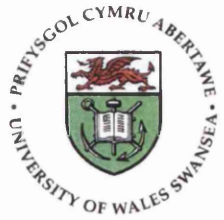
A locally conservative Galerkin (LCG) finite element method for convection-diffusion and Navier-Stokes equations.

Thomas, Craig George



A locally conservative Galerkin (LCG) finite element method for convection-diffusion and Navier-Stokes equations

Craig George Thomas
B.Eng., M.Res., A.M.I.Mech.E.

Thesis submitted to the University of Wales Swansea
in candidature for the degree of Doctor of Philosophy

September 30^{th} 2006

Civil and Computational Engineering Centre
School of Engineering
University of Wales Swansea
Singleton Park, Swansea SA2 8PP
Wales, United Kingdom

What I have accomplished in life, is the result of the unconditional love and years of hard work and sacrifice of my loving parents,
Alison and George Thomas.

This work is dedicated to both of you.

I mam e dad

Contents

Acknowledgments

I would like to express my most sincere and utmost gratitude to Dr. P. Nithiarasu - my supervisor and mentor, for his invaluable guidance, support, and wisdom. It has been a great privilege to have worked under tutelage of such an accomplished academic and kind gentleman. Thank you dear Arasu.

I am also grateful to the staff at the School of Engineering and to my office co-workers for their friendship and support. A special thanks is given here, to Prof. K. Morgan, Alex, Kim, and Rhodri. Thanks too, to little Guiness for keeping me company during the late nights of work.

Finally my deepest gratitude goes to my mam and dad, my sister Natalie, and my partner Jodie. Your constant encouragement and love has been my sustenance throughout. Thankyou for being there for me in times of need and advice, thankyou also for sharing with me in times of joy and celebration. God bless you all.

Summary

In this thesis, an element-wise locally conservative Galerkin (LCG) finite element method is presented. The LCG method has been shown here to be successful in solving equations of scalar-transport, and the incompressible Navier-Stokes equations.

The LCG approach facilitates an element-by-element solution and obtains a continuous and unique nodal solution from the surrounding element contributions, via averaging. A simple numerical flux establishes continuity at the edges between neighbouring elements. This allows the system of discrete equations to be solved over each elemental sub-domain, greatly simplifying the solution procedure. The method explicitly establishes local element-wise conservation, and after the averaging procedure a residual flux appears on the global boundary. It is this flux which gives the LCG method global conservation, regardless of prescribed boundary conditions.

Aspects research are: the mathematical formulation; explicit and implicit discretisations; edge flux calculation procedures; development and implementation of Petrov-Galerkin and characteristic based methods; and finally matrix-free LCG methods for steady and unsteady incompressible flows. Evaluation of all the proposed LCG methods has been given, showing the methods to be accurate and robust.

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The finite element (FE) approximation

A common theme occurring in proceeding chapters of this thesis, is the use of the finite element approximation, to provide the general framework for the development of a number of novel and appealing numerical schemes. These new schemes have been shown to have excellent success in solving scalar variable and Navier-Stokes equations [1, 2, 3, 4, 5]. Before going into the full details in the following chapters, a concise account of the finite element (FE) approximation is given below.

1.1.1 A brief historical overview

From a historical context, various authors have given conflicting opinions on the origins of the technique [6, 7, 8]. A paper by Felippa [9] researches as far back as the 1930s. Here, the origins of the FE method have been traced to a pre-computer tool introduced by an aero-elasticity group, based at the National Physics laboratory, London. The method underwent a period of major development from its pioneers in the 1950s [10, 11, 12, 13], before being coined with the name *finite element* by Clough [14], at a conference held in Pittsburgh, 1960. The FE method was further extended to the field of continuum mechanics by O.C. Zienkiewicz and Cheung in 1965 [15]. It was Zienkiewicz who first realised the general potential of the FE method outside the field of solid mechanics. Two years later

Zienkiewicz published the first text book on the FE method for structural and continuum mechanics [16].

1.1.2 Some general remarks on the FE method and the standard Galerkin approximation

The use of the FE method today, is widespread in many different fields of study. It enjoys much popularity due to its advantages in modeling complex geometries, and its natural enforcement of Neumann (flux-type) boundary conditions. Of the many choices for the weighting function, the Galerkin method (or Bubnov-Galerkin method) [17, 18] is the most standard. Here, the interpolation functions are chosen as the weighting functions, giving symmetrical matrices for modeling self-adjoint diffusive-type problems. Zienkiewicz *et al.* note that this is a most optimal approximation, with regards to giving the minimum error in the energy norm [19, 20, 21, 22, 23].

1.1.3 The FE method for convection-dominated problems

It is well known, that using the standard Galerkin spatial discretisation in the solution of convection-dominated problems, introduces *negative diffusion* into the solution procedure [24], causing instability. The instability is due to a central-difference type approximation of the non self-adjoint convective terms. This gives non-symmetrical terms in the obtained discrete matrix-system, and leads to spurious oscillations in the computed solution. Unless the velocity is of negligible value, as in creeping flows (or zero for the case of Stokes flow), then a suitable method of stabilisation is needed.

There are a number of well-known stabilisation methods, commonly used in finite elements, which will eliminate or at least significantly reduce the oscillations caused by the convective-transport terms. For solving scalar-variable transport problems, using the FE approach, such methods include: the Petrov-Galerkin (PG) method [23, 25, 26]; the Streamline-Upwind Petrov-Galerkin (SUPG) method [27, 28]; the Galerkin/Least Squares (GLS) method [29, 30]; and the Finite Increment Calculus (FIC) method [31]. Both PG and SUPG methods employ a modified weighting function, that is no longer identical to the interpolation function. The modified weighting function introduces an additional diffusion term

into the final discrete formulation. This extra term gives consistent convective-stabilisation, and is sometimes referred to as *artificial* or *balancing diffusion* [23]. The SUPG method is relatively superior of the two methods for two (or three) space-dimensions, as the balancing diffusion added is anisotropic - acting only in the direction of the streamlines. The GLS formulation produces similar results via a combination of the standard Galerkin and least squares approximation. The FIC procedure directly obtains the balancing diffusion by considering a finite length, and using a backwards Taylor series to give an extra stabilisation term. In more recent developments of stabilizing techniques, Hughes [32] identifies stabilised methods as approximate subgrid scale (SGS) models [33, 34, 35, 36], additionally the residual-free bubble function [37] has been implemented for advection-diffusion equations.

When time dependent equations are solved, high-order time schemes, such as the Taylor Galerkin (TG) method [38] and the characteristic Galerkin (CG) [39] method, are commonly used in FE analysis. For these latter schemes, performing a high-order time discretisation (which precedes the spatial one), naturally introduces a consistent balancing diffusion. The second-order TG method is the finite element equivalent of the original finite difference (FD) Lax-Wendroff scheme [40]. Here, the temporal derivative appearing in the governing equation, is approximated with a truncated Taylor series expansion - correct to second order. Higher-order variations of the TG method are discussed in detail by Donea and Huerta [38, 41].

For the CG method, first introduced by Löhner *et al.* [39], the temporal derivative is discretised along the problem *characteristic.* This gives a semi-discrete form that is completely self-adjoint in nature, and thus the Galerkin spatial approximation is optimal. In the original paper [39], it was shown that the need for mesh updating in the CG procedure, can be avoided - at a cost of conditional stability, by the use of a local Taylor expansion. This in turn gives a *simple explicit* CG procedure of second-order accuracy in time. For the case of convection-diffusion with scalar variables, the TG and the *simple explicit* CG schemes give an identical discrete equation. Further work and applications of the *simple explicit* CG method are given in [23, 42, 43]. Additionally, a recent paper presents an excellent and comprehensive overview of the all the most common variations of the CG

method(s) [44].

1.1.4 Using the FE method with discontinuous interpolation

In recent years, there has been a high level of interest amongst researchers on the Discontinuous Galerkin (DG) Method. The DG method can be regarded as a generalization of the finite volume (FV) method [45], as it assumes a discontinuous approximate solution. The method was first introduced by Reed and Hill in 1973 [46], to model the first-order neutron transport equation. Its potential was recognised by La Saint and Raviart [47], who in the following year, published the first numerical analysis of the method for the linear transport equation. A grand collection of papers on the various DG methods - illustrating the high level of activity in this area, is given by Cockburn *et al.* [48]. This reference is singularly pointed out as a valuable introduction to the DG literature and gives a concise summary of the current *state of the art.* Key literature on the DG method(s) from this book, includes the works: [49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64]. Motivated by the method of Bassi and Rebay [65], Cockburn and Shu have also developed a local discontinuous Galerkin (LDG) method [66] for solving general convection-diffusion problems. The method has symmetric algebraic equations, it is element-wise conservative and uses discontinuous basis functions - making it ideally suited for *hp*-adaptivity. For further contributions to the LDG method and applications see references [67, 68, 69, 70, 71, 72, 73].

Generally speaking; despite all the many favourable properties of the various DG methods, including: a very flexible framework for grid adaptation; local element-wise conservation; natural extension of the method to any order approximation; discontinuity capturing at domain interfaces; and smaller equation sets to solve, the DG methods are more expensive than the standard global (continuous) Galerkin method. The excessive number of degrees of freedom and the higher cost associated with it, are often blamed for DG's lack of interest within the engineering industry [74]. Thus, it is not surprising to hear that researchers are looking for a DG method with a structure of continuous Galerkin method, in an effort to increase the appeal of the DG method [75]. However it is believed that this approach is heading in the wrong direction of progress. Clearly, it would be easier to modify

a continuous Galerkin structure to adopt a discontinuous path [1, 2, 3, 4, 5], rather than adopting DG methods to develop a global Galerkin structure. This way, existing industrial codes can be modified to accommodate the changes without resorting to complete revision of the codes. This would make implementation more efficient on both a practical and commercial scale, in turn making them more attractive to prospective industrial clients.

1.2 The characteristic based split (CBS) scheme

As noted by Nithiarasu *et al.* [76], characteristic-based methods are not new to numerical modelling. In fact, they have received considerable research and much development, being widely employed in the past to solve both convection-diffusion problems and the Navier-Stokes equations [39, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86].

The characteristic based split (CBS) scheme for compressible and incompressible flow problems, was first introduced into the finite element literature in 1995 by Zienkiewicz and Codina and co-workers [42, 87, 88]. The CBS scheme is essentially a fractional time-stepping algorithm, based on an original FD velocity-projection scheme for solving incompressible flows by Chorin [89]. Since its introduction to the computational and numerical methods community, the CBS scheme has received great interest and has undergone extensive research for both incompressible and compressible flows [23, 43, 44, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99].

The CBS scheme has also been extended to investigate other applications such as shallow water flows [23, 100], thermal and porous medium flows [101, 102, 103, 104]. More recently the extension has included transient flow problems [23, 96, 97, 98]; steady and unsteady turbulent-incompressible flows [23, 105]; problems of viscous-elastic flow [23, 106]; and a special matrix free fractional step method for the Stokes problem has been applied to solve static and dynamic incompressible solid mechanics [76]. All of which, prove the scheme's robustness and applicability for different simulation scenarios. In this thesis some technical investigations [99, 107] are carried out, for improving the results for compressible flow, using the global Galerkin CBS scheme.

The basic three-step temporal scheme starts with the characteristic discretisation

of the momentum equation components. Since each momentum equation contains more than one characteristic variable, the CG procedure is not easy to implement. The approach recommended by the original authors - and generally adopted since, is to split each momentum equation by removing the pressure-gradient term. The remaining equation for each velocity component is one of simple convection-diffusion for a scalar-variable, and is thus easily discretised using the CG method. Solving these equations gives an intermediate velocity field, which needs to be corrected once the pressure field is known. In the second step, the pressure field is solved independently, using a pressure equation based on continuity (mass conservation). Once pressure has been evaluated, the intermediate velocities are corrected in the third step. If other scalar variables such as temperature, concentration, etc. are required, then these are normally solved during a separate fourth step - allowing quick adaptability to different problems.

If the pressure is removed completely from the momentum equation, then the fractional step method introduces a first-order error into the momentum equation [108]. As an alternative to the recommended approach described, it is certainly possible to keep the pressure gradient in the momentum equation, by treating it as a source-type quantity. This has been shown to be a more accurate alternative [108], however its appeal has not been as attractive as the recommended approach. This is due to the Ladyshenskaya-Bubska-Brezzi (LBB) constraints [109, 110, 111], which apply when incompressibility (or near incompressibility) is encountered. This is discussed in detail by Zienkiewicz *et al.* [23, 87] - where a mathematical analysis is given for the two different CBS approaches, with respect to the LBB condition . When the momentum equation is solved with the pressure term included, instability is unavoidable if equal-order interpolations (shape functions) are used for both the velocity and pressure fields. By removing the pressure term from the momentum equation circumvents the LBB condition, both enhancing pressure stability of the CBS scheme and freely allowing the arbitrary choice of interpolating functions.

For incompressible flows, an implicit solution for the pressure equation (Step 2) was used in the past - leading to a *semi-implicit* CBS scheme. Early attempts to obtain a fully explicit CBS algorithm for incompressible flows, were unsuccessful and suffered from severe time-step restrictions. A major contribution in overcoming problems in this area has been

made by Nithiarasu [23, 96, 97, 98, 76], who has applied artificial compressibility methods, previously used in FD and FV techniques [112, 82, 85, 113, 114], to the finite element context of the CBS scheme. In the artificial compressibility (AC) method implemented, the sonic velocity is replaced with a locally calculated artificial compressibility parameter based on the convective and diffusive velocities. The resulting matrix free CBS-AC procedure is fully explicit, and has been proven to be both accurate and efficient, with manageable local and global time-step restrictions. Although such a scheme is only valid if a steady-state solution to the problem exists, unsteady flows have also been computed successfully via a dual time-stepping approach [23, 96, 97, 98, 76, 113, 114, 115]. The semi-discrete CBS-AC algorithm for the Navier-Stokes equations, is used in this thesis as the temporal stencil for a locally conservative Galerkin (LCG) method for solving both steady and unsteady incompressible flows.

1.3 Local conservation, and conservative schemes

Local conservation is a desirable property in the numerical modelling of engineering problems. Finite volume (FV) methods are known to conserve fluxes for both a cell-centred volume and a dual-cell volume around a node. Additionally, discontinuous Galerkin (DG) methods are known to be element-wise conservative. For the general finite element (FE) method, there is a general feeling of ambiguity when it comes to describing its conservation properties.

The basic finite element (FE) form for an equation of the type $\partial F_i/\partial x_i$, may be written for a sub-domain surrounding a node as

$$\int_{\Omega} -\frac{\partial w_a}{\partial x_i}\tilde{F}_i d\Omega + \int_{\Gamma} w_a \tilde{F}_i d\Gamma = 0 \tag{1.1}$$

For the standard global (continuous) Galerkin approximation, the sum of the derivatives of the weight function surrounding an inside node, a, is zero. Thus the flux is conserved locally at the node, making the global Galerkin FE method conservative for patch of elements surrounding a node. This is standard, but other definitions of conservation for the global Galerkin method do not seem as transparent.

A significant effort has been made by various authors to highlight the conservation properties of the global Galerkin method [116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126]. As written by Hughes *et al.* [125], local element-wise conservation emanates from the property that the weighting function can be set exactly to unity on the sub-domain of interest and zero everywhere else. It is clear to see how DG methods are locally conservative, since their discontinuous basis functions allow them to achieve this property. However, this statement implies that the continuous global Galerkin FE method, is not always globally conservative. Only in the absence of Dirichlet boundary conditions is this statement true. In the above mentioned works (referenced in this paragraph), it is proposed that the continuous global Galerkin FE method can be made globally conservative by a postprocessing computation of flux on the Dirichlet part of the boundary. Using this flux, rather than strongly enforcing the Dirichlet conditions enables the continuous global Galerkin method to be globally conservative. The recent work of Hughes and co-workers [125, 126], has helped greatly in clarifying a lot of unnecessary confusion, on the global conservation properties of the standard global Galerkin method. More importantly though, this work has opened up new possibilities with regard to local element-wise conservation. A similar approach applied to a sub-domain or element allows the desirable property of local element-wise conservation to be gained by practitioners of the global Galerkin FE method.

Though the work of Hughes and co-workers rekindled the conservation issue of global finite element methods, element-by-element solution of the conservation equations, using a standard Galerkin structure was not addressed. Recently, a locally conservative Galerkin (LCG) scheme has been introduced by Nithiarasu [1], for solving convection-diffusion problems. The LCG method has similar advantages to DG methods without the need for solving additional variables. The previously mentioned paper proved that an element-by-element solution is possible via postprocessing the fluxes at every time step. The method uses the property of local conservation at steady state conditions in order to define a flux at element boundaries. This flux allows the computational domain to be broken down into a series of elemental domains, each with its own Neumann boundary conditions based on this computed flux. Although in the very first LCG paper [1] only linear elements were considered, and a small number of relatively simple problems solved. Its appeal prompted

a significant research effort to be implemented in this thesis work.

A large effort has been made in this thesis in understanding how the LCG method works. This has included a rigorous description and analysis of the scheme, in which surprising similarities were found with the global Galerkin method. Also the method has been applied to quadratic elements in order to improve the flux-calculation accuracy at the element interfaces. In addition to the above, stabilised LCG schemes have also been implemented using already established methods used for the global Galerkin method. These have included both the SUPG and CG methods. Success in 2D convection-dominated problems has led to implementation of a matrix-free LCG-CBS method for the solution of incompressible flow equations. Again both linear and quadratic versions have been implemented. Finally the problem of transient flow has been addressed by implementing a reliable dual time-stepping CBS procedure [96], giving a LCG-CBS scheme for unsteady flows.

1.4 Organisation of the thesis

The major objective of this thesis, is to develop a locally conservative Galerkin (LCG) method for solving both scalar conservation equations and the Navier-Stokes equation. The latter, requires the implementation of the characteristic based split (CBS) algorithm. As the CBS scheme with artificial compressibility (AC) played a key role in the development of an characteristic based LCG method for solving fluid dynamic problems, probationary research was carried out on the general CBS scheme and a number of small projects were carried out for compressible flow.

Chapter 2 introduces the governing equations for scalar transport and the general Navier-Stokes equations. Non-dimensional forms are given along with the Navier-Stokes equations for incompressible flow. The standard schemes for solving scalar equations using the global finite element (FE) method are then given, these include the Galerkin, Petrov-Galerkin, and characteristic Galerkin (CG). The Chapter closes with the discussion of the general CBS scheme for compressible flow.

Chapter 3 is the largest chapter of the thesis, it is solely dedicated to developing LCG methods for solving equations of scalar transport and incompressible flow. Here, the

discussion of the LCG method starts with the essential fundamentals: The basic discretisation procedure is described first for explicit and implicit methods, next the numerical flux calculation is derived for both linear and quadratic elements, this is followed by an analysis of the nodal equations for the LCG method and comparisons are made to the standard global Galerkin method. To allow convection dominated problems to be solved by the LCG method, standard methods of stabilisation - discussed in Chapter 2 for the global Galerkin method, are applied to the LCG procedure. Chapter 3 finishes with the implementation of the CBS scheme (with an artificial compressibility parameter), into the LCG formulation. The result is a fully explicit CBS-LCG scheme suitable for solving incompressible flow problems. Finally the chapter closes with a dual time-stepping approach, used to recover the transient solution of unsteady flow.

Chapter 4 discusses the influence of a stream-wise element-size calculation on the solution of compressible flow problems. Additionally, the use of a residual smoothing routine, is shown to allow the solutions of compressible flow simulations to be carried out on the same code for $0.01 \leq Ma \leq 3.00$. Although this work is not directly related to the LCG discussion, the results of this work lead to the stream-wise element-size being applied to optimise the SUPG-LCG method, for convection dominated transport. Additionally, work on the compressible CBS code facilitated in the development of a CBS-LCG code with an artificial compressibility parameter.

The explicit and implicit LCG methods are evaluated in Chapters 5 and 6, for diffusive and convection dominated transport problems respectively. In Chapter 5, steady-state problems of heat conduction are solved in two and three dimensions. Problems with internal sources are also considered. Chapter 6 validates the implementation of the SUPG and characteristic Galerkin methods into the LCG procedure. This is carried out using a number of well know convection-diffusion problems. Chapter 6 closes with details of the mesh convergence properties of the explicit and implicit CG based LCG methods. The general convection-diffusion equation with source term is used, with convergence rates being evaluated on grids of linear and quadratic triangular elements.

The solution of the simplified incompressible flow equations using the CBS-LCG scheme, is evaluated in Chapter 7 for steady flow. A number of well know benchmark test

cases are used to prove good performance of both linear and quadratic elements. Problems include: cavity flow, flow past a circular-cylinder, channel flow, and flow past a backward facing step. Where possible, the results have been validated by both experimental and numerical data.

Chapter 8 shows how the LCG method can be successfully used to solve transient problems. To examine the performance of the explicit characteristic based LCG method, the convection of a product-cosine hill in a purely rotational velocity-field is considered in the first section. The second section evaluates the implementation of the dual time-stepping CBS-LCG approach, for unsteady flows. The case of transient flow past a circular-cylinder at a Reynolds number of 100 is solved. It is shown that the unsteady CBS-LCG scheme gives good performance in accurately modelling the vortex shedding at this Reynolds number. This is a difficult problem to solve and its use demonstrates the robustness of the LCG approach.

Finally Chapter 9, concludes this thesis by outlining the results of this thesis and discusses the potential for future work.

Chapter 2

Governing Equations and Solution Procedures

2.1 Introduction

In this Chapter, details of the general governing equations modelled throughout the thesis are introduced. The discussion starts in Section 2.2, with the general scalar convection-diffusion equation. Next, the general Navier-Stokes equations are given. Here, non-dimensional variables are introduced into the equation set, to obtain the dimensionless Navier-Stokes equations for compressible flow. Section 2.2 closes with the non-conservative form of the Navier-Stokes equations, for incompressible fluid flow.

In section 2.3, a number of finite element discretisations are considered for solving the general scalar convection-diffusion equation. The global Galerkin method for spatial discretisation is applied first. Well known methods of stabilising the oscillations for convection-dominated problems are discussed next. These methods include the streamline-upwind Petrov-Galerkin (SUPG) method [27, 28], and the *simple explicit* characteristic Galerkin (CG) method [39, 42, 44]. Consideration is given for both explicit and implicit methods, with and without a lumped-mass matrix. The incorporation of linear and quadratic triangular elements is also discussed.

Section 2.4 presents the characteristic-based-split (CBS) algorithm of Zienkiewicz

et al. [42, 87, 88], for a general compressible flow simulation. This section firstly gives details on the recommended temporal discretisation and splitting procedure. Once the full four-step temporal scheme is discussed, details of the spatial discretisation procedure are given, to obtain a fully discrete form using the standard global (continuous) Galerkin method. Fully discrete matrix forms of the CBS scheme are given for general and inviscid compressible flow problems along with a shock capturing viscosity method for transonic and supersonic flow [91, 127]. Details of both local time-stepping and element-size calculation, are given next. In this section, the use of a flow dependent local element-size calculation in the streamline direction [128],- for computing the local (nodal) time-steps, is proposed for CBS calculations. Finally, in addition to the proposed stream-wise element-size calculation, this section concludes by addressing the issue of simulating inviscid flows at low Mach numbers by employing a variable smoothing approach.

2.2 General governing equations

In this section the general governing equations [23, 129, 130, 131, 132] are given. Firstly, governing equations are stated for the general convective-diffusive transport of a scaler variable. Next, the most general form of the Navier-Stokes equation set is given. This set also includes the constitutive laws for the case of a perfect gas, so to obtain a closed set of equations. Finally in this section, the Navier-Stokes equations for the case of an incompressible fluid are given. In all cases, Einstein's summation convention is implied if the equations are written in indicial notation.

2.2.1 General scalar convection-diffusion equation

The variation of a scalar variable ϕ within a spatial domain Ω and closed surface Γ, is governed by the following scalar equation, written in conservation form:

$$\beta\frac{\partial\phi}{\partial t}+\frac{\partial F_i}{\partial x_i}=S_i \tag{2.1}$$

where β is a constant. Here, F_i and S_i respectively represent the conservation flux component and source term component, in the direction x_i of a Cartesian co-ordinate system.

If source terms are negligible, the local intensity of ϕ varies only through the effect of the fluxes from its surroundings [132]. Generally F_i consists of both convective and diffusive flux components, but it can also be purely diffusive in cases, for example, such as heat conduction. The flux term of the convection diffusion equation is defined as

$$F_i = \left(u_i \phi - k \frac{\partial \phi}{\partial x_i} \right) \tag{2.2}$$

where k is the diffusion coefficient and u_i are the constant velocity components.

The solution of this equation is sought over Ω. To complete the initial-boundary value problem the following information on initial and boundary conditions is also required:

$$\begin{aligned} \phi(\mathbf{x}, t=0) &= \phi_0(\mathbf{x}) \quad \forall \mathbf{x} \in \Omega \\ \phi &= \overline{\phi} \quad on \quad \Gamma_\phi \quad and \\ F_n = n_i F_i &= \overline{F}_n \quad on \quad \Gamma_f \end{aligned} \tag{2.3}$$

Here the Dirichlet and Neumann partitions form the domain boundary, Γ, of Ω, i.e.

$$\Gamma = \Gamma_\phi \cup \Gamma_f \tag{2.4}$$

and n_i denotes the component of the unit outward normal vector to Γ in the the direction x_i.

2.2.2 The Navier-Stokes equations

The conservative form of the governing equations, for the general compressible flow of a viscous fluid, can be written in a convenient and compact fashion as

$$\frac{\partial \mathbf{A}}{\partial t} + \frac{\partial \mathbf{B_i}}{\partial x_i} + \frac{\partial \mathbf{C_i}}{\partial x_i} + \mathbf{D} = 0 \tag{2.5}$$

Here

$$\mathbf{A^T} = (\rho, \quad \rho u_1, \quad \rho u_2, \quad \rho u_3, \quad \rho E) \tag{2.6}$$

is the vector of independent variables,

$$\mathbf{B_i^T} = (\rho u_i, \quad \rho u_1 u_i + \delta_{1i} p, \quad \rho u_2 u_i + \delta_{2i} p, \quad \rho u_3 u_i + \delta_{3i} p, \quad u_i(\rho E + p)) \tag{2.7}$$

is the convective flux vector, and

$$\mathbf{C_i^T} = (0, \quad -\tau_{1i}, \quad -\tau_{2i}, \quad -\tau_{3i}, \quad q_i - \tau_{ij} u_j) \tag{2.8}$$

defines the diffusion flux vector. Finally the vector

$$\mathbf{D^T} = (0, \quad \rho g_1, \quad \rho g_2, \quad \rho g_3, \quad \rho(g_i u_i - q_H)) \tag{2.9}$$

contains the source terms.

