i}{2} - \frac{1}{2}\tilde{T}|\tilde{\Lambda}|\Delta U^* \tag{3.28}$$

The last of these is usually the most convenient for implementation purposes. Note that in (3.26) and (3.27), an upwinded interface flux is obtained as correction to either of the cell fluxes. This approach is thus sometimes referred to as flux-difference splitting.

A solution to the one-dimensional shock tube problem shown in figure 3.6 obtained using Godunov's approach with Roe's approximate Riemann solver appears in figure 3.7. From right to left, the recognisable features are a shock,

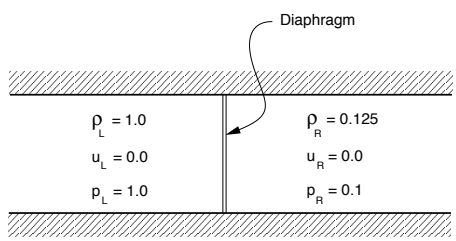


Figure 3.6: 1D shock tube problem

contact surface and expansion wave, corresponding to a certain time of the x-t diagram shown in figure 3.2. As for the basic Godunov scheme, the method

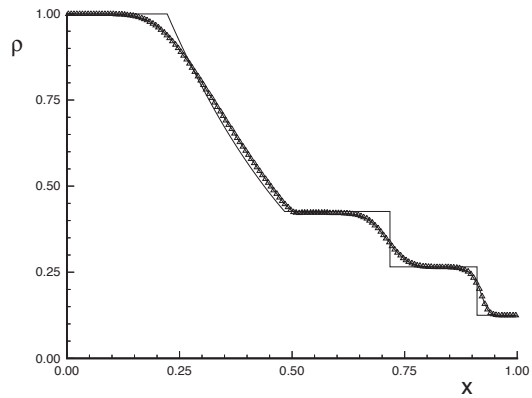


Figure 3.7: 1D Euler solution for the density profile in a shock tube computed with Godunov's approach and Roe's approximate Riemann solver (triangles) compared with the exact solution (lines).

is first-order accurate, resulting in a dissipated solution. For this case Roe's approximate Riemann solver also produces a monotonic solution. Due to the linearisation of the expansion, however, Roe's approximation no longer ensures that the entropy condition is satisfied. As a consequence expansion shocks are possible. To prevent this an additional entropy fix is sometimes used [19]. Note that the test case shown is slightly ironic, as a sequence of small Riemann problems have been used to solve one big Riemann problem. The small problems were only solved approximately, however.

3.4 Streamwise upwind Petrov Galerkin

The Galerkin finite-element method was originally applied to structural and heat-conduction problems, where it was very successful. Its application to convection-dominated problems, however, produced highly oscillatory solutions. This might be anticipated, since a Galerkin discretisation of the convection term in the linear convection equation produces a scheme similar to that derived from central finite-difference methods. The latter produce solutions which are dominated by dispersion errors when applied to convective problems.

The initial remedy to the problem followed the approach first used in finite differences, the addition of an artificial diffusion operators. Depending on the design of the artificial diffusion operator, an upwind scheme can also be produced. For example, consider the convection equation:

$$u_t + a u_x = 0 \tag{3.29}$$

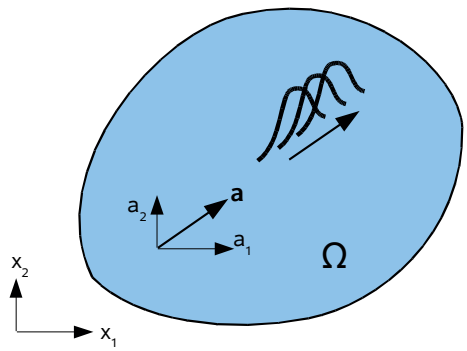


Figure 3.8: Domain Ω for a two-dimensional convection problem with wave speed $\mathbf{a} = [a_1, a_2]$.

A semi-discrete central discretisation with artificial dissipation could be constructed as:

$$\frac{\partial u_i}{\partial t} = -\frac{a}{2\Delta x}(u_{i+1} - u_{i-1}) + \frac{k}{\Delta x^2}(u_{i+1} - 2u_i + u_{i-1}) \tag{3.30}$$

with the artificial dissipation coefficient, k, chosen as $k = \frac{a\Delta x}{2}$, this becomes

$$\frac{\partial u_i}{\partial t} = -\frac{a}{\Delta x}(u_i - u_{i-1}) \tag{3.31}$$

which is the first-order upwind scheme. This type of equivalence can also be seem in the expressions for the flux computed with Roe's approximate Riemann solver (examine appendix 3.3 equation (3.28)).

In fact it is possible to do much better that the first-order upwind scheme, and derive a scaled artificial dissipation coefficient, $\tilde{k}$, which will produce nodally exact finite-element solutions for 1D linear steady convection-diffusion problems [20]. A further improvement on standard upwinding methods can be obtained by adding the scaled artificial dissipation anisotropically, as upwinding is only required in the direction of wave propagation. This can be accomplished by introducing a tensorial form for the artificial diffusion coefficient:

$$\tilde{\mathbf{k}} = \tilde{k}\frac{a_i a_j}{\|\mathbf{a}\|^2} \tag{3.32}$$

where a_i are the components of the wave speed vector $\mathbf{a}$ (figure 3.8). For example, if the wave speed vector is directed along the x_1 axis in a 2D flow:

$$\tilde{\mathbf{k}} = \tilde{k}\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \tag{3.33}$$

indicating that the artificial dissipation will only act in the x_1 equation in proportion to the x_1 convection term and not in the x_2 direction. This alignment

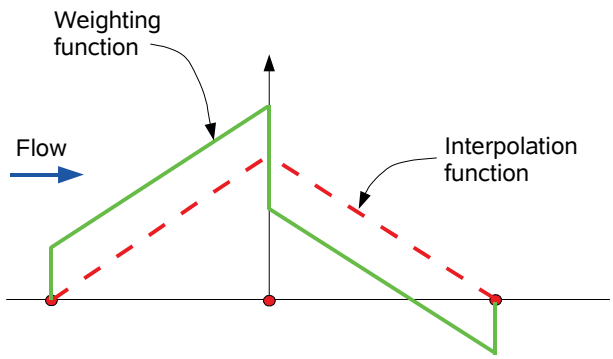


Figure 3.9: Interpolation function and perturbed weighting function for 1D elements with linear interpolation

occurs independently of local mesh directions. Correspondingly, the method is inherently multidimensional, unlike finite-volume and difference methods which consider upwinding in specific directions aligned with the mesh. In practice, this can result in a reduction in the diffusion associated with convection oblique to the mesh directions.

Finally, an equivalent alternative to adding an artificial diffusion term is to perturb the weighting function on the convection term in the variational form. This results in a so-called Petrov-Galerkin formulation, where the weighting functions are different from the interpolation functions. As an example, consider:

$$\begin{aligned} au_x - \nu u_{xx} &= 0 \quad for \quad x \in (0, L) \\ u(0) &= 0 \\ u(L) &= 0 \end{aligned} \tag{3.34}$$

A variational form of the problem with added scaled artificial dissipation is:

$$\int_0^L \left[wau_x + w_x(\nu + \tilde{k})u_x \right] dx = 0 \tag{3.35}$$

where we have used integration by parts on the (diffusion + artificial dissipation) term, and set $w = 0$ at $x = 0$ and $x = L$ for the application of Dirichlet boundary conditions. We can write equivalently:

$$\int_0^L \left[(w + \frac{\tilde{k}}{a} w_x) au_x + w_x \nu u_x \right] dx = 0 \tag{3.36}$$

A plot of the interpolation function and perturbed weighting function for the convection term is shown in figure 3.9.

For a bounded region Ω in multiple dimensions, the artificial diffusion term in (3.35) can be expressed:

$$\int_\Omega \nabla w \cdot \tilde{\mathbf{k}} \nabla u d\Omega = \int_\Omega \nabla w (\tilde{k} \frac{a_i a_j}{\|\mathbf{a}\|^2}) \nabla u d\Omega \tag{3.37}$$

$$= \int_\Omega \frac{\tilde{k}}{\|\mathbf{a}\|^2} (\mathbf{a} \cdot \nabla w)(\mathbf{a} \cdot \nabla u) d\Omega \tag{3.38}$$

so that the required perturbation to the weighting function on the convection term is:

$$\frac{\tilde{k}}{\|\mathbf{a}\|^2} (\mathbf{a} \cdot \nabla w) \tag{3.39}$$

In 1982, Brooks and Hughes incorporated these ideas into the design of the Streamwise-upwind Petrov-Galerkin (SUPG) method. An additional design criteria which separated this approach from previous methods was the requirement that it be residual-based in order to admit the exact solution. Thus the weighting function perturbation is applied to all terms. For general steady convection-diffusion-reaction problem with a source term s in a bounded region Ω:

$$\mathbf{a} \cdot \nabla u - \nabla \cdot (\nu \nabla u) + \sigma u = s \tag{3.40}$$

with the residual operator, $\mathcal{R}(u)$, and differential operator $\mathcal{L}(u)$ defined as:

$$\mathcal{R}(u) = \mathbf{a} \cdot \nabla u - \nabla \cdot (\nu \nabla u) + \sigma u - s = \mathcal{L}(u) - s \tag{3.41}$$

we can express the SUPG formulation as:

$$\int_\Omega \left[w + \frac{\tilde{k}}{\|\mathbf{a}\|^2} (\mathbf{a} \cdot \nabla) w \right] \mathcal{R}(u) d\Omega = 0 \tag{3.42}$$

or alternatively as:

$$\int_\Omega w \mathcal{L}(u) d\Omega + \sum_e \int_{\Omega_e} \left[(\mathbf{a} \cdot \nabla) w \right] \tau \mathcal{R}(u) d\Omega_e = \int_\Omega w s d\Omega \tag{3.43}$$

where $\tau = \tilde{k} / \|\mathbf{a}\|^2$ and the final term is specified to act only on element interiors, thus avoiding the discontinuity in the weighting function at the element boundaries. The final summation term is normally referred to as the stabilisation term.

Figure 3.10 shows results for the shock tube problem considered in the previous section computed using a SUPG discretisation for the Euler equations with a linear basis (second-order accurate) and cubic basis (third-order accurate). Generally the accuracy is good, and improves as the order is increased. However, the less expensive second-order accurate method produces significant oscillations near the shock. This type of behaviour is also typical for finite-difference and finite-volume discretisations of second or higher order accuracy.

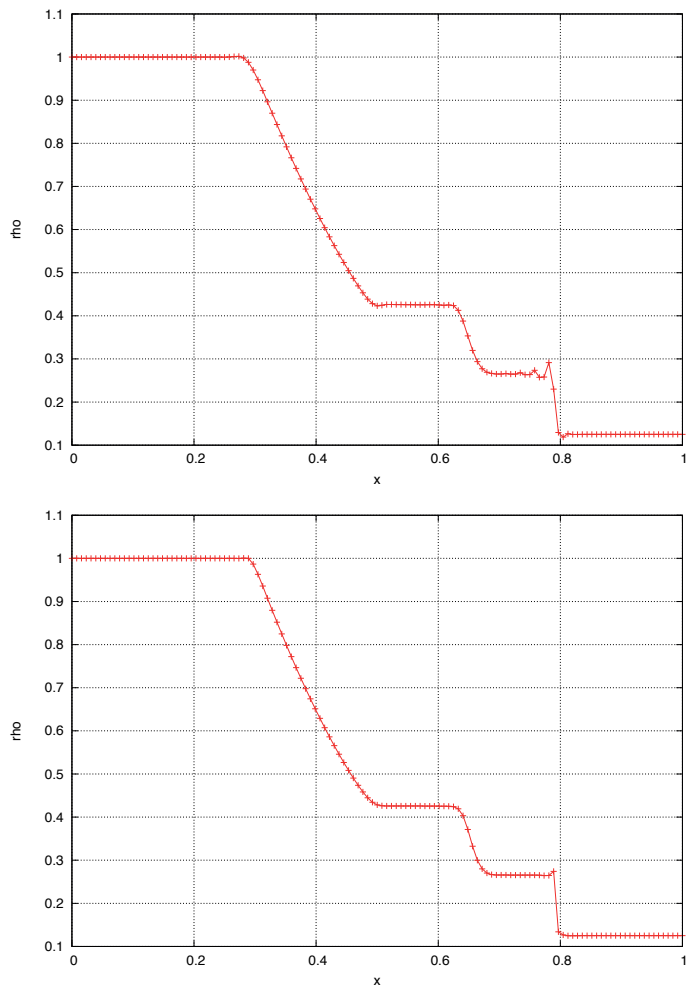


Figure 3.10: Shock tube problem computed with a SUPG formulation with a linear basis (top) and quadratic basis (bottom).

Since such oscillations are normally undesirable, techniques have evolved to control and prevent them. These will be considered in chapter 4.

During the 1980's and 1990's the SUPG method gradually evolved into the Galerkin least-squares (GLS) method, the latter being equivalent to SUPG for linear elements, but having a symmetric stabilisation term for which it is easier to prove stability. The GLS variant was ultimately developed into a very effective discretisation for the compressible Navier-Stokes equations in a series of papers by Hughes and his co-workers [24, 27, 25, 26, 22, 21, 23, 44, 45].

In the late 1990's, the class of stabilised methods which included SUPG and GLS were generalised to include a multiscale interpretation. This has led to a interesting new approach to incorporating general physical concepts (including upwinding) into numerical discretisations. Its formulation will be considered in part 2 of these notes, where the computation of turbulence is discussed.

Chapter 4

Controlling oscillations

As first-order schemes are insufficiently accurate for practical purposes, generalisations to second and higher order schemes have naturally been a subject of interest. Since conventional higher-order central and upwind methods produce oscillatory solutions near discontinuities, researchers have produced conditions which ensure non-oscillatory behaviour in order to help them design more suitable schemes. This chapter will review these conditions and give an example of a higher-order non-oscillatory scheme.

4.1 Monotone conservative schemes

Godunov's original method had several useful properties. Firstly, it produced non-oscillatory solutions, and secondly, it avoided unphysical expansion shocks, which are possible in the context of the Euler equations, but violate the second law of thermodynamics. It is therefore said to satisfy the *entropy condition*. Non-oscillatory schemes which satisfy the entropy condition are called monotone conservative schemes. In the following we will first examine how invalid discontinuous solutions arise, and then give the general mathematical conditions for a monotone conservative scheme.

4.1.1 Undesirable discontinuous solutions

Something similar to an expansion shock can occur in solutions to the inviscid Burgers equation, $u_t + uu_x = 0$. The behaviour of this equation is similar to that of the linear convection equation, except that the local convection speed is defined by the solution. Therefore, wave steepening, as shown in figure 4.1, can occur. This results in a convergence of characteristics as shown in the right part of the figure, leading to the formation of a shock at point A. If one begins with a discontinuous solution, where the upstream velocity is larger as shown in figure 4.2 (left), then the shock simply propagates through the space-time domain as shown. If the upstream velocity is smaller than the downstream ve-

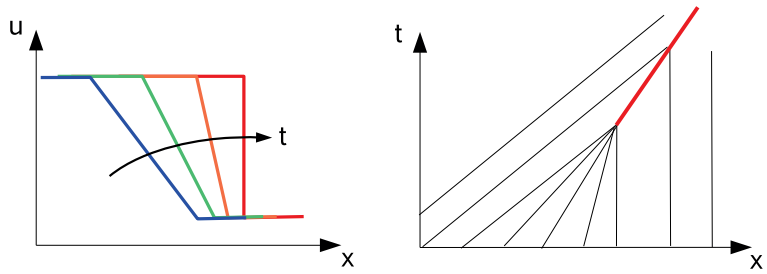


Figure 4.1: Wave steepening and shock formation in Burgers' equation

locity, then there are two possibilities as shown in figure 4.2 (right). First, the discontinuity could continue to propagate, but this would require information being emanated from it. Otherwise, a continuous transition could be obtained. When the inviscid Burgers equation is viewed as the limit of vanishing viscosity of the viscous Burgers equation, $u_t + uu_x = \nu u_x x$, then the continuous solution is the correct one. In the Euler equations, the analogous discontinuous solution where there is an increase in velocity and reduction in density across the discontinuity is called an expansion shock. In order to select the desired continuous solution instead, an entropy condition may be stated as for a shock with speed C,

$$(u)_{upstream} > C > (u)_{downstream} \tag{4.1}$$

Where the Godunov approach is not used, artificial dissipation is sometimes employed to enforce the entropy condition.

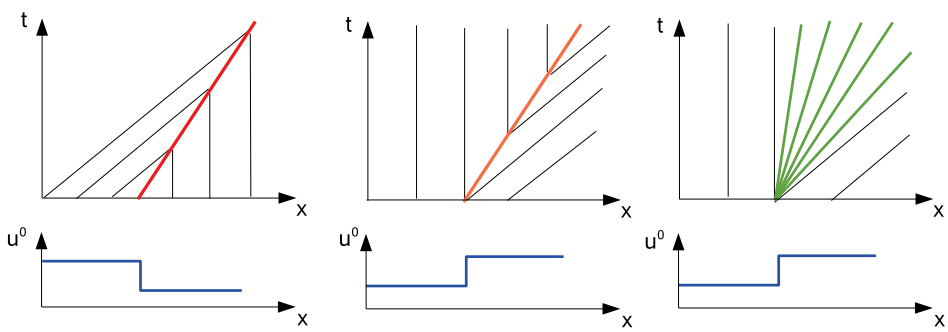


Figure 4.2: Solutions to the Burgers equation for discontinuous initial conditions

4.1.2 Conditions for monotone conservative schemes

A conservative scheme which advances the solution u at point i from time level n to $n+1$:

$$u^{n+1} = u^n - \frac{\Delta t}{\Delta x}(f_{i+1/2} - f_{i-1/2}) \tag{4.2}$$

is monotone when

$$if \quad u^n_{A\ i} \geq u^n_{B\ i} \quad \forall i, \quad then \qquad u^{n+1}_{A\ i} \geq u^{n+1}_{B\ i} \quad \forall i \tag{4.3}$$

where u_A and u_B are separate numerical solutions for u. For a general monotone conservative scheme:

$$u^{n+1}_i = H(u^n_1, u^n_2, \ldots u^n_{i-1}, u^n_i, u^n_{i+1} \ldots) \tag{4.4}$$

this implies:

$$\frac{\partial H}{\partial u_k} \geq 0 \quad \text{for all } k \tag{4.5}$$

Although the behaviour of monotone conservative schemes is definitely desirable, in practice the monotonicity conditions are quite restrictive. Consequently, weaker measurements of non-oscillatory behaviour are normally used for the design of higher-order schemes, such as the the monotonicity-preserving condition described in the next section.