In all the above equations ρ denotes the density; u_i represents the Cartesian components of the velocity vector; E is the total specific energy (internal and kinetic); δ_{ij} is the Kronecker delta function and is equal to unity when $i = j$; and zero when $i \neq j$; p represents the pressure; g_i is the acceleration due to gravitational forces; q_H is the heat generation per unit mass; and q_i is the heat flux. The latter is related to the local temperature gradient by Fourier's law of heat conduction:

$$q_i = -k \frac{\partial T}{\partial x_i}; \tag{2.10}$$

Here k is the thermal conductivity and T is the temperature. The deviatoric stress components τ_{ij} are related to velocity gradients by

$$\tau_{ij} = \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3} \frac{\partial u_k}{\partial x_k} \delta_{ij} \right) \tag{2.11}$$

where μ is the dynamic viscosity and is dependent on the temperature. For air, the relation is given according to Sutherland's law [132] as:

$$\mu = \mu(T) = \frac{1.45 T^{\frac{3}{2}}}{T + 110} * 10^{-6} \tag{2.12}$$

with T in Kelvin

In order to obtain a closed set of equations, two further equations are required and these come from equations of state:

$$p = p(\rho, T), \qquad and \qquad e = e(\rho, T) \tag{2.13}$$

In the study of aerodynamic flow it is often assumed that the fluid (air) behaves as a perfect gas thus allowing the perfect gas law to be applied. It is given as

$$p = \rho RT \tag{2.14}$$

where R is the universal gas constant. The equation set is completed by a final equation that relates temperature to internal energy. For a perfect gas with constant specific heats, the following relation is used.

$$e = c_v T \tag{2.15}$$

Where e is the internal energy per unit mass and c_v is the specific heat at constant volume. e can be found from the total energy per unit mass, E, using

$$E = e + \frac{u_i u_i}{2} \tag{2.16}$$

These relations are only approximation for viscous flows since these flows are energy dissipative and as a result will always increase the entropy of the flow system. However, assuming isentropic conditions the speed of sound can be defined as

$$c^2 = \frac{\partial p}{\partial \rho} = \frac{\gamma p}{\rho} = \gamma RT \tag{2.17}$$

where c is the speed of sound and γ is the ratio of specific heats at constant pressure and constant volume, i.e.

$$\gamma = \frac{c_p}{c_v} \tag{2.18}$$

It should be noted that the transient density term in the continuity equation, may be replaced by the relation given by Equation (2.17), under the assumption of isentropic conditions, i.e.,

$$\frac{\partial \rho}{\partial t} = \frac{\partial \rho}{\partial p}\frac{\partial p}{\partial t} = \frac{1}{c^2}\frac{\partial p}{\partial t} \tag{2.19}$$

The above relation, given in Equation (2.19), is often the basis of many artificial compressibility schemes for incompressible flow calculations [96, 113]. However, this approximation is not used in compressible flow calculations.

To complete the initial-boundary value problem, information on initial conditions throughout the problem domain is required.. Additionally, Dirichlet and Neumann type conditions are required on the respective portions of the domain-boundary: $\Gamma = \Gamma_\phi \cup \Gamma_f$. The exact form of which, will depend upon the problem being simulated.

2.2.3 The dimensionless Navier-Stokes equations for compressible flow

The implementation of coding a computational fluid dynamics (CFD) program, is greatly aided when the governing Navier-Stokes equations are expressed in a non-dimensional format [23, 129, 132]. The choice of scales used to non-dimensionalise the Navier-Stokes equations vary depending on the nature of the flow. For a compressible fluid flow, the following non-dimensional scales are generally used:

$$\begin{aligned} x_i^* &= \frac{x_i}{L}; \quad u_i^* = \frac{u_i}{u_\infty}; \quad \rho^* = \frac{\rho}{\rho_\infty}; \quad t^* = \frac{t u_\infty}{L}; \quad p^* = \frac{p}{\rho_\infty u_\infty^2}; \quad E^* = \frac{E}{u_\infty^2}; \\ {q_H}^* &= \frac{q_H L}{u_\infty^3}; \quad T^* = \frac{T c_p}{u_\infty^2}; \quad c^{*2} = \frac{c^2}{u_\infty^2}; \quad g_i^* = \frac{g_i L}{u_\infty^2}; \quad \mu^* = \frac{\mu}{\mu_\infty}; \quad k^* = \frac{k}{k_\infty} \end{aligned} \tag{2.20}$$

where a superscript *, indicates a non-dimensional quantity; a subscript infinity, $_\infty$, represents a reference free stream value; and L is a reference length. Substituting the above scales into Equations (2.5-2.9) gives the non-dimensional form of:

Continuity

$$\frac{\partial \rho^*}{\partial t^*} = -\frac{\partial U_i^*}{\partial x_i^*} \tag{2.21}$$

Momentum

$$\frac{\partial U_i^*}{\partial t^*} = -\frac{\partial}{\partial x_j^*}(u_j^* U_i^*) - \frac{\partial p^*}{\partial x_i^*} + \frac{1}{Re}\frac{\partial \tau_{ij}^*}{\partial x_j^*} - \rho^* g_i^* \tag{2.22}$$

Energy

$$\frac{\partial}{\partial t^*}(\rho^* E^*) = -\frac{\partial}{\partial x_j^*}u_j^*(\rho^* E^* + p^*) + \frac{1}{Re}\frac{\partial}{\partial x_i^*}(\tau_{ij}^* u_j^*) + \frac{1}{RePr}\frac{\partial}{\partial x_i^*}(k^*\frac{\partial T^*}{\partial x_i^*}) - \rho^*(g_i^* u_i^* - {q_H}^*) \tag{2.23}$$

In the above non-dimensional equations the mass flow fluxes are

$$U_i^* = \rho^* u_i^* \tag{2.24}$$

in addition:

$$\tau_{ij}^* = \mu^*\left(\frac{\partial u_i^*}{\partial x_j^*} + \frac{\partial u_j^*}{\partial x_i^*} - \frac{2}{3}\frac{\partial u_k^*}{\partial x_k^*}\delta_{ij}\right) \tag{2.25}$$

Re and Pr are the Reynolds and Prandtl numbers respectively and are given by

$$Re = \frac{u_\infty L}{\nu_\infty} \quad Pr = \frac{\mu_\infty c_p}{k_\infty} \tag{2.26}$$

Here, $\nu_\infty = \mu_\infty/\rho_\infty$ is the kinematic viscosity. It is clear that these non-dimensional forms of the compressible flow equations automatically cater for both viscous and inviscid Euler flows by simple controlling of the value of the parameter Re, the Reynolds number. Applying the non-dimensional scales to the perfect gas law gives:

Equation of state

$$p^* = \rho^* R^* T^* \tag{2.27}$$

where

$$R^* = \frac{R}{c_p} = \left(\frac{\gamma - 1}{\gamma}\right) \tag{2.28}$$

Once the dimensionless total energy, E^*, has been calculated, the dimensionless temperature, T^*, can be evaluated from

$$T^* = \gamma\left(E^* - \frac{1}{2}u_i^* u_i^*\right) \tag{2.29}$$

It is clear from here that when T^* is known, pressure can be found from density (and viceversa) using Equation (2.27). Furthermore, the non-dimensional sonic velocity is found from

$$c^* = \gamma R^* T^* = (\gamma - 1)\, T^* \tag{2.30}$$

and the local Mach number is given by

$$M_a = \frac{\sqrt{u_i^* u_i^*}}{c^*} \tag{2.31}$$

2.2.4 Relations for isentropic flow at the stagnation point

In the computations of inviscid compressible-flow the following isentropic relations are useful in validating quantitative data retrieved for the stagnation point(s). The ratio of stagnation temperature, T_o^*, to free-stream temperature, T_∞^*, is given by

$$\frac{T_o^*}{T_\infty^*} = \left[1 + \left(\frac{\gamma - 1}{2}\right) {M_a}^2\right] \tag{2.32}$$

Using the perfect gas law (2.27) and the isentropic relation $p^* \propto \rho^{*\gamma}$, the stagnation to free-stream ratios for density and pressure

$$\frac{\rho_o^*}{\rho_\infty^*} = \left[1 + \left(\frac{\gamma - 1}{2}\right) {M_a}^2\right]^{\left(\frac{1}{\gamma - 1}\right)} \tag{2.33}$$

and

$$\frac{p_o^*}{p_\infty^*} = \left[1 + \left(\frac{\gamma - 1}{2}\right) {M_a}^2\right]^{\left(\frac{\gamma}{\gamma - 1}\right)} \tag{2.34}$$

Tabulated values of gas flow functions [129] can be used in addition where the presence of shocks prevent the relations given by Equations (2.32) to (2.34) being used directly.

2.2.5 The Navier-Stokes equations for incompressible flow

For the case of incompressible flow, the fluid density remains constant. This allows the Navier-Stokes equations to be simplified before using non-dimensional scales. Neglecting

source terms, the *non-conservation* form of Equations (2.5) to (2.8) can be written in a detailed indicial fashion as

Continuity

$$\frac{\partial \rho}{\partial t} + \rho \frac{\partial u_i}{\partial x_i} + u_i \frac{\partial \rho}{\partial x_i} = 0 \tag{2.35}$$

Momentum

$$\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} + \frac{1}{\rho}\frac{\partial p}{\partial x_i} - \frac{1}{\rho}\frac{\partial \tau_{ij}}{\partial x_j} = 0 \tag{2.36}$$

Energy

$$\frac{\partial E}{\partial t} + u_j \frac{\partial E}{\partial x_j} - \frac{1}{\rho}\frac{\partial}{\partial x_i}\left(k\frac{\partial T}{\partial x_i}\right) + \frac{1}{\rho}\frac{\partial u_j p}{\partial x_j} - \frac{1}{\rho}\frac{\partial (u_j \tau_{ij})}{\partial x_j} = 0 \tag{2.37}$$

For an incompressible fluid the density is a constant, and the fluid is divergence free (i.e. $\nabla \cdot u = 0$). Thus, assuming a Newtonian fluid with constant viscosity and thermal conductivity, the governing system of equations for incompressible flow simplify to the following non-dimensional form

Continuity

$$\frac{\partial u_i^*}{\partial x_i^*} = 0 \tag{2.38}$$

Momentum

$$\frac{\partial u_i^*}{\partial t^*} = -u_j^* \frac{\partial u_i^*}{\partial x_j^*} + \frac{1}{Re}\frac{\partial^2 u_i^*}{\partial x_j^{*2}} - \frac{\partial p^*}{\partial x_i^*} \tag{2.39}$$

Energy

$$\frac{\partial T^*}{\partial t^*} = -u_j^* \frac{\partial T^*}{\partial x_j^*} + \frac{1}{RePr}\frac{\partial^2 T^*}{\partial x_j^{*2}} \tag{2.40}$$

where

$$x_i^* = \frac{x_i}{L}; \quad u_i^* = \frac{u_i}{u_\infty}; \quad t^* = \frac{t u_\infty}{L}; \quad p^* = \frac{p}{\rho u_\infty^2}; \quad T^* = \frac{(T - T_\infty)}{(T_w - T_\infty)} \tag{2.41}$$

are the non-dimensional scales [103] used to produce the dimensionless form of the equations. It is noticed at this point that, except for the pressure derivative, the simplified momentum equation takes on an identical format to the temperature (energy) equation. This will prove useful when applying a new approach discussed in the next chapter for scalar convection-diffusion equations to solve incompressible flow problems.

2.3 Galerkin, Petrov-Galerkin and high-order time schemes for the general scalar convection-diffusion equation

In this section, a number of discretisations are considered for solving Equation (2.1) for general scalar convection-diffusion problems. As a starting point, the global Galerkin method for spatial discretisation is applied first. This yields a discrete finite element form of the governing equation, and for pure diffusive problems this method is most optimal [19, 20, 21, 22, 23]. Using the standard global Galerkin method for convection dominated problems, leads to spurious oscillations in the computed solution. To overcome difficulties at elevated values of Péclet number, suitable stabilisation of the convection terms is required.

There are a number of well-known stabilisation methods commonly used in finite elements, which will eliminate or at least significantly reduce the oscillations caused by the convective-transport terms. Such methods include: the Petrov-Galerkin (PG) method [23, 25, 26]; the streamline-upwind Petrov-Galerkin (SUPG) method [27, 28]; the Galerkin/Least Squares (GLS) method [29, 30]; and the Finite Increment Calculus (FIC) method [31]. More recent developments of stabilizing techniques for advection-diffusion equations, include the subgrid scale (SGS) model [32, 33, 34, 35, 36] and the residual-free bubble function method [37]. For time dependent equations, high-order time schemes, such as the Taylor Galerkin (TG) method [38, 41] and the characteristic Galerkin (CG) [39, 42, 44] method, are commonly used. In addition, there are many methods which use discontinuous basis functions that are suitable for convection-dominated scalar transport [48]. In this thesis problems will be solved using both the SUPG and CG methods. For the SUPG method, special weighting for transient terms is not used. In addition, for this study third and higher order terms are neglected. This latter point is also applies to the CG method.

2.3.1 Standard FE spatial-discretisation (global Galerkin method)

The transient term of Equation (2.1) can be discretised using a simple Euler difference to give the explicit semi-discrete form, which is written as

$$\beta\frac{\phi^{n+1}-\phi^{n}}{\Delta t}=\beta\frac{\Delta\phi}{\Delta t}=-\frac{\partial F_i{}^{n}}{\partial x_i}+S_i \tag{2.42}$$

The standard finite element procedure involves the spatial discretisation of Ω into non-overlapping elements or subregions [20, 21]. The variation of the scalar variable over an n-node element is then approximated by:

$$\phi\approx\tilde{\phi}=\sum_{a=1}^{n}N_a\phi_a=\mathbf{N}\mathbf{\Phi} \tag{2.43}$$

where $\tilde{\phi}$ is an approximation of the scalar quantity and subscript a indicates nodes. Using the method of weighted residuals, Equation (2.42) is written as:

$$\beta\int_{\Omega}w_a\frac{\Delta\tilde{\phi}}{\Delta t}d\Omega=-\int_{\Omega}w_a\frac{\partial\tilde{F}_i{}^{n}}{\partial x_i}d\Omega+\int_{\Omega}w_a\tilde{S}_i d\Omega \tag{2.44}$$

where $w_a(\mathbf{x})$ are the n arbitrary weighting functions. In a standard Galerkin approach the weights, w_a, are chosen to be equal to the interpolation functions N_a. Performing integration by parts to the non-source term in the RHS of Equation (2.44) gives:

$$\beta\int_{\Omega}N_a\frac{\Delta\tilde{\phi}}{\Delta t}d\Omega=\int_{\Omega}\frac{\partial N_a}{\partial x_i}\tilde{F}_i{}^{n}d\Omega-\int_{\Gamma}N_a\tilde{F}_i{}^{n}d\Gamma n_i+\int_{\Omega}N_a\tilde{S}_i d\Omega \tag{2.45}$$

where n_i are the components of the outward boundary normal. Equation (2.45) is completed by the assembly of the individual elemental equation systems. Using boundary conditions appropriate to the type of problem being solved, a continuous solution is sought over the domain. The explicit matrix form of Equation (2.45) is written as:

$$\beta[\mathbf{M}]\{\Delta\mathbf{\Phi}\}=\Delta t([\mathbf{K}]\{\mathbf{\Phi}\}^{n}+\{\mathbf{f}\}^{n}) \tag{2.46}$$

An implicit solution procedure is also possible and is obtained by treating the flux term of Equation (2.42) implicitly to give

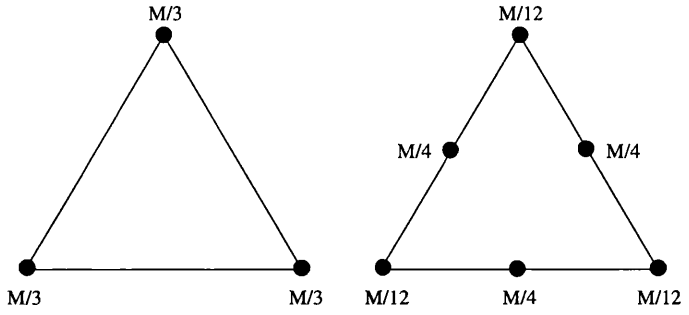


Figure 2.1: Mass lumping for linear and quadratic triangles

$$[\beta\mathbf{M} + \Delta t\mathbf{K}]\{\mathbf{\Phi}\}^{n+1} = [\mathbf{M}]\{\mathbf{\Phi}\}^{n} + \Delta t\{\mathbf{f}\}^{n} \tag{2.47}$$

Assuming the diffusion coefficient and velocity components are all constant, the matrices in Equations (2.46) and (2.47) are given by

$$\begin{aligned} \mathbf{M} &= \int_{\Omega} \mathbf{N}^{\mathbf{T}}\mathbf{N}d\Omega, \qquad \mathbf{K} = \int_{\Omega} \frac{\partial \mathbf{N}^{\mathbf{T}}}{\partial x_i}\left(u_i\mathbf{N} - k\frac{\partial \mathbf{N}}{\partial x_i}\right)d\Omega \quad and \\ \mathbf{f} &= \int_{\Omega} \mathbf{N}^{\mathbf{T}}\mathbf{N}d\Omega\{\mathbf{S_i}\}^{n} - \int_{\Gamma} \mathbf{N}^{\mathbf{T}}\left(u_i\mathbf{N} - k\frac{\partial \mathbf{N}}{\partial x_i}\right)n_i d\Gamma\{\mathbf{\Phi}\}^{n} \end{aligned} \tag{2.48}$$

In both the explicit and the implicit form (given by Equations (2.46) and (2.47) respectively), the mass matrix, $\mathbf{M}$, may either be lumped, or kept as a consistent mass matrix during the assembly of the individual elemental equations. To avoid ambiguity between consistent and lumped mass matrices, a lumped mass matrix is denoted by $\mathbf{M_L}$

For simple linear-triangular elements $\mathbf{M_L}$ can be found by summing up the mass-matrix coefficients of each row and placing along the diagonal. Lumping quadratic-triangular elements in this way gives zero mass terms on the matrix diagonal corresponding to the nodes on the element's vertices. This prevents the inversion of the $\mathbf{M_L}$. Figure (2.1) shows a recommended lumping procedure for quadratic elements [21], which is used in this work. This procedure considers one quadratic element as 4 linear elements for lumping the mass matrix.

2.3.2 Streamline-upwind Petrov-Galerkin (SUPG) method

The Petrov-Galerkin methodology, with $w_a \neq N_a$, was first used by Christie *et al.* [25] for the one-dimensional convection-diffusion equation, and by Heinrich et al. [26] for the general two-dimensional case. However, the direct application of the upwinding Petrov-Galerkin formulation to multidimensional problems suffers from excessive *cross-wind* diffusion perpendicular to the flow direction [23]. This problem is eliminated by the SUPG method of Brooks and Hughes [27, 28], where the added artificial diffusion is anisotropic - acting only in the direction of the streamlines. To have a consistent weighted residual equation, the modified weighting function is applied to all terms in the governing equation.

For SUPG stabilisation [28] in two-dimensions, the following weighting function is used

$$w_a = N_a + \frac{\alpha h}{2} \frac{u_i}{|\mathbf{u}|} \frac{\partial N_a}{\partial x_i} \tag{2.49}$$

where for linear finite element triangles the optimal value of α is given by

$$\alpha = \alpha_{opt} = \coth(Pe) - \frac{1}{(Pe)} \tag{2.50}$$

and the element Péclet number, Pe, is calculated from

$$Pe = \frac{|\mathbf{u}|h}{2k} = \frac{\sqrt{u_i u_i}h}{2k} \tag{2.51}$$

here h is the element-size, there are a number of ways of calculating element-size. For a known, unchanging velocity field, the element-size can be computed in the streamline direction and will only need to be performed once at the pre-processing stage.

The choice of the optimal stabilizing parameters for quadratic triangular elements follows the work of both Donea [41] and Codina [133] for one-dimensional quadratic elements. In 1D, separate stabilizing parameters are calculated for centre nodes and end nodes of the elements. For 2D triangular finite elements, the nodes at the element's vertices are treated as end nodes, and the mid-side nodes are treated as centre nodes. The optimal parameter for the mid-side node is the same as linear elements and is given by Equation (2.50). For the vertex nodes the optimal value of α is given by

$$\alpha_{opt} = \frac{(2Pe-1)+(-6Pe+7)e^{-2Pe}+(-6Pe-7)e^{-4Pe}+(2Pe+1)e^{-6Pe}}{(Pe+3)+(-7Pe-3)e^{-2Pe}+(7Pe-3)e^{-4Pe}-(Pe+3)e^{-6Pe}} \tag{2.52}$$

Using normal Galerkin weighting on the transient terms and the SUPG weighting for the RHS of Equation (2.44) gives

$$\beta \int_\Omega N_a \frac{\Delta\tilde{\phi}}{\Delta t} d\Omega = -\int_\Omega \left(N_a + \frac{\alpha h}{2}\frac{u_i}{|\mathbf{u}|}\frac{\partial N_a}{\partial x_i}\right)\left(\frac{\partial \tilde{F}_j}{\partial x_j}^n - \tilde{S}_j\right) d\Omega \tag{2.53}$$

Using the same notation as in the previous sub-section and neglecting higher-order terms, the explicit and implicit matrix forms of Equation (2.53) are written as:

$$\beta[\mathbf{M}]\{\Delta\mathbf{\Phi}\} = \Delta t\left([\mathbf{K}]\{\mathbf{\Phi}\}^n + [\mathbf{K_{stab1}}]\{\mathbf{\Phi}\}^n + \{\mathbf{f}\}^n + \{\mathbf{f_{stab1}}\}^n\right) \tag{2.54}$$

$$[\beta\mathbf{M} + \Delta t\mathbf{K}]\{\mathbf{\Phi}\}^{n+1} = [\mathbf{M}]\{\mathbf{\Phi}\}^n + \Delta t\left([\mathbf{K_{stab1}}]\{\mathbf{\Phi}\}^n + \{\mathbf{f}\}^n + \{\mathbf{f_{stab1}}\}^n\right) \tag{2.55}$$

where matrices $\mathbf{M}$, $\mathbf{K}$, and $\mathbf{f}$ are defined previously. The additional matrices used in Equations (2.54) and (2.55) are

$$\begin{aligned} \mathbf{K_{stab1}} &= \int_\Omega \left(\frac{\alpha h}{2}\frac{u_i}{|\mathbf{u}|}\right)\frac{\partial \mathbf{N^T}}{\partial x_i} u_j \frac{\partial \mathbf{N}}{\partial x_j} d\Omega \quad \textit{and} \\ \mathbf{f_{stab1}} &= \int_\Omega \left(\frac{\alpha h}{2}\frac{u_i}{|\mathbf{u}|}\right)\frac{\partial \mathbf{N^T}}{\partial x_i}\mathbf{N} d\Omega \{\mathbf{S_j}\}^n \end{aligned} \tag{2.56}$$

2.3.3 Characteristic Galerkin (CG) scheme

In the characteristic Galerkin (CG) method [39], the temporal derivative is discretised first - along the problem *characteristic*. A semi-discrete form is then produced that is fully self-adjoint, allowing the Galerkin spatial approximation to be the most optimal choice in FE methods [23]. Interest in characteristic based methods is huge [23, 39, 42, 44, 77, 78, 79, 80, 81, 82, 83, 84, 86, 88] and there are a number of variations of the CG method available. Here in this thesis, the *simple explicit* characteristic Galerkin procedure, as originally

introduced by Löhner, Morgan and Zienkiewicz [39], is employed. This attractive version appears to be the most popular, as it avoids the necessity of remeshing by using a local Taylor expansion to return to original coordinates. It has also provided the basis of the characteristic-based-split (CBS) algorithm [23, 42, 88] for computational fluid dynamics. A recent paper by Nithiarasu *et al.* [44] presents a comprehensive overview of the commonest characteristic based methods.

The implementation of the *simple explicit* CG method [39] is very straightforward and far more flexible than the SUPG described previously. Repeating Equation (2.1) for convenience

$$\beta\frac{\partial\phi}{\partial t} = -\frac{\partial}{\partial x_i}\left(u_i\phi - k\frac{\partial\phi}{\partial x_i}\right) + S_i = -\frac{\partial F_i}{\partial x_i} + S_i \tag{2.57}$$

Temporal discretisation of Equation (2.57), using the simple characteristic based procedure, gives the following form for the convection-diffusion problem

$$\beta\frac{\phi^{n+1}-\phi^n}{\Delta t} = -\left(\frac{\partial F_i}{\partial x_i} - S_i\right)^n + \frac{\Delta t}{2}u_k\frac{\partial}{\partial x_k}\left(\frac{\partial F_i}{\partial x_i} - S_i\right)^n + O(\Delta t^2) \tag{2.58}$$

Using the method of weighted residuals and neglecting higher order terms, Equation (2.58) is written as:

$$\beta\int_\Omega N_a\frac{\Delta\tilde{\phi}}{\Delta t}d\Omega = -\int_\Omega N_a\left(\frac{\partial\tilde{F}_i}{\partial x_i} - \tilde{S}_i\right)^n d\Omega + \int_\Omega N_a\frac{\Delta t}{2}u_k\frac{\partial}{\partial x_k}\left(\frac{\partial\tilde{F}_i}{\partial x_i} - \tilde{S}_i\right)^n d\Omega \tag{2.59}$$

In this form the Galerkin weighting, $w_a = N_a$, is optimal and will produce non-oscillatory results, subject to stability conditions. In this work the boundary term arising from integration by parts of the second-order differential term in Equation (2.59) was neglected [23]. In a familiar fashion to the previous sections, the explicit and implicit matrix forms of Equation (2.53) are written as:

$$\beta[\mathbf{M}]\{\Delta\mathbf{\Phi}\} = \Delta t\left([\mathbf{K}]\{\mathbf{\Phi}\}^n + [\mathbf{K_{stab2}}]\{\mathbf{\Phi}\}^n + \{\mathbf{f}\}^n + \{\mathbf{f_{stab2}}\}^n\right) \tag{2.60}$$

$$[\beta\mathbf{M} + \Delta t\mathbf{K}]\{\mathbf{\Phi}\}^{n+1} = [\mathbf{M}]\{\mathbf{\Phi}\}^{n} + \Delta t\left([\mathbf{K_{stab2}}]\{\mathbf{\Phi}\}^{n} + \{\mathbf{f}\}^{n} + \{\mathbf{f_{stab2}}\}^{n}\right) \tag{2.61}$$

where the new matrices are defined for the CG method as

$$\mathbf{K_{stab2}} = \int_{\Omega}\left(\frac{\Delta t}{2}u_i\right)\frac{\partial \mathbf{N^T}}{\partial x_i}u_j\frac{\partial \mathbf{N}}{\partial x_j}d\Omega \quad and$$
$$\mathbf{f_{stab2}} = \int_{\Omega}\left(\frac{\Delta t}{2}u_i\right)\frac{\partial \mathbf{N^T}}{\partial x_i}\mathbf{N}d\Omega\{\mathbf{S_j}\}^{n} \tag{2.62}$$

2.4 The characteristic based split (CBS) algorithm for compressible flow

In this section, the characteristic-based-split (CBS) algorithm of Zienkiewicz *et al.* [42, 87, 88], is given for a general compressible flow simulation. The CBS scheme is essentially a fractional time-stepping algorithm, based on an original FD velocity-projection scheme described by Chorin [89]. Since its original introduction to the FE method community, the CBS scheme been proved to be a successful numerical tool for the computation of a wide range of flow problems of compressible and incompressible nature [23, 43, 44, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99].