4.2 Monotonicity-preserving schemes

As discussed in the previous chapter, the Euler equations may be viewed as a system which expresses the convection of entropic, vortical and acoustic waves. A reasonable model equation is therefore the linear convection equation:

$$u_t + a u_x = 0 \tag{4.6}$$

which has the solutions of the form $u(x,t) = u_o(x - at)$, in other words, the initial condition is propagated without attenuation with a wave speed a. Numerical approximations to this solution, however, can contain oscillations. This may be viewed as the consequence a numerical method which generates new extrema in the solution (figure 4.3). Expressing conditions which prevent the generation

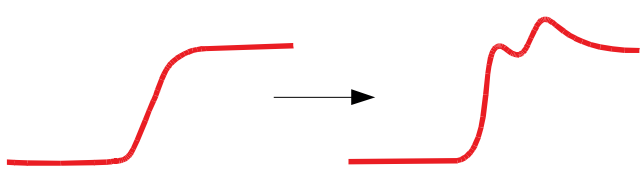


Figure 4.3: Generation of new extrema in the numerical solution

of new extrema is easiest when considering monotone functions. For example,

if the initial condition is monotone (i.e. if $u(x,0)$ is a strictly increasing or strictly decreasing function of x), then the desired numerical solution should also remain monotone. Consequently, a scheme which does not generate new extrema is said to be monotonicity-preserving.

Single-step numerical schemes for the linear convection equation can ultimately be written as an expression which advances the solution at grid point i from time level n to $n+1$. For linear discretisations which can be written

$$u_i^{n+1} = \sum_k b_k u_{i+k}^n \tag{4.7}$$

monotonicity preservation requires all b_k to be positive. As a proof, consider the monotonically increasing solution for u at time level n shown in figure 4.4. For discretisations defined using (4.7) the difference in solution between neigh-

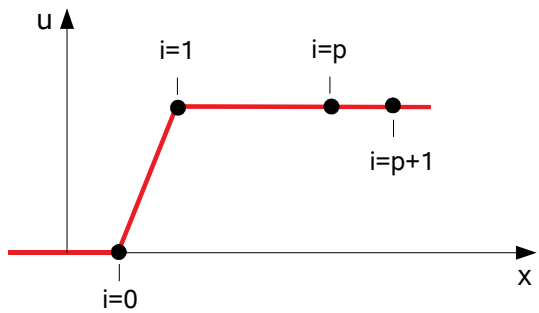


Figure 4.4: Proof of monotonicity preservation

bouring points at the next time level, $n+1$ is:

$$u_{p+1}^{n+1} - u_p^{n+1} = \sum_k b_k (u_{p+1+k}^n - u_{p+k}^n) \tag{4.8}$$

if b_k is negative for $k = -p$, then $(u_{p+1}^{n+1} - u_p^{n+1})$ is negative and the solution is no longer monotonic. Since p is arbitrary, none of the b_k can be negative.

Using the requirement that $b_k \geq 0$, one can quickly verify that the first-order upwind scheme

$$u_i^{n+1} = (1 - \frac{a\Delta t}{\Delta x}) u_i^n + (\frac{a\Delta t}{\Delta x}) u_{i-1}^n \tag{4.9}$$

is monotonicity-preserving provided the stability condition is satisfied,
i.e. $0 \leq a\Delta t/\Delta x \leq 1$. In fact, Godunov [12] showed that linear monotonicity-preserving schemes can be no more than first-order accurate. This is referred to as Godunov order-barrier theorem.

In the solution of the Euler equations, we would like to have monotonicity-preserving behaviour in the characteristic variables. Note that this is not the same as saying no new extrema can be generated in variables such as pressure for example, as multiple waves can still interact. The order-barrier theorem,

however, indicates that we are limited to first-order accuracy if we want to use linear schemes. The solution is to instead use discretisations which are non-linear, even when applied to linear equations.

4.3 Total variation diminishing schemes

A condition which includes monotonicity preservation is based on a property of scalar conservation laws derived by Lax [35]. Scalar conservation laws of the form:

$$u_t + f_x = 0 \tag{4.10}$$

have the property:

$$\frac{\partial}{\partial t}(TV) \;=\; \frac{\partial}{\partial t}\int \left|\frac{\partial u}{\partial x}\right| dx \;\leq\; 0 \tag{4.11}$$

An analogous discrete condition was formulated by Harten [14]:

$$(TV)^{n+1} \leq (TV)^n \tag{4.12}$$

where:

$$(TV)^n = \sum_i \left|u_{i+1}^n - u_i^n\right| \tag{4.13}$$

The discrete version of the TVD condition is often used in the design of numerical schemes. The motivation for doing so is illustrated by considering linear discretisations of the form:

$$u_i^{n+1} = \sum_k b_k u_{i+k}^n \tag{4.14}$$

which are TVD when the are monotonicity preserving:

$$b_k \geq 0 \qquad \forall k \tag{4.15}$$

and consistent:

$$\sum_k b_k \leq 1 \tag{4.16}$$

To prove these properties, consider the advancement of a monotone initial solution u_i^0 which is constant for $\pm\infty$ to a non-monotone solution u_i^1 , so that $b_k \geq 0\ \forall k$ is not satisfied (figure 4.5). Then:

$$(TV)^1 = \sum_i \left|u_i^1 - u_{i-1}^1\right| > (TV)^0 \tag{4.17}$$

Now consider a non-decreasing u_i^0, and assume $b_k \geq 0\ \forall k$. Then u_i^1 is non-decreasing, so that:

$$\begin{aligned}(TV)^1 &= \sum_i (u_i^1 - u_{i-1}^1) = \sum_i \sum_k b_k (u_{i+k}^0 - u_{i-1+k}^0) \\ &= \sum_k b_k \sum_j (u_j^0 - u_{j-1}^0) = \sum_k b_k (TV)^0 \end{aligned} \tag{4.18}$$

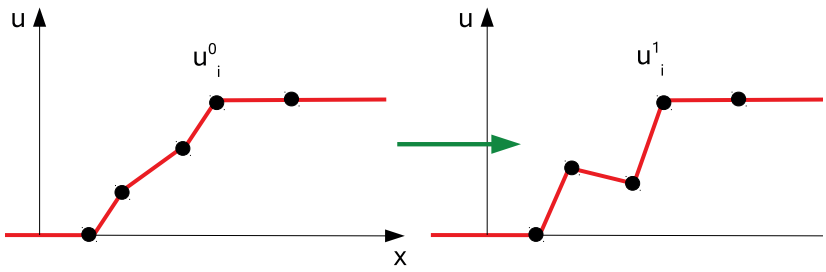


Figure 4.5: Proof of monotonicity preservation in TVD schemes

We therefore require $\sum_k b_k \leq 1$ to maintain $(TV)^1 \leq (TV)^0$.

Equations (4.13) and (4.12) can thus be used for the design of non-oscillatory discretisations, but if we limit our attention to second-order finite-volume or finite-difference methods, we can express them in a more convenient way. Harten also showed that three-point semi-discrete schemes of the form:

$$\frac{\partial u_i}{\partial t} = -\frac{1}{\Delta x}\left(C^-_{i+\frac{1}{2}}(u_{i+1} - u_i) + C^+_{i-\frac{1}{2}}(u_i - u_{i-1})\right) \tag{4.19}$$

will be TVD ([14],[19]) provided:

$$\begin{aligned} C^+_{i+\frac{1}{2}} &\geq 0 \\ C^-_{i+\frac{1}{2}} &\leq 0 \end{aligned} \tag{4.20}$$

If explicit time integration is used, maintenance of the TVD property for a fully-discrete scheme will also require:

$$\frac{\Delta t}{\Delta x}\left(C^+_{i+\frac{1}{2}} - C^-_{i+\frac{1}{2}}\right) \leq 1 \tag{4.21}$$

It therefore only remains to define schemes whose coefficients satisfy these properties.

4.4 Non-linear limiting

The result that linear schemes which are monotonicity-preserving can be no more than first-order accurate can be circumvented by designing non-linear schemes. These typically include non-linear limiters which are designed to satisfy the non-oscillatory conditions described in the previous sections.

One way to increase the order of a finite-volume scheme, for example, is to assume the solution is piecewise linear as shown in figure 4.6. Such a method can generate the situation shown in cell i, however, which implies the introduction of a new extrema. We can detect this situation by comparing the slopes of the neighbouring cells and looking for a change in sign. The new slope in cell i

can then be limited to a value of zero. This limiting is a non-linear operation, and is typically implemented using a limiter function ϕ, written in terms of neighbouring cell gradients. Schemes where the limiting is based on solution

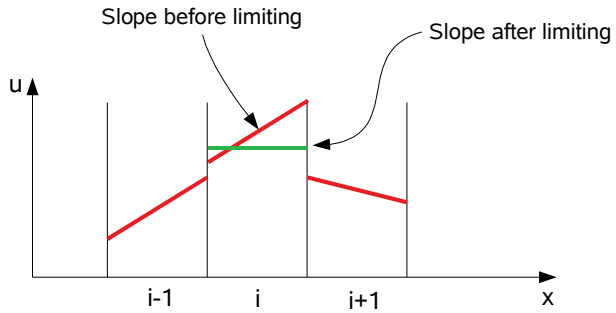


Figure 4.6: Generation and prevention of new extrema in the numerical solution

gradients are called slope-limited schemes. It is also possible to base limiter design on local flux changes and limit the numerical flux function. Such methods are referred to as flux-limited schemes.

4.5 An example non-oscillatory higher-order scheme

We now demonstrate the construction of a higher-order non-oscillatory scheme using the TVD conditions and flux limiting. The starting point is the first-order accurate Godunov method using Roe's approximate Riemann solver, as described in section 3.2. This produces a semi-discrete method of the form:

$$\frac{\partial u_i}{\partial t} = -\frac{1}{\Delta x}\left(f_{i+\frac{1}{2}} - f_{i-\frac{1}{2}}\right) \tag{4.22}$$

where

$$f_{i+\frac{1}{2}} = \frac{1}{2}\left(f_{i+1} + f_i\right) - \frac{1}{2} \mid a_{i+\frac{1}{2}} \mid \left(u_{i+1} - u_i\right) \tag{4.23}$$

and

$$a_{i+\frac{1}{2}} = \begin{cases} \frac{f_{i+1}-f_i}{u_{i+1}-u_i} & \text{if } u_{i+1} \neq u_i \\ \\ \frac{\partial f}{\partial u} & \text{if } u_{i+1} = u_i \end{cases} \tag{4.24}$$

This scheme can be verified to have the TVD property. As $\frac{1}{2}\left(f_{i+1} + f_i\right)$ represents a second-order estimation of the interface flux, we can identify the second term on the right-hand side of (4.23) as the error which reduces the scheme to first order. In the construction of a second-order TVD scheme, we desire to eliminate as much of this term as possible, in a manner consistent with the conditions (4.20). To this end, a corrected flux can be defined as [55]:

$$\hat{f}_i = f_i + g_i \tag{4.25}$$

where g_i represents an anti-diffusive flux which cancels first-order error. The value of g_i is to be limited non-linearly, however, so that the TVD conditions are satisfied:

$$g_i = \phi_i \; g_{i+\frac{1}{2}} \tag{4.26}$$

$$= \phi_i \; \frac{\mid a_{i+\frac{1}{2}} \mid}{2} (u_{i+1} - u_i) \tag{4.27}$$

The limiter function, ϕ, is defined in terms of the ratio of neighbouring fluxes:

$$\phi_i = \phi_i (r_i) \tag{4.28}$$

where

$$r_i = \frac{g_{i-\frac{1}{2}}}{g_{i+\frac{1}{2}}} \tag{4.29}$$

To make it easier to find a TVD scheme, we use the modified flux definition in the same way as the flux was used in the first-order scheme. Expressions for $f_{i+\frac{1}{2}}$ equivalent to (4.23) are given by:

$$f_{i+\frac{1}{2}} = f_i - \frac{1}{2}\left(\mid a_{i+\frac{1}{2}} \mid - a_{i+\frac{1}{2}}\right)(u_{i+1} - u_i) \tag{4.30}$$

$$= f_{i+1} - \frac{1}{2}\left(\mid a_{i+\frac{1}{2}} \mid + a_{i+\frac{1}{2}}\right)(u_{i+1} - u_i) \tag{4.31}$$

Replacing $\partial g / \partial u$ by $(g_{i+1} - g_i)/(u_{i+1} - u_i)$, the corrected interface flux can be written:

$$\hat{f}_{i+\frac{1}{2}} = f_i + \frac{1}{2}\left(a_{i+\frac{1}{2}} - \mid a_{i+\frac{1}{2}} \mid\right)(u_{i+1} - u_i)$$
$$+ g_i - \frac{1}{2}\left(\left|\frac{g_{i+1} - g_i}{u_{i+1} - u_i}\right| - \frac{g_{i+1} - g_i}{u_{i+1} - u_i}\right)(u_{i+1} - u_i) \tag{4.32}$$

$$= f_{i+1} - \frac{1}{2}\left(a_{i+\frac{1}{2}} + \mid a_{i+\frac{1}{2}} \mid\right)(u_{i+1} - u_i)$$
$$+ g_{i+1} - \frac{1}{2}\left(\left|\frac{g_{i+1} - g_i}{u_{i+1} - u_i}\right| + \frac{g_{i+1} - g_i}{u_{i+1} - u_i}\right)(u_{i+1} - u_i) \tag{4.33}$$

Averaging the above expressions, we get:

$$\hat{f}_{i+\frac{1}{2}} = \frac{f_{i+1} + f_i}{2} - d_{i+1/2} \tag{4.34}$$

where

$$d_{i+1/2} = g_{i+\frac{1}{2}} - \frac{(g_{i+1} + g_i)}{2} + \frac{1}{2} \mid g_{i+1} - g_i \mid \text{sign}\left(g_{i+\frac{1}{2}}\right) \tag{4.35}$$

The form (4.35) is particularly convenient for implementation. In order to analyse the new flux definition, (4.27) and (4.29) can be used to alternatively

express the average of (4.32) and (4.33) as:

$$\hat{f}_{i+\frac{1}{2}} = \frac{f_{i+1} + f_i}{2} - \left[\frac{|\, a_{i+\frac{1}{2}} \,|}{2} \left(1 - \frac{1}{2} \left(\frac{\phi_{i+1}}{r_{i+1}} + \phi_i \right) + \frac{1}{2} \left| \frac{\phi_{i+1}}{r_{i+1}} - \phi_i \right| \right) \right] (u_{i+1} - u_i) \tag{4.36}$$

The three-point scheme coefficients can also be obtained by substituting (4.32) and (4.33) into (4.19) to obtain:

$$C^-_{i+\frac{1}{2}} = \frac{a_{i+\frac{1}{2}}}{2} - \frac{|\, a_{i+\frac{1}{2}} \,|}{2} \left(1 - \frac{1}{2} \left(\frac{\phi_{i+1}}{r_{i+1}} - \phi_i \right) + \frac{1}{2} \left| \frac{\phi_{i+1}}{r_{i+1}} - \phi_i \right| \right) \tag{4.37}$$

$$C^+_{i+\frac{1}{2}} = \frac{a_{i+\frac{1}{2}}}{2} + \frac{|\, a_{i+\frac{1}{2}} \,|}{2} \left(1 + \frac{1}{2} \left(\frac{\phi_{i+1}}{r_{i+1}} - \phi_i \right) + \frac{1}{2} \left| \frac{\phi_{i+1}}{r_{i+1}} - \phi_i \right| \right) \tag{4.38}$$

It can be easily verified that $C^+_{i+\frac{1}{2}} \geq 0$ and $C^-_{i+\frac{1}{2}} \leq 0$. However, the solutions produced by the discretisation are influenced by the choice of ϕ. For example, (4.36) reverts to the first-order flux when $\phi = 0$. (4.36) also indicates that in order to maintain second-order accuracy in regions of low gradients, we require $\phi = 1$ when $r = 1$. We also require $\phi \to 0$ as $r \to 0$ for the expression to remain bounded. For the fully-discrete scheme, condition (4.21) requires:

$$\left| \frac{\phi_{i+1}}{r_{i+1}} - \phi_i \right| \leq \frac{2}{\nu} - 2 \tag{4.39}$$

where

$$\nu = \frac{|\, a_{i+\frac{1}{2}} \,| \, \Delta t}{\Delta x} \tag{4.40}$$

which indicates that the choice of limiter function affects the maximum allowable time step.

The design of limiter functions for flux-limited schemes was considered in detail by [46]. By examining limited forms of the Lax-Wendroff and the Warming and Beam upwind schemes, Sweby defined regions which maintain the TVD property while preserving second-order accuracy whenever possible (Figure 4.7). Limiters which lie on the lower part of the region tend to be diffusive, while those on the upper part of the region tend to be compressive, or tend to turn sine waves into square waves.

Application to the system of Euler equations

The modified flux TVD scheme can be applied to the Euler equations by replacing $(u_{i+1} - u_i)$ with the changes in the characteristic variables $\Delta U^* = T^{-1} \Delta U$, where T is the matrix of eigenvectors of the conservative flux Jacobian.

For example, if a Riemann problem is solved for the first-order flux using the procedure outlined in Appendix 3.3, the difference between the second- and first-order fluxes is given by:

$$g^*_{m,i+\frac{1}{2}} = \frac{|\lambda_m|}{2} \Delta U^*_m \tag{4.41}$$

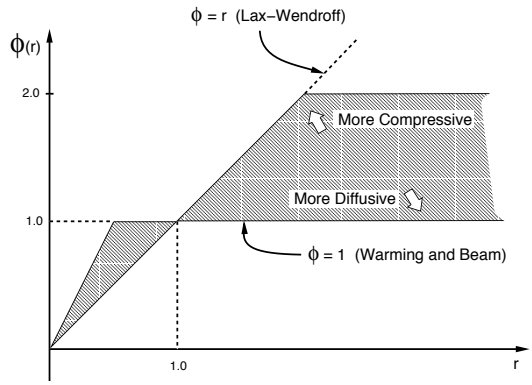


Figure 4.7: Second-order limiter regions defined by Sweby.

where m refers to the index of the characteristic variable under consideration. One can then define $r_{m,i}$ as $g^*_{m,i-\frac{1}{2}}/g^*_{m,i+\frac{1}{2}}$ and obtain $g^*_{m,i} = \phi_{m,i} g^*_{m,i+\frac{1}{2}}$. Expression (4.35) can be used to obtain the net diffusive flux in characteristic components, $d^*_{m,i+\frac{1}{2}}$. This can be returned to conservative components via:

$$d_{i+\frac{1}{2}} = T\, d^*_{i+\frac{1}{2}} \tag{4.42}$$

and added to the conservative flux average across the cell. In multiple dimensions, the characteristic decomposition is normally based on the matrix $A \cdot \vec{n} = A_x n_x + A_y n_y + A_z n_z$. If used in a finite volume scheme, the resulting $d_{i+\frac{1}{2}}$ then needs to be scaled by the magnitude of the face area vector, S:

$$F_{i+\frac{1}{2}} \cdot S = \left(\frac{F_{i+1} + F_i}{2} \right) \cdot S \; - \; d_{i+\frac{1}{2}} |S| \tag{4.43}$$

Computed results

We return to the shock tube problem discussed in section 3.2 but now apply the TVD scheme derived above, implemented within a finite-volume discretisation. Figure 4.8 compares results for limiters occupying different parts of the TVD region. Each solution remains monotone, and is clearly more accurate than the original Godunov/Roe scheme. The highest portion of the TVD region traced by the SuperBee limiter gives the most compressive solution.

The different limiters are now compared for a transonic airfoil case. A NACA0012 profile with an angle of attack of 1.25 degrees and a Mach number of 0.8 is considered. The 298x39 C-Mesh shown in figure 4.9 is used for all computations. For reference, the results from the JST central scheme described in the introduction are also shown. The shock capturing trends are similar to the 1D case, which is perhaps not surprising as the grid lines are

nearly aligned with the shock. The JST scheme's free coefficients have been adjusted to produce an oscillation-free solution for in this particular case. It is seen to be competitive, although the shocks are more smeared. The detailed representation of the shock is inferior, however. This can be seen by examining the entropy variation along the airfoil surface. As can be seen from figure 4.11, in both cases false entropy is generated at the leading edge and convected downstream due to the high gradients in that region. The JST scheme has less of this type of initial error. For the entropy change at the shock, however, the prediction from the TVD scheme is within 2% of the Rankine-Hugoniot value ($\Delta s = 1.08 \times 10^{-2}$) while the JST scheme over-predicts it by 33%. The expense of the upwind approach thus seems justified when the details of shock behaviour are of interest.

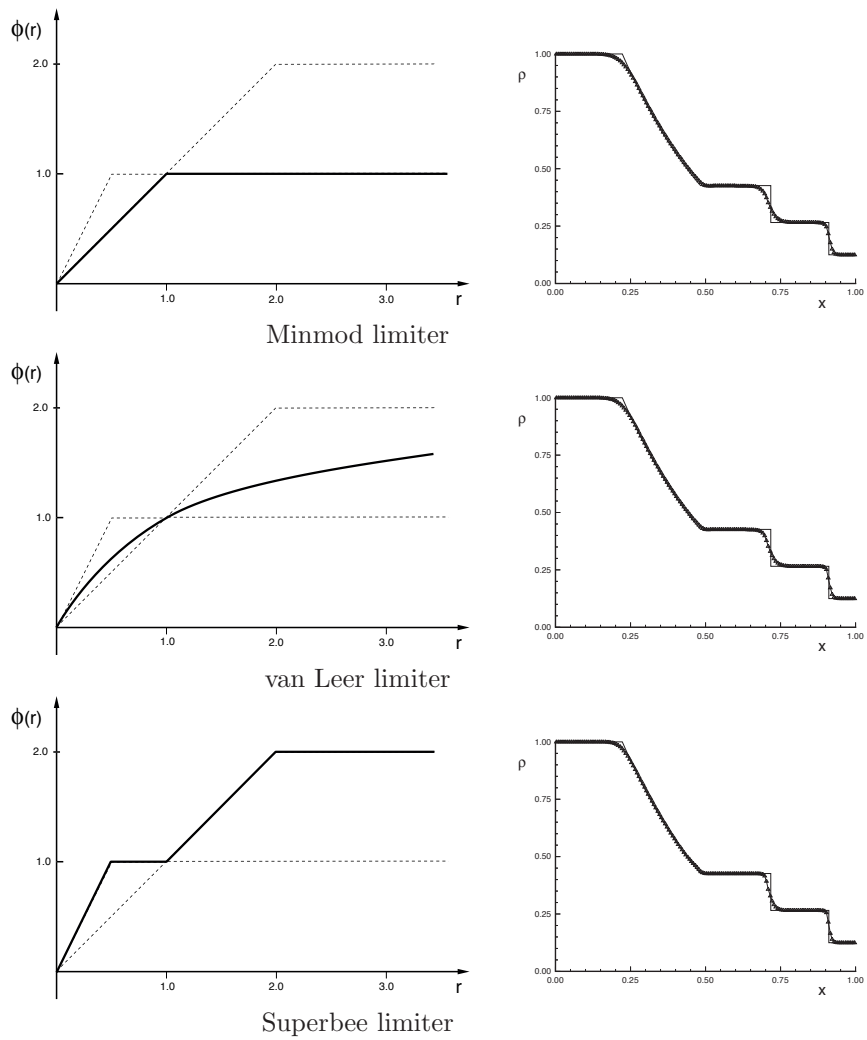


Figure 4.8: TVD solutions for 1D shock tube problem using different limiters

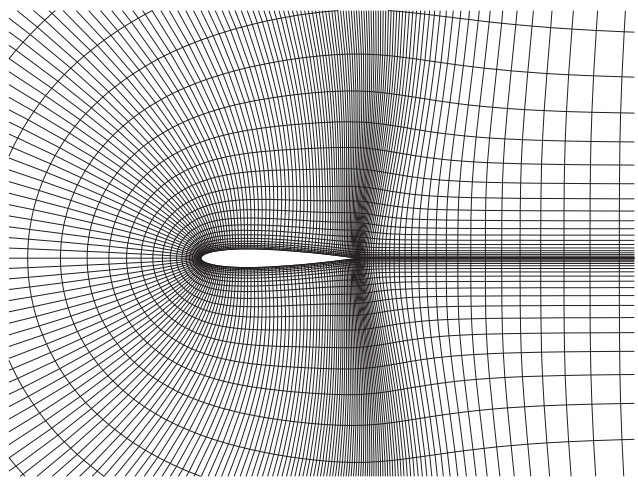


Figure 4.9: The 298x39 C-Mesh for NACA0012 Airfoil

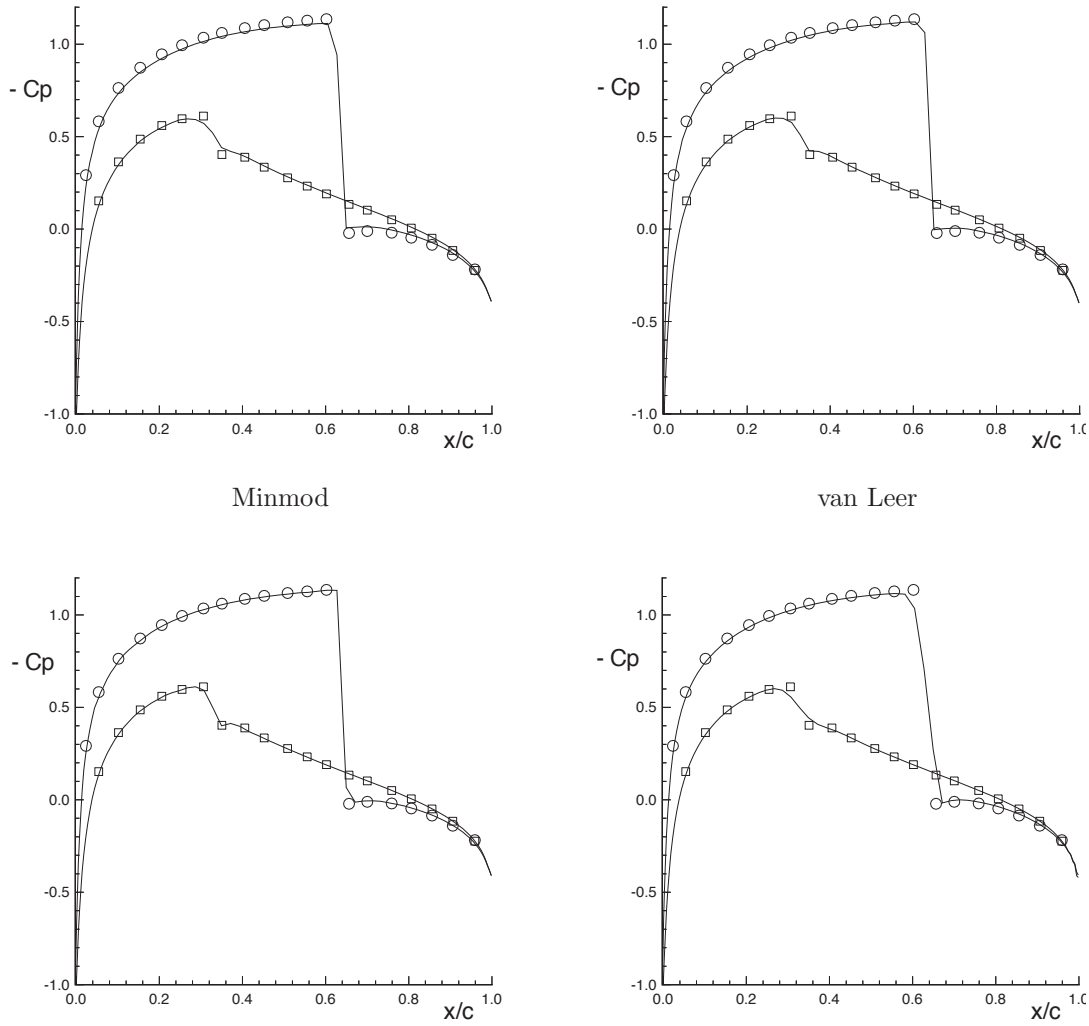


Figure 4.10: TVD solutions for NACA0012 using different limiters and the JST scheme. The solid lines indicate computed solutions, and the symbols the solution computed on a very fine mesh.

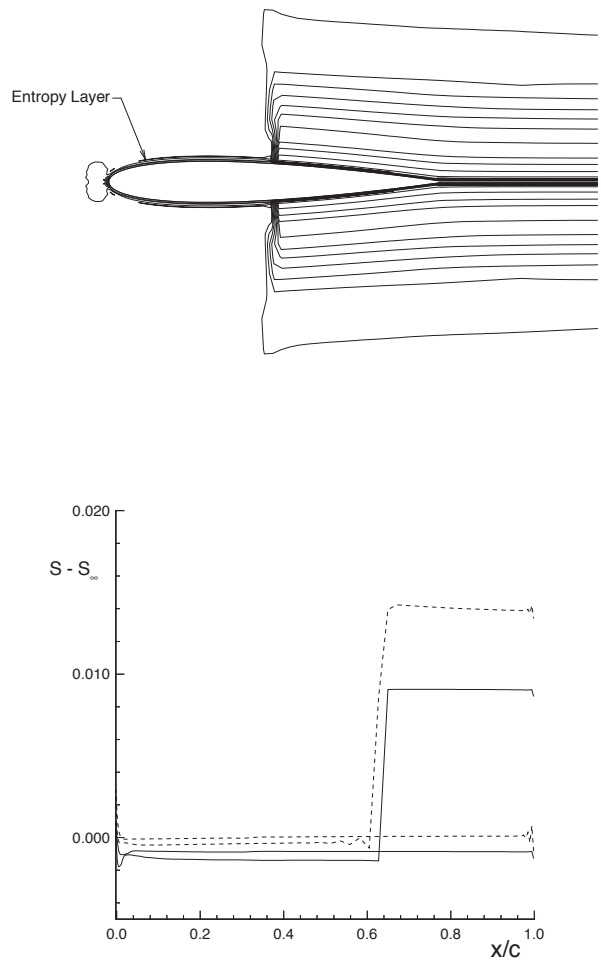


Figure 4.11: Entropy contours for a $\alpha = 0$ case (top) and entropy profiles for the $\alpha = 1.25$ case (bottom) from TVD/Superbee (solid) and optimised JST (dashed) computations

Bibliography

[1] R. Abgrall and M. Mezine. Construction of second order accurate monotone and stable residual distribution schemes for unsteady flow problems. *Journal of Computational Physics*, 188(1):16–55, June 2003.

[2] Cockburn B., G. E. Karniadakis, and C.-W. Shu. The development of discontinuous Galerkin methods. In Cockburn B., G. E. Karniadakis, and C.-W. Shu, editors, *Discontinuous Galerkin Methods.Theory, Computation and Applications*, volume 11 of *Lecture Notes in Computational Science and Engineering*, pages 3–50. Springer-Verlag, 2000.

[3] de Maerschalck B., Gerritsma M. I., and Proot M. M. J. Space-time least-squares spectral elements for convection-dominated unsteady flows. *AIAA journal*, 44(3):558–565, 2006.

[4] Pavel Bochev and James Hyman. Principles of mimetic discretizations of differential operators. In Douglas N. Arnold, Pavel B. Bochev, Richard B. Lehoucq, Roy A. Nicolaides, Mikhail Shashkov, Douglas N. Arnold, and Fadil Santosa, editors, *Compatible Spatial Discretizations*, volume 142 of *The IMA Volumes in Mathematics and its Applications*, pages 89–119. Springer New York, 2006.

[5] J. P. Boris and D. L. Book. Flux corrected transport I: SHASTA, a fluid transport algorithm that works. *J. Comput. Phys.*, 11(1):38–69, 1973.

[6] J. P. Boris and D. L. Book. Flux corrected transport III: Minimal-error FCT algorithms. *J. Comput. Phys.*, 20(4):397–431, 1976.

[7] J. P. Book D. L. Boris and K. Hain. Flux corrected transport II: Generalizations of the method. *J. Comput. Phys.*, 18(3):248–283, 1975.

[8] Franco Brezzi, Konstantin Lipnikov, and Valeria Simoncini. A family of mimetic finite difference methods on polygonal and polyhedral meshes. *Mathematical Models and Methods in Applied Sciences*, 15(10):1533–1552, 2005.

[9] A. N. Brooks and T. J. R. Hughes. Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on

the incompressible Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 32(1-3):199–259, September 1982.

[10] Tim Colonius. Modeling artificial boundary conditions for compressible flow. *Annu. Rev. Fluid Mech.*, 36:315–345, 2004.

[11] J. Glimm and P. D. Lax. Decay of solutions of systems of nonlinear hyperbolic conservation laws. *Memoirs of the Amer. Math. Soc*, (101), 1970.

[12] S. K. Godunov. A difference method for numerical calculation of discontinuous solutions of the equations of hydrodynamics. *Mat. Sb.*, 47:271–306, 1959.

[13] J. B. Goodman and LeVeque R. J. On the accuracy of stable schemes for 2d scalar conservation laws. *Mathematics of Computation*, 45(171):15–21, July 1985.

[14] A. Harten. High resolution schemes for hyperbolic conservation laws. *Journal of Computational Physics*, 50(2):235–269, may 1983.

[15] A. Harten. Uniformly high order accurate essentially non-oscillatory schemes, III. *Journal of Computational Physics*, 71(2):231–303, 1987.

[16] A. Harten and S. Osher. Uniformly high-order accurate nonoscillatory schemes, I. *SIAM Journal on Numerical Analysis*, 24(2):279–309, 1987.

[17] R. Hartmann and P. Houston. Adaptive discontinuous Galerkin finite element methods for the compressible Euler equations. *Journal of Computational Physics*, 183(2):508–532, December 2002.

[18] C Hirsch. *Numerical computation of internal and external flows. Volume 1: Fundamentals of Numerical Discretization*, volume Series in Numerical Methods in Engineering. Wiley, 1990.

[19] C Hirsch. *Numerical computation of internal and external flows. Volume 2: Computational methods for inviscid and viscous flows*, volume Series in Numerical Methods in Engineering. Wiley, 1990.

[20] T. J. R. Hughes. A simple scheme for developing upwind finite elements. *IJNME*, 12:1359–1365, 1978.

[21] T. J. R. Hughes and L. P. Franca. A new finite element formulation for computational fluid dynamics: VII. The stokes problem with various well-posed boundary conditions: Symmetric formulations that converge for all velocity/pressure spaces. *Computer Methods in Applied Mechanics and Engineering*, 65(1):85–96, November 1987.

[22] T. J. R. Hughes, L. P. Franca, and M. Balestra. A new finite element formulation for computational fluid dynamics: V. Circumventing the Babuška-Brezzi condition: A stable Petrov-Galerkin formulation of the stokes problem accommodating equal-order interpolations. *Computer Methods in Applied Mechanics and Engineering*, 59(1):85–99, November 1986.

[23] T. J. R. Hughes, L. P. Franca, and G. M. Hulbert. A new finite element formulation for computational fluid dynamics: VIII. The Galerkin/least-squares method for advective-diffusive equations. *Computer Methods in Applied Mechanics and Engineering*, 73(2):173–189, May 1989.

[24] T. J. R. Hughes, L. P. Franca, and M. Mallet. A new finite element formulation for computational fluid dynamics: I. Symmetric forms of the compressible Euler and Navier-Stokes equations and the second law of thermodynamics. *Computer Methods in Applied Mechanics and Engineering*, 54(2):223–234, February 1986.