This section firstly gives details on the recommended temporal discretisation and splitting procedure [23, 42]. The temporal scheme starts with the splitting of the momentum equation - where the pressure-gradient term is completely removed. The remaining momentum equation for each velocity component is one of simple convection-diffusion for a scalar-variable, and is thus easily discretised using the CG method described previously. Solving these equations gives an intermediate velocity field, which is corrected using the pressure field. In the second step, the density/pressure field is solved independently, using an equation based on continuity. For the case of solving compressible flow problems, the isentropic approximation is avoided by solving the density field in the second step, and calculating pressure once both energy and density are known. In this case, after Step 2, the intermediate velocities are corrected in the third step, using the pressure field from the

previous time-step. Also for compressible flow, the energy coupling remains and a fourth and final step is required to solve the energy equation. Obtaining the energy field, allows the temperature and pressure fields to be computed, as well as other variables such as the local sonic velocity and Mach number. Once the full temporal scheme is discussed, details of the spatial discretisation procedure are given, to obtain a fully discrete form using the global (continuous) Galerkin method.

The characteristic-based scheme introduces consistent convection stabilisation, which is similar to the other available schemes such as SUPG and GLS [134, 135, 136]. In the CBS scheme though, the convection stabilisation terms are controlled by the time step, which in turn is based on the stability criteria involving the local element-size h. Details of both local time-stepping and element-size calculation, are given next. It is not clear, however, whether the local element-size calculation methods have any significant influence on the solution. The standard element-sizes commonly employed are calculated as part of the pre-processing stage and stored for use during the time stepping operation. Once calculated, these element-sizes are not altered during the time stepping process. This method of evaluating the element-sizes is computationally straight forward and inexpensive.

If only inviscid (convective) problems are considered, the time-step only needs to be based on the element-size in the direction of the streamline [128]. This may be completely different from the standard minimum h calculation [44]. In this section, the use of a flow dependent local element-size calculation in the streamline direction - for computing the local (nodal) time-steps, is proposed for calculations using CBS. Here, an updating of element-sizes is required at each time step during the transient stages of the calculation. Computing the element-size in the streamline direction is computationally more expensive than the standard method, especially for large scale problems. However, the advantages gained by using a stream lined element-size calculation, should not be overlooked. Finally, in addition to the effect of element-size calculation, this section concludes by addressing the issue of simulating inviscid flows at low Mach numbers by employing a variable smoothing approach.

2.4.1 Temporal-discretisation - the splitting of the momentum equation

Dropping the asterisk, *, for clarity and ignoring source terms the non-dimensional Equations (2.21), (2.22), and (2.23) respectively simplify to

Continuity

$$\frac{\partial \rho}{\partial t} = \frac{1}{c^2}\frac{\partial p}{\partial t} = -\frac{\partial U_i}{\partial x_i} \tag{2.63}$$

Momentum

$$\frac{\partial U_i}{\partial t} = -\frac{\partial}{\partial x_j}(u_j U_i) - \frac{\partial p}{\partial x_i} + \frac{1}{Re}\frac{\partial \tau_{ij}}{\partial x_j} \tag{2.64}$$

Energy

$$\frac{\partial}{\partial t}(\rho E) = -\frac{\partial}{\partial x_j}u_j(\rho E + p) + \frac{1}{Re}\frac{\partial}{\partial x_i}(\tau_{ij}u_j) + \frac{1}{RePr}\frac{\partial}{\partial x_i}(k\frac{\partial T}{\partial x_i}) \tag{2.65}$$

Direct application of the CG method to solve the momentum equation, Equation (2.64), is not easy to implement, since there is more than one characteristic speed. This is due to both the velocity and pressure variables appearing in the same equation. There are essentially two main alternatives given by Zienkiewicz *et al.* [23]. The first, is to keep the pressure gradient in the momentum equation, by treating it as a source-type quantity. The second, and most commonly used approach, is to remove it totally from the momentum equation and solve for an intermediate velocity field. This latter approach is taken here, with the explanation being based on the description of Nithiarasu *et al.* [44].

Rewriting the Equation (2.64) without the pressure gradient term and performing the CG procedure, gives an approximation for the intermediate velocity field:

$$\begin{aligned} \Delta U_i^\dagger = U_i^\dagger - U_i^n &= -\Delta t\left(\frac{\partial}{\partial x_j}(u_j U_i) - \frac{1}{Re}\frac{\partial \tau_{ij}}{\partial x_j}\right)^n \\ &+ \frac{\Delta t^2}{2}u_k^n\frac{\partial}{\partial x_k}\left(\frac{\partial}{\partial x_j}(u_j U_i)\right)^n + O(\Delta t^3) \end{aligned} \tag{2.66}$$

Here, a superscript † indicates an intermediate quantity. Also third-order spatial-derivative terms, due to the temporal discretisation of the deviatoric stresses, are neglected. The

intermediate velocity is only an approximation, it is corrected in Step 3 with the pressure gradient terms being calculated. The correction equation is simply

$$\Delta U_i = \Delta U_i^{\dagger} - \Delta t \frac{\partial p}{\partial x_i}^{n+\theta_2} + \frac{\Delta t^2}{2} u_k^n \frac{\partial}{\partial x_k} \frac{\partial p}{\partial x_i}^{n+\theta_2} \tag{2.67}$$

with

$$\frac{\partial p}{\partial x_i}^{n+\theta_2} = \left(\frac{\partial p}{\partial x_i}^{n} + \theta_2 \frac{\partial \Delta p}{\partial x_i} \right) \tag{2.68}$$

Here the parameter θ_2 has a range of $(0 \leq \theta_2 \leq 1)$, where $\theta_2 = 0$ for a fully explicit scheme, and $\theta_2 > 0$ for a semi implicit scheme

In the second step density is calculated. Using the continuity relation given by Equation (2.63), density and pressure can be evaluated from

$$\Delta \rho = \frac{1}{c^2} \Delta p = -\Delta t \frac{\partial U_i^{n+\theta_1}}{\partial x_i} \tag{2.69}$$

where $U_i^{n+\theta_1} = \theta_1 U_i^{n+1} + (1-\theta_1) U_i^n$, and the parameter θ_1 has a range of $(0.5 \leq \theta_1 \leq 1)$. Substituting Equation (2.67) into Equation (2.69), and neglecting high-order terms gives

$$\Delta \rho = \frac{1}{c^2} \Delta p = -\Delta t \left[\frac{\partial U_i}{\partial x_i}^{n} + \theta_1 \frac{\partial \Delta U_i^{\dagger}}{\partial x_i} - \Delta t \theta_1 \left(\frac{\partial^2 p}{\partial {x_i}^2}^{n} + \theta_2 \frac{\partial^2 \Delta p}{\partial {x_i}^2} \right) \right] \tag{2.70}$$

which is the density/pressure equation used in Step 2. It is noted here, that for compressible flow problems, the density field is usually solved for in Step 2 - with the pressure being obtained after the energy field is solved. This avoids having to make an isentropic assumption for compressible flow. The term containing the sonic velocity has, however, been purposely left in Equation 2.70, to help the description of an artificial compressibility scheme discussed in Chapter 3.

The calculation of the energy equation, Equation 2.65, follows Step 3. Applying the characteristic time discretisation gives Step 4, the final step of the scheme, as

$$\begin{aligned} \Delta(\rho E) = \Delta t \left(-\frac{\partial}{\partial x_j} u_j(\rho E + p) + \frac{1}{Re} \frac{\partial}{\partial x_i} (\tau_{ij} u_j) + \frac{1}{RePr} \frac{\partial}{\partial x_i} (k \frac{\partial T}{\partial x_i}) \right)^n \\ + \frac{\Delta t^2}{2} u_k^n \frac{\partial}{\partial x_k} \left[\frac{\partial}{\partial x_j} u_j(\rho E + p)^n \right] \end{aligned} \tag{2.71}$$

In summary, the steps of the CBS scheme are:

1. Solve equation (2.66) for ΔU_i^*.
2. Solve equation (2.70) for $\Delta\rho$ or Δp. (In compressible flow calculations, $\Delta\rho$ is normally solved for first and Δp after Step 4).
3. Solve equation (2.67) for ΔU_i.
4. Solve equation (2.71) for $\Delta\rho E$, then obtain T, c, etc.

2.4.2 Global Galerkin spatial-discretisation

In this sub-section Equations (2.66), (2.70), (2.67), and (2.71) are discretised in space using the standard global Galerkin finite element procedure. The computational domain is discretised into a mesh of non-overlapping elements. The variation of each of the variables in each element is approximated by the following standard spatial discretisation

$$\theta \approx \tilde{\theta} = \sum_{a=1}^{n} N_a \theta_a = \mathbf{N\Theta} \tag{2.72}$$

where $\tilde{\theta}$ is an approximation of a general variable θ, N is the interpolation (shape) function, and subscript a indicates a nodal term. Hence

$$U_i \approx \tilde{U}_i = \mathbf{NU_i}, \qquad \rho \approx \tilde{\rho} = \mathbf{N}\rho, \qquad p \approx \tilde{p} = \mathbf{Np}, \qquad \rho E \approx \tilde{\rho E} = \mathbf{NE} \tag{2.73}$$

An arbitrary choice of shape function can be made for the approximation of the velocity, pressure/density, and energy fields. In this work, it is assumed that all variable approximations use identical shape functions.

As mentioned before, the optimal choice of weighting function for the semi-discrete equation is Galerkin, $w_a = N_a$, when using a characteristic-based discretisation. The Galerkin weak form of intermediate momentum, Step 1, is

$$\int_\Omega N_a \Delta\tilde{U}_i^{\dagger} d\Omega = - \Delta t \int_\Omega N_a \left(\frac{\partial}{\partial x_j}(u_j U_i) - \frac{1}{Re}\frac{\partial \tau_{ij}}{\partial x_j} \right)^n d\Omega + \frac{\Delta t^2}{2} \int_\Omega N_a u_k^n \frac{\partial}{\partial x_k} \left(\frac{\partial}{\partial x_j}(u_j \tilde{U}_i) \right)^n d\Omega \quad (2.74)$$

Performing integration by parts on both second-order (stress and stabilisation) terms gives

$$\int_\Omega N_a \Delta\tilde{U}_i^{\dagger} d\Omega = - \Delta t \int_\Omega \left(N_a \frac{\partial}{\partial x_j}(u_j \tilde{U}_i) + \frac{1}{Re}\frac{\partial N_a}{\partial x_j}\tau_{ij} \right)^n d\Omega - \frac{\Delta t^2}{2} \int_\Omega \frac{\partial}{\partial x_k}(N_a u_k^n) \frac{\partial}{\partial x_j}(u_j \tilde{U}_i)^n d\Omega + \frac{\Delta t}{Re} \int_\Gamma N_a (\tau_{ij})^n n_j d\Gamma \quad (2.75)$$

The boundary terms from the integration by parts of the second-order stabilizing terms in the above equations are neglected as they are equal to zero on the boundaries [42]. Replacing continuous velocity fields with the approximations given by Equation (2.73), leads to the final matrix form

$$[\mathbf{M_1}]\{\Delta \mathbf{U_i}^{\dagger}\} = \Delta t \left(-[\mathbf{C_1}]\{\mathbf{U_i}\}^n - \{\mathbf{K}_\tau\}^n - [\mathbf{K_1}]\{\mathbf{U_i}\}^n + \{\mathbf{f_1}\}^n\right) \quad (2.76)$$

where

$$\mathbf{M_1} = \int_\Omega \mathbf{N^T N} d\Omega, \quad \mathbf{C_1} = \int_\Omega \mathbf{N^T} \frac{\partial}{\partial x_j} \mathbf{N} u_j d\Omega, \quad \mathbf{K}_\tau = \frac{1}{Re} \int_\Omega \frac{\partial}{\partial x_j} \mathbf{N^T} \tau_{ij} d\Omega$$
$$\mathbf{K_1} = \frac{\Delta t}{2} \int_\Omega u_k \frac{\partial}{\partial x_k} \mathbf{N^T} \frac{\partial}{\partial x_j} \mathbf{N} u_j d\Omega \quad \mathbf{f_1} = \frac{1}{Re} \int_\Gamma \mathbf{N^T} \tau_{ij} n_j d\Gamma \quad (2.77)$$

The Galerkin weak form of density/pressure equation, Step 2 is

$$\int_\Omega N_a \Delta\tilde{\rho} d\Omega = \int_\Omega N_a \frac{1}{c^2} \Delta\tilde{p} d\Omega = - \Delta t \int_\Omega N_a \frac{\partial}{\partial x_i} \left[\tilde{U_i}^n + \theta_1 \Delta\tilde{U_i}^{\dagger} - \Delta t \theta_1 \left(\frac{\partial \tilde{p}}{\partial x_i}^n + \theta_2 \frac{\partial \Delta\tilde{p}}{\partial x_i} \right) \right] d\Omega \quad (2.78)$$

Integration by parts of the RHS gives

$$\begin{aligned}\int_\Omega N_a \Delta\tilde{\rho} d\Omega &= \int_\Omega N_a \frac{1}{c^2}\Delta\tilde{p} d\Omega = \\ &+ \Delta t \int_\Omega \frac{\partial N_a}{\partial x_i}\left[\tilde{U_i}^n + \theta_1 \Delta\tilde{U_i}^\dagger - \Delta t\theta_1\left(\frac{\partial\tilde{p}}{\partial x_i}^n + \theta_2\frac{\partial\Delta\tilde{p}}{\partial x_i}\right)\right] d\Omega \\ &- \Delta t \int_\Gamma N_a\left[\tilde{U_i}^n + \theta_1\Delta\tilde{U_i}^\dagger - \Delta t\theta_1\left(\frac{\partial\tilde{p}}{\partial x_i}^n + \theta_2\frac{\partial\Delta\tilde{p}}{\partial x_i}\right)\right] n_i d\Gamma \end{aligned} \tag{2.79}$$

Spatial approximation leads to the final matrix form of Step 2 as

$$\begin{aligned}[\mathbf{M_1}]\{\Delta\rho\} = [\mathbf{M_2}]\{\Delta\mathbf{p}\} &= \Delta t\left(\mathbf{C_2}\left(\{\mathbf{U_i}\}^n + \theta_1\{\Delta\mathbf{U_i}^\dagger\}\right) - \{\mathbf{f_2}\}\right) \\ &- \Delta t^2\theta_1\left([\mathbf{K_2}]\left(\{\mathbf{p}\}^n + \theta_2\{\Delta\mathbf{p}\}\right) + \{\mathbf{f_3}\}\right)\end{aligned} \tag{2.80}$$

where

$$\mathbf{M_2} = \int_\Omega \mathbf{N^T}\frac{1}{c^2}\mathbf{N} d\Omega, \quad \mathbf{C_2} = \int_\Omega\left(\frac{\partial}{\partial x_i}\mathbf{N^T}\right)\mathbf{N} d\Omega, \quad \mathbf{K_2} = \int_\Omega \frac{\partial}{\partial x_i}\mathbf{N^T}\frac{\partial}{\partial x_i}\mathbf{N} d\Omega,$$
$$\mathbf{f_2} = \int_\Gamma \mathbf{N^T N}\left(\{\mathbf{U_i}\}^n + \theta_1\{\Delta\mathbf{U_i}^\dagger\}\right) n_i d\Gamma, \quad \mathbf{f_3} = \int_\Gamma \mathbf{N^T}\frac{\partial}{\partial x_i}\mathbf{N}\left(\{\mathbf{p}\}^n + \theta_2\{\Delta\mathbf{p}\}\right) n_i d\Gamma \tag{2.81}$$

The weak form of momentum correction, Step 3, is

$$\begin{aligned}\int_\Omega N_a\Delta\tilde{U_i} d\Omega &= \int_\Omega N_a \Delta\tilde{U_i}^\dagger \\ &- \Delta t\left[\int_\Omega N_a\frac{\partial}{\partial x_i}\left(\tilde{p} + \theta_2\Delta\tilde{p}\right) d\Omega - \frac{\Delta t}{2}\int_\Omega N_a u_k^n \frac{\partial}{\partial x_k}\frac{\partial\tilde{p}}{\partial x_i}^n d\Omega\right]\end{aligned} \tag{2.82}$$

and again performing integration by parts gives

$$\begin{aligned}\int_\Omega N_a\Delta\tilde{U_i} d\Omega &= \int_\Omega N_a \Delta\tilde{U_i}^\dagger d\Omega \\ &+ \Delta t\left[\int_\Omega \frac{\partial N_a}{\partial x_i}\left(\tilde{p}^n + \theta_2\Delta\tilde{p}\right) d\Omega - \frac{\Delta t}{2}\int_\Omega \frac{\partial N_a}{\partial x_k} u_k^n \frac{\partial\tilde{p}}{\partial x_i}^n d\Omega\right] \\ &- \Delta t\left[\int_\Gamma N_a\left(\tilde{p}^n + \theta_2\Delta\tilde{p}\right) n_i d\Gamma\right]\end{aligned} \tag{2.83}$$

Inserting the spatial approximations gives the final matrix form of Step 3 as

$$[\mathbf{M_1}]\{\Delta \mathbf{U_i}\} = [\mathbf{M_1}]\{\Delta \mathbf{U_i}^{\dagger}\} + \Delta t\left([\mathbf{C_2}]\left(\{\mathbf{p}\}^n + \theta_2\{\Delta \mathbf{p}\}\right) - [\mathbf{K_3}]\{\mathbf{p}\}^n - \mathbf{f_4}\right) \tag{2.84}$$

where

$$\mathbf{K_3} = \left(\frac{\Delta t}{2}\right)\int_\Omega \frac{\partial}{\partial x_k}\mathbf{N^T} u_k \frac{\partial}{\partial x_j}\mathbf{N} d\Omega, \quad \mathbf{f_4} = \int_\Gamma \mathbf{N^T N}\left(\{\mathbf{p}\}^n + \theta_2\{\Delta \mathbf{p}\}\right) n_i d\Gamma, \tag{2.85}$$

Finally the weak form of energy equation, Step 4,

$$\begin{aligned}\int_\Omega N_a \Delta(\rho\tilde{E}) d\Omega &= \Delta t \int_\Omega N_a \left(-\frac{\partial}{\partial x_j}u_j((\rho\tilde{E}) + \tilde{p}) + \frac{1}{Re}\frac{\partial}{\partial x_i}(\tau_{ij}u_j) + \frac{1}{RePr}\frac{\partial}{\partial x_i}(k\frac{\partial T}{\partial x_i})\right)^n d\Omega \\ &+ \frac{\Delta t^2}{2}\int_\Omega N_a u_k^n \frac{\partial}{\partial x_k}\left[\frac{\partial}{\partial x_j}u_j((\rho\tilde{E}) + \tilde{p})^n\right]^n d\Omega \end{aligned} \tag{2.86}$$

Using integration by parts

$$\begin{aligned}\int_\Omega N_a \Delta(\rho\tilde{E}) d\Omega = &- \Delta t \int_\Omega N_a \frac{\partial}{\partial x_j}u_j((\rho\tilde{E}) + \tilde{p})^n d\Omega \\ &- \frac{\Delta t}{Re}\left(\int_\Omega \frac{\partial}{\partial x_i}N_a(\tau_{ij}u_j)d\Omega + \frac{1}{Pr}\int_\Omega \frac{\partial}{\partial x_i}N_a(k\frac{\partial \tilde{T}}{\partial x_i})d\Omega\right)^n \\ &- \frac{\Delta t^2}{2}\int_\Omega u_k \frac{\partial}{\partial x_k}N_a \frac{\partial}{\partial x_j}u_j((\rho\tilde{E}) + \tilde{p})^n d\Omega \\ &+ \frac{\Delta t}{Re}\int_\Gamma N_a(\tau_{ij}u_j + \frac{1}{Pr}k\frac{\partial T}{\partial x_i})^n n_i d\Gamma \end{aligned} \tag{2.87}$$

Spatial approximations of the variables give

$$[\mathbf{M_1}]\{\Delta\rho\mathbf{E}\} =$$

$$\Delta t\left([\mathbf{C_1}]\left(\{\rho\mathbf{E}\} + \{\mathbf{p}\}\right)^n - \{\mathbf{K_{\tau 2}}\}^n - [\mathbf{K_4}]\{\mathbf{T}\} - [\mathbf{K_1}]\left(\{\rho\mathbf{E}\} + \{\mathbf{p}\}\right)^n - \mathbf{f_5}\right) \tag{2.88}$$

where

$$\begin{aligned} \mathbf{f_5} &= \frac{1}{Re}\int_\Gamma \mathbf{N^T}(\tau_{ij}u_j + \frac{1}{Pr}k\frac{\partial T}{\partial x_i})n_i d\Gamma, \\ \mathbf{K}_{\tau 2} &= \frac{1}{Re}\int_\Omega \frac{\partial}{\partial x_j}\mathbf{N^T}\tau_{ij}u_j d\Omega, \qquad \mathbf{K_4} = \frac{k}{RePr}\mathbf{K_2} \end{aligned} \tag{2.89}$$

The first Characteristic Galerkin step is always in explicit form, however Step 2 can be either explicit or implicit depending on the choice of the the parameter θ_2. For the fully explicit form, $(0.5 \leq \theta_1 \leq 1)$ and $\theta_2 = 0$, is commonly used.

Further, at very large values of Re, the diffusive terms become negligible. The fully explicit form for solving the Euler equation set is summarised as

Step 1

$$[\mathbf{M_1}]\{\Delta \mathbf{U_i}^\dagger\} = \Delta t\left([\mathbf{C_1}]\{\mathbf{U_i}\}^n - [\mathbf{K_1}]\{\mathbf{U_i}\}^n - \{\mathbf{f_1}\}^n\right) \tag{2.90}$$

Step 2

$$\begin{aligned} [\mathbf{M_1}]\{\Delta\rho\} = [\mathbf{M_2}]\{\Delta \mathbf{p}\} &= \Delta t\left(\mathbf{C_2}\left(\{\mathbf{U_i}\}^n + \theta_1\{\Delta \mathbf{U_i}^\dagger\}\right) - \{\mathbf{f_2}\}\right) \\ &- \Delta t^2\theta_1\left([\mathbf{K_2}]\{\mathbf{p}\}^n + \{\mathbf{f_3}\}\right) \end{aligned} \tag{2.91}$$

Step 3

$$[\mathbf{M_1}]\{\Delta \mathbf{U_i}\} = [\mathbf{M_1}]\{\Delta \mathbf{U_i}^\dagger\} + \Delta t\left([\mathbf{C_2}]\{\mathbf{p}\}^n - [\mathbf{K_3}]\{\mathbf{p}\}^n - \{\mathbf{f_4}\}\right) \tag{2.92}$$

Step 4

$$[\mathbf{M_1}]\{\Delta\rho\mathbf{E}\} = \Delta t\left([\mathbf{C_1}]\left(\{\rho\mathbf{E}\} + \{\mathbf{p}\}\right)^n - [\mathbf{K_1}]\left(\{\rho\mathbf{E}\} + \{\mathbf{p}\}\right)^n - \{\mathbf{f_5}\}\right) \tag{2.93}$$

Where all matrices shown are the same as the terms appearing in Equations 2.76, 2.80, 2.84, and 2.88 to solve the Navier-Stokes equation set. In both equation sets, the mass matrix $\mathbf{M_1}$ is lumped, to reach a faster steady state and facilitates the practice of local time-stepping discussed next.

2.4.3 Local time-stepping and calculation of element-size

If only a steady-state solution is required, then the efficiency of explicit time-stepping algorithms, such as the CBS scheme, can be greatly increased by implementing a local time-stepping approach [44, 88]. Here, each node in the mesh has its own individually calculated minimum time-step, based on local element-size and flow conditions. The process results in different time-step values being used at different nodes. Using local time-stepping gives a more rapid convergence, and fewer iterations are required, than when using a globally minimum time-step to reach steady-state conditions.

The local time-step at each node a, is calculated from

$$\Delta t = Sf \times \min(\Delta t_{convection}, \Delta t_{diffusion}) \tag{2.94}$$

here, Sf is a factor of safety which can normally vary between 0.1 and 1.0, but its value depends on the problem being solved and specific mesh used. For the the case of general compressible flow the local convective and diffusive time-steps are calculated from

$$\Delta t_{convection} = \frac{h_a}{|\mathbf{u}| + c} \tag{2.95}$$

and

$$\Delta t_{diffusion} = \frac{{h_a}^2}{2Re} \tag{2.96}$$

respectively. Here h_a is the minimum element-size value for node a. Additionally, c and $\mathbf{u}$ are sonic and fluid velocities, and Re is the Reynolds number. Generally, h_a is calculated in two-dimensions (see Figure 2.2) as the minimum mid-height of all the surrounding connecting elements, e, i.e.

$$h_a = min\left(2area/oppositesidelength\right)_e \tag{2.97}$$

When solving the Euler equations, (Equations 2.90, 2.91, 2.92, and 2.93) viscous terms are not present. In this case the local time-step depends solely on Equation 2.95. It is possible in this case to improve the accuracy of the h_a calculation by taking into account the

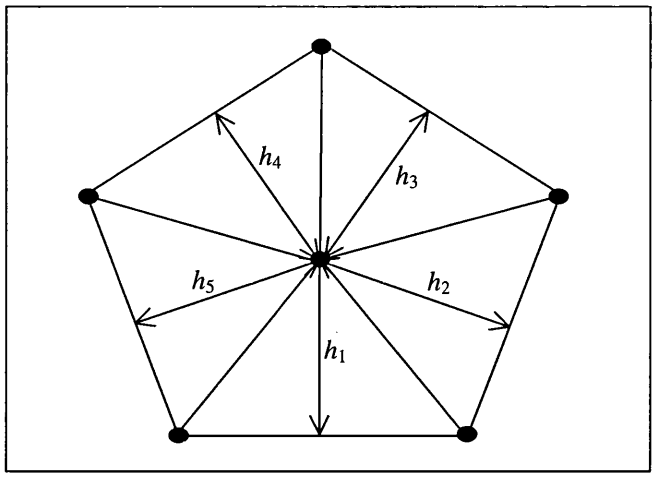


Figure 2.2: Standard element-size calculation.

direction of the resultant velocity at node a, as shown in Figure 2.3. The following equation, given by Shakib for SUPG [128], is used to calculate h_a in the streamline direction:

$$h_a = min\left(\frac{2}{\sum_{n=1}^{3} |\mathbf{s}_n \nabla N_n|}\right)_e \tag{2.98}$$

where $\mathbf{s}_n$ is the unit vector in the streamline direction and N_n is the shape function, at each node n of an element e. It is noted that h_a is still the minimum value of element-size amongst the elements connected to node a, but the values of h_a for each of the elements are influenced by the direction of the convective velocity at node a.

Using the proposed local element-size calculation, given by Equation (2.98), for each node should give higher accuracy than the standard element calculation since the value of h_a may be considerably different than the minimum mid-height calculation given by Equation (2.97). Computing the element-size in the streamline direction is computationally more expensive than the standard method, especially for large scale problems. However, the advantage of increasing accuracy, when using a streamlined element size calculation should be considered.

A recommended calculation for the steady-state tolerance, is given by Nithiarasu [44] as

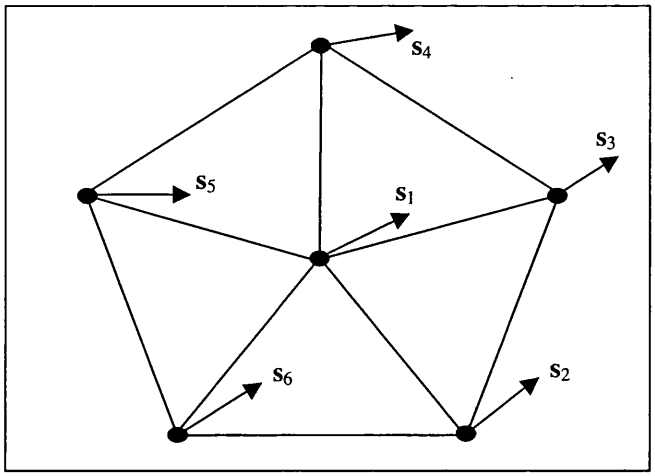


Figure 2.3: Streamline element-size calculation is based on unit vector in the streamline direction.

$$error = \sqrt{\frac{\sum_{a=1}^{nodes}(\frac{{\rho_a}^{n+1}-{\rho_a}^{n}}{\Delta t})^2}{\sum_{a=1}^{nodes}({\rho_a}^{n+1})^2}} \tag{2.99}$$

The above equation is non-dimensionalised using a characteristic time scale of $(u_\infty/L)^{-1}$. The non-dimensional tolerance is reduced to $e \leq 10^{-5}$ to obtain a steady state.