[25] T. J. R. Hughes and M. Mallet. A new finite element formulation for computational fluid dynamics: III. The generalized streamline operator for multidimensional advective-diffusive systems. *Computer Methods in Applied Mechanics and Engineering*, 58(3):305–328, November 1986.

[26] T. J. R. Hughes and M. Mallet. A new finite element formulation for computational fluid dynamics: IV. A discontinuity-capturing operator for multidimensional advective-diffusive systems. *Computer Methods in Applied Mechanics and Engineering*, 58(3):329–336, November 1986.

[27] T. J. R. Hughes, M. Mallet, and A. Mizukami. A new finite element formulation for computational fluid dynamics: II. Beyond SUPG. *Computer Methods in Applied Mechanics and Engineering*, 54(3):341–355, March 1986.

[28] Thomas J. R. Hughes, Gerald Engel, Luca Mazzei, and Mats G. Larson. The continuous galerkin method is locally conservative. *Journal of Computational Physics*, 163(2):467 – 488, 2000.

[29] Thomas J.R. Hughes and Garth N. Wells. Conservation properties for the galerkin and stabilised forms of the advection-diffusion and incompressible navier-stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 194(9-11):1141 – 1159, 2005.

[30] M. Y. Hussaini, B. van Leer, and J. van Rosendale. *Upwind and High-Resolution Schemes*. Springer-Verlag, 1997.

[31] A. Jameson. Analysis and design of numerical schemes for gas dynamics 1: Artificial diffusion, upwind biasing, limiters and their effect on accuracy and multigrid convergence. *International Journal of Computational Fluid Dynamics*, 4:171–218, 1995.

[32] A. Jameson, W. Schmidt, and E. Turkel. Numerical solutions of the Euler equations by finite volume methods with Runge-Kutta time stepping schemes. *AIAA*, Paper(81-1259):1, 1981.

[33] Bo-Nan Jiang. *The Least-squares Finite Element Method: Theory and Applications in Computational Fluid Dynamics and Electromagnetics*. Springer, 1998.

[34] C. M. Klaija, J.J.W. van der Vegt, and H. van der Ven. Space–time discontinuous Galerkin method for the compressible Navier–Stokes equations. *Journal of Computational Physics*, 217(2):589–611, September 2006.

[35] P. Lax. Hyperbolic systems of conservation laws and the mathematical theory of shock waves. *SIAM Regional Series on Applied Mathematics*, 11, 1973.

[36] M-S. Liou. Ten years in the making - AUSM family. (2001-2521), 2001.

[37] Guido Lodato, Pascale Domingo, and Luc Vervisch. Three-dimensional boundary conditions for direct and large-eddy simulation of compressible viscous flows. *Journal of Computational Physics*, 227(10):5105–5143, 2008.

[38] L. G. Margolin and M. Shashkov. Finite volume methods and the equations of finite scale: A mimetic approach. *International Journal for Numerical Methods in Fluids*, 56(8):991–1002, 2008.

[39] Jan Nordstrom and Mark H. Carpenter. High-order finite difference methods, multidimensional linear problems, and curvilinear coordinates. *Journal of Computational Physics*, 173(1):149 – 174, 2001.

[40] S. Osher and F. Solomon. Upwind difference schemes for hyperbolic systems of conservation laws. *Mathematics of Computation*, 38(158):339–374, April 1982.

[41] T. J. Poinsot and S. K. Lele. Boundary conditions for direct simulations of compressible viscous flows. *Journal of Computational Physics*, 101:104–129, 1992.

[42] Wolfgang Polifke, Clifton Wall, and Parviz Moin. Partially reflecting and non-reflecting boundary conditions for simulation of compressible viscous flow. *Journal of Computational Physics*, 213(1):437–449, 2006.

[43] P. L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43(2):357–372, October 1981.

[44] F. Shakib and T. J. R. Hughes. A new finite element formulation for computational fluid dynamics: IX. Fourier analysis of space-time Galerkin/least-squares algorithms. *Computer Methods in Applied Mechanics and Engineering*, 87(1):35–58, May 1991.

[45] F. Shakib, T. J. R. Hughes, and Zdenek J. A new finite element formulation for computational fluid dynamics: X. The compressible Euler and Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 89(1-3):141–219, August 1991.

[46] P. K. Sweby. High resolution schemes using flux limiters for hyperbolic conservation laws. *SIAM Journal on Numerical Analysis*, 21(5):995–1011, Oct 1984.

[47] F. Taghaddosi, W.G. Habashi, G. Guévremont, and D. Ait-Ali-Yahia. An adaptive least-squares method for the compressible Euler equations. *International Journal for Numerical Methods in Fluids*, 31(7):1121–1139, November 1999.

[48] Hughes T.J.R., Engel G., Mazzei L., and Larson M.G. The continuous Galerkin method is locally conservative. *Journal of Computational Physics*, 163(2):467–488, September 2000.

[49] B. van Leer. *Towards the Ultimate Conservative Difference Scheme. I. The quest of monotonicity*, volume Lecture Notes in Physics. Springer, 1973.

[50] B. van Leer. Towards the ultimate conservative difference scheme. II. Monotonicity and conservation combined in a second-order scheme. *Journal of Computational Physics*, 14(4):361–370, March 1974.

[51] B. van Leer. Towards the ultimate conservative difference scheme. III. Upstream-centered finite-difference schemes for ideal compressible flow. *Journal of Computational Physics*, 23(3):263–275, March 1977.

[52] B. van Leer. Towards the ultimate conservative difference scheme. IV. A new approach to numerical convection. *Journal of Computational Physics*, 23(3):276–299, March 1977.

[53] B. van Leer. Flux-vector splitting for the Euler equations. In *8th International Conference on Numerical Methods in Fluid Dynamics*, pages 507–512. Springer-Verlag, 1982.

[54] B. van Leer. Towards the ultimate conservative difference scheme, V. A second-order sequel to Godunov's method. *Journal of Computational Physics*, 135(2):229–248, August 1997.

[55] V. Venkatakrishnan and A. Jameson. Computation of unsteady transonic flows by the solution of Euler equations. *AIAA journal*, 26(8):974–981, 1986.

[56] P. Wesseling. *Principles of computational fluid dynamics*. Springer, 2001.

[57] S. T. Zalesak. Fully multidimensional flux-corrected transport algorithms for fluids. *Journal of Computational Physics*, 31(3):335–362, 1979.

Appendix A

The Euler equations

The Euler equations are a first-order nonlinear system which describe the conservation of mass, momentum and energy for frictionless, non-heat-conducting flows. As well as describing the convection of entropy and vorticity, the Euler equations include all inviscid compressibility effects, and can be used to predict the behaviour of acoustic waves. In addition, they can be used to describe discontinuous flow features, such as compression shocks and contact surfaces. It is possible to express the Euler equations in a number of different forms, some of which are more convenient, depending on the solution method under consideration. The most frequently-used forms are described below.

A.1 Forms of the Euler Equations

A.1.1 Integral Conservation Form

The integral conservative form is the most natural for the derivation of finite-volume methods. It can be written as:

$$\frac{\partial}{\partial t}\int_V U \; dV \; + \; \int_S \vec{F}\cdot\vec{n} \; dS = 0 \tag{A.1}$$

where V is the volume and S is its surface area, with outward unit normal $\vec{n}$ (figure A.1) The algebraic column vector of the conservative variables is given by:

$$U = \begin{bmatrix} \rho \\ \vec{m} \\ \rho E \end{bmatrix} \tag{A.2}$$

where ρ is the density, and E is the total energy per unit mass. $\vec{m}$ is the momentum vector, defined by $\vec{m} = \rho\vec{u}$, where $\vec{u}$ is the the fluid velocity. The normal component of the fluxes through the surface of the control volume may

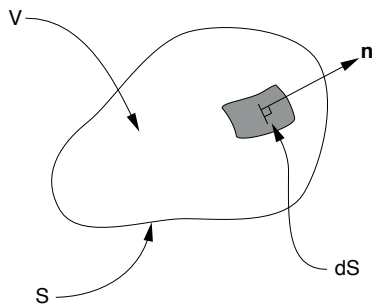


Figure A.1: Arbitrary Control Volume

be expressed:

$$\vec{F} \cdot \vec{n} = \begin{bmatrix} \rho(\vec{u} \cdot \vec{n}) \\ \vec{m}(\vec{u} \cdot \vec{n}) + p\vec{n} \\ \rho E(\vec{u} \cdot \vec{n}) + p\vec{u} \cdot \vec{n} \end{bmatrix} \tag{A.3}$$

where p is the static pressure. When expanded for three dimensions, the above form a system of five nonlinear equations in six variables. In order to close the system, a thermodynamic relation for pressure is required. Throughout these notes we will consider a calorically ideal gas, for which the static pressure may be related to total energy by:

$$\rho E = \frac{p}{\gamma - 1} + \frac{\rho}{2}|\vec{u}|^2 \tag{A.4}$$

In addition to the discontinuities mentioned above, solutions to the Euler equations may also include unphysical expansion shocks. These are discarded from consideration by through the introduction of the second law of thermodynamics, which results in:

$$\frac{\partial s}{\partial t} + \left(\vec{u} \cdot \vec{\nabla}\right) s \;\; \geq \;\; 0 \tag{A.5}$$

i.e. as a fluid element is convected through the flow field, its entropy, s, can only increase or remain constant.

A.1.2 Differential Conservation Form

The differential conservation form of the Euler equations can be derived from the above integral form using the divergence theorem, which applied to the flux integral gives:

$$\int_S \vec{F} \cdot \vec{n} \;\, dS \;\; = \;\; \int_V \vec{\nabla} \cdot \vec{F} \;\, dV \tag{A.6}$$

Provided that V is fixed in time, we can write:

$$\int_V \left(\frac{\partial U}{\partial t} + \vec{\nabla} \cdot \vec{F} \;\right) dV \;\; = \;\; 0 \tag{A.7}$$

As the size of the control volume is arbitrary, this can only be true if for all points within it:

$$\frac{\partial U}{\partial t} + \vec{\nabla} \cdot \vec{F} \;=\; 0 \tag{A.8}$$

This is known as a conservation form as it is still in terms of the conservative variable vector.

A.1.3 Quasi-Linear Conservative-Variable Form

A number of numerical methods make use of a quasi-linear form of equation(A.8):

$$\frac{\partial U}{\partial t} + \left(\frac{\partial \vec{F}}{\partial U}\right) \cdot \vec{\nabla} U \;=\; 0 \tag{A.9}$$

For the flow of an ideal gas in three dimensions, this may be expressed:

$$\frac{\partial U}{\partial t} + A_x \frac{\partial U}{\partial x} + A_y \frac{\partial U}{\partial y} + A_z \frac{\partial U}{\partial z} \;=\; 0 \tag{A.10}$$

where for $\vec{u} = (u, v, w)^t$ and γ is the ratio of specific heats, the Jacobians of the flux vector may be written as:

$$A_x = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ -u^2 + \frac{(\gamma-1)}{2}\vec{u}^2 & (3-\gamma)u & -(\gamma-1)v & -(\gamma-1)w & (\gamma-1) \\ -uv & v & u & 0 & 0 \\ -uw & w & 0 & u & 0 \\ -u[\gamma E - (\gamma-1)\vec{u}^2] & \gamma E - \frac{(\gamma-1)}{2}(\vec{u}^2 + 2u^2) & -(\gamma-1)uv & -(\gamma-1)uw & \gamma u \end{bmatrix}$$

$$A_y = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ -uv & v & u & 0 & 0 \\ -v^2 + \frac{(\gamma-1)}{2}\vec{u}^2 & -(\gamma-1)u & (3-\gamma)v & -(\gamma-1)w & (\gamma-1) \\ -vw & 0 & w & v & 0 \\ -v[\gamma E - (\gamma-1)\vec{u}^2] & -(\gamma-1)uv & \gamma E - \frac{(\gamma-1)}{2}(\vec{u}^2 + 2v^2) & -(\gamma-1)vw & \gamma v \end{bmatrix}$$

$$A_z = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ -uw & w & 0 & u & 0 \\ -vw & 0 & w & v & 0 \\ -w^2 + \frac{(\gamma-1)}{2}\vec{u}^2 & -(\gamma-1)u & -(\gamma-1)v & (3-\gamma)w & (\gamma-1) \\ -w[\gamma E - (\gamma-1)\vec{u}^2] & -(\gamma-1)uw & -(\gamma-1)vw & \gamma E - \frac{(\gamma-1)}{2}(\vec{u}^2 + 2w^2) & \gamma w \end{bmatrix}$$

The quasi-linear form is often the starting point for conservative finite-difference methods, although special care must still be taken with metric evaluation to ensure discrete conservation when curvilinear meshes are used.

A.1.4 Primitive-Variable Form

The primitive variable form can be useful for dealing with certain types of boundary conditions. Its simple structure also makes it a good starting point for the derivation of characteristic forms. Defining a column vector of the primitive variables:

$$U_p = \begin{bmatrix} \rho \\ \vec{u} \\ p \end{bmatrix} \tag{A.11}$$

The previous quasi-linear form of the Euler equations can also be expressed:

$$\frac{\partial U_p}{\partial t} + A_{x_p}\frac{\partial U_p}{\partial x} + A_{y_p}\frac{\partial U_p}{\partial y} + A_{x_p}\frac{\partial U_p}{\partial z} = 0 \tag{A.12}$$

where in this case, the Jacobian matrices have the much simpler form:

$$A_{x_p} = \begin{bmatrix} u & \rho & . & . & . \\ . & u & . & . & \frac{1}{\rho} \\ . & . & u & . & . \\ . & . & . & u & . \\ . & \rho c^2 & . & . & u \end{bmatrix}$$

$$A_{y_p} = \begin{bmatrix} v & . & \rho & . & . \\ . & v & . & . & . \\ . & . & v & . & \frac{1}{\rho} \\ . & . & . & v & . \\ . & . & \rho c^2 & . & v \end{bmatrix}$$

$$A_{z_p} = \begin{bmatrix} w & . & . & \rho & . \\ . & w & . & . & . \\ . & . & w & . & . \\ . & . & . & w & \frac{1}{\rho} \\ . & . & . & \rho c^2 & w \end{bmatrix}$$

where c is the local speed of sound. The flux Jacobians of the primitive variables are related to those of the conservative variables by a similarity transform:

$$A_p = M^{-1} A M \tag{A.13}$$

where the Jacobian of the transformation from conservative to primitive variables is defined by:

$$M = \frac{\partial U}{\partial U_p} = \begin{bmatrix} 1 & . & . & . & . \\ u & \rho & . & . & . \\ v & . & \rho & . & . \\ w & . & . & \rho & . \\ \frac{\vec{u}^2}{2} & \rho u & \rho v & \rho w & \frac{1}{(\gamma-1)} \end{bmatrix} \tag{A.14}$$

A.1.5 Characteristic-Variable Form / Compatibility Relations

An important property of the Euler equations is their ability to represent the propagation of waves. Mathematically, this is defined by the existence of characteristic or wave-front surfaces which separate regions of the flow which have been affected by a wave from those which have not. This behaviour is most clearly illustrated by casting the equations into their characteristic-variable form. For simplicity, we first consider the equations in one dimension, for which the primitive-variable form may be expressed:

$$\frac{\partial U_p}{\partial t} + A_p \frac{\partial U_p}{\partial x} = 0 \tag{A.15}$$

where:

$$A_p = \begin{bmatrix} u & \rho & . \\ . & u & \frac{1}{\rho} \\ . & \rho c^2 & u \end{bmatrix} \tag{A.16}$$

The flux Jacobian matrix, A_p, can be diagonalised by its matrix of right eigenvectors, defined by:

$$L = \begin{bmatrix} 1 & \frac{\rho}{2c} & \frac{\rho}{2c} \\ . & \frac{1}{2} & -\frac{1}{2} \\ . & \frac{\rho c}{2} & \frac{\rho c}{2} \end{bmatrix} \tag{A.17}$$

which has the inverse:

$$L^{-1} = \begin{bmatrix} 1 & . & -\frac{1}{c^2} \\ . & 1 & \frac{1}{\rho c} \\ . & -1 & \frac{1}{\rho c} \end{bmatrix} \tag{A.18}$$

to give the diagonal matrix of eigenvalues:

$$\Lambda = L^{-1} A_p L = \begin{bmatrix} u & . & . \\ . & u+c & . \\ . & . & u-c \end{bmatrix} \tag{A.19}$$

By pre-multiplying (A.15) with L^{-1}:

$$L^{-1}\frac{\partial U_p}{\partial t} + (L^{-1} A_p L) L^{-1} \frac{\partial U_p}{\partial x} = 0 \tag{A.20}$$

We write the one-dimensional Euler equations as:

$$L^{-1}\frac{\partial U_p}{\partial t} + \Lambda L^{-1} \frac{\partial U_p}{\partial x} = 0 \tag{A.21}$$

The system has been reduced to three partially decoupled non-linear convection equations, known as the compatibility relations. These may be expressed in

terms of characteristic or Riemann variables defined by:

$$\delta U^* \;=\; L^{-1}\delta U_p \;=\; \begin{bmatrix} \delta\rho - \frac{1}{c^2}\delta p \\ \delta u + \frac{1}{\rho c}\delta p \\ \delta u - \frac{1}{\rho c}\delta p \end{bmatrix} \tag{A.22}$$

So that:

$$\frac{\partial U^*}{\partial t} + \Lambda\frac{\partial U^*}{\partial x} \;=\; 0 \tag{A.23}$$

The first of the above compatibility relations describes the propagation of entropy waves with the flow speed, u. The second and third compatibility relations describe propagation of acoustic waves, the first travelling with a speed $u + c$, and the second with a speed $u - c$.

To derive compatibility relations for multi-dimensional flows, we first note that the A_x, A_y and A_z matrices cannot be simultaneously diagonalised. We therefore consider an arbitrary but specified direction of propagation $\vec{k}$, with a flux Jacobian defined by a linear combination of the A_x, A_y, and A_z matrices. Working with the primitive variables, we can write:

$$\vec{A_p}\cdot\vec{k} = A_{x_p}k_x + A_{y_p}k_y + A_{z_p}k_z \tag{A.24}$$

so that the compatibility relations are given by:

$$L_k^{-1}\frac{\partial U_p}{\partial t} + (L_k^{-1}\vec{A_p}L_k)L_k^{-1}\vec{\nabla}U_p \;=\; 0 \tag{A.25}$$

In this case, the inverse of the matrix of right eigenvectors of the combined flux Jacobian is given by:

$$L_k^{-1} = \begin{bmatrix} 1 & . & . & . & -\frac{1}{c^2} \\ . & -k_z & . & k_x & . \\ . & k_y & -k_x & . & . \\ . & k_x & k_y & k_z & \frac{1}{\rho c} \\ . & -k_x & -k_y & -k_z & \frac{1}{\rho c} \end{bmatrix} \tag{A.26}$$

and the characteristic variables are given by:

$$\delta U^* \;=\; L_k^{-1}\delta U_p \;=\; \begin{bmatrix} \delta\rho - \frac{1}{c^2}\delta p \\ -k_z\delta u + k_x\delta w \\ k_y\delta u - k_x\delta v \\ k_x\delta u + k_y\delta v + k_z\delta w + \frac{1}{\rho c}\delta p \\ -k_x\delta u - k_y\delta v - k_z\delta w + \frac{1}{\rho c}\delta p \end{bmatrix} \tag{A.27}$$

leading to the following compatibility relations:

$$\left(\frac{\partial}{\partial t} + \vec{u}\cdot\vec{\nabla}\right)U_1^* = 0 \tag{A.28}$$

$$\left(\frac{\partial}{\partial t} + \vec{u}\cdot\vec{\nabla}\right)U_2^* = \vec{1_y}\cdot\left(\vec{k}\times\nabla p/\rho\right) \tag{A.29}$$

$$\left(\frac{\partial}{\partial t} + \vec{u}\cdot\vec{\nabla}\right) U_3^* = \vec{1_z}\cdot\left(\vec{k}\times\nabla p/\rho\right) \tag{A.30}$$

$$\left(\frac{\partial}{\partial t} + [\vec{u} + c\vec{1_k}]\cdot\vec{\nabla}\right) U_4^* = -c\,\vec{l}\cdot(\vec{l}\cdot\vec{\nabla})\vec{u} - c\,\vec{m}\cdot(\vec{m}\cdot\vec{\nabla})\vec{u} \tag{A.31}$$

$$\left(\frac{\partial}{\partial t} + [\vec{u} - c\vec{1_k}]\cdot\vec{\nabla}\right) U_5^* = -c\,\vec{l}\cdot(\vec{l}\cdot\vec{\nabla})\vec{u} - c\,\vec{m}\cdot(\vec{m}\cdot\vec{\nabla})\vec{u} \tag{A.32}$$

where $\vec{l}$ and $\vec{m}$ are vectors perpendicular to $\vec{k}$. The first compatibility relation, which describes the convection of entropy with the flow, is unchanged from the one-dimensional case. The second and third compatibility relations describe the convection of vorticity with the flow. The last two relations describe the propagation of acoustic waves. Unlike the one-dimensional relations, however, the changes in the last four wave variables are now functions of the local gradients in pressure or velocity normal to the chosen direction of propagation.

Appendix B

Finite-volume methods

Both finite-difference and finite-element methods have been widely applied to fluid dynamic problems. Finite-difference methods are simple to design and analyse, but their application to complex geometries can be awkward, particularly when considering higher orders of accuracy on irregular or unstructured meshes. Finite-element methods deal easily with such issues, but can require more computational effort due to the cost of integration. For fluid dynamics problems there is a third approach, however, which retains some of the flexibility of finite-element methods but is relatively simple to implement, and relatively fast in execution. This approach, known as the finite-volume method, is currently the most popular for engineering fluid dynamics.

The finite-volume method is a natural approach to the discretisation of *conservation laws*, such as the mass, momentum and energy equations encountered in fluid dynamics. Finite-volume methods mimic these laws by *exactly*[1] conserving mass, momentum or energy, no matter what the size of the mesh. This property is known as *discrete conservation*, and has been shown to enhance both stability and accuracy, particularly when computing the dynamics of discontinuities (such as shock waves) on coarse meshes. When formulated in conservative variables, finite-element methods also have the discrete conservation property [28, 29], but finite-difference methods require special treatment to achieve it [39].

Although finite-volume methods were initially developed by physical analogy (which partly accounts for their popularity), their close connection to discrete conservation links them to a larger class of methods designed to exactly reproduce key mathematical properties of their underlying physical models. These are known as *mimetic methods* and are currently receiving considerable attention from the numerical analysis community [4, 8, 38].

[1] To machine precision

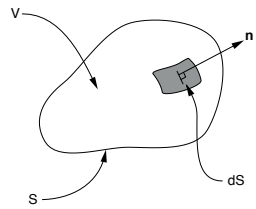


Figure B.1: A control volume of finite size V

B.1 Finite-volume methods for fluid dynamics

The starting point for the derivation of finite-volume methods is the integral form of the conservation laws. These can be derived by considering the conservation of mass, momentum and energy within a volume of finite size, such as that shown in figure B.1. For such a volume this results in:

$$\frac{\partial}{\partial t}\int_V W \; dV \;\; + \;\; \int_S \vec{F}\cdot\vec{n} \; dS = 0 \tag{B.1}$$

where W is known as the *state vector* defined by:

$$W = \begin{bmatrix} \rho \\ \rho\vec{u} \\ \rho E \end{bmatrix} \tag{B.2}$$

which contains the unknown conservative variables density (ρ), momentum ($\rho\vec{u}$), and total specific energy (ρE). $\vec{F}$ is the vector containing the fluxes of these quantities through the boundaries of the volume. Basically, equation (B.1) can be read as:

The time rate of change of a quantity within the control volume is equal and opposite to the flux of the quantity out of the control volume

For the equations of fluid flow, in the mass conservation equation $\vec{F}$ is simply the conserved variable ρ times the local velocity vector. In the momentum and energy equations, $\vec{F}$ also includes pressure and viscous terms.

A finite-volume method is obtained by applying (B.1) directly to the finite-sized cells which make up the computational mesh. Referring to figure B.2, if we define our unknowns, $\tilde{W}_{ij}$,as the volume average of W in each cell:

$$\tilde{W}_{ij} = \frac{1}{V_{ij}}\int_{V_{ij}} W \; dV_{ij} \tag{B.3}$$

we can apply (B.1) to each cell to obtain the following set of semi-discretised

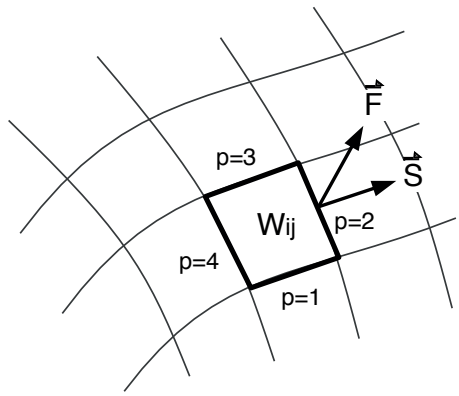


Figure B.2: Cell-centred finite volume discretisation

equations:

$$\frac{\partial}{\partial t}\tilde{W}_{ij}V_{ij} + \sum_{p=1}^{P} \vec{F}_p \cdot \vec{S}_p = 0 \tag{B.4}$$

where P is the number of faces which define the cell, (four in figure B.2), and $\vec{S}_p$ is the area vector of a face:

$$\vec{S}_p = \vec{n}_p \cdot \mid S_p \mid \tag{B.5}$$

where $\vec{n}$ is the face normal vector and $\mid S_p \mid$ is the face area. In general $\vec{F}_p$ can be a function of several of the $\tilde{W}_{ij}$ values in the vicinity of the face p. If we consider a face perpendicular to the i-direction of a structured mesh, for example (figure B.3), a typical form for $\vec{F}_p$ would be:

$$\vec{F}_p = \vec{F}_p(w_{i-m,j}, ...w_{i,j}, ...w_{i+m,j}) \tag{B.6}$$

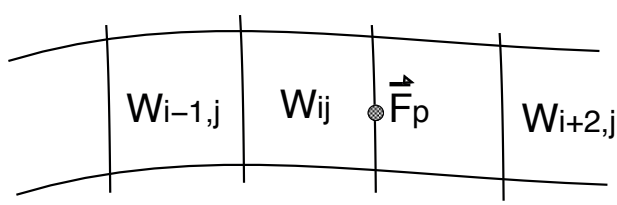


Figure B.3: Flux evaluation on a structured mesh

Different finite-volume methods can be categorised in terms of the choice of control volume and the method used for flux evaluation, as described in the following sections.

B.1.1 Choice of control volume

In order to maintain the discrete conservation property, the choice of control volume for the evaluation of $\tilde{W}_{ij}$ is subject to the following conditions:

- Each volume must be closed ($\int \vec{n} \cdot dS = 0$)
- The value of flux, $\vec{F}_p$, evaluated at any surface must be independent of the volume under consideration
- The volumes must cover the entire domain
- Any overlapping face must be common to two control volumes

There remain, however, a number of ways to define a control volume. In addition to the *cell-centred* discretisation of figure B.2, it is possible to define *vertex-centred* methods, such as that shown in figure B.4, where cells surrounding the vertices are used to define control volumes. Volume definitions which do not follow the grid lines are also possible (figure B.5). Finally, it is also easy to define control volumes on unstructured meshes. These may be either cell-centred or vertex centred, as shown in figure B.6.

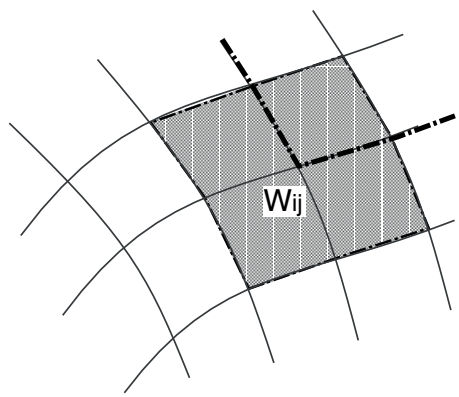


Figure B.4: Vertex-centred finite-volume discretisation

B.1.2 Choice of flux evaluation

To complete a finite-volume discretisation we need to express the surface fluxes $\vec{F}$ as a function of the unknown volume averaged values, $\tilde{W}_{ij}$. The simplest approach is to use the average the $\tilde{W}_{ij}$ values on either side of the face to compute $\vec{F}$. This leads to a scheme similar to central differencing, and typically requires the addition of artificial dissipation to the flux definition to prevent oscillations in the presence of advection. Alternatively, the fluxes can be based on values of $\tilde{W}_{ij}$ taken from one side of the face, corresponding to an upwind

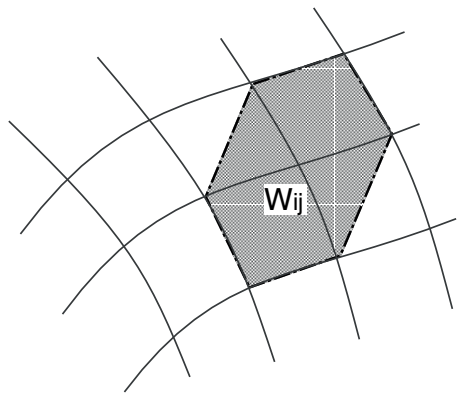


Figure B.5: Mc Donald finite-volume discretisation

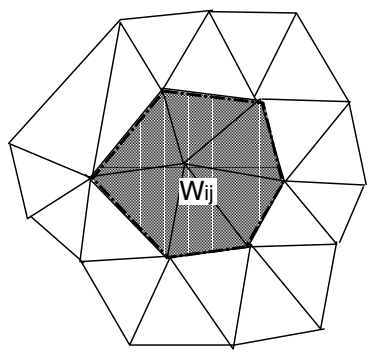


Figure B.6: Unstructured vertex-centred finite-volume discretisation

scheme. This is a natural approach when the propagation direction of solution information clear. Upwinding is effective if only first-order accuracy is desired, but care must be taken in the design of higher-order schemes, particularly on unstructured meshes. This topic has received considerable attention from those in the field of method design [30]. In any case, provided that the definition of flux on given face is unique, the finite-volume method will retain the discrete conservation property, for any flux evaluation method.

Appendix C

Spectral and finite-element methods

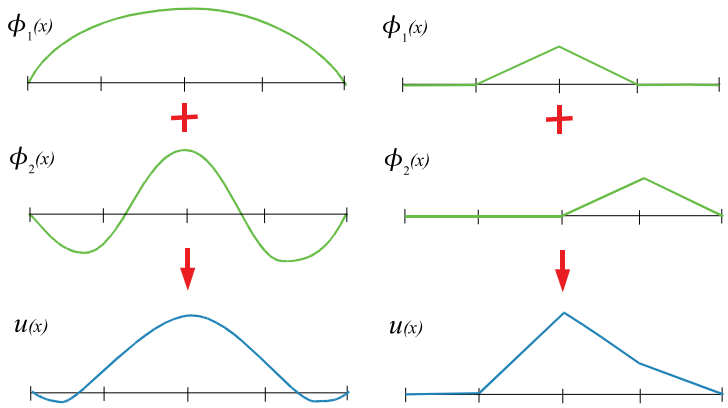
C.1 Approximating the solution with functions

In contrast to approximating the solution using point values and finite-difference expressions for the required derivatives, we now seek to approximate the solution using a combination of pre-specified basis functions. For a one-dimensional problem, for example, we could approximate the solution using a combination of global functions, which span the entire problem domain, as shown in figure C.1(a). Alternatively, we could choose local functions (those which are zero outside of specific regions), as shown in C.1(b). In both cases, we could write the approximate solution as:

$$\hat{u}(x) = \sum_{i=1}^{N} a_i \phi_i(x) \tag{C.1}$$

where $\hat{u}$ indicates the approximation for the exact solution u, N is the number of basis functions used, and the a_i are the amplitudes of the basis functions, $\phi_i(x)$. With this approach, the a_i (often called *degrees of freedom*), are the discrete unknowns, and are determined by the numerical solution method. Methods which use global basis functions are usually referred to as *spectral methods*. The use of local basis functions, on the other hand, is the key concept behind *finite-element methods*.

Once the functions are chosen, there are a number of approaches to determining their amplitudes. One approach is to define an equivalent problem which considers the minimisation of some scalar measure of the solution, $I[u]$. In structural problems, for example, this could be the total potential energy. $I[u]$ is called a *functional*, since it is itself a function of a function (in this case the function which defines $u(x)$). If $I[u]$ is to be minimised (or maximised), then a system of equations for a_i can be generated by substituting (C.1) into



(a) Global functions (b) Local functions

Figure C.1: Approximation using global and local functions

the definition of $I[u]$, and then requiring the derivatives of $I[u]$ with respect to the unknown a_i to be zero:

$$\frac{\partial I[u]}{\partial a_i} = 0, \qquad i = 1, ..., N \tag{C.2}$$

This is known as the Rayleigh-Ritz approach, and has its roots in the *calculus of variations*, the branch of mathematics concerned with finding the extrema of functionals.

C.2 The method of weighted residuals

In certain problems, it is difficult to define a functional which allows application of the Rayleigh-Ritz method. It is still possible to construct procedures for finding the a_i, however, using an approach known as the *method of weighted residuals* (MWR). This produces a weak form of the governing equations, which can be used to build a system of equations for a_i. Since the MWR is more generally applicable than the Rayleigh-Ritz approach, we will employ it for all of the examples considered in the following sections. For a broader view on this subject, the interested reader is encouraged to read more extensive texts, such as [**?**, **?**, **?**].

To make our description more concrete, we will make use of the following sample problem on the 1D domain Ω spanning the region from $x = 0$ to $x = 1$:

$$u_{xx} = f \qquad \text{on} \quad \Omega \tag{C.3}$$

$$u(0) = g \tag{C.4}$$

$$u_x(1) = q \tag{C.5}$$

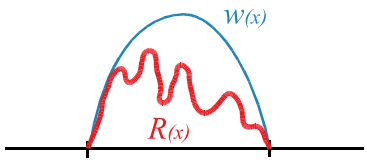


Figure C.2: Testing the residual in a small region

Equation (C.3) is referred to as the *strong form* of the governing equation. Equations (C.4) and (C.5) define Dirichlet and Neumann boundary conditions, respectively.

It is possible to define another form of the governing equation, known as the *weak form*, using the method of weighted residuals. This requires the weighted integrals of the governing equation's *residual*, $R(x) = u_{xx} - f$ to be zero for all suitable weighting functions, w:

$$\int_0^1 w(u_{xx} - f)dx = 0 \tag{C.6}$$

By suitable, we mean all weighting functions which allow the integral in (C.6) to be meaningful (we shall elaborate on this shortly).

It should be mentioned that equivalent weak forms of the governing equations can sometimes be derived by approaches which consider the minimisation of a functional. As a result, the weak form is often called the *variational form*, even when it has been derived using the MWR.

We will describe how the weak form can be used to generate a system of equations for the unknown a_i in section C.4. Before doing so, we will demonstrate that we do not loose anything by shifting our focus to the weak form.

C.3 Equivalence of the strong and weak forms

Spectral and finite-element methods work by solving the weak form of the governing equation rather than the strong form. It is thus important to confirm that the weak form has the same exact solution as the strong form. Clearly, the residual of the exact solution to the strong form, u, is zero, meaning that it will also satisfy the weak form. However, is it possible for the weak form to be satisfied by other solutions?

To visualise why this cannot be true [1], consider a trial solution which produces a positive residual over some small region of the domain (figure C.2). Since we are to express (C.6) for all suitable weighting functions, we must include a weighting function which is purely positive over this region, and zero everywhere else. Using such a weighting function would result in a non-zero

[1] More formal proofs of the equivalence of the strong and weak forms can be found in [**?**] and [**?**].