2.4.4 Shock capturing viscosity

At transonic and supersonic speeds, an additional shock capturing dissipation is necessary to capture shocks and to smooth local oscillations in the vicinity of shocks. A recommended shock capturing viscosity method for compressible inviscid flow problems, given by [91, 127], is adopted here. For a scalar variable field ϕ the smoothed values, ϕ_s are computed by

$$\left[\frac{\phi_s^{n+1}-\phi^{n+1}}{\Delta t}\right] = \mathbf{M_L^{-1}}\frac{C_e S_e}{\Delta t_e}\left(\mathbf{M}-\mathbf{M_L}\right)\phi^n \tag{2.100}$$

Here S_e is the element 'pressure switch' and is taken to be the mean of the element nodal switches S_i, which in turn are given by

$$S_i = \frac{|\Sigma_e (p_i - p_k)|}{\Sigma_e |p_i - p_k|} \tag{2.101}$$

C_e is a user specified constant ranging from 0.0 to 2.0 and Δt_e is the local element time step.

2.4.5 Variable smoothing

Residual based methods been applied previously by Nithiarasu *et al.* [91], to improve the performance of a residual-based shock capturing method in the hypersonic region. What is proposed here, however, is novel. To solve compressible flow problems at Mach numbers $Ma \leq 0.8$, without removing the coupling between energy and the other transport variables, a variable smoothing procedure is used in the place of artificial shock capturing diffusion. This gives a flexible compressible CBS scheme which can be used in all flow regimes from sub-sonic to hypersonic.

The following equation defines the variable smoothing applied to the transport variables, $\{\mathbf{\Phi}\}$, on a two dimensional grid

$$\{\mathbf{\Phi}\} = \left[\frac{1}{1+0.5\alpha}\{\mathbf{\Phi}\} + \frac{\alpha}{1+0.5\alpha}\mathbf{M_L^{-1}}[\mathbf{M} - \mathbf{M_D}]\{\mathbf{\Phi}\}\right] \tag{2.102}$$

where α is a variable smoothing parameter varies between 0 and 0.05, $\mathbf{M}$ is the consistent mass matrix, $\mathbf{M_D}$ is the consistent mass matrix without non-diagonal terms and $\mathbf{M_L}$ is the lumped mass matrix. By increasing α the weighting on the node in question is decreased while the influence of the surrounding nodes is increased.

2.5 Summary

In the present chapter, the most general form of the scaler convection-diffusion equation and the Navier-Stokes equation set have been given, along with constitutive relations for closure. This was followed by the presentation of non-dimensional quantities and parameters for obtaining the dimensionless Navier-Stokes equations for compressible and incompressible flow problems. In addition, relations for computing the stagnation values of transport variables were also given.

The second section outlined the Galerkin form of the method of weighted residuals, which forms the basis of finite element analysis. First-order-time Petrov-Galerkin and second-order-time characteristic Galerkin global finite methods, were then reviewed for solving convection dominated problems.

The CBS algorithm, for solving compressible flow problems, was discussed in the last section. In particular, details were given for both temporal and spatial discretisation. This includes using a shock capturing viscosity method for transonic and supersonic flow. A more accurate time-step calculation based on the element-size in the streamline direction was discussed. Finally a direct variable smoothing procedure for low Mach number flows was described.

Chapter 3

Proposed Locally Conservative Galerkin (LCG) Method for Solving Conservation Equations

3.1 Introduction

As mentioned in Chapter 1, a locally conservative Galerkin (LCG) scheme has been recently introduced by Nithiarasu [1], for solving convection-diffusion problems. The LCG method described in the paper, proved that an element-by-element solution is possible via a postprocessing of edge-fluxes at every time step. This gives the LCG method similar advantages to DG methods without the need for solving additional variables. The method uses the property of local conservation at steady state conditions in order to define a numerical flux at element boundaries. This allows the computational domain to be broken down into a series of elemental domains, each with its own Neumann-type boundary conditions based on the computed numerical flux. In the very first LCG paper [1] only linear elements were considered, and a small number of relatively simple problems were solved. However, its appeal prompted a significant research effort to be implemented in this thesis work.

A large effort has been made in this thesis in understanding how the LCG method works. This chapter starts by dedicating Section 3.2 to give the details on basics of the

standard LCG procedure. Here, a simple conservation equation - for general convection-diffusion transport, is used to illustrate the fundamentals of the LCG spatial-discretisation method. In the following section, Section 3.3, attention is turned towards deriving the numerical edge-flux calculation procedure. Local conservation plays an important role in developing a small post-processing calculation. This calculation, which is implemented at the end of each time-step, gives the required numerical flux to provide continuity between neighbouring elements at the next time-step. The original procedure described for linear elements [1] is expanded, along with proposals for increasing the accuracy of the flux by using higher-order elements. Details are given for implementing this flux in both a lumped and consistent form.

It was considered important, that the effective nodal equation obtained for the LCG method, should be analysed. In Section 3.4, a direct comparison of the nodal equation is made between the LCG method and the standard Global Galerkin method. Both a 1D and a 2D analysis of the LCG method are considered, both of which give surprising similarities to the global Galerkin method. When a lumped mass is used and elements are of a similar size, the two different methods yield an identical nodal equation - even for unstructured meshes. In the LCG method however, once the averaging procedure is complete, the internal edge fluxes cancel and a residual flux is left on the global boundary. This gives the LCG method explicit global conservation - even in the absence of Neumann-type boundary conditions.

Stabilised LCG schemes are implemented in Section 3.5. Established methods used in the global Galerkin context are considered. These include both the SUPG method of Brooks and Hughes [28] and the *simple explicit* CG method of Löhner *et al.* [39]. In Chapter 2 these methods were discussed in detail for the global Galerkin method. Here it is shown how they can be used in an elemental formation to give suitable stabilisation for solving convection dominated problems using the LCG method. Both linear and quadratic triangular element applications are discussed.

Before closure, the last section - Section 3.6, outlines the development of a matrix-free LCG-CBS method - used for the solution of the incompressible flow equations. Again both linear and quadratic versions are implemented. The fully explicit CBS time discretisation with artificial compressibility [76, 82, 96, 97, 98, 112, 113, 114], is used to obtain a

semi-discrete form, before spatially discretizing the equations, using the LCG method. This section closes with details on addressing unsteady flows, using a reliable dual time-stepping CBS procedure [96, 115], giving a LCG-CBS scheme, suitable for transient flow simulation.

3.2 Locally conservative Galerkin (LCG) discretisation

In this section a novel and attractive LCG approach for spatially discretizing the convection-diffusion equation, is introduced. For simplicity, it is assumed that the governing equation is describing a diffusion-dominated problem. Thus, for this section, convective-stabilisation terms are not used. In addition source terms are also neglected. Details on using source terms and existing stabilisation methods for convection-dominated problems, within the LCG frame-work, is discussed in a later section of this chapter.

The governing equation for the convective-diffusive transport of a scalar variable ϕ, is given by Equation (2.1). In the absence of source terms this simplifies to

$$\beta\frac{\partial\phi}{\partial t}+\frac{\partial F_i}{\partial x_i}=0 \tag{3.1}$$

with flux term, F_i, given (by Equation (2.2)) as

$$F_i=\left(u_i\phi-k\frac{\partial\phi}{\partial x_i}\right) \tag{3.2}$$

where k is the diffusion coefficient and u_i are the constant velocity components. A forward (Euler) difference replaces the LHS term of Equation (3.1) to give the semi-discrete form

$$\beta\frac{\phi^{n+1}-\phi^n}{\Delta t}=-\frac{\partial F_i}{\partial x_i}^n \tag{3.3}$$

The terms (n) and $(n+1)$ represent the current time level and the next time level respectively.

The first stages of the LCG procedure for a simple convection-diffusion equation [1], follows the familiar global Galerkin method of weighted residuals, up to the point of global elemental assembly. However, the aim here is to obtain a discrete equation for solving over individual elements. Using the same approximation (given by Equation (2.43)) for the global Galerkin method, the variation of the scalar variable over an element is

$$\phi \approx \tilde{\phi} = \sum_{a=1}^{n} N_a \phi_a = \mathbf{N}\mathbf{\Phi} \tag{3.4}$$

and a solution to the following residual equation is to be found

$$\beta \int_{\Omega} w_a \frac{\Delta\tilde{\phi}}{\Delta t} d\Omega = -\int_{\Omega} w_a \frac{\partial \tilde{F}_i}{\partial x_i}^n d\Omega \tag{3.5}$$

Performing integration by parts to the RHS term of Equation (3.5) gives

$$\beta \int_{\Omega} w_a \frac{\Delta\tilde{\phi}}{\Delta t} d\Omega = \int_{\Omega} \frac{\partial w_a}{\partial x_i} \tilde{F}_i^n d\Omega - \int_{\Gamma} w_a \tilde{F}_i^n d\Gamma n_i \tag{3.6}$$

where n_i are the components of the outward boundary normal. As mentioned in the previous chapter, the standard global Galerkin method uses the interpolation functions as the weighting functions. Element contributions of Equation (3.6) are then assembled into a global matrix system and a continuous solution is retrieved.

The LCG method still uses the Galerkin approximation (i.e. $w_a = N_a$), but solves the discrete equation system elementally. In the LCG method the variable and its fluxes are explicitly conserved over each of the individual elements. This is achieved through the calculation of an accurate flux at the element boundaries. The computed flux also ensures that continuity between neighbouring elements is maintained. Such a process is equivalent to treating the global domain as a group of elemental sub-domains; each with its own set of time-dependent Neumann boundary conditions prescribed at each time step.

With an accurate value of flux being available at the element edges, Equation (3.6) is rewritten in the LCG form for solving over an elemental sub-domain Ω_e as

$$\beta \int_{\Omega_e} \mathbf{N}^{\mathbf{T}}\mathbf{N} \frac{\Delta\mathbf{\Phi}}{\Delta t} d\Omega_e = \int_{\Omega_e} \frac{\partial \mathbf{N}^{\mathbf{T}}}{\partial x_i} \tilde{F}_i^n d\Omega_e - \int_{\Gamma_e} \mathbf{N}^{\mathbf{T}} \hat{\tilde{F}}_i^n d\Gamma_e n_i \tag{3.7}$$

where the subscript e, stands for an element. Here, the boundary integral term of Equation (3.7) is approximated with a computed value of numerical-flux, $\hat{\tilde{F}}_i$, across the element edges. A simple procedure for estimating this flux is described in the proceeding section.

The matrix form of Equation (3.7) is written as

$$\beta[\mathbf{M_e}]\{\Delta\mathbf{\Phi}\} = \Delta t([\mathbf{K_e}]\{\mathbf{\Phi}\}^n + \{\hat{\mathbf{f}}_\mathbf{e}\}^n) \tag{3.8}$$

and the system of simultaneous equations are solved over individual elements independently of surrounding element equation sets. Equation (3.8) is the explicit, elemental-matrix form of the conservation Equation (3.1). As with the global Galerkin method, an implicit solution procedure is obtained by treating the flux term in Equation (3.1) implicitly, thus

$$[\beta \mathbf{M_e} + \Delta t \mathbf{K_e}]\{\mathbf{\Phi}\}^{n+1} = [\mathbf{M_e}]\{\mathbf{\Phi}\}^n + \Delta t \{\hat{\mathbf{f}}_\mathbf{e}\}^n \tag{3.9}$$

In both the explicit and implicit form (given by Equations (3.8) and (3.9) respectively) the elemental mass and 'stiffness' matrices are given as

$$\mathbf{M_e} = \int_{\Omega_e} \mathbf{N^T N} d\Omega_e, \qquad \mathbf{K_e} = \int_{\Omega_e} \frac{\partial \mathbf{N^T}}{\partial x_i} \left(u_i \mathbf{N} - k \frac{\partial \mathbf{N}}{\partial x_i} \right) d\Omega_e \tag{3.10}$$

For linear and quadratic triangular elements, the maximum sizes of $[\mathbf{M_e}]$ and $[\mathbf{K_e}]$ are only (3×3) and (6×6), respectively - thus greatly reducing storage. The elemental mass matrix, $[\mathbf{M_e}]$, may either be lumped, or kept as a consistent mass matrix. The mass lumping procedures for linear and quadratic triangular-elements were discussed in the last chapter, for the global Galerkin method. These lumping techniques, which are summarized graphically in Figure (2.1), also apply here. The definition of forcing vector, $\{\hat{\mathbf{f}}_\mathbf{e}\}$, that contains the numerical flux will be given in the proceeding section. It is noted here, however, that $\{\hat{\mathbf{f}}_\mathbf{e}\}$ is always evaluated at the nth time level regardless of the time discretisation.

Due to the individual discrete equation system for each element, the nodal solution obtained will be non-unique. A situation of multiple solutions being calculated at global mesh nodes is eliminated by taking an arithmetic mean of nodal values obtained from different elemental contributions to a node. For a node a, connected to a number of different elements, ne, there will be ne elemental values of the scalar variable ϕ at node a. The average nodal value of ϕ_a is simply

$$\phi_a = \frac{1}{ne} \sum_{e=1}^{ne} (\phi_a)_e \tag{3.11}$$

The averaging procedure is necessary, as it provides a unique continuous solution throughout the domain. A continuous solution is needed globally to make a post-processing step, (discussed in the next section) for calculating the interface flux.

3.3 Local conservation and calculation of edge flux for the LCG method

In this section, local conservation is used to define the numerical flux calculation that provides continuity between elements. The numerical flux on the element edge generally has two components, one convective and one diffusive. The latter is based on approximating the scalar variable's gradients using a post-processing calculation made at the end of each time-step. When using linear elements the flux calculation is based on an element-averaged diffusive flux. By using quadratic interpolation functions, accuracy is improved by allowing the diffusive fluxes to be nodally calculated. This section finishes with details on how to simply retrieve the gradients for both linear and quadratic elements.

3.3.1 Conservation properties of the LCG method

Conservation at a local or element-wise level emanates from the property that the weighting function can be set exactly to unity on the element of interest and zero elsewhere [125]. At steady state conditions, the elemental form of Equation (3.6) reduces to:

$$\int_{\Gamma_e} w_a \hat{F_i}^n d\Gamma_e n_i = \int_{\Omega_e} \frac{\partial w_a}{\partial x_i} F_i^n d\Omega_e \tag{3.12}$$

Noting that for the LCG method, in each individual elemental sub-domain Ω_e, we can define the weighting function for node a as

$$\begin{aligned} w_a(\mathbf{x}) &= 1 \quad \mathbf{x} \in \Omega_e \\ w_a(\mathbf{x}) &= 0 \quad otherwise \end{aligned} \tag{3.13}$$

a condition for conservation of ϕ and its gradients is found:

$$\int_{\Gamma_e} F_i^n d\Gamma_e n_i = 0 \tag{3.14}$$

A simple procedure to enforce the flux boundary condition given by Equation (3.14) at the element edges, is to make the flux crossing a common edge, shared by two elements, to be

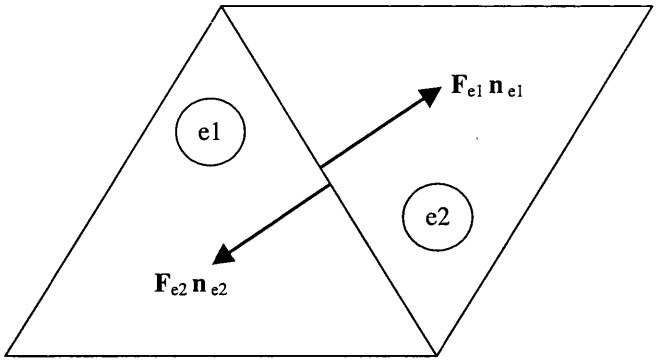


Figure 3.1: Flux crossing a common edge between two elements.

equal and to act in opposing directions as shown in Figure (3.1). Using the computed edge flux on the element boundaries of adjoining elements, the following condition is enforced:

$$F_{e1}n_{e1} = F_{e2}n_{e2} \tag{3.15}$$

where $F_{e1} = F_{e2}$ and n_{e1} and n_{e2} are the outward normals from the edges of the respective elements. The subscripts are defined in Figure (3.1).

3.3.2 Calculation of edge fluxes

With the above condition for local conservation obtained, the numerical flux can be realized. All that is needed to enforce the condition given by Equation (3.15), is a small post-processing calculation to be made at the end of each time-step. In a previous paper [1] it was proposed that nodal values of the scalar variable, ϕ, and its gradients, $\frac{\partial \phi}{\partial x_i}$, could be used to define the interface fluxes. This is possible once a globally continuous solution is recovered from the end of the previous time-step. As discussed in the last section, a globally continuous solution is obtained from Equation (3.11), which computes an average of the connecting element solutions for each node. The post-processing calculation uses the nodal values of the continuous solution and its gradients, to provide an accurate interface flux. It is this flux which establishes connectivity between elements at the next time-step. Thus in this work, we define the forcing vector appearing in Equations (3.8) and (3.9) as:

$$\{\hat{\mathbf{f}}_\mathbf{e}\}^n = -\int_{\Gamma_e} \mathbf{N}^\mathbf{T} u_i \mathbf{N} d\Gamma_e n_i \{\mathbf{\Phi}\}^n + \int_{\Gamma_e} \mathbf{N}^\mathbf{T} k \mathbf{N} d\Gamma_e n_i \left\{ \frac{\partial \hat{\mathbf{\Phi}}}{\partial \mathbf{x_i}} \right\}^n \tag{3.16}$$

A number of post-processing approaches, to evaluate the first derivative of the scalar variable at the node, are available. These are: (1) Simple averaging of gradients over elements connected to a node [1]; (2) Area-weighted averaging of the gradients [21]; and (3) Super-convergence patch recovery (SPR), which is used in adaptivity to recover nodal values of stresses [137]. In this work, the Simple averaging of gradients was only considered to evaluate the gradients. For cases where convection of the scalar variable is present, the convective flux component was based on nodal values of the continuous solution of ϕ, retrieved from the previous time-step n. Together the diffusive and convective components give a nodal flux approximation.

3.3.3 Approximating nodal values of gradient in linear finite elements

For linear triangular finite elements the gradient of the scalar variable is a constant within each element. In a previous paper [1], a mean average of the constant gradients, over the elements connected to a node, was computed to give a nodal approximation of the variable's gradient. To exemplify, Figure (3.2) shows a group of linear triangular elements attached to a common node a. The gradient of the scalar variable at node a in Figure (3.2) is computed as an average of the constant elemental values connected to node a. i.e.

$$\left(\frac{\partial \hat{\phi}}{\partial x_i} \right)_a = \frac{1}{ne} \sum_{e=1}^{ne} \left(\frac{\partial \phi}{\partial x_i} \right)_e \tag{3.17}$$

These nodal gradient values were then used to estimate the diffusive component of flux crossing the edges. In general the nodal fluxes, $\left(\hat{F}_i\right)_a$, will contain both a nodal diffusive flux component, and a nodal convective flux component, i.e.,

$$\left(\hat{F}_i\right)_a = \left(\hat{F}_i\right)_a \left(k_i \left(\frac{\partial \hat{\phi}}{\partial x_i} \right)_a, u_i \phi_a \right) \tag{3.18}$$

Once nodal values of flux are recovered, flux on an edge between linear elements was computed as an average of the two nodes forming the edge. The edge flux was then used on the

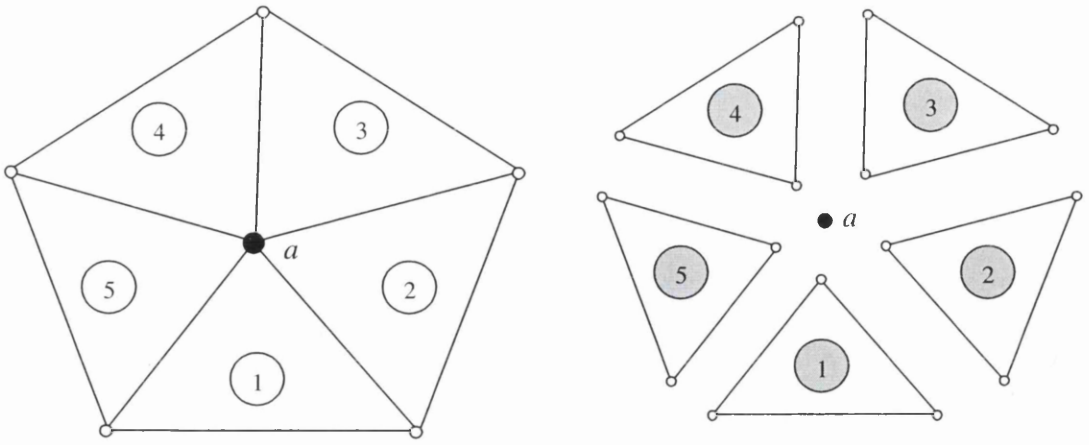


Figure 3.2: A patch of linear elements. Element averaged gradient calculation of the scalar variable at node a.

boundaries of an element to enforce the condition given by Equation (3.15).

It is noted that treating the flux in a consistent form was not considered for the scalar convection-diffusion problems investigated in reference [1]. Only the lumping of nodal fluxes was used. In addition only linear triangular finite elements were used. When using linear elements the flux calculation is based on an element-averaged diffusive flux. By using higher-order interpolation functions it is possible to improve the accuracy of edge fluxes by allowing the diffusive fluxes to be nodally estimated. This gives way to a more accurate computation of the fluxes at each time-step.

3.3.4 Using quadratic finite elements to obtain nodal values of gradient

If higher-order elements are used, then the gradient, $\frac{\partial \phi}{\partial x_i}$, is not a constant for each element. In particular, for a six-node quadratic-triangular element, the gradient is given by Equation (3.4) as

$$\frac{\partial \phi}{\partial x_i}(x_i) \approx \frac{\partial \tilde{\phi}}{\partial x_i}(x_i) = \sum_{n=1}^{6} \frac{\partial N_n}{\partial x_i}(x_i)\,\phi_n \tag{3.19}$$

Here, the gradient varies linearly through the element domain, before being discontinuous at the edges between neighbouring elements. An actual value of $\frac{\partial \phi}{\partial x_i}$ can be found at any

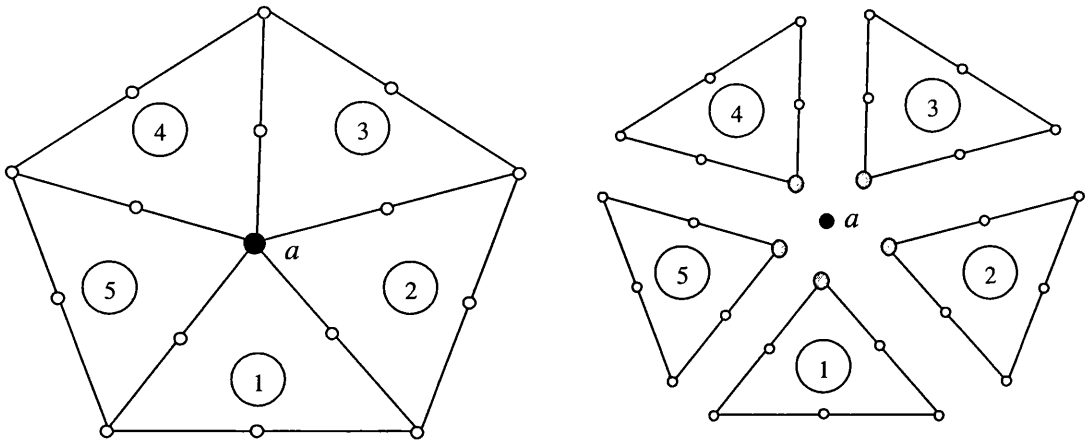


Figure 3.3: A patch of quadratic elements. nodal averaged gradient calculation of the scalar variable at node a.

point within the element, by inserting the coordinates, x_i, of the point of interest into the derivatives of the shape functions.

Thus, by using the nodal coordinates the gradients of the scalar variable can be computed at the node of interest, in each of the connecting elements. Figure (3.3) shows a group of quadratic elements connected together at node a in Ω. In order to compute an approximation for $\frac{\partial \phi}{\partial x_i}$ at node a, elemental computations are made of $\frac{\partial \phi}{\partial x_i}$ in each of the connecting elements, at the point corresponding to node a (marked grey in Figure (3.3). Different nodal values of $\left(\frac{\partial \phi}{\partial x_i}\right)_a$ are obtained for each of the connecting elements. Taking a mean average of all the obtained nodal values from connecting elements:

$$\left(\frac{\partial \hat{\phi}}{\partial x_i}\right)_a = \frac{1}{ne} \sum_{e=1}^{ne} \left\{ \left(\frac{\partial \phi}{\partial x_i}\right)_a \right\}_e \tag{3.20}$$

gives the required nodal value that is used to compute diffusive edge flux. In Equation (3.20), ne is the total number of elements connected to node, a.

Figure 3.4 shows an edge of a quadratic element with the computed nodal fluxes labeled relative to the local numbering of each edge node. For the general convection-diffusion problem (assuming a constant velocity field and diffusivity), the flux on the edge of a quadratic triangle is given by

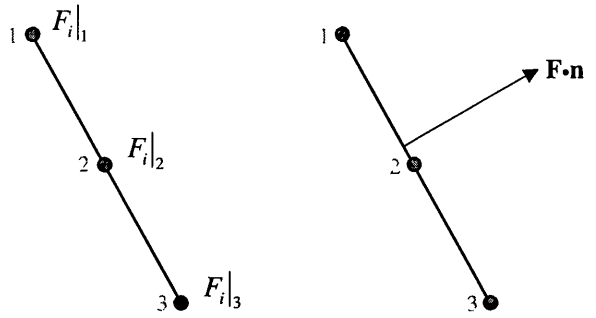


Figure 3.4: Local edge node numbering for nodal fluxes and edge normal flux representation.

$$\begin{Bmatrix} \left(\hat{F}_i\right)_1 \\ \left(\hat{F}_i\right)_2 \\ \left(\hat{F}_i\right)_3 \end{Bmatrix} = \frac{l_{edge}}{30} \begin{bmatrix} 4 & 2 & -1 \\ 2 & 16 & 2 \\ -1 & 2 & 4 \end{bmatrix} n_i \left(-u_i \begin{Bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{Bmatrix} + k \begin{Bmatrix} \left(\frac{\hat{\partial\phi}}{\partial x_i}\right)_1 \\ \left(\frac{\hat{\partial\phi}}{\partial x_i}\right)_2 \\ \left(\frac{\hat{\partial\phi}}{\partial x_i}\right)_3 \end{Bmatrix} \right) \tag{3.21}$$

where the numbering refers to edge nodes defined in Figure (3.4), and l_{edge} is the length of the edge of the element. This flux can be simplified further. Treating the edge as a one-dimensional quadratic element, the following lumped form can be used in the computation of the total interface flux along an element edge.

$$\begin{Bmatrix} \left(\hat{F}_i\right)_1 \\ \left(\hat{F}_i\right)_2 \\ \left(\hat{F}_i\right)_3 \end{Bmatrix} = \frac{l_{edge}}{6} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 1 \end{bmatrix} n_i \left(-u_i \begin{Bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{Bmatrix} + k \begin{Bmatrix} \left(\frac{\hat{\partial\phi}}{\partial x_i}\right)_1 \\ \left(\frac{\hat{\partial\phi}}{\partial x_i}\right)_2 \\ \left(\frac{\hat{\partial\phi}}{\partial x_i}\right)_3 \end{Bmatrix} \right) \tag{3.22}$$

The simplified form is based on the lumped mass matrix for a one-dimensional quadratic element [20]. As mentioned earlier, the edge flux ensures continuity between elements, allowing the discrete equation to be solved element by element.