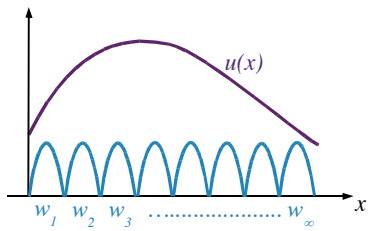


Figure C.3: An exact solution can be obtained using an infinite number of tests of the weak form

result for the integral in (C.6), meaning that such a solution could not be a solution of the weak form. Note that the application of the weighted integral, as in the case of figure C.2, provides essentially a "test" of the local residual. For this reason, weighting functions are also referred to as *test functions*.

C.4 Solving for a_i; the Galerkin method

The weak form produces the exact continuous solution when all suitable weighting functions are considered, in other words an infinite-dimensional set of functions (figure C.3). We can also express any exact solution, u, using (C.1) with $N = \infty$ and a group of functions ϕ_i, from a similar infinite-dimensional set.

In our case, however, we wish to use finite-dimensional approximations for u, i.e. (C.1) with finite N. We therefore need only to test the weak form using a finite-dimensional set of weighting functions w_i. Specifically, for a solution interpolation defined by (C.1), no more than N weighting functions need be used, resulting in N equations similar to (C.6) which can be used to solve for the unknown a_i. Solving for the mode amplitudes in this way is known as the *Galerkin method* [2].

The most popular Galerkin method is to choose the weighting functions to be equivalent to the solution basis functions, i.e. $w_i = \phi_i(x)$ (figure C.4). This is formally known as the *Bubnov-Galerkin method*, although the Bubnov is often omitted. Bubnov-Galerkin are popular as they maximise the reuse of information, and produce symmetric matrices for certain classes of problems. All the examples given in these notes will be based on the Bubnov-Galerkin method, but the extension to Petrov-Galerkin methods (those which use weighting functions which are different from the solution expansion functions) is straightforward.

[2] The Galerkin method can be also related to an older approach based on the MWR. If we test (C.6) using delta functions as weighting functions the integration is bypassed, which results in a system which directly enforces a zero residual at a set of points. This approach is known as the collocation method [**?**].

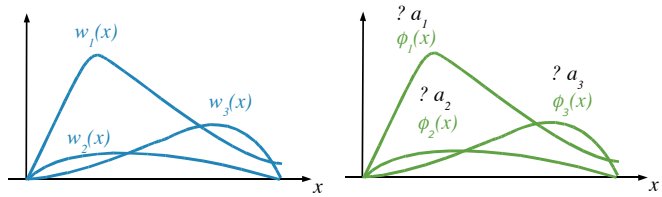


Figure C.4: In the Bubnov-Galerkin method, the weighting functions w_i are chosen to be the same as those used to expand the solution, ϕ_i

C.5 Suitable choices for w and u

As we shall see later on, the local piecewise linear functions shown in figure C.1(b) can provide both accuracy and flexibility. If we would like to use them for our sample problem, however, we are faced with the issue that the evaluation of $\hat{u}_{xx}$ would result in zero, except in the kinks of the curves where it would be ∞. This means that if (C.6) is used, we would be forced to reject piecewise linear functions in favour of those with higher continuity. This issue can be circumvented, however, by re-expressing (C.6) using integration by parts:

$$wu_x\Big|_0^1 - \int_0^1 w_x u_x dx - \int_0^1 wf dx = 0 \tag{C.7}$$

This formulation only requires us to evaluate first derivatives, which are well defined for the functions in C.1(b). This form is also better suited for a Bubnov-Galerkin discretisation, as the continuity requirements for the weighting functions and solution basis functions are similar. Note that it also isolates the boundary value of u_x which allows it it be used to enforce the application of Neumann boundary conditions. In this sense, Neumann boundary conditions are natural for this form.

On the other hand, this particular weak form does not allow the application of the Dirichlet boundary condition (C.4). It will have to be applied using an equation which sets the value of the mode amplitudes on the boundary. Such additional equations are called essential boundary conditions. If we already have N equations by testing the weak form with N weighting functions, then adding equations for E essential boundary conditions will result in more equations $(N + E)$ than unknowns (N). Generally the best solution is to to remove E of the weighting functions which are non-zero on the Dirichlet boundary from consideration so that a system of N equations is finally obtained.

At this point we are ready to complete our definition of a "suitable" function. A basic requirement for the method to work is that the weak form remains integrable. This is not a trivial requirement. For example, if $u_x = \frac{1}{x^2}$, and w is chosen as x, the second definite integral in (C.7) would evaluate to ∞. For a Bubnov-Galerkin discretisation of (C.7), we can define the set of suitable

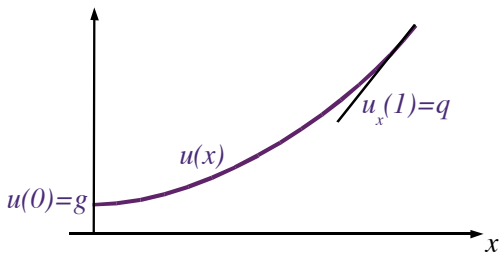


Figure C.5: Solution of the sample problem

functions as those which have square-integrable derivatives, defined as:

$$\int_0^1 (\phi_x)^2 dx \ < \ \infty \tag{C.8}$$

Note that square integrability of the derivative can be a stronger restriction than integrability of the function itself.

C.6 An example spectral method

We will now proceed to find an approximate solution to the sample problem of section C.2, which has the form illustrated in figure C.5. We will use a spectral method, that is, one employing basis functions defined over the complete domain. The basis functions will be chosen from the "modal p-type expansion" defined in [**?**]:

$$\begin{aligned} \phi_1(x) &= 1 - x \\ \phi_2(x) &= x \\ \phi_3(x) &= x - x^2 \\ \phi_4(x) &= -2x + 6x^2 - 4x^3 \end{aligned}$$

These functions allow the interpolation of polynomial solutions of up to third order, and are illustrated in figure C.6. We will express our approximate solution as:

$$\hat{u}(x) = \sum_{i=1}^{4} a_i \phi_i(x) \tag{C.9}$$

and employ a Bubnov-Galerkin method for which $w_i = \phi_i$.

C.6.1 The basic system

We can obtain a system for the four unknowns in the problem by testing the integrated-by-parts weak form of the equations (C.7) with $w_{1\to4}$. Noting that

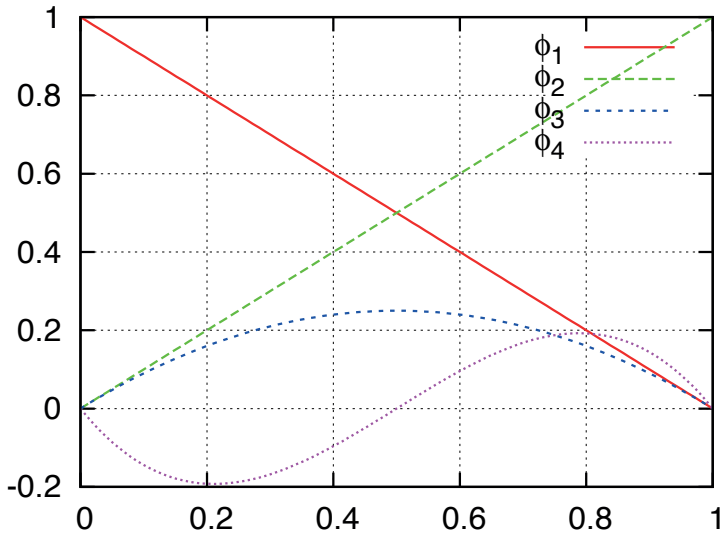


Figure C.6: Modal basis functions

only $w_1 = \phi_1$ is non-zero at $x = 0$, and only $w_2 = \phi_2$ is non-zero at $x = 1$, this results in the following basic system of equations:

$$w_1(0)u_x(0) - \int_0^1 w_{1_x} \sum_{i=1}^{N} a_i \phi_{i_x} dx = \int_0^1 w_1 f dx \tag{C.10}$$

$$w_2(1)u_x(1) - \int_0^1 w_{2_x} \sum_{i=1}^{N} a_i \phi_{i_x} dx = \int_0^1 w_2 f dx \tag{C.11}$$

$$- \int_0^1 w_{3_x} \sum_{i=1}^{N} a_i \phi_{i_x} dx = \int_0^1 w_3 f dx \tag{C.12}$$

$$- \int_0^1 w_{4_x} \sum_{i=1}^{N} a_i \phi_{i_x} dx = \int_0^1 w_4 f dx \tag{C.13}$$

C.6.2 Dirichlet boundary condition

We can start by applying the Dirichlet boundary condition at $x = 0$. Examining the solution basis functions, we see they are all zero at $x = 0$ except for ϕ_1, which has a value of 1. Expressing the Dirichlet boundary condition using (C.9) we obtain the essential condition $a_1 = g$. Based on the discussion in the previous section, the test with $w_1 = \phi_1$ should be removed from consideration as w_1 is the only weighting function that is non-zero on the the Dirichlet boundary. As we already know a_1, however, and have only three remaining unknowns (a_2, a_3

and a_4), a test with w_1 will not be required. We then have:

$$a_1 = g \tag{C.14}$$

$$w_2(1)u_x(1) - \int_0^1 w_{2_x} \sum_{i=1}^{N} a_i \phi_{i_x} dx = \int_0^1 w_2 f dx \tag{C.15}$$

$$- \int_0^1 w_{3_x} \sum_{i=1}^{N} a_i \phi_{i_x} dx = \int_0^1 w_3 f dx \tag{C.16}$$

$$- \int_0^1 w_{4_x} \sum_{i=1}^{N} a_i \phi_{i_x} dx = \int_0^1 w_4 f dx \tag{C.17}$$

C.6.3 Neumann boundary condition

The Neumann condition is natural for this weak form, so we can apply it directly by noting $u_x = q$ at $x = 1$. We then have

$$a_1 = g \tag{C.18}$$

$$w_2(1)q - \int_0^1 w_{2_x} \sum_{i=1}^{N} a_i \phi_{i_x} dx = \int_0^1 w_2 f dx \tag{C.19}$$

$$- \int_0^1 w_{3_x} \sum_{i=1}^{N} a_i \phi_{i_x} dx = \int_0^1 w_3 f dx \tag{C.20}$$

$$- \int_0^1 w_{4_x} \sum_{i=1}^{N} a_i \phi_{i_x} dx = \int_0^1 w_4 f dx \tag{C.21}$$

C.6.4 The final system

We now complete the system of equations by substituting the assumed solution (C.9) into the system and noting for a Bubnov-Galerkin method $w_i = \phi_i$.

$$\begin{aligned} a_1 &= g \\ \phi_2(1)q - \int_0^1 \phi_{2_x} \left(a_1\phi_{1_x} + a_2\phi_{2_x} + a_3\phi_{3_x} + a_4\phi_{4_x}\right) dx &= \int_0^1 \phi_2 f dx \\ - \int_0^1 \phi_{3_x} \left(a_1\phi_{1_x} + a_2\phi_{2_x} + a_3\phi_{3_x} + a_4\phi_{4_x}\right) dx &= \int_0^1 \phi_3 f dx \\ - \int_0^1 \phi_{4_x} \left(a_1\phi_{1_x} + a_2\phi_{2_x} + a_3\phi_{3_x} + a_4\phi_{4_x}\right) dx &= \int_0^1 \phi_4 f dx \end{aligned}$$

The above can be expressed as a matrix equation for the unknowns a_i as:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ K_{21} & K_{22} & K_{23} & K_{24} \\ K_{31} & K_{32} & K_{33} & K_{34} \\ K_{41} & K_{42} & K_{43} & K_{44} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} = \begin{bmatrix} g \\ F_2 \\ F_3 \\ F_4 \end{bmatrix} \tag{C.22}$$

where, for example, all terms like:

$$K_{23} = -\int_0^1 [\ \phi_{2_x}(x)\ \phi_{3_x}(x)\ ]\ dx \tag{C.23}$$

$$F_2 = \int_0^1 [\ \phi_2(x)\ f(x)\ ]\ dx - \phi_2(1)q \tag{C.24}$$

can be evaluated using known data. With the exception of the first row, which contains an essential condition, each row in the K matrix corresponds to a test of the weak form with a single weighting function. K is often referred to as the *stiffness matrix*, due to the role it plays in structural problems [3].

For the system to have a solution, the $\phi_i(x)$ must be linearly independent. This means that it is not possible to express one $\phi_i(x)$ as a linear combination of the others. The definition of K_{23} given by (C.23) also demonstrates that if one has an orthogonal basis, i.e. one for which:

$$\int_\Omega [\ \phi_i(x)\ \phi_j(x)\ ]\ d\Omega = 0; \qquad i \neq j \tag{C.25}$$

the resulting K matrix will be diagonal, and trivial to invert. When non-orthogonal bases are used, those which are closer to being orthogonal will tend to produce K matrices with increased diagonal dominance.

C.6.5 Observations and results

Spectral solutions to the sample problem with $g = 0, q = \frac{1}{2}$ and $f = x$ obtained using increasing numbers of basis functions are shown in figure C.7. In this case the exact solution is given by $u(x) = g + qx + \frac{1}{6}(x^3 - 3x)$, shown using diamond symbols in the plot. The spectral solution converges quickly, and becomes exact when $\phi_2(x), \phi_3(x)$ and $\phi_4(x)$ are used. This occurs since once $\phi_4(x)$ is added, the solution function space is rich enough to represent the cubic variation of the exact solution without error. This phenomena is known as *superconvergence*.

C.7 An example finite-element method

When a domain has a complex shape, it can be difficult to define global functions appropriate for a spectral method. Furthermore, many realistic problems include sudden local changes in the boundary conditions, or in the governing equations themselves. To deal with these issues it is easiest to use basis functions which are only non-zero on limited parts of the domain. This is the concept behind the finite-element method. The basic procedure can be summarised as:

[3] Note that in the construction of K there is an analogy with the finite-difference example considered in section **??**. In that case, to apply a Dirichlet boundary condition the first equation corresponding to the PDE evaluated at the boundary point was removed and replaced with an expression specifying the solution at the point. Thus the "test" of the PDE was not performed at the boundary, since it would lead to redundant information. The analogy does not hold for the method used to apply the Neumann boundary condition, however.

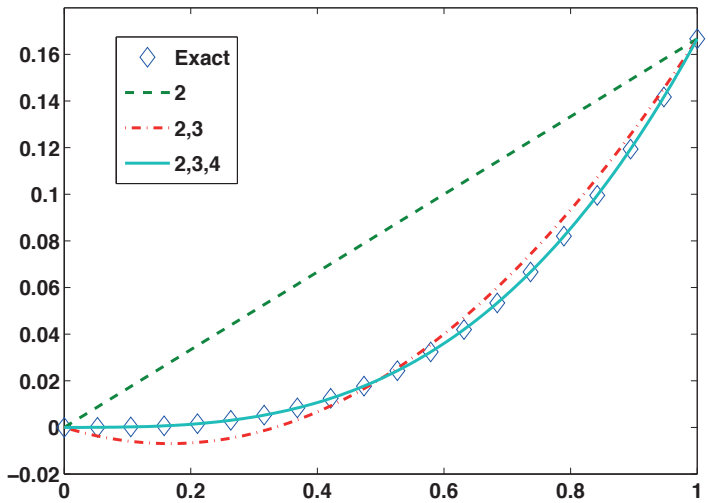


Figure C.7: Spectral solution using an increasing number of basis functions from the modal p-type expansion

1. Divide the domain into elements
2. Define basis functions which span a small number of elements
3. Compute per-element contributions to the weak form
4. Assemble the element contributions into a global matrix
5. Apply the boundary conditions and solve the resulting system

We will consider each of these steps in turn by solving our sample problem.

Step 1

We begin by splitting the domain into three elements. These are chosen to have an equal width of $h = 1/3$, although unequal element sizes could also be used. For complex domains, division into elements is normally done using dedicated grid-generation software.

Step 2

We define four basis functions, from the group of local piecewise linear functions in figure C.1(b). For this one-dimensional problem, we choose to have the functions span two elements, with the exception of $\phi_1(x)$ and $\phi_4(x)$, which are truncated and exist only in the boundary elements (figure C.8).

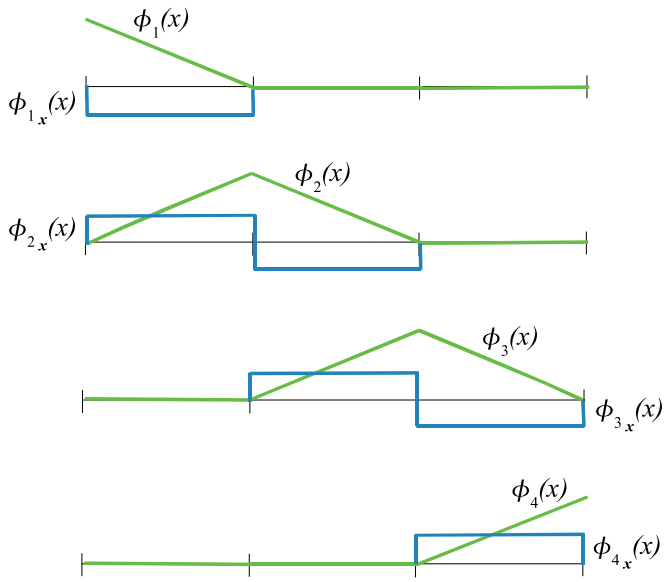


Figure C.8: Basis functions (green) for a three-element discretisation of the sample problem. The derivatives of the basis functions are shown in red.

Step 3

Since grid generation codes produce a representation of the problem domain in terms of a group of elements, it makes sense to assemble the system of equations by looping over elements and computing their individual contributions to the K and F matrices. This of course can be done by evaluating the integrals in physical coordinates on an element-by-element basis.

However, most large finite-element codes use a limited number of element types (e.g. triangle, hexahedron, etc..) to which all elements in a mesh can be topologically related. Since some quantities remain fixed for a given element type, it makes sense to consider a master element in Cartesian coordinates and relate individual elements to the master element using a coordinate transform.

Returning to our one-dimensional problem, each of the elements in the mesh can be treated generically if we consider local versions of the basis functions defined on the master element shown in figure C.9. In the figure ϕ_L indicates the portion of the basis function centred on the left which is contained in the element, while ϕ_R indicates the portion of the basis function centred on the right. For the case shown, the master element starts at $\xi = 0$ and ends at $\xi = 1$. When we compute the contribution from a given element in the mesh, it will have some arbitrary width h. Therefore we need to evaluate the integral in a way that accounts for this. If the element in physical space has a length h and starts at x_1, then $\xi = \frac{(x-x_1)}{h}$. x derivatives can then be obtained from the chain

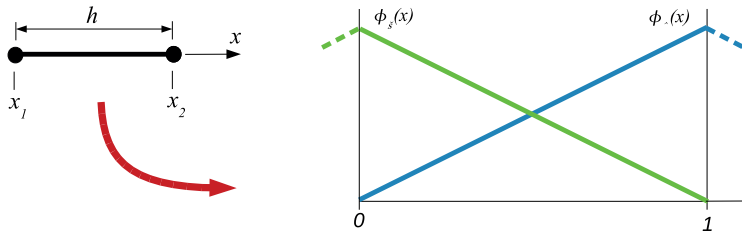


Figure C.9: Physical element (left) and master element (right) with two basis functions

rule, so that $\phi_x = \phi_\xi \xi_x$, where $\xi_x = \frac{1}{h}$. To change integration coordinates, we note that given the physical length is h and the transformed length is 1, so that $dx = hd\xi$. The element matrix, K_e, and vector, F_e are then given by:

$$Ke = \begin{bmatrix} \frac{1}{h}\int_0^1 \phi_{L_\xi}(\xi)\,\phi_{L_\xi}(\xi)\,d\xi & \frac{1}{h}\int_0^1 \phi_{L_\xi}(\xi)\,\phi_{R_\xi}(\xi)\,d\xi \\ \frac{1}{h}\int_0^1 \phi_{R_\xi}(\xi)\,\phi_{L_\xi}(\xi)\,d\xi & \frac{1}{h}\int_0^1 \phi_{R_\xi}(\xi)\,\phi_{R_\xi}(\xi)\,d\xi \end{bmatrix} \tag{C.26}$$

$$Fe = \begin{bmatrix} -h\int_0^1 \phi_L(\xi)\,f(\xi)\,d\xi \\ -h\int_0^1 \phi_R(\xi)\,f(\xi)\,d\xi \end{bmatrix} \tag{C.27}$$

In simple cases the integrals which define the element matrices and vectors can be evaluated analytically. In practice, however, quadrature is often used, i.e.:

$$\int_\Omega f(x)\,dx \approx \sum_{ip=1}^{N_{ip}} k_{ip}\,f(x_{ip}) \tag{C.28}$$

where N_{ip} is the number of quadrature (integration) points used in the interval and k_{ip} is the weight associated with a given quadrature point. Using quadrature reduces implementation complexity and increases flexibility, allowing, for example, non-analytic input data to be used. The number and placement of quadrature points can be chosen to exactly integrate polynomials up to a given order.

Step 4

The next step is to add the per-element contributions to the global matrix and vector. Each row in the global system corresponds to a test with a certain weighting function ϕ_i. Since ϕ_i normally spans more than one element, more than one element will contribute to a given row. On the other hand, a given element will be normally touched by more than one test function, so integration of the weak forms active within an element will result in a local 'element" matrix and vector. Constructing the global matrix and vector by adding the contributions of the element matrices and vectors is called *assembly*.

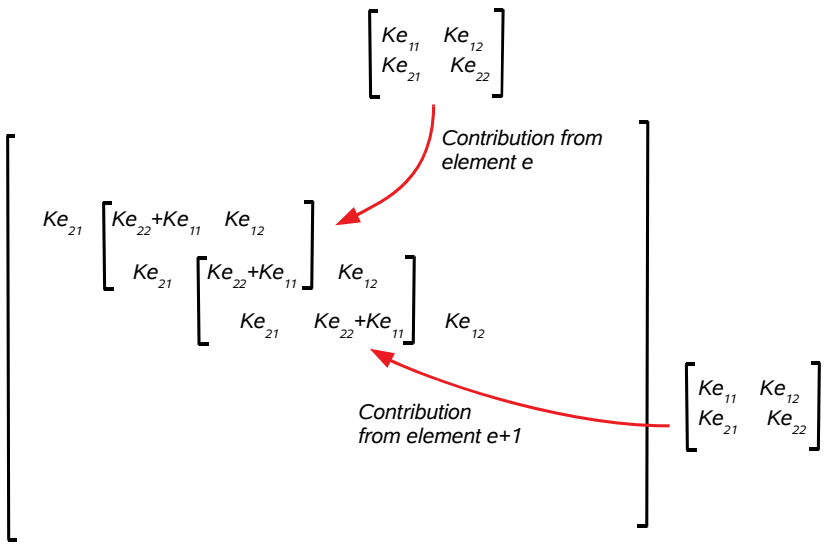


Figure C.10: Overlapping element arrays in global matrix assembly

Consider element 2. Its ϕ_L is actually ϕ_2 and its ϕ_R is ϕ_3. The first row of its element matrix thus corresponds to row 2 of the global matrix and its first column to column 2. It also makes contributions to row 3. In the next element (element 3), ϕ_L is actually ϕ_3 and ϕ_R is ϕ_4. Element 3 thus accounts for the the second half of the equation associated with the weighting function ϕ_3. The result is that the element matrices must be added to the global matrix in an overlapped fashion, as shown in figure C.10. A similar overlapping occurs in the right-hand side vector.

Since we chose our functions to span only two elements, using many elements will result in a matrix which is sparse and banded. In other words, it will be filled mostly with zeros except along the diagonal, and in row-column combinations corresponding to adjacent elements. Sparse matrices can be solved very efficiently using iterative or specialised direct techniques. For the PDE considered the stiffness matrix will also be symmetric. This is a consequence of our choice of the integrated-by-parts weak form combined with a Bubnov-Galerkin method, which leads to symmetry between the weighting functions and the solution basis functions. Symmetric matrices are generally easier to solve than unsymmetric ones.

Step 5

Using the same arguments as in the previous section, $a_1 = g$, and w_1 is redundant. We therefore eliminate the first row of the K matrix, and then eliminate the first column by inserting the known value for a_1 in each equation and

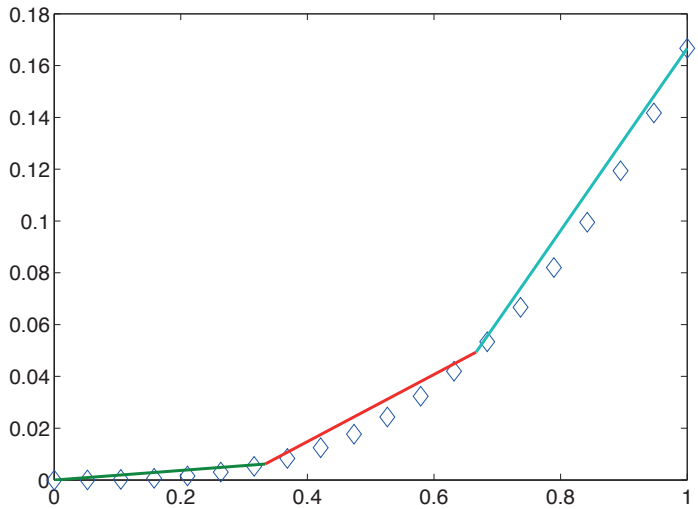


Figure C.11: Finite-element solution of the sample problem using three elements

transferring the result to the right-hand side. Finally, we need to consider the Neumann boundary condition at $x = 1$. This can be done by adding $\phi_4(1)q$ to F_4. We can then solve the remaining system.

Results

The results for a three-element discretisation of our sample problem with $g = 0, q = \frac{1}{2}$ and $f = x$ is shown in figure C.11. Remarkably, the solution is nodally exact. This type of superconvergence occurs due to the combination of piecewise linear basis functions with this particular weak form (see [**?**] for details). Convergence rates for finite-element discretisations of more general problems are discussed in the next section.

Contrast with the spectral approach

Note that the global matrix obtained in the finite-element test problem could have been obtained by using a spectral method with functions which are zero everywhere except in the vicinity of two elements, as shown in the upper two plots of figure C.1 (b). However, performing integration element-wise, followed by global assembly, has certain advantages. If a spectral approach is used, every integral requires evaluation over the complete domain, which is generally expensive when quadrature is used. Secondly, in a parallel computing environment, all functions and unknowns need to be present on all processors. thirdly, the

boundary terms would have to be retained on all functions. In contrast, with a finite-element method integration is only performed locally. Furthermore, information exchanged across processors can be limited to that associated with the elements touching the inter-processor boundaries. Finally, only the elements adjacent to the physical boundaries need to be involved in the application of boundary conditions.

C.8 Convergence rates

For problems where superconvergence does not occur, the convergence rate of spectral and finite-element discretisations is normally given by:

$$|e| = \left(\int_\Omega (u - \hat{u})^2 \, d\Omega \right)^{1/2} \leq Ch^p \tag{C.29}$$

where C is a constant determined by the discretisation and problem data, h is a measure of the domain (spectral) or element (FEM) size, and p depends on the basis functions chosen for the discretisation.

For the piecewise-linear functions used in the previous section, normally $p = 2$. For the spectral method of section C.6, the value of p is typically the order of the highest polynomial in the basis plus one.

It is possible to combine the spectral and finite-element methods, for example by using the basis (C.9) within each element of a group of elements which span the domain. Such *spectral-element methods* also converge with a rate given by (C.29). They can be refined either by introducing more elements (h refinement) or adding more basis functions (p refinement).

Since (C.29) is exponential in p, p refinement can be more effective than h refinement in spectral-element methods, especially when the solution is "smooth" (changes gradually between elements). When the solution is "rough", (changes rapidly between elements), h refinement tends to be more effective. h refinement may also be less expensive for a given number of degrees of freedom, as p refinement can lead to a reduction in conditioning of the assembled matrices, making them more expensive to solve (see [**?**] for details).

C.9 FEM in multiple dimensions

The finite-element method in multiple dimensions works with the same steps as in the one-dimensional case, but with some additional procedures associated with the division of the domain into elements, and the evaluation of integrals and gradients on elements of varying geometry. In this section we discuss the main issues associated with division of the domain into elements. Appendix E describes two approaches to dealing with arbitrary element geometries in multiple dimensions.

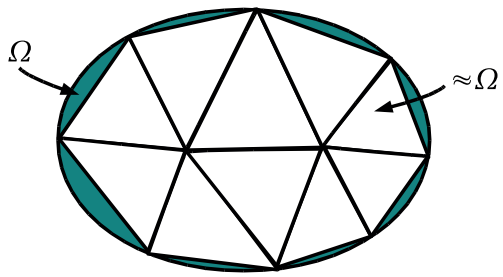


Figure C.12: Approximate domain implied by straight element edges

C.9.1 Division of the domain into elements

Classical discretisation methods such as the finite-difference method consider the boundary of the domain to be defined by a sequence of points. This provides no definition of the functional form of the boundary, and thus many grid generation packages represent the boundaries of the domain using a series of straight-line segments. Consequently, it is common for finite-element methods to use elements with edges defined by straight-line segments, resulting in approximations of the physical domain, as shown in figure C.12.

However, the finite-element method is not inherently limited to this representation. Curved element boundaries, typically expressed in terms of polynomials, can also been used. These are normally combined with higher-order representations of the interior solution, although this is not strictly necessary. For problems which are sensitive to boundary geometry (e.g. the flow around a curved surface) large increases in solution accuracy have been observed simply by improving the representation of the domain boundary. An example is shown in figure C.13, where the piecewise-linear geometry representation on the left shows artificial jumps in the isodensity contours as the boundary is approached.

Recently it was realised that the limitations associated with current grid-generation packages could be bypassed if finite-element programs worked directly with the interpolation functions used in the initial computer-aided design (CAD) process [**?**]. These are typically non-uniform rational B-splines (NURBS). Employing them directly with a finite-element method allows an exact representation of the boundary, independent of the number of elements used to discretise the domain (figure C.14). Having an exact boundary definition accessible also makes adaptation easier, as the exact coordinates of new boundary nodes are easily generated. A related approach is to use standard polynomial approximations for the solution, but an exact NURBS representation for the boundary. This ensures that the polynomial order of the interpolation is maintained in physical space, so that the optimal order of convergence is obtained [**?**, **?**].

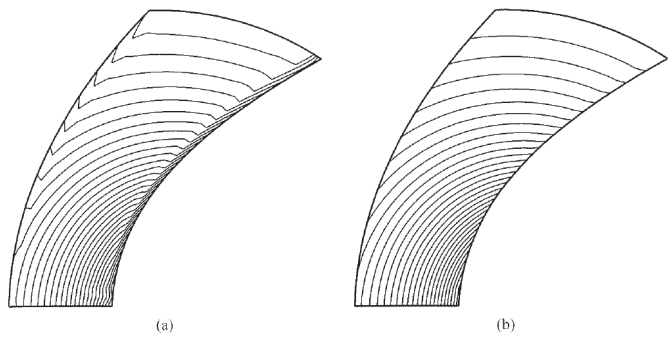


Figure C.13: Isodensity contours of a Ringleb flow. (a) discretised with piecewise linear solution and geometry representations, (b) discretised with piecewise linear solution and quadratic geometry representations (from [**?**]).

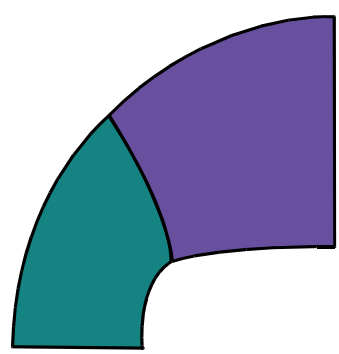


Figure C.14: A two-element discretisation of a curved domain (from [**?**])

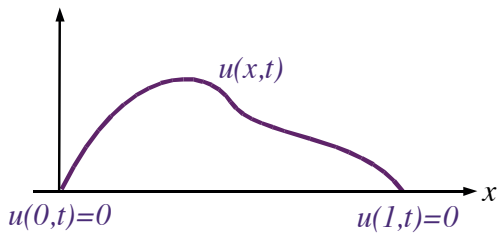


Figure C.15: Solution of the unsteady sample problem at a time t

C.10 Unsteady problems

There are two common approaches to discretising unsteady problems using spectral and finite-element methods. In the first, known as the semi-discrete approach, the time derivatives are initially treated as additional unknowns and the spectral or finite-element method is only used to discretise the problem in space. The time derivatives are then discretised using any appropriate method, including those based on finite-difference approximations. The second approach to unsteady problems is known as the fully-discrete approach, in which both time and space are discretised using the same spectral or finite-element method. Both approaches are described in the following sections in the context of the problem below:

$$u_t - u_{xx} = 0 \qquad \text{on} \quad \Omega \tag{C.30}$$

$$u(0,t) = 0 \tag{C.31}$$

$$u(1,t) = 0 \tag{C.32}$$

which has a solution with the form illustrated in figure C.15.

C.10.1 Semi-discrete approach

In the semi-discrete approach, the time derivatives are initially left as continuous variables in the formulation. This results in a system of ordinary differential equations which can be integrated using conventional time-marching techniques. Consider the discretisation of our sample unsteady problem. We begin by assuming:

$$\hat{u}(x,t) = \sum_{i=1}^{N} a_i(t)\ \phi_i(x) \tag{C.33}$$

and then derive a weak form of the PDE by considering the MWR applied in the spatial domain:

$$\int_\Omega w\ u_t\ d\Omega\ +\ \int_\Omega w_x\ u_x\ d\Omega = 0 \tag{C.34}$$

where we have used the fact that $w = 0$ on both Dirichlet boundaries as specified by (C.31) and (C.32). (C.33) can be substituted into (C.34), to obtain a system of the form:

$$M_{ij} \frac{da_j}{dt} + K_{ij}\, a_j = 0 \tag{C.35}$$

where the stiffness matrix, K_{ij}, has elements determined by expressions similar to (C.23) or (C.26), and the *mass matrix* M_{ij} is defined by:

$$M_{ij} = \int_\Omega \phi_i(\xi)\, \phi_j(\xi)\, d\Omega \tag{C.36}$$

Equation (C.35) is a system of ODEs which can be integrated using standard time-marching techniques. For example, using the explicit Euler method would result in:

$$a_j^{n+1} = a_j^n + (\Delta t) M_{ij}^{-1} K_{ij}\, a_j^n \tag{C.37}$$

where Δt is the time step and n is indicates the n^{th} time level. In this case, using an orthogonal basis is advantageous, since then M_{ij} is diagonal, and is thus trivial to invert.