3.4 A comparative analysis of the LCG method with the standard global Galerkin method

In this section a direct comparison of the nodal equation is made between the LCG method and the standard global Galerkin method. The analysis is crucial in understanding

how the LCG scheme works. To start, a simple one-dimensional patch is considered for convection-diffusion problems. The matrix form of the elemental residual equation is then given and a final nodal equation is retrieved that includes the relation for the computed numerical flux. It is shown that the final nodal equation is exactly the same as that obtained by the standard global Galerkin method, for inside nodes. The extension to two-dimensions is considered next. Again it is shown that the LCG and global Galerkin methods are equivalent for inside nodes. The former however possesses a residual flux on the boundary that gives it explicit global conservation - even in the absence of Neumann-type boundary conditions

3.4.1 The LCG nodal-equation for one-dimensional convection-diffusion

To demonstrate the equivalence of the LCG method with the standard global Galerkin method for inside nodes, a one-dimensional patch is considered for convection-diffusion problems. To simplify the presentation of the analysis, it is assumed that the governing convection-diffusion equation, Equation (3.1), does not contain any source terms. In addition the LHS mass matrix for each scheme is lumped for comparison of each method's nodal-equation.

The nodal equation for node a using the standard global Galerkin method, is obtained after assembling elemental contributions of Equation (2.46) into a global matrix system. After assembly the nodal equation is written as

$$\beta\left(\frac{h_1+h_2}{2}\right)\frac{\Delta\phi_a}{\Delta t}=\frac{u}{2}(\phi_{a-1}-\phi_{a+1})-k\left(\frac{\phi_{a-1}}{h_1}-\frac{\phi_a}{h_1}-\frac{\phi_a}{h_2}+\frac{\phi_{a+1}}{h_2}\right) \tag{3.23}$$

The discrete form of the governing equations, using the LCG method, is given by Equation (3.7) - for solving over each individual element e. For a mesh of linear one-dimensional elements, the elemental-matrix form of Equation (3.7) becomes

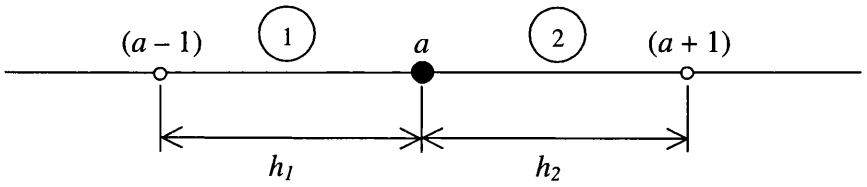


Figure 3.5: A one-dimensional patch of linear elements, with common node a.

$$\beta\left(\frac{h_e}{2\Delta t}\right)\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\begin{Bmatrix}\Delta\phi_l\\ \Delta\phi_r\end{Bmatrix}^{n+1} = \frac{u}{2}\begin{bmatrix}-1 & -1\\ +1 & +1\end{bmatrix}\begin{Bmatrix}\phi_l\\ \phi_r\end{Bmatrix}^{n} - \frac{k}{h_e}\begin{bmatrix}+1 & -1\\ -1 & +1\end{bmatrix}\begin{Bmatrix}\phi_l\\ \phi_r\end{Bmatrix}^{n} - u\begin{bmatrix}-1 & 0\\ 0 & +1\end{bmatrix}\begin{Bmatrix}\phi_l\\ \phi_r\end{Bmatrix}^{n} + k\begin{bmatrix}-1 & 0\\ 0 & +1\end{bmatrix}\begin{Bmatrix}\left(\frac{\hat{\partial\phi}}{\partial x}\right)_l\\ \left(\frac{\hat{\partial\phi}}{\partial x}\right)_r\end{Bmatrix}^{n} \tag{3.24}$$

Here, h_e is the length of element e, and the subscripts l and r are used to identify the LHS-node and RHS-node of element e respectively.

Figure (3.5) shows an internal one-dimensional patch with an inside node - denoted by a. As mentioned in a previous section, at the end of each time-step there will be multiple solutions obtained for an inside node - one given by each of the connecting elements. A unique solution for each node is only obtained after the averaging process of Equation (3.11) is completed. In the specific case of the one-dimensional patch illustrated in Figure (3.5), there will be nodal equation given by each of the elements sharing node a. In element 1, node a is on the RHS and the corresponding nodal equation is

$$\beta\left(\frac{h_1}{2\Delta t}\right)\Delta\phi_a = \frac{u}{2}(\phi_{a-1}+\phi_a) - \frac{k}{h_1}(-\phi_{a-1}+\phi_a) - u(\phi_a) + k\left(\frac{\hat{\partial\phi}}{\partial x}\right)_a \tag{3.25}$$

For element 2, node a is on the LHS giving a corresponding nodal equation of

$$\beta\left(\frac{h_2}{2\Delta t}\right)\Delta\phi_a = \frac{u}{2}(-\phi_a-\phi_{a+1}) - \frac{k}{h_2}(\phi_a-\phi_{a+1}) + u(\phi_a) - k\left(\frac{\hat{\partial\phi}}{\partial x}\right)_a \tag{3.26}$$

In both Equation (3.25) and Equation (3.26) the numerical flux term $\left(\frac{\hat{\partial\phi}}{\partial x_i}\right)_a$ appears. For 1D linear elements, it is given (by Equation (3.17)) as the average of constant gradients from the two elements attached to node a i.e.

$$\left(\frac{\hat{\partial\phi}}{\partial x}\right)_a = \frac{1}{2}\left(\left(\frac{-\phi_{a-1}+\phi_a}{h_1}\right)+\left(\frac{-\phi_a+\phi_{a+1}}{h_2}\right)\right) \tag{3.27}$$

For constant element-size $h_1 = h_2 = \Delta x$, it can be seen in Equation (3.27) that the averaging of constant gradients procedure gives

$$\left(\frac{\hat{\partial\phi}}{\partial x}\right)_a = \left(\frac{\phi_{a+1}-\phi_{a-1}}{2\Delta x}\right) + O(\Delta x^2) \tag{3.28}$$

which is a second-order accurate central difference approximation for the numerical flux $\left(\frac{\hat{\partial\phi}}{\partial x_i}\right)$ at node a.

Substituting the relation for the computed flux, Equation (3.27), into Equation (3.25) and Equation (3.26) gives

$$\begin{aligned}\beta\left(\frac{h_1}{2\Delta t}\right)\Delta\phi_a = &\frac{u}{2}(\phi_{a-1}+\phi_a) - \frac{k}{h_1}(-\phi_{a-1}+\phi_a)\\ &-u(\phi_a) + k\left(\left(\frac{-\phi_{a-1}+\phi_a}{2h_1}\right)+\left(\frac{-\phi_a+\phi_{a+1}}{2h_2}\right)\right)\end{aligned} \tag{3.29}$$

and

$$\begin{aligned}\beta\left(\frac{h_2}{2\Delta t}\right)\Delta\phi_a = &\frac{u}{2}(-\phi_a-\phi_{a+1}) - \frac{k}{h_2}(\phi_a-\phi_{a+1})\\ &+u(\phi_a) - k\left(\left(\frac{-\phi_{a-1}+\phi_a}{2h_1}\right)+\left(\frac{-\phi_a+\phi_{a+1}}{2h_2}\right)\right)\end{aligned} \tag{3.30}$$

for elements 1 and 2 respectively. Simplification gives the final equation for node a in element 1 as

$$\beta\left(\frac{h_1}{2\Delta t}\right)\Delta\phi_a = \frac{u}{2}(\phi_{a-1}-\phi_a) - \frac{k}{2h_1}(-\phi_{a-1}+\phi_a) + \frac{k}{2h_2}(-\phi_a+\phi_{a+1}) \tag{3.31}$$

and the final equation for node a in element 2 as

$$\beta\left(\frac{h_2}{2\Delta t}\right)\Delta\phi_a = \frac{u}{2}(\phi_a - \phi_{a+1}) - \frac{k}{2h_1}(-\phi_{a-1} + \phi_a) + \frac{k}{2h_2}(-\phi_a + \phi_{a+1}) \tag{3.32}$$

Since no assembly is carried out during the LCG process - only averaging, essentially $\beta\left(\frac{h_e}{2\Delta t}\right)\Delta\phi_a$ is being approximated using

$$\beta\left(\frac{h_e}{2\Delta t}\right)\Delta\phi_a = \beta\left(\frac{(h_1+h_2)/2}{2\Delta t}\right)\Delta\phi_a = \frac{\left(\beta\left(\frac{h_1}{2\Delta t}\right)\Delta\phi_a\right)_1 + \left(\beta\left(\frac{h_2}{2\Delta t}\right)\Delta\phi_a\right)_2}{2} \tag{3.33}$$

Thus after the elemental averaging procedure, the effective nodal equation at a is

$$\beta\left(\frac{h_1+h_2}{2}\right)\frac{\Delta\phi_a}{\Delta t} = \frac{u}{2}(\phi_{a-1} - \phi_{a+1}) - k\left(\frac{\phi_{a-1}}{h_1} - \frac{\phi_a}{h_1} - \frac{\phi_a}{h_2} + \frac{\phi_{a+1}}{h_2}\right) \tag{3.34}$$

Clearly it is evident, from comparing Equation (3.34) to the nodal equation obtained from the global Galerkin method, Equation (3.23), that the two methods are indeed identical for internal nodes.

3.4.2 Analysis of the LCG nodal-equation in two-dimensions

The extension of the analysis to two-dimensions, is made by examining the 2D patch of linear triangular-elements given in Figure (3.6). As can be seen, all elements share a common node 1 at the centre of the patch. Figure (3.6) also gives the notation definition for a general element e in this patch.

To start, the following elementary details [20] are recalled for a general linear triangular-element with local (area) coordinates: For each node a of element e, the shape function is given as

$$N_a = (A_a + B_a x + C_a y) \tag{3.35}$$

where the constants A_a, B_a, and C_a are defined in terms of the coordinates of nodes b and c and element area A_e as

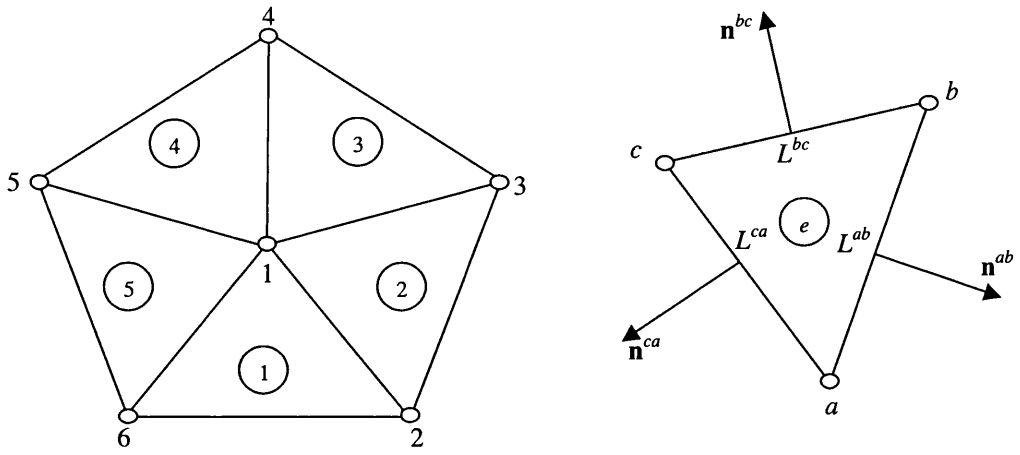


Figure 3.6: Left: A two-dimensional patch of linear triangular elements, sharing a common node at patch-centre. Right: a local general element with notations used for analysis.

$$A_a = \frac{x_b y_c - x_c y_b}{2A_e}; \qquad B_a = \frac{y_b - y_c}{2A_e}; \qquad C_a = \frac{x_c - x_b}{2A_e} \tag{3.36}$$

A cyclic permutation of a, b, c gives the values A_b, A_c, B_b, etc. On an edge between two element-boundary nodes a and b the components of the outwards normal are

$$\mathbf{n}^{ab} = (n_1^{ab}, n_2^{ab}) = \left(\frac{y_b - y_a}{L^{ab}}, \frac{x_a - x_b}{L^{ab}} \right) \tag{3.37}$$

with $L^{ab} = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2}$ denoting the length of side a-b. Such details are very standard and are generally restricted to introductory texts on finite element analysis. However, it is important that they are included here, to aid in the analysis of the nodal equation obtained for Node 1 using the LCG method.

Using the basic definitions and the notation of Figure (3.6), the elemental-matrix form of Equation (3.7) becomes

Table 3.1: Connectivity matrix for 2D element patch given in Figure 3.6

element	a	b	c
1	6	2	1
2	2	3	1
3	1	3	4
4	1	4	5
5	6	1	5

$$
\begin{aligned}
& \beta\left(\frac{A_e}{3\Delta t}\right)\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix}\begin{Bmatrix}\Delta\phi_a\\\Delta\phi_b\\\Delta\phi_c\end{Bmatrix} \\
= \; & \frac{A_e}{3}\left(u_x\begin{bmatrix}B_a&B_a&B_a\\B_b&B_b&B_b\\B_c&B_c&B_c\end{bmatrix}+u_y\begin{bmatrix}C_a&C_a&C_a\\C_b&C_b&C_b\\C_c&C_c&C_c\end{bmatrix}\right)\begin{Bmatrix}\phi_a\\\phi_b\\\phi_c\end{Bmatrix}^n \\
- \; & A_e\left(k_x\begin{bmatrix}B_aB_a&B_aB_b&B_aB_c\\B_bB_a&B_bB_b&B_bB_c\\B_cB_a&B_cB_b&B_cB_c\end{bmatrix}+k_y\begin{bmatrix}C_aC_a&C_aC_b&C_aC_c\\C_bC_a&C_bC_b&C_bC_c\\C_cC_a&C_cC_b&C_cC_c\end{bmatrix}\right)\begin{Bmatrix}\phi_a\\\phi_b\\\phi_c\end{Bmatrix}^n \\
+ \; & \left(\frac{L^{ab}n_i^{ab}}{6}\begin{bmatrix}2&1&0\\1&2&0\\0&0&0\end{bmatrix}+\frac{L^{bc}n_i^{bc}}{6}\begin{bmatrix}0&0&0\\0&2&1\\0&1&2\end{bmatrix}+\frac{L^{ca}n_i^{ca}}{6}\begin{bmatrix}2&0&1\\0&0&0\\1&0&2\end{bmatrix}\right)\begin{Bmatrix}\left(\hat{F}_i\right)_a\\\left(\hat{F}_i\right)_b\\\left(\hat{F}_i\right)_c\end{Bmatrix}^n
\end{aligned}
\tag{3.38}
$$

for a linear triangular element. Here, the LHS mass matrix has been lumped, and

$$
\begin{Bmatrix}\left(\hat{F}_i\right)_a\\\left(\hat{F}_i\right)_b\\\left(\hat{F}_i\right)_c\end{Bmatrix}=\left(-u_i\begin{Bmatrix}\phi_a\\\phi_b\\\phi_c\end{Bmatrix}+k_i\begin{Bmatrix}\left(\frac{\partial\hat{\phi}}{\partial x_i}\right)_a\\\left(\frac{\partial\hat{\phi}}{\partial x_i}\right)_b\\\left(\frac{\partial\hat{\phi}}{\partial x_i}\right)_c\end{Bmatrix}\right) \tag{3.39}
$$

defines the numerical flux.

From Equation 3.38 the nodal equation for node a in a general element is

$$\begin{aligned}\beta\left(\frac{A_e}{3\Delta t}\right)\Delta\phi_a &= \frac{A_e}{3}\left(u_xB_a+u_yC_a\right)\left(\phi_a+\phi_b+\phi_c\right)^n \\ &- A_e\left(\left(k_xB_a\right)\left(B_a\phi_a+B_b\phi_b+B_c\phi_c\right)+\left(k_yC_a\right)\left(C_a\phi_a+C_b\phi_b+C_c\phi_c\right)\right)^n \\ &+ \frac{L^{ab}n_i^{ab}}{6}\left(2\left(\hat{F}_i\right)_a+1\left(\hat{F}_i\right)_b\right)^n+\frac{L^{ca}n_i^{ca}}{6}\left(2\left(\hat{F}_i\right)_a+1\left(\hat{F}_i\right)_c\right)^n \end{aligned} \quad (3.40)$$

Using Equation (3.36) and Equation (3.37), the normals and the shape function derivatives are rewritten in terms of nodal coordinates to give

$$\begin{aligned}\beta\left(\frac{A_e}{3\Delta t}\right)\Delta\phi_a &= \frac{1}{6}\left[u_x(y_b-y_c)+u_y(x_c-x_b)\right]\left[\phi_a+\phi_b+\phi_c\right]^n \\ &- \frac{1}{4A_e}\left[k_x(y_b-y_c)\right]\left[(y_b-y_c)\phi_a+(y_c-y_a)\phi_b+(y_a-y_b)\phi_c\right]^n \\ &- \frac{1}{4A_e}\left[k_y(x_c-x_b)\right]\left[(x_c-x_b)\phi_a+(x_a-x_c)\phi_b+(x_b-x_a)\phi_c\right]^n \\ &+ \left[\frac{(y_b-y_a)}{6}\right]\left[2\left(\hat{F}_x\right)_a+1\left(\hat{F}_x\right)_b\right]^n+\left[\frac{(x_a-x_b)}{6}\right]\left[2\left(\hat{F}_y\right)_a+1\left(\hat{F}_y\right)_b\right]^n \\ &+ \left[\frac{(y_a-y_c)}{6}\right]\left[2\left(\hat{F}_x\right)_a+1\left(\hat{F}_x\right)_c\right]^n+\left[\frac{(x_c-x_a)}{6}\right]\left[2\left(\hat{F}_y\right)_a+1\left(\hat{F}_y\right)_c\right]^n \end{aligned} \quad (3.41)$$

There are two more element-nodal equations for node b and node c, which are found by expanding out rows 2 and 3 of Equation 3.38, respectively. In Figure (3.6) a general patch of elements are given. For the patch of Figure (3.6), there are five element-nodal equations are for node 1, one given by each of the connecting elements. The exact form and structure of each element-nodal equation, will be different and depends on the connectivity for the element. The connectivity matrix for the 2D patch of triangular elements is given in Table 3.1. Using this connectivity matrix, the element-nodal equation for node 1, given by the matrix equation of element 1, is

$$\left(\beta\left(\frac{A_{e_1}}{3\Delta t}\right)\Delta\phi_1\right)_{e_1} = \frac{1}{6}\left[u_x(y_6-y_2)+u_y(x_2-x_6)\right]\left[\phi_6+\phi_2+\phi_1\right]^n$$
$$-\frac{1}{4A_{e_1}}\left[k_x(y_6-y_2)\right]\left[(y_2-y_1)\phi_6+(y_1-y_6)\phi_2+(y_6-y_2)\phi_1\right]^n$$
$$-\frac{1}{4A_{e_1}}\left[k_y(x_2-x_6)\right]\left[(x_1-x_2)\phi_6+(x_6-x_1)\phi_2+(x_2-x_6)\phi_1\right]^n$$
$$+\left[\frac{(y_1-y_2)}{6}\right]\left[1\left(\hat{F}_x\right)_2+2\left(\hat{F}_x\right)_1\right]^n+\left[\frac{(x_2-x_1)}{6}\right]\left[1\left(\hat{F}_y\right)_2+2\left(\hat{F}_y\right)_1\right]^n$$
$$+\left[\frac{(y_6-y_1)}{6}\right]\left[1\left(\hat{F}_x\right)_6+2\left(\hat{F}_x\right)_1\right]^n+\left[\frac{(x_1-x_6)}{6}\right]\left[1\left(\hat{F}_y\right)_6+2\left(\hat{F}_y\right)_1\right]^n \quad (3.42)$$

The element-nodal equation for node 1, for element 2 is

$$\left(\beta\left(\frac{A_{e_2}}{3\Delta t}\right)\Delta\phi_1\right)_{e_2} = \frac{1}{6}\left[u_x(y_2-y_3)+u_y(x_3-x_2)\right]\left[\phi_2+\phi_3+\phi_1\right]^n$$
$$-\frac{1}{4A_{e_2}}\left[k_x(y_2-y_3)\right]\left[(y_3-y_1)\phi_2+(y_1-y_2)\phi_3+(y_2-y_3)\phi_1\right]^n$$
$$-\frac{1}{4A_{e_2}}\left[k_y(x_3-x_2)\right]\left[(x_1-x_3)\phi_2+(x_2-x_1)\phi_3+(x_3-x_2)\phi_1\right]^n$$
$$+\left[\frac{(y_1-y_3)}{6}\right]\left[1\left(\hat{F}_x\right)_3+2\left(\hat{F}_x\right)_1\right]^n+\left[\frac{(x_3-x_1)}{6}\right]\left[1\left(\hat{F}_y\right)_3+2\left(\hat{F}_y\right)_1\right]^n$$
$$+\left[\frac{(y_2-y_1)}{6}\right]\left[1\left(\hat{F}_x\right)_2+2\left(\hat{F}_x\right)_1\right]^n+\left[\frac{(x_1-x_2)}{6}\right]\left[1\left(\hat{F}_y\right)_2+2\left(\hat{F}_y\right)_1\right]^n \quad (3.43)$$

The element-nodal equation for node 1, for element 3 is

$$\left(\beta\left(\frac{A_{e_3}}{3\Delta t}\right)\Delta\phi_1\right)_{e_3} = \frac{1}{6}\left[u_x(y_3-y_4)+u_y(x_4-x_3)\right]\left[\phi_1+\phi_3+\phi_4\right]^n$$
$$-\frac{1}{4A_{e_3}}\left[k_x(y_3-y_4)\right]\left[(y_3-y_4)\phi_1+(y_4-y_1)\phi_3+(y_1-y_3)\phi_4\right]^n$$
$$-\frac{1}{4A_{e_3}}\left[k_y(x_4-x_3)\right]\left[(x_4-x_3)\phi_1+(x_1-x_4)\phi_3+(x_3-x_1)\phi_4\right]^n$$
$$+\left[\frac{(y_3-y_1)}{6}\right]\left[2\left(\hat{F}_x\right)_1+1\left(\hat{F}_x\right)_3\right]^n+\left[\frac{(x_1-x_3)}{6}\right]\left[2\left(\hat{F}_y\right)_1+1\left(\hat{F}_y\right)_3\right]^n$$
$$+\left[\frac{(y_1-y_4)}{6}\right]\left[2\left(\hat{F}_x\right)_1+1\left(\hat{F}_x\right)_4\right]^n+\left[\frac{(x_4-x_1)}{6}\right]\left[2\left(\hat{F}_y\right)_1+1\left(\hat{F}_y\right)_4\right]^n \quad (3.44)$$

The element-nodal equation for node 1, for element 4 is

$$\begin{aligned}
\left(\beta\left(\frac{A_{e_4}}{3\Delta t}\right)\Delta\phi_1\right)_{e_4} &= \frac{1}{6}\left[u_x(y_4-y_5)+u_y(x_5-x_4)\right]\left[\phi_1+\phi_4+\phi_5\right]^n \\
&-\frac{1}{4A_{e_4}}\left[k_x(y_4-y_5)\right]\left[(y_4-y_5)\phi_1+(y_5-y_1)\phi_4+(y_1-y_4)\phi_5\right]^n \\
&-\frac{1}{4A_{e_4}}\left[k_y(x_5-x_4)\right]\left[(x_5-x_4)\phi_1+(x_1-x_5)\phi_4+(x_4-x_1)\phi_5\right]^n \\
&+\left[\frac{(y_4-y_1)}{6}\right]\left[2\left(\hat{F}_x\right)_1+1\left(\hat{F}_x\right)_4\right]^n+\left[\frac{(x_1-x_4)}{6}\right]\left[2\left(\hat{F}_y\right)_1+1\left(\hat{F}_y\right)_4\right]^n \\
&+\left[\frac{(y_1-y_5)}{6}\right]\left[2\left(\hat{F}_x\right)_1+1\left(\hat{F}_x\right)_5\right]^n+\left[\frac{(x_5-x_1)}{6}\right]\left[2\left(\hat{F}_y\right)_1+1\left(\hat{F}_y\right)_5\right]^n
\end{aligned} \tag{3.45}$$

The element-nodal equation for node 1, for element 5 is

$$\begin{aligned}
\left(\beta\left(\frac{A_{e_5}}{3\Delta t}\right)\Delta\phi_1\right)_{e_5} &= \frac{1}{6}\left[u_x(y_5-y_6)+u_y(x_6-x_5)\right]\left[\phi_6+\phi_1+\phi_5\right]^n \\
&-\frac{1}{4A_{e_5}}\left[k_x(y_5-y_6)\right]\left[(y_1-y_5)\phi_6+(y_5-y_6)\phi_1+(y_6-y_1)\phi_5\right]^n \\
&-\frac{1}{4A_{e_5}}\left[k_y(x_6-x_5)\right]\left[(x_5-x_1)\phi_6+(x_6-x_5)\phi_1+(x_1-x_6)\phi_5\right]^n \\
&+\left[\frac{(y_1-y_6)}{6}\right]\left[1\left(\hat{F}_x\right)_6+2\left(\hat{F}_x\right)_1\right]^n+\left[\frac{(x_6-x_1)}{6}\right]\left[1\left(\hat{F}_y\right)_6+2\left(\hat{F}_y\right)_1\right]^n \\
&+\left[\frac{(y_5-y_1)}{6}\right]\left[2\left(\hat{F}_x\right)_1+1\left(\hat{F}_x\right)_5\right]^n+\left[\frac{(x_1-x_5)}{6}\right]\left[2\left(\hat{F}_y\right)_1+1\left(\hat{F}_y\right)_5\right]^n
\end{aligned} \tag{3.46}$$

The effective nodal equation for node 1 is found from the average of the elemental-nodal equations obtained from the connecting elements. Here the averaging approximation is given as

$$\begin{aligned}
\beta\left(\frac{A_e}{3\Delta t}\right)\Delta\phi_1 &\approx \frac{\beta}{3\Delta t}\left(\frac{A_{e_1}+A_{e_2}+A_{e_3}+A_{e_4}+A_{e_5}}{5}\right)\Delta\phi_1 \\
&\approx \frac{1}{5}\sum_{n=1}^{5}\left(\beta\left(\frac{A_{e_n}}{3\Delta t}\right)\Delta\phi_1\right)_{e_n}
\end{aligned} \tag{3.47}$$