C.10.2 Fully-discrete approach

In the fully-discrete approach, we simply consider time as an extra dimension, and apply a spectral or finite-element discretisation in the multidimensional sense. For a finite-element discretisation of our sample problem, we would assume a solution of the form:

$$\hat{u}(x,t) = \sum_{i=1}^{N} a_i\, \phi_i(x,t) \tag{C.38}$$

and discretise (C.34) on the space-time domain Q shown in figure C.16. To do so we apply the MWR on the space-time domain Q (figure C.17) (a slab of width Δt):

$$\int_Q w\,(u_t - u_{xx})\, dQ = 0 \tag{C.39}$$

and integrate by parts:

$$\int_\Omega \left(wu\Big|_0^{\Delta t} - \int_0^{\Delta t} w_t\, u\, dt \right) dx - \int_0^{\Delta t} \left(wu_x\Big|_0^1 - \int_\Omega w_x\, u_x\, dx \right) dt = 0$$

Applying the Dirichlet BCs at $x = 0$ and $x = 1$, we are left with

$$\int_{\Omega^{n+1}} w\, u\, dx - \int_{\Omega^n} w\, u\, dx - \int_Q w_t\, u\, dQ + \int_Q w_x\, u_x\, dQ = 0 \tag{C.40}$$

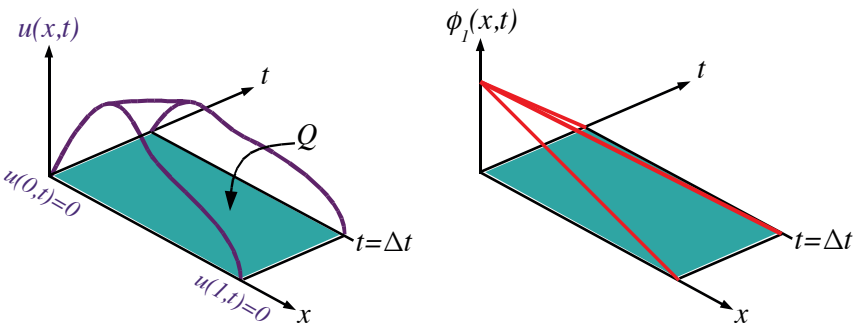


Figure C.16: A space-time domain solution (left) and basis function (right)

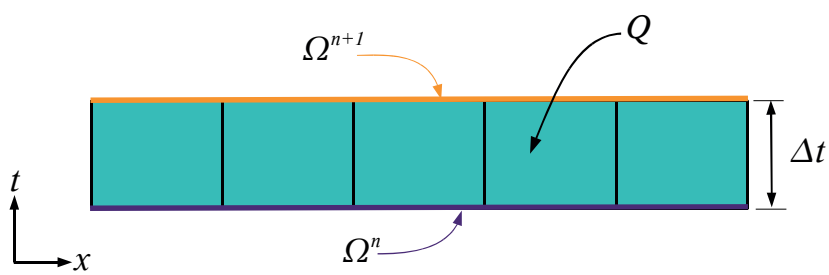


Figure C.17: A Space-time slab, Q. Its lower boundary is given by the spatial domain Ω at time step n, and its upper boundary by Ω at time step $n+1$.

Figure C.18: Space-time slab for an expanding spatial domain

There is usually no need to consider more than one time step in the time direction due to causality (the past does not depend on the future). Q is normally referred to as a *space-time slab*. On the lower boundary, Ω^n, the initial condition is applied as a Dirichlet condition. By virtue of the fact that the problem is parabolic in the time direction, no boundary condition is required on Ω^{n+1}. The solution at the next times step is just the computed solution on Ω^{n+1}, which is used as an initial condition for the next time step.

Space-time methods are flexible, in that they provide a straightforward way to deal with changing domains (figure C.18). They also provide some additional choices for adaptation, such as increasing the number of time basis functions locally, so that time accuracy is enhanced in only a portion of the domain. Space-time methods are relatively expensive, however, if p-refinement is used globally to increase the order accuracy, as this has a direct impact on the size of the matrices which must be solved within a time march loop. Furthermore, their stability analysis using Fourier analysis techniques can be laborious, typically requiring the use of symbolic manipulation programs.

C.11 Further Developments

Historically, finite-element methods were first applied to structural problems, but the last few decades have seen their application to a wide range of PDEs. In certain cases, this has required the development of techniques which go beyond those we have just discussed.

A case in point is the application of finite-element methods to problems involving advection, typically encountered in fluid mechanics. The standard Bubnov-Galerkin approach is insufficient in this regard, as it tends to produce oscillatory solutions in the presence of advective phenomena. This problem was successfully overcome by employing weighting functions which differ from the solution basis functions, known as the *Petrov-Galerkin method*. It was later shown that this was comparable to introducing additional terms to a Bubnov-Galerkin method which account for unresolved components of the solution, leading to the concept of *variational multiscale methods*. These have since been applied to a range of challenging problems.

A number of other successful approaches to finite-element discretisation have also been developed. These include *least-squares methods* and *discontinuous Galerkin methods*. The interested reader is encouraged to investigate their respective advantages by consulting the latest scientific literature.

Appendix D

Common notation for spectral and finite-element methods

Requirements such as (C.8) arise frequently in the study of spectral and finite-element methods, so it makes sense to use compact notation to express them. In this appendix, definitions are provided for the most common notation encountered in the literature.

D.1 $C^n(\Omega)$

We say that a function is in $C^n(\Omega)$ when it has continuous derivatives up to order n in domain Ω. The functions in figure C.1(b) are therefore in $C^0(\Omega)$. Often we omit the reference to Ω and just say the functions are in C^0, or are C^0 continuous.

D.2 $L^2(\Omega)$

A function $f(x)$ is in $L^2(\Omega)$ if it is square integrable in Ω:

$$\int_\Omega f^2 d\Omega < \infty \tag{D.1}$$

D.3 $H^n(\Omega)$

A function $f(x)$ is in $H^n(\Omega)$ if it and its derivatives up to order n are square integrable in Ω. For example, the following would be true for a function in H^1:

$$\int_\Omega \left[f^2 + f_x^2\right] d\Omega < \infty \tag{D.2}$$

$H^n(\Omega)$ defines the *Sobolev space* of functions, which is widely used in the analysis of partial differential equations.

D.4 $\{\cdot|....\}$

This notation is used to describe a particular set of functions, also known as a *function space*. The braces indicate that we are providing a set description. To the left of the | we give the notation for a member of the set, and to the right of the | we give the properties of the members. For example, we can define the set of suitable weighting functions for the weak form (C.7) as:

$$\mathcal{W} = \{w \mid w \in H^1,\ w(0) = 0\} \tag{D.3}$$

Since $\mathcal{W}$ describes a Sobolev space, but with some additional conditions, $\mathcal{W}$ may be referred to as a *Sobolev subspace*.

D.5 $(\cdot, \cdot)_\Omega$

This is a short form for symmetric, bilinear forms. Basically:

$$(a, b)_\Omega \equiv \int_\Omega (a\ b)\ d\Omega \tag{D.4}$$

By symmetric, we mean:

$$(a, b) = (b, a) \tag{D.5}$$

whereas bilinear refers to the following type of linearity for both a and b:

$$(c_1 a_1 + c_1 a_2, b) = c_1(a_1, b) + c_2(a_2, b) \tag{D.6}$$

D.6 A precise statement of the sample problem

Using the notation introduced in the previous section, we can now give a precise statement of the weak formulation (C.7) for the sample problem:

For $\mathcal{W} = \{w \mid w \in H^1,\ w(0) = 0\}$ and $\mathcal{U} = \{u \mid u \in H^1,\ u(0) = g\}$, find $u \in \mathcal{U}$ such that for all $w \in \mathcal{W}$:

$$w(1)q - (w_x, u_x)_\Omega = (w, f)_\Omega \tag{D.7}$$

In the above, we have substituted $w(0) = 0$ and $u_x(1) = q$, corresponding to the Dirichlet and Neumann boundary conditions.

Appendix E

Computations with arbitrary finite-element geometries

E.1 Isoparametric approach

Computing on elements with arbitrary geometries can be done using an *isoparametric approach*, where arbitrary elements in physical space are transformed into a simple master element of equivalent topology (figure E.1). Computations can then be performed in the coordinates of the master element. To take the physical geometry of an element into account, a generalised transformation between the master coordinates (ξ, η) and the physical coordinates (x, y) can be used.

When working with a master element, multidimensional basis functions can easily be defined by using *tensor products* of one-dimensional basis functions.

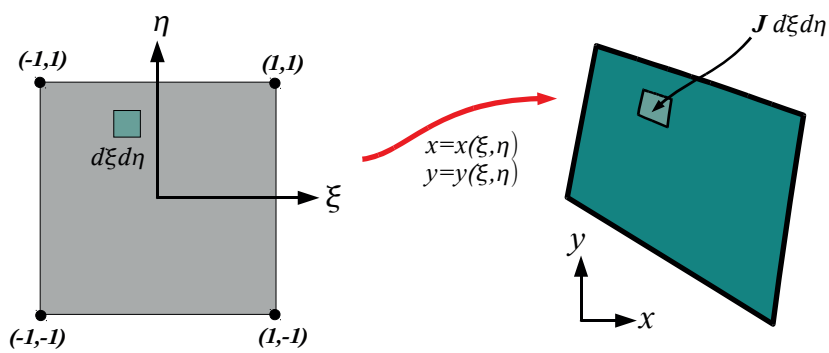


Figure E.1: A master element (left) and a physical element (right)

For example, consider the master element shown in the left half of figure E.1. A two-dimensional piecewise linear basis can be defined by using the following functions:

$$\phi_1(\xi,\eta) = \frac{1}{4}(1-\xi)(1-\eta), \quad \phi_2(\xi,\eta) = \tfrac{1}{4}(1+\xi)(1-\eta), \tag{E.1}$$
$$\phi_3(\xi,\eta) = \frac{1}{4}(1+\xi)(1+\eta), \quad \phi_4(\xi,\eta) = \tfrac{1}{4}(1-\xi)(1+\eta)$$

These are essentially products of the 1D piecewise linear basis we considered previously, expressed in ξ and η coordinates.

In order to compute element matrices and vectors in master coordinates, we need to relate the operations of differentiation and integration to equivalent operations in physical coordinates. This can be done by realising that the physical coordinates can be considered to be known functions when expressed in (ξ,η) coordinates, that is $x = x(\xi,\eta)$ and $y = y(\xi,\eta)$. The differential lengths in both coordinate systems can then be related to each other by the chain rule:

$$\begin{aligned} dx &= x_\xi\, d\xi + x_\eta\, d\eta \\ dy &= y_\xi\, d\xi + y_\eta\, d\eta \end{aligned} \tag{E.2}$$

where quantities such as $x_\xi = \frac{\partial x}{\partial \xi}, y_\eta = \frac{\partial y}{\partial \eta}$... are known as metrics of the transformation. (E.2) can be written in matrix format as:

$$\begin{bmatrix} dx \\ dy \end{bmatrix} = \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix} \begin{bmatrix} d\xi \\ d\eta \end{bmatrix} \tag{E.3}$$

We could have also started by considering differential lengths on the master element to obtain:

$$\begin{bmatrix} d\xi \\ d\eta \end{bmatrix} = \begin{bmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{bmatrix} \begin{bmatrix} dx \\ dy \end{bmatrix} \tag{E.4}$$

By comparison, the second matrix of metrics is equivalent to the inverse of the first. This leads to:

$$\begin{bmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{bmatrix} = J^{-1} \begin{bmatrix} y_\eta & -x_\eta \\ -y_\xi & x_\xi \end{bmatrix} \tag{E.5}$$

where $J = (x_\xi y_\eta - y_\xi x_\eta)$ is called the Jacobian of the transformation.

J represents the ratio of areas measured in the two coordinate systems. Specifically, an area of integration $d\xi\ d\eta$ on the master element corresponds to an area $J\ d\xi\ d\eta$ on the physical element. Correspondingly, integrals on the physical element (pe) are related to those on the master element (me) by:

$$\int_{\Omega_{pe}} f(x,y)\, dx\, dy = \int_{\Omega_{me}} f(\xi,\eta)\, J\, d\xi\, d\eta \tag{E.6}$$

In summary, (E.2), (E.5) and (E.6) can be used to relate the operations of differentiation and integration on the master element to those on the physical

element. Before using these relations, however, we need to determine the transformation metrics x_ξ, x_η, y_ξ and y_η. To do so we need explicit definitions for $x = x(\xi, \eta)$ and $y = y(\xi, \eta)$. A common approach is to construct a functional definitions using the solution basis:

$$x(\xi, \eta) = \sum_{i=1}^{N} b_i \, \phi_i(\xi, \eta)$$
$$y(\xi, \eta) = \sum_{i=1}^{N} c_i \, \phi_i(\xi, \eta) \tag{E.7}$$

although any appropriate basis can be used. The amplitudes of the coordinate basis functions, b_i and c_i, can be determined by requiring the interpolation to reproduce the physical coordinates of the element. If the physical elements have straight-line edges, and the basis (E.2) is used, the coefficients are simply given by $b_i = x_i$ and $c_i = y_i$, where x_i, y_i are the physical coordinates of the vertex where $\phi_i(\xi, \eta) = 1$ (note that for (E.2) all other ϕ are zero at this point).

Bringing everything together, the procedure for evaluating integrals and derivatives on arbitrary elements is:

1. Define $x(\xi, \eta)$ and $y(\xi, \eta)$ using physical element coordinates and (E.7)
2. Take derivatives of (E.7) to find $x_\xi, x_\eta, y_\xi, y_\eta$
3. Compute $J = (x_\xi y_\eta - y_\xi x_\eta)$
4. Use (E.5) to determine $\xi_x, \eta_x, \xi_y, \eta_y$
5. Evaluate the required solution derivatives by taking gradients in (ξ, η), then applying the chain rule (E.2)
6. Evaluate contributions to integrals using (E.6)

When integration is performed by quadrature, steps 2-6 above are repeated for each quadrature point. Note that all of the above operations are performed on the master element. The isoparametric approach is widely used since it is convenient from an implementation point of view, but complex transformations due to curved boundaries change the polynomial order of the interpolation in physical space. Maintaining the correct order of convergence (p in equation (C.29) then requires that the curvature of the elements be not too large. In practice this means that a minimum level of refinement in h is required before the optimal rate of convergence is obtained for a given p.

E.2 Physical coordinate approach

Basis functions can also be created and evaluated directly in physical coordinates. This has the advantage that the order of polynomial interpolation is preserved when elements with curved boundaries are used. To illustrate the

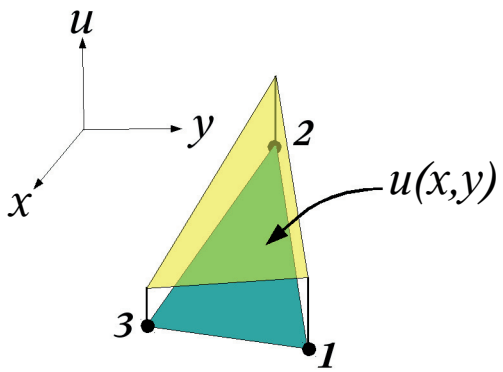


Figure E.2: Linear solution over a two-dimensional triangle element

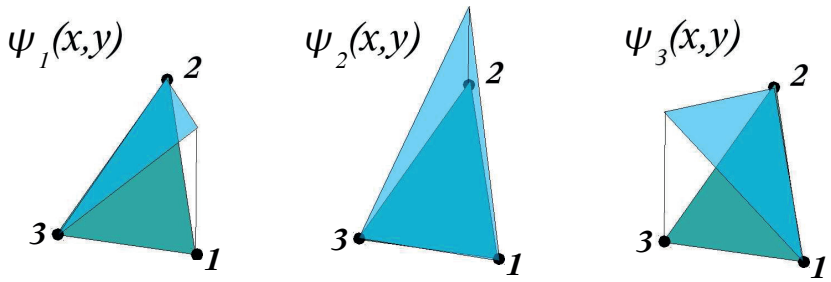


Figure E.3: Local functions for interpolating a linear solution within an element

procedure we will consider the construction of a piecewise linear basis for triangular elements with straight edges. As shown in figure E.2, a linear solution over a two-dimensional element e can be represented by the function:

$$u_e(x, y) = a + bx + cy \tag{E.8}$$

For convenience, we would like to express this as a combination of three "shape" functions, each of which has the value of the solution at one node and is zero at the others, as shown in figure E.3. This requirement can be expressed as:

$$u_1 = a + b\,x_1 + c\,y_1 \tag{E.9}$$
$$u_2 = a + b\,x_2 + c\,y_2 \tag{E.10}$$
$$u_3 = a + b\,x_3 + c\,y_3 \tag{E.11}$$

Solving this system for a, b and c leads to:

$$a = \frac{1}{2A_e}\left[u_1(x_2y_3 - x_3y_2) + u_2(x_3y_1 - x_1y_3) + u_3(x_1y_2 - x_2y_1)\right] \tag{E.12}$$
$$b = \frac{1}{2A_e}\left[u_1(y_2 - y_3) + u_2(y_3 - y_1) + u_3(y_1 - y_2)\right] \tag{E.13}$$

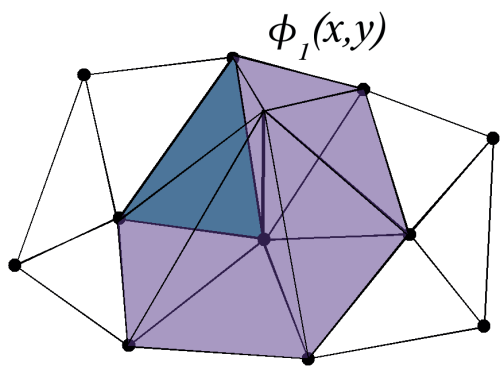


Figure E.4: Definition of a nodal basis function ϕ_i by aggregating shape functions from each surrounding element

$$c = \frac{1}{2A_e}\left[u_1(x_3 - x_2) + u_2(x_1 - x_3) + u_3(x_2 - x_1)\right] \tag{E.14}$$

where A_e is the area of the element, given by half the determinant of the system coefficient matrix:

$$A_e = \frac{1}{2}\left(x_2y_3 + x_1y_2 + x_3y_1 - x_3y_2 - x_1y_3 - x_2y_1\right) \tag{E.15}$$

Substituting the expressions for a, b and c into (E.8) and grouping terms in u_1, u_2 and u_3 gives:

$$u_e(x,y) = u_1\,\psi_1(x,y) + u_2\,\psi_2(x,y) + u_3\,\psi_3(x,y) \tag{E.16}$$

where the functions ψ_1, ψ_2, ψ_3 are given by:

$$\psi_1(x,y) = \tfrac{1}{2A_e}\left[(x_2y_3 - x_3y_2) + (y_2 - y_3)x + (x_3 - x_2)y\right] \tag{E.17}$$

$$\psi_2(x,y) = \tfrac{1}{2A_e}\left[(x_3y_1 - x_1y_3) + (y_3 - y_1)x + (x_1 - x_3)y\right] \tag{E.18}$$

$$\psi_3(x,y) = \tfrac{1}{2A_e}\left[(x_1y_2 - x_2y_1) + (y_1 - y_2)x + (x_2 - x_1)y\right] \tag{E.19}$$

The basis functions for the finite element method, $\phi_i(x,y)$ are then constructed by adding the ψ functions from surrounding elements which are non-zero at the vertex i, as shown in figure E.4.