Examination of all the equations for node 1, clearly shows that by adding together Equations (3.42) to (3.46), the edge fluxes will cancel each other out. This is due to the local conservation of the scheme. Thus, the resulting nodal equation for node 1 is

$$\frac{\beta}{3\Delta t}\left(\left(A_{e_1}+A_{e_2}+A_{e_3}+A_{e_4}+A_{e_5}\right)\right)\Delta\phi_1 =$$

$$
\begin{aligned}
&+\frac{u_x}{6}\left[(y_6-y_3)\phi_2+(y_2-y_4)\phi_3+(y_3-y_5)\phi_4+(y_4-y_6)\phi_5+(y_5-y_2)\phi_6\right]^n\\
&-\frac{k_x}{4}\left[\left(\left[\frac{(y_6-y_2)^2}{A_{e_1}}\right]+\left[\frac{(y_2-y_3)^2}{A_{e_2}}\right]+\left[\frac{(y_3-y_4)^2}{A_{e_3}}\right]+\left[\frac{(y_4-y_5)^2}{A_{e_4}}\right]+\left[\frac{(y_5-y_6)^2}{A_{e_5}}\right]\right)\phi_1\right]^n\\
&-\frac{k_x}{4}\left[\left(\left[\frac{(y_6-y_2)(y_1-y_6)}{A_{e_1}}\right]+\left[\frac{(y_2-y_3)(y_3-y_1)}{A_{e_2}}\right]\right)\phi_2\right]^n\\
&-\frac{k_x}{4}\left[\left(\left[\frac{(y_2-y_3)(y_1-y_2)}{A_{e_2}}\right]+\left[\frac{(y_3-y_4)(y_4-y_1)}{A_{e_3}}\right]\right)\phi_3\right]^n\\
&-\frac{k_x}{4}\left[\left(\left[\frac{(y_3-y_4)(y_1-y_3)}{A_{e_3}}\right]+\left[\frac{(y_4-y_5)(y_5-y_1)}{A_{e_4}}\right]\right)\phi_4\right]^n\\
&-\frac{k_x}{4}\left[\left(\left[\frac{(y_4-y_5)(y_1-y_4)}{A_{e_4}}\right]+\left[\frac{(y_5-y_6)(y_6-y_1)}{A_{e_5}}\right]\right)\phi_5\right]^n\\
&-\frac{k_x}{4}\left[\left(\left[\frac{(y_5-y_6)(y_1-y_5)}{A_{e_5}}\right]+\left[\frac{(y_6-y_2)(y_2-y_1)}{A_{e_1}}\right]\right)\phi_6\right]^n\\
&+\frac{u_y}{6}\left[(x_3-x_6)\phi_2+(x_4-x_2)\phi_3+(x_5-x_3)\phi_4+(x_6-x_4)\phi_5+(x_2-x_5)\phi_6\right]^n\\
&-\frac{k_y}{4}\left[\left(\left[\frac{(x_2-x_6)^2}{A_{e_1}}\right]+\left[\frac{(x_3-x_2)^2}{A_{e_2}}\right]+\left[\frac{(x_4-x_3)^2}{A_{e_3}}\right]+\left[\frac{(x_5-x_4)^2}{A_{e_4}}\right]+\left[\frac{(x_6-x_5)^2}{A_{e_5}}\right]\right)\phi_1\right]^n\\
&-\frac{k_x}{4}\left[\left(\left[\frac{(x_2-x_6)(x_6-x_1)}{A_{e_1}}\right]+\left[\frac{(x_3-x_2)(x_1-x_3)}{A_{e_2}}\right]\right)\phi_2\right]^n\\
&-\frac{k_x}{4}\left[\left(\left[\frac{(x_3-x_2)(x_2-x_1)}{A_{e_2}}\right]+\left[\frac{(x_4-x_3)(x_1-x_4)}{A_{e_3}}\right]\right)\phi_3\right]^n\\
&-\frac{k_x}{4}\left[\left(\left[\frac{(x_4-x_3)(x_3-x_1)}{A_{e_3}}\right]+\left[\frac{(x_5-x_4)(x_1-x_5)}{A_{e_4}}\right]\right)\phi_4\right]^n\\
&-\frac{k_x}{4}\left[\left(\left[\frac{(x_5-x_4)(x_4-x_1)}{A_{e_4}}\right]+\left[\frac{(x_6-x_5)(x_1-x_6)}{A_{e_5}}\right]\right)\phi_5\right]^n\\
&-\frac{k_x}{4}\left[\left(\left[\frac{(x_6-x_5)(x_5-x_1)}{A_{e_5}}\right]+\left[\frac{(x_2-x_6)(x_1-x_2)}{A_{e_1}}\right]\right)\phi_6\right]^n
\end{aligned}
\tag{3.48}
$$

Equations (3.48) is identical to the nodal equation for node 1, retrieved using the global Galerkin method for the same 2D patch of Figure (3.6). Thus proving equivalence of the nodal-equations of the LCG and global Galerkin methods, in two-dimensions for inside nodes, when a lumped mass is used. The former however, is always locally and globally

conservative.

3.4.3 Remarks on the analysis of the LCG method

1. For the LCG method, a post-processed flux is calculated at every time-step, for each node, to provide Neumann-type boundary conditions on the edge of each element sub-domain.

2. The diffusion portion of this nodal flux is taken as an average of gradients from connected elements. It was shown for 1D linear elements that the diffusive flux calculation is equivalent to the second-order accurate central-difference approximation in finite difference methods.

3. Examination of all the equations for node 1, clearly shows that by adding together Equations (3.42) to (3.46), the edge fluxes will cancel each other out. This statement is valid for any grid, whether structured or unstructured, uniform or non-uniform.

4. The approximation given by Equation (3.47) also applies to both structured and unstructured grids as long as the elements surrounding the inside node are of equal size. A similar condition applies to Equation (3.33) for the 1D case.

5. Assuming Equation (3.47) to be a valid approximation, the LCG method gives an identical nodal equation as the standard GG method for inside nodes, when a lumped mass is used. This is true at both transient and steady states. Although the nodal equations are identical, the LCG requires additional computation.

6. If the elements surrounding an internal node differ greatly in size, then Equations (3.33) and (3.47) are no-longer good approximations, thus the lumped-mass LCG and lumped-mass global Galerkin methods are not strictly equivalent for the transient state.

7. Due to the coupling of nodal equations when a consistent mass is used, it is difficult to prove that the LCG and global Galerkin methods are identical for inside nodes for transient state, when a consistent mass-matrix is employed.

8. After the averaging procedure, a residual post-processed flux appears along the domain boundary for an LCG method, regardless of the prescribed boundary conditions.

9. Not only is the LCG method locally (element-wise) conservative, but it is also globally conservative - even in the absence of Neumann conditions along the global boundary.

3.5 LCG methods for convection dominated flows

In the previous chapter - dealing with the global Galerkin method, suitable stabilisation techniques were discussed for solving convection-dominated problems. It is shown in this section, that standard methods such as: the SUPG method [28, 41] and the Characteristic Galerkin (CG) scheme [39, 42, 44], can be readily used in LCG discretisation; Thus providing stabilisation of the non-self-adjoint convective terms [23], at elevated values of Péclet numbers. The extra stabilizing or higher order terms resulting from the discretisation are treated explicitly and locally - with no global assembly.

3.5.1 SUPG stabilised LCG methods

Incorporating the SUPG method [28, 41] into the proposed LCG formulation is simple and straight-forward. Equation (2.42) gives the semi-discrete form of the general convection-diffusion equation as

$$\beta \frac{\phi^{n+1} - \phi^n}{\Delta t} = -\frac{\partial F_j}{\partial x_i}^n + S_j \tag{3.49}$$

Spatially discretizing Equation (3.49) using the LCG method, gives a fully discrete form for solving over individual domains.

To obtain a SUPG stabilised LCG method, we start with the residual equation for an elemental-domain:

$$\beta \int_{\Omega_e} w_a \frac{\Delta \tilde{\phi}}{\Delta t} d\Omega_e = -\int_{\Omega_e} w_a \left(\frac{\partial \tilde{F}_j}{\partial x_j}^n - \tilde{S}_j \right) d\Omega_e \tag{3.50}$$

The weighting function on the RHS of Equation (3.50) is given by Equation (2.49) for SUPG stabilisation, giving:

$$\beta \int_{\Omega_e} N_a \frac{\Delta \tilde{\phi}}{\Delta t} d\Omega_e = - \int_{\Omega_e} \left(N_a + \frac{\alpha h}{2} \frac{u_i}{|\mathbf{u}|} \frac{\partial N_a}{\partial x_i} \right) \left(\frac{\partial \tilde{F}_j}{\partial x_j}^n - \tilde{S}_j \right) d\Omega_e \tag{3.51}$$

Here α is a parameter, whose optimal value is given by Equation (2.50) for linear triangular elements. For quadratic triangular elements the optimal parameter is given by Equations (2.50) and (2.52) for mid-side and vertex nodes respectively. Assuming a constant velocity field, the element-size, h, is best calculated in the stream-wise direction. This is done during the pre-processing, using the element portion of Equation (2.96). To remain consistent with the global Galerkin SUPG method (discussed in the last chapter), no special weighting is used on the LHS of Equation (3.51); i.e. $w_a = N_a$ here.

To provide continuity between elements in the LCG method, the numerical flux, $\hat{\tilde{F}}_i^n$, is introduced by introducing integration by parts

$$- \int_{\Omega_e} N_a \frac{\partial \tilde{F}_j}{\partial x_j}^n d\Omega_e = \int_{\Omega_e} \frac{\partial N_a}{\partial x_j} \tilde{F}_j^{\,n} d\Omega_e - \int_{\Gamma_e} N_a \hat{\tilde{F}}_j^n d\Gamma_e n_j \tag{3.52}$$

into Equation (3.51) giving

$$\begin{aligned} \beta \int_{\Omega_e} N_a \frac{\Delta \tilde{\phi}}{\Delta t} d\Omega_e &= \int_{\Omega_e} \frac{\partial N_a}{\partial x_j} \tilde{F}_j^{\,n} d\Omega_e - \int_{\Gamma_e} N_a \hat{\tilde{F}}_j^n d\Gamma_e n_i + \int_{\Omega_e} N_a \left(\tilde{S}_j \right)^n d\Omega_e \\ &- \int_{\Omega_e} \left(\frac{\alpha h}{2} \frac{u_i}{|\mathbf{u}|} \frac{\partial N_a}{\partial x_i} \right) \left(\frac{\partial \tilde{F}_j}{\partial x_j} - \tilde{S}_j \right)^n d\Omega_e \end{aligned} \tag{3.53}$$

Neglecting third- and higher-order terms, the explicit and implicit elemental-matrix forms of Equation (3.53) are written as:

$$\beta [\mathbf{M_e}] \{\Delta \mathbf{\Phi}\}^{n+1} = \Delta t \left([\mathbf{K_e}] \{\mathbf{\Phi}\} + [\mathbf{K_e^{supg}}] \{\mathbf{\Phi}\} + \{\hat{\mathbf{f}}_\mathbf{e}\} + \{\mathbf{f_e^{source}}\} + \{\mathbf{f_e^{supg}}\} \right)^n \tag{3.54}$$

and

$$\left(\beta [\mathbf{M_e}] + \Delta t [\mathbf{K_e}] \right) \{\mathbf{\Phi}\}^{n+1} = [\mathbf{M_e}] \{\mathbf{\Phi}\}^n + \Delta t \left([\mathbf{K_e^{supg}}] \{\mathbf{\Phi}\} + \{\hat{\mathbf{f}}_\mathbf{e}\} + \{\mathbf{f_e^{source}}\} + \{\mathbf{f_e^{supg}}\} \right)^n \tag{3.55}$$

respectively. The matrices: $[\mathbf{M_e}]$, $[\mathbf{K_e}]$, and $\{\hat{\mathbf{f}}_\mathbf{e}\}$ are given by Equations (3.11), and (3.16) respectively. The additional matrices appearing from the SUPG formulation are given by:

$$\begin{aligned}
[\mathbf{K_e^{supg}}] &= \int_{\Omega_e} \left(\frac{\alpha h}{2}\frac{u_i}{|\mathbf{u}|}\right) \frac{\partial \mathbf{N^T}}{\partial x_i} u_j \frac{\partial \mathbf{N}}{\partial x_j} d\Omega_e \\
\{\mathbf{f_e^{source}}\} &= \int_{\Omega_e} \mathbf{N^T N} d\Omega_e \{\mathbf{S_j}\}^n \\
\{\mathbf{f_e^{supg}}\} &= \int_{\Omega_e} \left(\frac{\alpha h}{2}\frac{u_i}{|\mathbf{u}|}\right) \frac{\partial \mathbf{N^T}}{\partial x_i} \mathbf{N} d\Omega_e \{\mathbf{S_j}\}^n
\end{aligned} \tag{3.56}$$

As mentioned previously the above system of simultaneous equations are solved for each of the elements in turn - independently of surrounding element equation sets.

3.5.2 Characteristic Galerkin (CG) based LCG methods

The SUPG stabilised LCG methods, developed above and given by Equations (3.54) and (3.55), are only first-order accurate in time. A higher-order time-accurate CG based LCG scheme for solving convection-dominated problems can also be easily implemented. This is done by using the *simple explicit* characteristic based procedure [39, 42, 44]. As with the global Galerkin method, temporal discretisation of Equation (2.1) is carried out first. Repeating Equation (2.58) for convenience

$$\beta \frac{\phi^{n+1} - \phi^n}{\Delta t} = -\left(\frac{\partial F_i}{\partial x_i} - S_i\right)^n + \frac{\Delta t}{2} u_k \frac{\partial}{\partial x_k}\left(\frac{\partial F_i}{\partial x_i} - S_i\right)^n + O(\Delta t^2) \tag{3.57}$$

gives the second-order time accurate semi-discrete form. The weighted residual form of Equation (3.57) is written (with Galerkin weighting $w_a = N_a$) for an elemental-domain as

$$\beta \int_{\Omega_e} N_a \frac{\Delta \tilde{\phi}}{\Delta t} d\Omega_e = -\int_{\Omega_e} N_a \left(\frac{\partial \tilde{F}_i}{\partial x_i} - \tilde{S}_i\right)^n d\Omega_e + \int_{\Omega_e} N_a \frac{\Delta t}{2} u_k \frac{\partial}{\partial x_k}\left(\frac{\partial \tilde{F}_i}{\partial x_i} - \tilde{S}_i\right)^n d\Omega_e \tag{3.58}$$

Continuity between elements is established using the weak form, given by Equation (3.52). This allows the numerical flux, which crosses element edges, to be introduced into the element matrix system. Integration by parts is also used on the additional second-order

stabilizing terms, that appear due to discretizing along the characteristics. As with the global Galerkin method, the boundary contribution arising from integration by parts of the stabilizing terms in Equation (3.58) are ignored because the original residual is zero [23]. Performing the above on Equation (3.58) and rearranging, gives the characteristic Galerkin discrete form of Equation (2.1) as

$$\begin{aligned}\beta\int_{\Omega_e} N_a\frac{\Delta\tilde{\phi}}{\Delta t}d\Omega_e &= \int_{\Omega_e}\frac{\partial N_a}{\partial x_i}\tilde{F}_i{}^n d\Omega_e - \int_{\Gamma_e} N_a\hat{\tilde{F}}_i^n d\Gamma_e n_i + \int_{\Omega_e} N_a\left(\tilde{S}_i\right)^n d\Omega_e \\ &- \int_{\Omega_e}\left(\frac{\Delta t}{2}u_k\frac{\partial N_a}{\partial x_k}\right)\left(\frac{\partial\tilde{F}_i}{\partial x_i}-\tilde{S}_i\right)^n d\Omega_e \end{aligned} \tag{3.59}$$

Neglecting third- and higher-order terms

$$\beta[\mathbf{M_e}]\{\Delta\mathbf{\Phi}\}^{n+1} = \Delta t\left([\mathbf{K_e}]\{\mathbf{\Phi}\}+[\mathbf{K_e^{cg}}]\{\mathbf{\Phi}\}+\{\hat{\mathbf{f}}_\mathbf{e}\}+\{\mathbf{f_e^{source}}\}+\{\mathbf{f_e^{cg}}\}\right)^n \tag{3.60}$$

and

$$\left(\beta[\mathbf{M_e}]+\Delta t[\mathbf{K_e}]\right)\{\mathbf{\Phi}\}^{n+1} = [\mathbf{M_e}]\{\mathbf{\Phi}\}^n + \Delta t\left([\mathbf{K_e^{cg}}]\{\mathbf{\Phi}\}+\{\hat{\mathbf{f}}_\mathbf{e}\}+\{\mathbf{f_e^{source}}\}+\{\mathbf{f_e^{cg}}\}\right)^n \tag{3.61}$$

Give the explicit and implicit elemental-matrix forms of Equation (3.59) respectively. The matrices: $[\mathbf{M_e}]$, $[\mathbf{K_e}]$, $\{\hat{\mathbf{f}}_\mathbf{e}\}$, and $\{\mathbf{f_e^{source}}\}$ are given by Equations (3.11), (3.16), and (3.56). The additional matrices appearing from the characteristic Galerkin formulation are given by:

$$\begin{aligned}[\mathbf{K_e^{cg}}] &= \int_{\Omega_e}\left(\frac{\Delta t}{2}u_k\right)\frac{\partial\mathbf{N^T}}{\partial x_k}u_i\frac{\partial\mathbf{N}}{\partial x_i}d\Omega_e \\ \{\mathbf{f_e^{cg}}\} &= \int_{\Omega_e}\left(\frac{\Delta t}{2}u_k\right)\frac{\partial\mathbf{N^T}}{\partial x_k}\mathbf{N}d\Omega_e\{\mathbf{S_i}\}^n \end{aligned} \tag{3.62}$$

As with the SUPG stabilised LCG method, the element matrix equation systems given in Equations (3.60) and (3.61) are solved for each element in turn. The numerical flux provides the continuity between neighbouring elements.

3.6 Application of the LCG method for incompressible fluid dynamic problems, using the CBS scheme and artificial compressibility

The direct application of the LCG method for solving incompressible fluid dynamics problems is considered in this section. In the previous section it was shown how to obtain a second-order time accurate stabilised CG-LCG method for solving convection diffusion problems. On a similar lines one can produce stabilised scheme for solving fluid dynamic problems, using the Characteristic Based Split (CBS) temporal discretisation of the Navier-Stokes equations. In this work the non-conservation form of the Navier-Stokes equations are used.

3.6.1 Splitting of the momentum equation and temporal-discretisation using a characteristic based method

The non-conservation form of the Navier-Stokes equations were expressed in a non-dimensional format in the previous Chapter. Repeating Equations (2.38), (2.39), and (2.40):

Continuity

$$\frac{1}{c^2}\frac{\partial p}{\partial t} = -\rho\frac{\partial u_i}{\partial x_i} \tag{3.63}$$

Momentum

$$\frac{\partial u_i}{\partial t} = -u_j\frac{\partial u_i}{\partial x_j} + \frac{1}{Re}\frac{\partial^2 u_i}{\partial {x_j}^2} - \frac{\partial p}{\partial x_i} \tag{3.64}$$

Temperature

$$\frac{\partial T}{\partial t} = -u_j\frac{\partial T}{\partial x_j} + \frac{1}{RePr}\frac{\partial^2 T}{\partial {x_j}^2} \tag{3.65}$$

In the above equation set the superscript asterisk, *, which was used to indicate non-dimensional variables, have been dropped to simplify the presentation. In addition, the continuity equation includes the isentropic relation of Equation (2.19). Although the speed

of sound, c, is infinite for an incompressible fluid, the inclusion allows a fully explicit CBS-LCG scheme to be constructed. This will be implemented by replacing c with an artificial compressibility parameter [23, 76, 82, 96, 97, 98, 112, 113, 114].

The proposed CBS-LCG approach discussed in this section, starts with splitting up the momentum equation. Here the term containing pressure is dropped from the equation, and an auxiliary intermediate velocity field is solved for instead. The intermediate velocity field is corrected at a later stage, once the pressure field is obtained from a pressure (continuity) equation. An alternative approach is of course to treat the pressure term as a source quantity. This alternative has been shown to give slightly more accuracy in transient problems, but is considered a less flexible approach by practitioners of the CBS scheme. This is due to the restrictions imposed by the Ladyshenskaya-Bubska-Brezzi (LBB)condition [109, 110, 111], which apply when incompressibility (or near incompressibility) is encountered. Details on circumventing the the LBB condition for the CBS scheme are given by Zienkiewicz and co-workers [23, 87].

By dropping the pressure term, Equation (3.64) becomes

$$\frac{\partial u_i{}^{\dagger}}{\partial t} = -u_j \frac{\partial u_i}{\partial x_j} + \frac{1}{Re} \frac{\partial^2 u_i}{\partial {x_j}^2} \tag{3.66}$$

To make the following CBS-LCG spatial discretisation procedure consistent to the previous sections of this chapter, Equation (3.66) is written in a similar fashion to Equation (2.42), i.e.

$$\frac{\partial u_i{}^{\dagger}}{\partial t} + \frac{\partial F_j}{\partial x_j} = 0 \tag{3.67}$$

Noting $\frac{\partial u_i{}^{\dagger}}{\partial x_i} = 0$, the flux term, F_j, is defined as

$$F_j = \left(u_j u_i - \frac{1}{Re} \frac{\partial u_i}{\partial x_j} \right) \tag{3.68}$$

Applying the CG temporal discretisation (see Equation (3.57)) to Equation (3.67) gives the following semi-discrete form.

$$\frac{u_i{}^{\dagger} - u_i{}^{n}}{\Delta t} = -\left(\frac{\partial F_j}{\partial x_j} \right)^n + \frac{\Delta t}{2} u_k \frac{\partial}{\partial x_k} \left(\frac{\partial F_j}{\partial x_j} \right)^n + O(\Delta t^2) \tag{3.69}$$

This is Step 1 of the LCG-CBS scheme. As the pressure term was split from the momentum equation, an dagger mark, †, is used for the intermediate value of u_i at time $(n+1)$. The correct value of ${u_i}^{n+1}$ is given by

$$ {u_i}^{n+1} = {u_i}^{\dagger} - \Delta t \left(\frac{\partial p}{\partial x_i} \right)^n + \frac{\Delta t^2}{2} u_k \frac{\partial}{\partial x_k} \left(\frac{\partial p}{\partial x_i} \right)^n + O(\Delta t^3) \tag{3.70} $$

This is Step 3 of the scheme. Before the velocity correction can be applied, the pressure has to be calculated independently from another source in the second step. Using the continuity relation given by Equation (3.63), the pressure can be written as

$$ \frac{1}{c^2} \frac{\Delta p}{\Delta t} = -\rho \frac{\partial {u_i}^{n+1}}{\partial x_i} \tag{3.71} $$

The ${u_i}^{n+1}$ term appearing in this equation has to be eliminated, as it is unknown. Using Equation (3.70) and neglecting terms higher than second-order, the pressure equation is

$$ \frac{1}{c^2} \frac{\Delta p}{\Delta t} = -\rho \frac{\partial}{\partial x_i} \left({u_i}^{\dagger} - \Delta t \left(\frac{\partial p}{\partial x_i} \right)^n \right) \tag{3.72} $$

Equation (3.72) has been derived by assuming a pseudo density variation in the continuity equation. For incompressible fluid flow the sonic velocity, c, approaches infinity, and the transient term disappears from the equation. In the absence of a transient term, the resulting Poisson-type equation has to be solved directly using a matrix solution method. It is possible to avoid this and keep the fully explicit nature of the scheme, by assuming a small amount of compressibility. However, even if c is finite, its value may still be very large and the solution procedure becomes highly restricted by severe time-step limits [23, 96, 113].

An artificial compressibility parameter, β, that has the dimensions of speed, has been successfully used in the literature [23, 76, 82, 96, 97, 98, 112, 113, 114]. to replace the wave speed. Assuming a steady-state exists, a sufficiently low value for β is selected instead. This eliminates the restrictions imposed by c. Replacing the real wave speed in Equation (3.72) with β, the pressure equation to be used for Step 2 of a fully explicit incompressible flow solution is

$$ \left(\frac{1}{\beta^2} \right)^n \frac{\Delta p}{\Delta t} = -\rho \frac{\partial}{\partial x_i} \left({u_i}^{\dagger} - \Delta t \left(\frac{\partial p}{\partial x_i} \right)^n \right) \tag{3.73} $$

The value of β can be given as a constant throughout the domain, but the recommended approach - which is adopted here, is to calculate β locally - based on both convective and diffusive time-step restrictions [23, 76, 96, 97, 98, 113, 114]. This gives not only a scheme which is suitable for different Reynolds numbers but more importantly it accommodates different flow regimes (convection and diffusion dominated) within a problem at a particular Reynolds number. In this work the relation

$$\beta = \max\left(\epsilon, \nu_{conv}, \nu_{diff}, \nu_{therm}\right) \tag{3.74}$$

is employed for a general non-isothermal fluid flow. The constant ϵ, appearing in Equation (3.74), ensures that β does not approach zero, and typically takes the value of $0.1 \leq \epsilon \leq 0.5$. ν_{conv} is the local convective velocity ,ν_{diff} is the local diffusive velocity, and ν_{therm} is the local thermal velocity - used when the temperature field is solved for. These velocities are calculated from the non-dimensional relations

$$\nu_{conv} = \sqrt{u_i u_i} \qquad \nu_{diff} = \frac{1}{hRe} \qquad \nu_{therm} = \frac{1}{hPr} \tag{3.75}$$

The above relations are given by Nithiarasu [96] in order keep the relations simple while still giving as good performance as the pre-conditioned AC schemes. Malan *et al.* [113] discuss in detail the local velocity relations given for pre-conditioned AC schemes, which are calculated differently.

The temperature field is calculated in the fourth and final step of the scheme. To start, Equation (3.65) is rewritten in the familiar form

$$\frac{\partial T}{\partial t} + \frac{\partial G_j}{\partial x_j} = 0 \tag{3.76}$$

To avoid confusion with Equation (3.68) the flux term is denoted by G_j, and is given as

$$G_j = \left(u_j T - \frac{1}{RePr}\frac{\partial T}{\partial x_j}\right) \tag{3.77}$$

Applying the CG temporal discretisation gives the following semi-discrete form of Step 4

$$\frac{T^{n+1}-T^n}{\Delta t} = -\left(\frac{\partial G_j}{\partial x_j}\right)^n + \frac{\Delta t}{2} u_k \frac{\partial}{\partial x_k}\left(\frac{\partial G_j}{\partial x_j}\right)^n + O(\Delta t^2) \tag{3.78}$$

Summarizing, the four semi-discrete steps are:

1. Solve for the intermediate velocity field, $u_i{}^{\dagger}$, using equation (3.69)
2. Solve for the pressure field, p, using equation (3.73)
3. Solve for the correct velocity field, u_i, using equation (3.70)
4. Solve for the Temperature field, T, using equation (3.78)

3.6.2 LCG spatial discretisation of the CBS scheme for the non-conservation form of incompressible flow equations

In this sub-section Equations (3.69), (3.73), (3.70), and (3.78) are discretised in space using the proposed locally conservative Galerkin (LCG) finite element procedure. As with the global Galerkin the variation of each of the variables is approximated by the standard spatial discretisation (Equation (3.4)) as

$$u_i \approx \tilde{u_i} = \mathbf{N}\mathbf{u_i}, \qquad p \approx \tilde{p} = \mathbf{N}\mathbf{p}, \qquad T \approx \tilde{T} = \mathbf{N}\mathbf{T} \tag{3.79}$$

As mentioned, the LBB condition is circumvented when the pressure gradient is removed from the momentum equation [23, 42, 109, 110, 111]. This allows an arbitrary choice of shape functions to be made for approximating the velocity, pressure, and temperature fields. Here, all variable approximations use identical shape functions. The optimal choice of weighting function for the semi-discrete equations is Galerkin, $w_a = N_a$, when using a characteristic-based discretisation.

The elemental-residual equation for Step 1, with Galerkin weighting, is

$$\int_{\Omega_e} N_a \frac{\Delta \tilde{u_i}^{\dagger}}{\Delta t} d\Omega_e = -\int_{\Omega_e} N_a \left(\frac{\partial \tilde{F_j}}{\partial x_j}\right)^n d\Omega_e + \int_{\Omega_e} N_a \frac{\Delta t}{2} u_k \frac{\partial}{\partial x_k}\left(\frac{\partial \tilde{F_j}}{\partial x_j}\right)^n d\Omega_e \tag{3.80}$$

Performing integration by parts on the RHS of Equation (3.80) gives the following weak form

$$\begin{aligned}\int_{\Omega_e} N_a \frac{\Delta \tilde{u_i}^\dagger}{\Delta t} d\Omega_e &= \int_{\Omega_e} \frac{\partial N_a}{\partial x_j} \tilde{F_j}^n d\Omega_e - \int_{\Gamma_e} N_a \hat{\tilde{F}}_j^n d\Gamma_e n_j \\ &- \int_{\Omega_e} \left(\frac{\Delta t}{2} u_k \frac{\partial N_a}{\partial x_k} \right) \left(\frac{\partial \tilde{F_j}}{\partial x_j} \right)^n d\Omega_e \end{aligned} \tag{3.81}$$

As with before, for the LCG method for the scalar convection-diffusion equation, the boundary term is replaced with a numerical flux, $\hat{\tilde{F}}_j$, at time (n). The boundary terms from the integration by parts of the second order stabilizing terms are neglected [42]. Neglecting also third- and higher-order terms, the final matrix form of Step 1, to solve over each element is

$$[\mathbf{M1_e}]\{\Delta \mathbf{u_i}^\dagger\}^{n+1} = \Delta t \left([\mathbf{K1_e}]\{\mathbf{u_i}\} + [\mathbf{K1_e^{cg}}]\{\mathbf{u_i}\} + \{\mathbf{f\hat{l}_e}\} \right)^n \tag{3.82}$$

$$\begin{aligned}[\mathbf{M1_e}] &= \int_{\Omega_e} \mathbf{N^T N} d\Omega_e \qquad [\mathbf{K1_e}] = \int_{\Omega_e} \frac{\partial \mathbf{N^T}}{\partial x_j} \left(u_j \mathbf{N} - \frac{1}{Re} \frac{\partial \mathbf{N}}{\partial x_j} \right) d\Omega_e \\ [\mathbf{K1_e^{cg}}] &= \int_{\Omega_e} \left(\frac{\Delta t}{2} u_k \right) \frac{\partial \mathbf{N^T}}{\partial x_k} u_j \frac{\partial \mathbf{N}}{\partial x_j} d\Omega_e \\ \{\mathbf{f\hat{l}_e}\}^n &= -\int_{\Gamma_e} \mathbf{N^T} u_j \mathbf{N} d\Gamma_e n_j \{\mathbf{u_i}\}^n + \frac{1}{Re} \int_{\Gamma_e} \mathbf{N^T N} d\Gamma_e n_j \left\{ \frac{\partial \mathbf{u_i}}{\partial \mathbf{x_j}} \right\}^n \end{aligned} \tag{3.83}$$

The elemental-residual equation for Step 2 of a fully explicit incompressible flow solution is

$$\int_{\Omega_e} N_a \left(\frac{1}{\beta^2} \right)^n \frac{\Delta \tilde{p}}{\Delta t} d\Omega_e = -\rho \int_{\Omega_e} N_a \frac{\partial}{\partial x_i} \left(\tilde{u_i}^\dagger - \Delta t \left(\frac{\partial \tilde{p}}{\partial x_i} \right)^n \right) d\Omega_e \tag{3.84}$$

Performing integration by parts on the RHS of Equation (3.84) gives the following weak form

$$\begin{aligned}\int_{\Omega_e} N_a \left(\frac{1}{\beta^2} \right)^n \frac{\Delta \tilde{p}}{\Delta t} d\Omega_e = &+ \rho \int_{\Omega_e} \frac{\partial N_a}{\partial x_i} \left(\tilde{u_i}^\dagger - \Delta t \left(\frac{\partial \tilde{p}}{\partial x_i} \right)^n \right) d\Omega_e \\ &- \rho \int_{\Gamma_e} N_a \left(\tilde{u_i}^\dagger - \Delta t \left(\frac{\partial \tilde{p}}{\partial x_i} \right)^n \right) d\Gamma_e n_i \end{aligned} \tag{3.85}$$

the final matrix form of Step 2, to solve over each element is

$$[\mathbf{M2_e}]\{\Delta\mathbf{p}\}^{n+1} = \Delta t\left([\mathbf{C2_e}]\{\mathbf{u_i}^\dagger\} - \Delta t[\mathbf{K2_e}]\{\mathbf{p}\}^n + \{\hat{\mathbf{f2}}_\mathbf{e}\}\right) \tag{3.86}$$

$$\begin{aligned}
[\mathbf{M2_e}] &= \int_{\Omega_e} \mathbf{N^T}\left(\frac{1}{\beta^2}\right)^n \mathbf{N} d\Omega_e \qquad [\mathbf{C2_e}] = \int_{\Omega_e} \frac{\partial \mathbf{N^T}}{\partial x_i}\mathbf{N} d\Omega_e \\
[\mathbf{K2_e}] &= \int_{\Omega_e} \frac{\partial \mathbf{N^T}}{\partial x_i}\frac{\partial \mathbf{N}}{\partial x_i} d\Omega_e \\
\{\hat{\mathbf{f2}}_\mathbf{e}\}^n &= -\int_{\Gamma_e} \mathbf{N^T N} d\Gamma_e n_i \left(\{\mathbf{u_i}^\dagger\} - \Delta t\left\{\frac{\partial \mathbf{p}}{\partial \mathbf{x_i}}\right\}^n\right)
\end{aligned} \tag{3.87}$$

The elemental-residual equation for Step 3 of a fully explicit incompressible flow solution is

$$\begin{aligned}
\int_{\Omega_e} N_a \tilde{u_i}^{n+1} d\Omega_e = \int_{\Omega_e} N_a \tilde{u_i}^\dagger d\Omega_e &- \Delta t \int_{\Omega_e} N_a \left(\frac{\partial \tilde{p}}{\partial x_i}\right)^n d\Omega_e \\
&+ \int_{\Omega_e} N_a \frac{\Delta t^2}{2} u_k \frac{\partial}{\partial x_k}\left(\frac{\partial \tilde{p}}{\partial x_i}\right)^n d\Omega_e
\end{aligned} \tag{3.88}$$

Performing integration by parts on the RHS of Equation (3.88) gives the following weak form

$$\begin{aligned}
\int_{\Omega_e} N_a \tilde{u_i}^{n+1} d\Omega_e = \int_{\Omega_e} N_a \tilde{u_i}^\dagger d\Omega_e &+ \Delta t \int_{\Omega_e} \frac{\partial N_a}{\partial x_i}\tilde{p}^n d\Omega_e - \Delta t \int_{\Gamma_e} N_a \hat{\tilde{p}}^n d\Gamma_e n_i \\
&- \int_{\Omega_e} \frac{\Delta t^2}{2} u_k \frac{\partial N_a}{\partial x_k}\left(\frac{\partial \tilde{p}}{\partial x_i}\right)^n d\Omega_e
\end{aligned} \tag{3.89}$$

The boundary terms from the integration by parts of the second order stabilizing terms are neglected - also third- and higher-order terms, the final matrix form of Step 3, to solve over each element is

$$[\mathbf{M3_e}]\{\mathbf{u_i}\}^{n+1} = [\mathbf{M3_e}]\{\mathbf{u_i}\}^\dagger - \Delta t\left([\mathbf{K3_e}]\{\mathbf{p}\} + [\mathbf{K3_e^{cg}}]\{\mathbf{p}\} + \{\mathbf{f3_e}\}\right)^n \tag{3.90}$$

$$
\begin{aligned}
[\mathbf{M3_e}] &= \int_{\Omega_e} \mathbf{N}^{\mathbf{T}}\mathbf{N}d\Omega_e \qquad [\mathbf{K3_e}] = \int_{\Omega_e} \frac{\partial \mathbf{N}^{\mathbf{T}}}{\partial x_i}\mathbf{N}d\Omega_e \\
[\mathbf{K3_e^{cg}}] &= \int_{\Omega_e} \left(\frac{\Delta t}{2}u_k\right) \frac{\partial \mathbf{N}^{\mathbf{T}}}{\partial x_k}\frac{\partial \mathbf{N}}{\partial x_i}d\Omega_e \\
\{\mathbf{f3_e}\}^n &= -\int_{\Gamma_e} \mathbf{N}^{\mathbf{T}}\mathbf{N}d\Gamma_e n_i \{\mathbf{p}\}^n
\end{aligned} \tag{3.91}
$$

The elemental-residual equation for Step 4, with Galerkin weighting, is

$$
\int_{\Omega_e} N_a \frac{\Delta \tilde{T}^{n+1}}{\Delta t} d\Omega_e = -\int_{\Omega_e} N_a \left(\frac{\partial \tilde{G}_j}{\partial x_j}\right)^n d\Omega_e + \int_{\Omega_e} N_a \frac{\Delta t}{2} u_k \frac{\partial}{\partial x_k}\left(\frac{\partial \tilde{G}_j}{\partial x_j}\right)^n d\Omega_e \tag{3.92}
$$

Performing integration by parts on the RHS of Equation (3.75) gives the following weak form

$$
\begin{aligned}
\int_{\Omega_e} N_a \frac{\Delta \tilde{T}^{n+1}}{\Delta t} d\Omega_e &= \int_{\Omega_e} \frac{\partial N_a}{\partial x_j} \tilde{G}_j{}^n d\Omega_e - \int_{\Gamma_e} N_a \hat{\tilde{G}}_j^n d\Gamma_e n_j \\
&\quad - \int_{\Omega_e} \left(\frac{\Delta t}{2} u_k \frac{\partial N_a}{\partial x_k}\right)\left(\frac{\partial \tilde{G}_j}{\partial x_j}\right)^n d\Omega_e
\end{aligned} \tag{3.93}
$$

The final matrix form of Step 4, to solve over each element is

$$
[\mathbf{M4_e}]\{\Delta \mathbf{T}\}^{n+1} = \Delta t \left([\mathbf{K4_e}]\{\mathbf{T}\} + [\mathbf{K4_e^{cg}}]\{\mathbf{T}\} + \{\hat{\mathbf{f4_e}}\}\right)^n \tag{3.94}
$$

$$
\begin{aligned}
[\mathbf{M4_e}] &= \int_{\Omega_e} \mathbf{N}^{\mathbf{T}}\mathbf{N}d\Omega_e \qquad [\mathbf{K4_e}] = \int_{\Omega_e} \frac{\partial \mathbf{N}^{\mathbf{T}}}{\partial x_j}\left(u_j\mathbf{N} - \frac{1}{RePr}\frac{\partial \mathbf{N}}{\partial x_j}\right) d\Omega_e \\
[\mathbf{K4_e^{cg}}] &= \int_{\Omega_e} \left(\frac{\Delta t}{2}u_k\right) \frac{\partial \mathbf{N}^{\mathbf{T}}}{\partial x_k} u_j \frac{\partial \mathbf{N}}{\partial x_j} d\Omega_e \\
\{\hat{\mathbf{f4_e}}\}^n &= -\int_{\Gamma_e} \mathbf{N}^{\mathbf{T}} u_j \mathbf{N} d\Gamma_e n_j \{\mathbf{T}\}^n + \frac{1}{RePr}\int_{\Gamma_e} \mathbf{N}^{\mathbf{T}}\mathbf{N}d\Gamma_e n_j \left\{\frac{\partial \mathbf{T}}{\partial x_j}\right\}^n
\end{aligned} \tag{3.95}
$$

3.6.3 Local time-stepping

In the previous Chapter, the use of local time-stepping was discussed for increasing the efficiency (viz faster convergence rates) of the global Galerkin CBS scheme for solving

compressible flow problems. The use of local time-stepping is also possible when using the proposed LCG-CBS scheme with artificial compressibility. The next sub-section discusses how to retrieve the true transient solution when both artificial compressibility and local time-stepping are used.

The local time-step limitation at each node a, for the artificial compressibility LCG-CBS method is given by

$$\Delta t_a = \frac{h_a}{\sqrt{u_i u_i} + \beta} \tag{3.96}$$

for a viscous incompressible flow. Where h_a is the local element-size at node a. The value of h_a is given by Equation (2.95) as the minimum mid-height of all the surrounding connecting elements - as illustrated in Figure 2.2. The artificial compressibility parameter is calculated from Equation (3.74). The inclusion of β in Equation (3.96) allows the viscous wave speeds to be included into the local time-step limit. The calculated Δt_a is in practice multiplied by a safety factor, sf, where ($sf \leq 1.0$). The actual value of sf depends on the problem being simulated and the mesh used. The recommended approach [96], is to start with a maximum value and reduce until the scheme starts converging.

For examining the error of the solution for the AC-CBS scheme, for the global Galerkin method, the residual norm of the pressure residual was used [96, 97].

$$error = \frac{1}{nnode}\sqrt{\sum_{i=1}^{nnode}\left[\frac{1}{\beta^2}\frac{p^{n+1}-p^n}{\Delta t}\right]^2} \tag{3.97}$$

where *nnodes* is the total number of nodes in the mesh. It is also used to examine the convergence of the artificial compressibility LCG-CBS scheme, discussed here. The criterion for reaching steady-state is that this error should be reduced to a value of 1×10^{-5}.

3.6.4 Recovering a transient solution via a dual time-stepping approach

For the solution of transient incompressible-flow problems,using the proposed LCG-CBS scheme with artificial compressibility, a dual time-stepping procedure [115] is implemented. The method has been shown to be very successful for both pre-conditioned

artificial-compressibility finite volume schemes [113, 114], and the global Galerkin fully-explicit CBS scheme with artificial-compressibility [23, 76, 96, 97, 98]

Essentially, the dual time-stepping procedure transforms an unsteady flow calculation into a series of instantaneous steady states in real-time. A pseudo time-step is then used to iterate the solution within each instantaneous steady-state until the solution converges to a desired level of tolerance. By incorporating dual time-stepping, the error in the transient solution, introduced by both lumping the mass matrix and using the artificial compressibility parameter, is considerably reduced.

In order to recover the true transient solution, a real time term is added to the momentum equation. The CBS temporal discretisation supports two approaches: It can either be added to Step 1 (Equation 3.69), or Step 3 (Equation 3.70) of the scheme. To remain consistent with the global Galerkin CBS scheme [76, 96, 97, 98], the latter is chosen here. The addition of a true transient term, $\{\mathbf{u_i}\}^\tau$, leads to the following modified third step

$$\{\mathbf{u_i}\}^{n+1} = \{\mathbf{u_i}\}^\dagger - [\mathbf{M3_e}]^{-1}\Delta t\left([\mathbf{K3_e}]\{\mathbf{p}\} + [\mathbf{K3_e^{cg}}]\{\mathbf{p}\} + \{\mathbf{f3_e}\}\right)^n - \frac{\Delta t}{\Delta \tau}\{\Delta \mathbf{u_i}\}^\tau \quad (3.98)$$

where $\Delta\tau$ is the real time-step. When using the dual time-stepping scheme Δt becomes the pseudo time-step. In order to get a second-order real time accuracy, $\{\Delta \mathbf{u_i}\}^\tau$ is approximated with an implicit second-order backward-difference formula

$$\{\mathbf{u_i}\}^\tau = \frac{3\{\mathbf{u_i}\}^{m+1} - 4\{\mathbf{u_i}\}^m + \{\mathbf{u_i}\}^{m-1}}{2} \quad (3.99)$$

In Equation 3.99 $\{\mathbf{u_i}\}^{m+1}$ is the n^{th} pseudo-time level value within the pseudo-time loop. $\{\mathbf{u_i}\}^m$ is the steady-state solution at the last real time-step, $\{\mathbf{u_i}\}^{m-1}$ is the steady-state solution at one real time-step before the last. Clearly both the latter vectors need to be appropriately stored at the end of each real time-step. Due to the implicit nature of the scheme, the real time-step size is unrestricted and only governed by the quality of the transient solution required. The pseudo time-step however, is locally calculated and subject to the stability conditions discussed in the last section.

3.7 Summary

In this chapter, a locally conservative Galerkin (LCG) method, has been presented for the solution of the scalar conservation equations and the Navier-Stokes equations.

This Chapter gives a substantial introduction to this new and appealing scheme. The fundamentals of the method have been clearly defined. The element edge fluxes - which play a crucial role in obtaining an element-by-element solution have been presented, along with procedures for increasing accuracy via quadratic-order elements. Strategies for both explicit and implicit solutions strategies were also proposed - with and without a lumped-mass matrix.

A continuous solution throughout the domain is retrieved at the end of each time-step. This avoids multiple solutions at the nodes, but more importantly allows the numerical flux to be calculated for the next time-step. The diffusive component of the edge-fluxes, were shown to be easily computed from the continuous solution. Separate procedures developed for linear and quadratic elements, were discussed in detail.

Both the LCG and global Galerkin method were analysed and surprising familiarities were found. Particularly for the explicit lumped-mass versions of the two methods. Here, an identical discrete equation was obtained if the element areas are of equal size. The former method, however, possesses a residual flux on the global boundary after the averaging procedure is made. It is this residual flux, that gives the LCG method explicit global conservation - even in the absence of Neumann-type boundary conditions.

For convection-dominated problems and incompressible flow simulations, a number of LCG schemes were developed in this chapter. For convection-diffusion problems, procedures for incorporating the SUPG method and the *simple explicit* Characteristic Galerkin (CG) scheme were given. A fully explicit CBS scheme - using an artificial compressibility, was used as the temporal stencil for an LCG method for solving incompressible flows.

Closure of this chapter was completed with details for recovering the transient solution. A dual time-stepping procedure was implemented into the CBS-LCG scheme. By incorporating dual time-stepping into the scheme, the solution of unsteady flows is permitted.

Chapter 4

Influences of element-size calculation and variable smoothing on the global Galerkin CBS scheme for inviscid compressible flow

4.1 Introduction

In Chapter 2, the characteristic based split (CBS) algorithm - for solving compressible flow problems, was presented. As mentioned previously, the CBS scheme introduces consistent convection stabilisation, via the second-order temporal discretisation. In the CBS scheme, the convection stabilisation terms are controlled by the time-step, which is in turn based on the stability criteria involving the local element-sizes. The standard element-sizes employed in the past [23, 42, 43, 88, 90, 91, 96, 94], were calculated as part of the pre-processing stage and stored for use during the time-stepping operation. Once calculated, these element-sizes were not altered during the time-stepping process. This method of evaluating the element-sizes is computationally straight forward and inexpensive. It is not clear, however, whether local element-size calculation methods [128] will have any significant influence on the solution.

As discussed in Chapter 2, if the flow problem is purely convective, i.e. inviscid, then the local nodal time-step depends only on the convective and sonic velocities. SUPG schemes [23, 28, 128], often use a *directional* element-size in order to obtain optimal stabilisation of the convective terms. It is interesting to see, if taking into account the streamline direction when calculating the local element-size at a node, has any positive effect on computing inviscid compressible flow using CBS. Here, an updating of element-sizes is required at each time step during the transient stages of the calculation. Calculating the element-size in the streamline direction is computationally more expensive than the standard method, especially for large scale problems. However, the advantages gained using a streamlined element-size calculation should not be overlooked.

In this Chapter, therefore, the effect of using a flow-dependent local element-size in the streamline direction - for the calculation of local time-steps, is considered. The proposed stream-wise local element-size is calculated using Equation (2.98). Investigations are carried out on the effect of such an element-size calculation, by solving inviscid compressible flow past a NACA0012 aerofoil. In addition to the effect of the element-size calculation, the issue of simulating inviscid flows at low Mach numbers, is addressed. This is done, by employing the proposed variable smoothing approach given in Equation (2.102). The employed variable smoothing approach, permits oscillation free solutions at a Mach number as small as 0.01. The variable smoothing is particulary important to both CBS schemes, as it allows accurate and non-oscillatory low subsonic solutions to be obtained on the same non-isothermal code used to perform supersonic simulations - without having to remove the energy coupling. With a combination of the element-sizes in the streamline direction and the variable smoothing, inviscid solutions are obtained for Mach numbers ranging from 0.01 to 3.0.

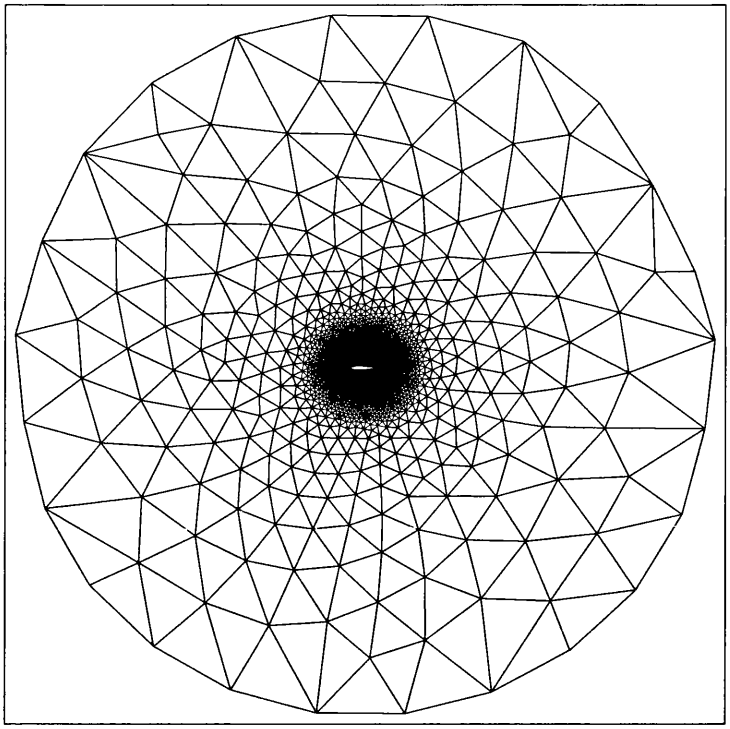
4.2 Inviscid compressible flow over a NACA0012 aerofoil

To investigate the performance of the local element-size calculation in the streamline direction (Equation 2.98) and the effect of the local variable smoothing (Equation 2.102), an example problem of inviscid flow past a NACA0012 aerofoil is considered in

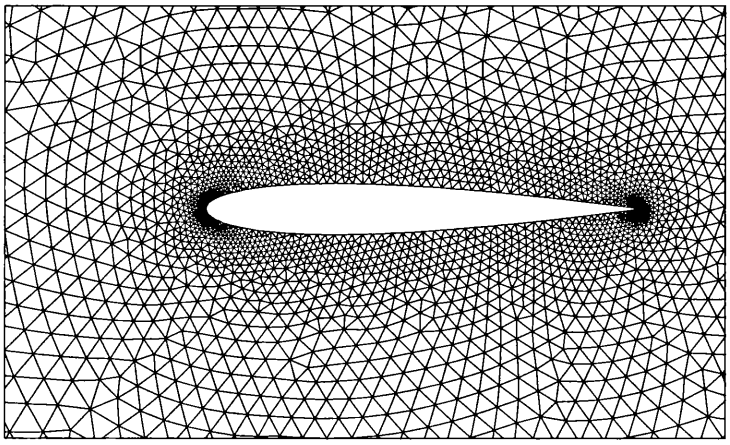
this section. Here, the problem has been solved for various Mach numbers in the range of $0.01 \leq Ma \leq 3.00$, using the CBS scheme. This covers all regimes - including subsonic, transonic, and supersonic flow. Results obtained using a CBS scheme with an element-size calculated in the streamline direction, are compared against: analytical solutions, benchmark data, and solutions obtained from the CBS scheme using a standard element-size calculation. The use of local variable smoothing in the CBS scheme, is applicable to both types of element-size calculation. In this section it is used, in the place of artificial shock capturing diffusion, to solve compressible flow problems at low mach numbers (i.e. $Ma \leq 0.8$), without removing the coupling between energy and the other transport variables.

The computational domain and mesh used in the analysis is shown in Figure 4.1. The domain is circular with a diameter equal to 25 times the chord length of the NACA0012 aerofoil. The leading edge of the aerofoil is at the geometric centre $(0.0, 0.0)$ of the domain. Inlet conditions are prescribed on the left half of the circular boundary, and exit conditions are prescribed on the right half. Standard practise [23, 132, 138] is adopted here on the correct implementation of boundary conditions for subsonic and supersonic flows. For $Ma \geq 1.0$, all velocity components and the density are prescribed at the inlet, no conditions are prescribed at the exit. For $Ma < 1.0$, all velocity components are prescribed at the inlet, and the density is prescribed at the exit. An unstructured mesh consisting of 7351 elements and 3753 nodes is used in the domain discretisation. A close-up view of the mesh in the vicinity of the aerofoil is also given in Figure 4.1(b). The mesh and other parameters used in the calculations are identical for both, the CBS scheme with standard element-size calculation and the CBS scheme with stream-wise element-size calculation.

Figure (4.2) gives the computed density contours using the proposed element-size calculation, for flows within the subsonic-transonic region. Results are given here, for Ma = 0.25, 0.50, 0.65, and 0.85. The variable smoothing parameter, α, defined in Equation (2.102) was activated when carrying out the computations for Mach numbers of 0.80 and below. Figure (4.3) gives the computed density contours using the proposed element-size calculation, for flows within the transonic-supersonic region, viz Ma = 0.95, 1.20, 2.00, and 3.00. From a qualitative point of view, all results shown are in excellent agreement with the results obtained using the standard element-size calculation. Considering the coarseness of

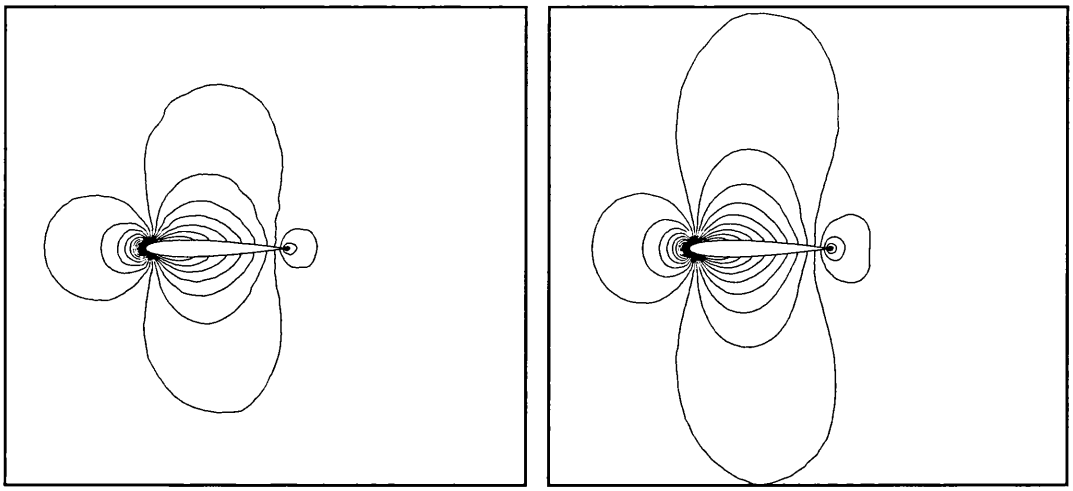


(a) Finite element mesh - whole domain



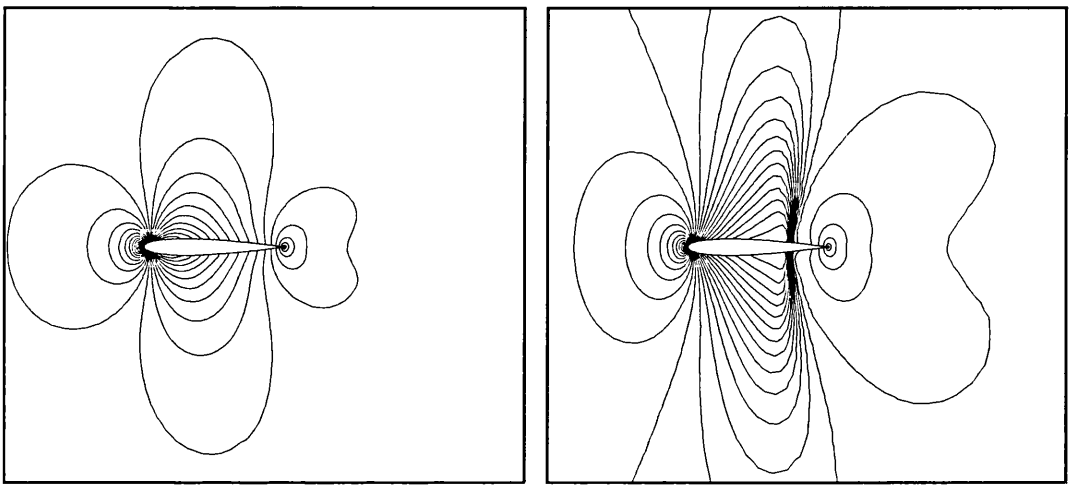
(b) Finite element mesh - detail of mesh close to aerofoil

Figure 4.1: Compressible flow over a NACA0012 aerofoil. Linear triangular finite element mesh with NACA0012 aerofoil profile, 7351 elements and 3753 nodes



(a) $Ma = 0.25$

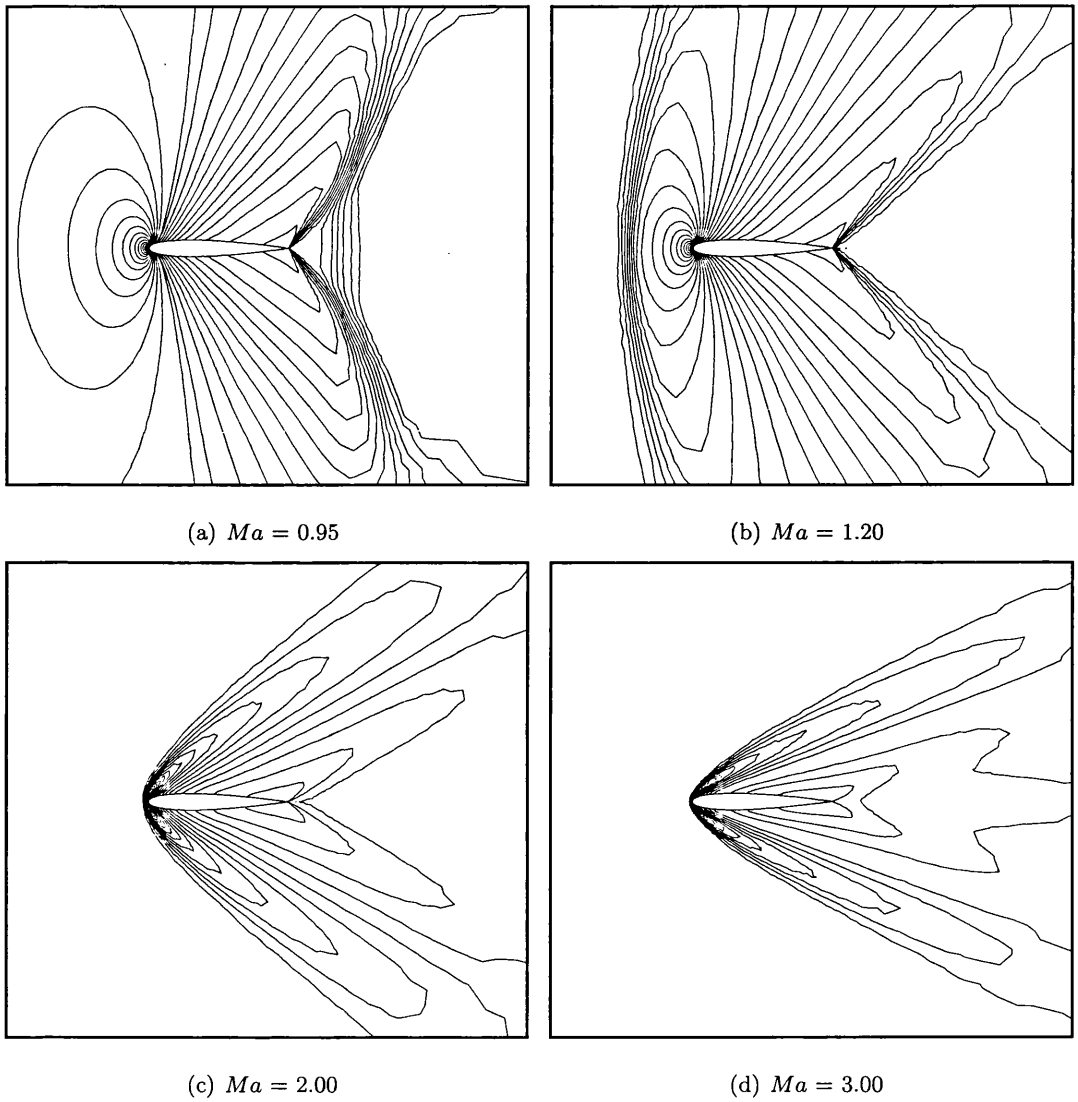
(b) $Ma = 0.50$



(c) $Ma = 0.65$

(d) $Ma = 0.85$

Figure 4.2: Compressible flow over a NACA0012 aerofoil. Contours of density at subsonic and transonic values of Ma, using proposed stream-wise element-size calculation.



(a) $Ma = 0.95$ (b) $Ma = 1.20$

(c) $Ma = 2.00$ (d) $Ma = 3.00$

Figure 4.3: Compressible flow over a NACA0012 aerofoil. Contours of density at transonic and supersonic values of Ma, using proposed stream-wise element-size calculation.

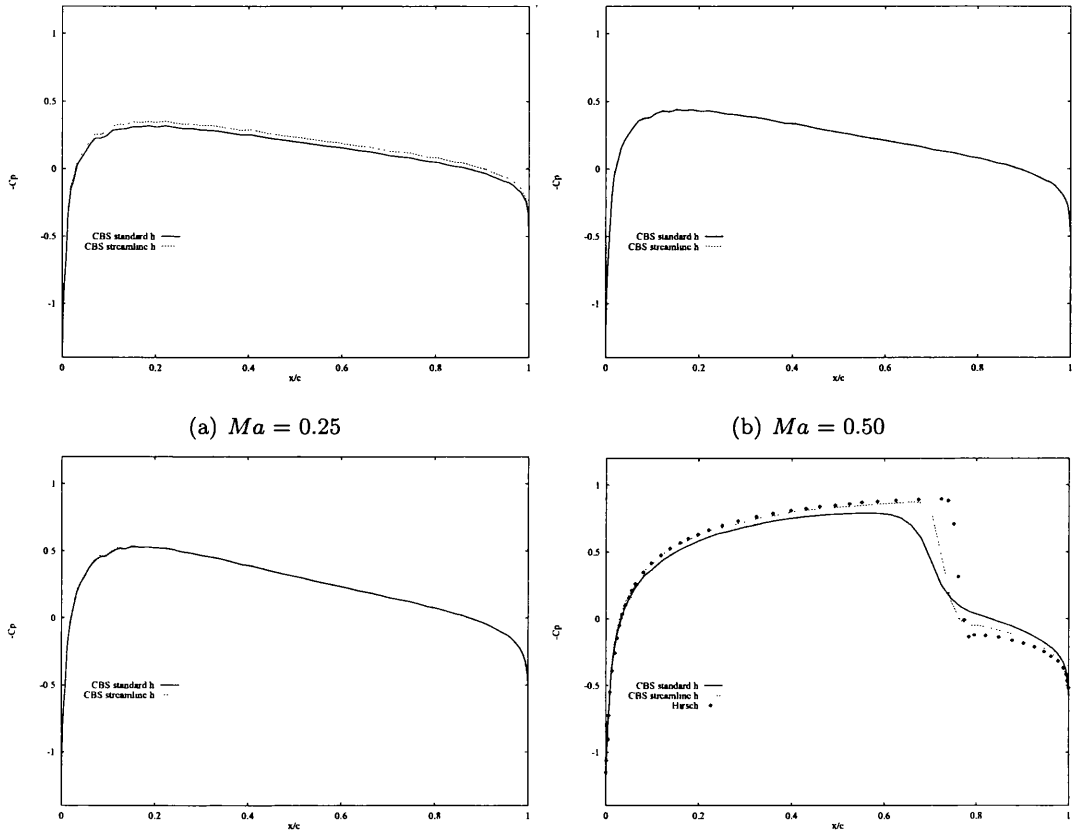
the mesh employed for the calculations, the contours are smooth and show good symmetry along the horizontal centreline that passes through the chord of the aerofoil.

A quantitative analysis of the results obtained using the proposed element-size calculation is made by examining the coefficient of pressure, given in Appendix A by Equation A.1, along the surface of the aerofoil. Figures (4.4) and (4.5) display the computed Cp plots corresponding to the flows given in Figures (4.2) and (4.3) respectively. The results have been compared with results obtained - using the same input data, from the CBS scheme with a standard element-size calculation. Where possible, the computed Cp plots for both CBS schemes have been supported by benchmark data from Hirsch [132] and Pulliam *et al.* [139].

In most cases, both the CBS schemes - with proposed and standard element-size calculations, closely agree. The exception is at the transonic region, particularly at $Ma = 0.85$. Here the standard CBS scheme is clearly more diffusive than the CBS scheme with the proposed element-size calculation, for the same specified input-parameters. It is obvious, from the comparison between the benchmark structured grid data given by [132] and present solution that the element-size calculation in the streamline direction substantially improves the results obtained at $Ma = 0.85$. Here, however, the influence of the interaction between the shock-capturing diffusion and element-size calculation is not ruled out, especially close to the shock. The general conclusion is that the element-size in the stream line direction gives optimal shock capturing viscosity and second order convection stabilisation. However, it is difficult to individually quantify the effects of these terms.

A detailed comparison of the density, temperature, and pressure variables - at the leading edge stagnation point, has been made for the CBS scheme, using both proposed and standard element-size calculations. The results of which are presented in Table (4.1). In addition, Table (4.1) gives the analytical values for the leading edge stagnation point. These are calculated using Equations (2.32) to (2.34), and tables of perfect gas flow functions - involving shock waves [129].

For convenience, the variation of stagnation values of density against the Mach number is shown in Figure (4.6). Here it can be seen quite clearly that superior accuracy over the standard CBS scheme is achieved when an element-size in the streamline direction



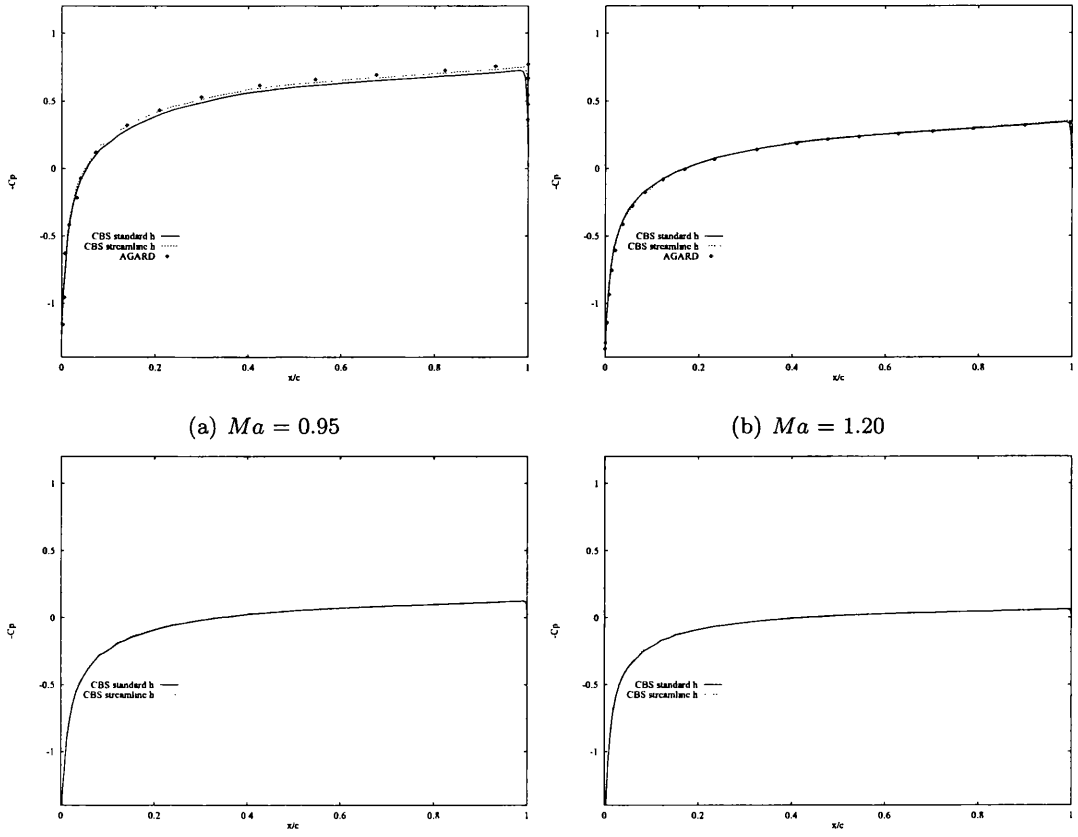
(a) $Ma = 0.25$

(b) $Ma = 0.50$

(c) $Ma = 0.65$

(d) $Ma = 0.85$

Figure 4.4: Compressible flow over a NACA0012 aerofoil. C_p distribution at subsonic and transonic values of Ma using the proposed stream-wise element-size calculation, compared with results obtained from using the standard element-size calculation.



(a) $Ma = 0.95$

(b) $Ma = 1.20$

(c) $Ma = 2.00$

(d) $Ma = 3.00$

Figure 4.5: Compressible flow over a NACA0012 aerofoil. C_p distribution at transonic and supersonic values of Ma using the proposed stream-wise element-size calculation, compared with results obtained from using the standard element-size calculation.

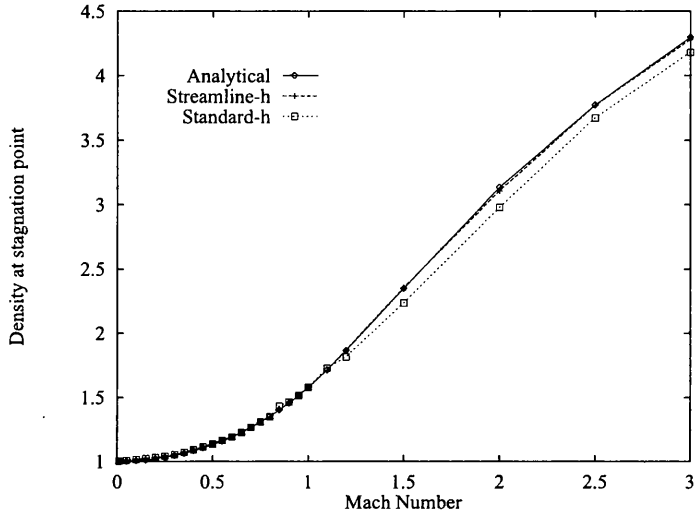


Figure 4.6: Compressible flow over a NACA0012 aerofoil. Plot of the density variable at the leading edge stagnation points as a function of Ma. Comparisons made with calculated values from the proposed and standard element-size calculations, and analytical solution.

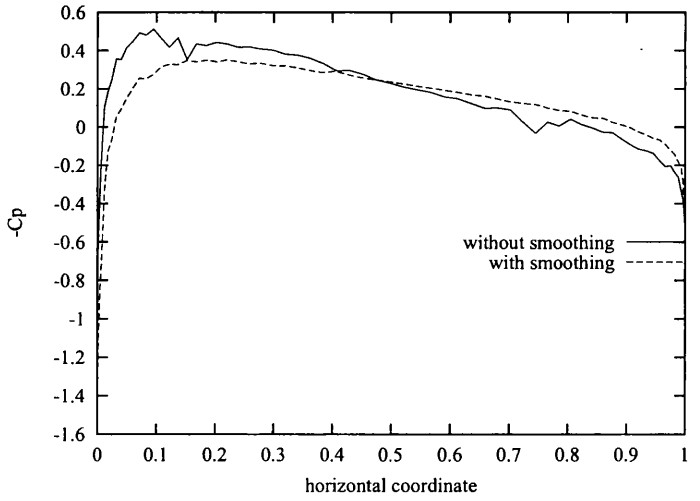


Figure 4.7: Compressible flow over a NACA0012 aerofoil. C_p distribution at $Ma = 0.25$ using variable smoothing, compared with the C_p distribution obtained from using zero smoothing.

is used, especially when the Mach number is above unity. The improvement in results at supersonic speeds may be attributed to the changes introduced by the time steps in the higher order stabilizing terms (Equations (2.66),(2.70),(2.67) and (2.71)) and shock capturing viscosity (Equation (2.100)). However, it has been shown previously that the value of shock capturing viscosity employed is almost nil at the stagnation points [91]. It is, therefore, conveniently argued that the second order terms of Equations (2.66),(2.70),(2.67) and (2.71) are responsible for the improved stagnation values. Among these terms, the second order pressure term (Equation (2.70)) multiplied by the time step directly influences the stagnation density values.

To illustrate the effectiveness of the variable smoothing at low Mach numbers, the solutions obtained at a Mach number of 0.25 with and without the variable smoothing, have been compared in Figure 4.7. For both results, the element-size calculated in the streamline direction was used. Without variable smoothing the non-isothermal code failed to give an accurate solution even with additional shock capturing, second-order diffusion. However, with variable smoothing an accurate oscillation free solution was obtained (see also the result in Figure 4.2(a)). As seen the smoothed solution is accurate and the solution without smoothing is oscillatory and wrong. It should be noted here, that smoothing is also required to get a non-oscillatory result at this Mach number when a standard element-size calculation is employed in the CBS scheme.

4.3 Summary

Compressible inviscid flow over a NACA0012 aerofoil, was simulated in this chapter. For inviscid flow, the local time-step criteria depends only on the local element-size and the sonic and convective velocities. This allows the standard element-size calculation, to be made more accurate by taking in to account the nodal stream-wise direction.

As a result, a comparative study was carried out in this chapter, to evaluate the performance of two local element-size calculations - used in the CBS algorithm. It was found that the proposed modification in the local element-size calculation, based on the streamline direction, improved accuracy, especially in the transonic region.

Table 4.1: Compressible flow over a NACA0012 aerofoil. Analytical and computed values of density, temperature, and pressure at the leading edge stagnation point. Computed results given for the CBS scheme using proposed element-size calculation and standard element-size calculation.

	Exact			S.line h			Std. h		
Ma	ρ_o	T_o	p_o	ρ_o	T_o	p_o	ρ_o	T_o	p_o
3.00	4.2970	0.7818	0.9611	4.2797	0.7705	0.9422	4.1796	0.7587	0.9277
2.50	3.7724	0.9028	0.9738	3.7680	0.8769	0.9438	3.6723	0.8514	0.8934
2.00	3.1321	1.1295	1.0105	3.1064	1.1181	0.9924	2.9789	1.1406	0.9708
1.50	2.3493	1.6147	1.0838	2.3538	1.6027	1.0778	2.2349	1.6331	1.0428
1.20	1.8653	2.2380	1.1931	1.8689	2.2292	1.1819	1.8135	2.2472	1.1675
1.10	1.7146	2.5686	1.2577	1.7099	2.5486	1.2451	1.7261	2.4771	1.2216
1.00	1.5770	3.0025	1.3536	1.5782	2.9606	1.3407	1.5796	2.9182	1.3170
0.95	1.5140	3.2742	1.4159	1.5179	3.2244	1.3984	1.5141	3.2048	1.3864
0.90	1.4550	3.5895	1.4929	1.4517	3.5628	1.4777	1.4630	3.4903	1.4590
0.85	1.4010	3.9654	1.5868	1.4018	3.9077	1.5651	1.4319	3.7943	1.5523
0.80	1.3510	4.4102	1.7020	1.3489	4.4506	1.7152	1.3479	4.3141	1.6614
0.75	1.3051	4.9467	1.8451	1.3081	4.9858	1.8634	1.3103	5.0028	1.8729
0.70	1.2630	5.6071	2.0233	1.2656	5.6497	2.0429	1.2673	5.6687	2.0526
0.65	1.2250	6.4201	2.2468	1.2264	6.4750	2.2689	1.2276	6.4964	2.2786
0.60	1.1900	7.4514	2.5317	1.1915	7.5389	2.5644	1.1928	7.5562	2.5752
0.55	1.1580	8.7686	2.9020	1.1638	8.9143	2.9642	1.1655	8.9417	2.9777
0.50	1.1300	10.500	3.3914	1.1357	10.659	3.4587	1.1369	10.692	3.4731
0.45	1.1040	12.852	4.0529	1.1118	13.036	4.1411	1.1130	13.073	4.1573
0.40	1.0820	16.125	4.9866	1.0895	16.326	5.6823	1.0904	16.368	5.0775
0.35	1.0620	20.919	6.3499	1.0700	21.119	6.4569	1.0710	21.167	6.4752
0.30	1.0460	28.278	8.4524	1.0535	28.497	8.5776	1.0540	28.552	8.5953
0.25	1.0320	40.000	11.943	1.0410	40.861	12.154	1.0412	40.796	12.137
0.20	1.0200	63.000	18.358	1.0280	63.390	18.631	1.0311	63.327	18.657
0.15	1.0110	112.27	32.811	1.0220	111.67	32.254	1.0227	112.20	32.785
0.10	1.0050	250.50	71.929	1.0150	251.41	72.429	1.0159	251.28	72.940
0.05	1.0001	1001.0	286.29	1.0073	1002.2	288.42	1.0071	1001.9	288.30
0.01	1.0000	25000.	7142.9	1.0014	25008.	7155.6	1.0026	24963.	7142.9

A variable smoothing algorithm was also introduced and shown to aid both schemes to enable efficient subsonic flow simulations. With the flow variable smoothing activated for Mach numbers of 0.80 or less, the scheme was able to produce results without oscillations. The success of using the variable smoothing procedure, was particulary important. This allowed the full range of flow types: $0.01 \leq Ma \leq 3.00$, to be investigated using the same compressible code. For subsonic flows, the only other change to the CBS code was turning on a small subroutine that prescribed density along the exit boundary when $Ma < 1.00$. This was automatic, preventing any major inconvenience.

Chapter 5

The LCG Method for Problems of Pure Diffusion

5.1 Introduction

In Chapter 3, a locally conservative Galerkin (LCG) method, was discussed. The LCG method allows an element-by-element solution to the discrete equations. Formulations were presented for both linear and quadratic triangular elements. When using linear elements the flux calculation is based on an element-averaged diffusive flux [1]. It was proposed in Chapter 3, that increased accuracy in the flux calculation may be obtained if higher-order elements are used. Using the quadratic interpolation functions, enables the diffusive fluxes to be nodally calculated along the element edges. This approach gives a more accurate computation of flux at each time-step, which is not possible when using linear elements.

In this chapter, a direct validation of the LCG formulation, is carried out for problems of pure diffusion. Here the implemented approaches for both linear and quadratic elements are thoroughly tested for problems of steady-state heat conduction. The robustness of the LCG methods are proved by performing tests on both structured and unstructured grids. Solutions and convergence data are given, using both linear and quadratic triangular finite elements. Additionally, the performance of both the implicit and explicit variations of the proposed LCG approaches are considered, all with and without a lumped mass matrix.

The application of the LCG method in more than two space dimensions is proved by considering the same heat conduction problem in 3D. Unstructured meshes of linear tetrahedral elements are used. Again, the validation of both the implicit and explicit variations of the proposed LCG approaches, with and without a lumped mass matrix, is given. This chapter closes with a 2D analysis of the LCG methods' performance when a source term is present in the governing equation.

5.2 Steady-state heat conduction

In this section, the LCG methods are analysed for their competency in solving linear heat conduction problems, in two- and three-space dimensions. For problems of heat conduction [140], the scalar variable ϕ in Equation (3.1) is replaced by the temperature T, and the coefficient β is replaced with ρc_p - the product of density and specific heat at constant pressure. The flux is defined by Fourier's law as

$$F_i = -k\frac{\partial T}{\partial x_i} \tag{5.1}$$

with k representing the thermal conductivity .

5.2.1 Two-dimensional steady-state heat conduction

The first problem considered in this chapter, is the solution of the steady-state temperature distribution within a unit-square plate - subject to Dirichlet boundary conditions on all its four sides. The problem definition is shown Figure 5.1. A temperature of 500°C is prescribed along the top edge of the plate. On the other three sides, a constant temperature of 100°C is enforced. The thermal conductivity, density and specific heat are all assumed to be equal to unity in this problem. An analytical solution for this problem exists and is given by Holman [141] as

$$T = T_{side} + (T_{top} - T_{side})\frac{2}{\pi}\sum_{n=1}^{\infty}\frac{(-1)^{n+1}+1}{n}sin\left(\frac{n\pi x_1}{w}\right)\frac{sinh\left(\frac{n\pi x_2}{w}\right)}{sinh\left(\frac{n\pi H}{w}\right)} \tag{5.2}$$

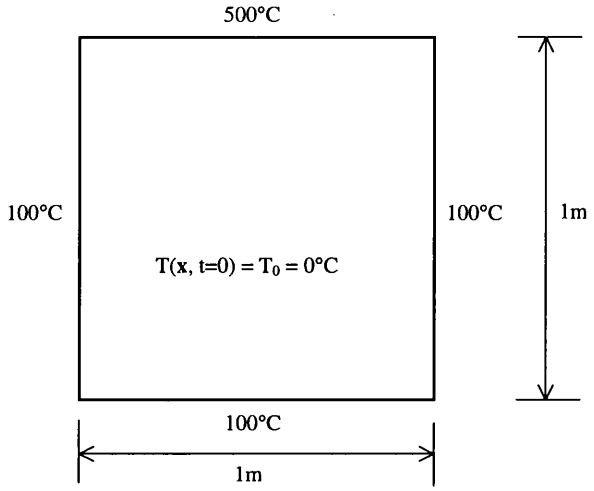


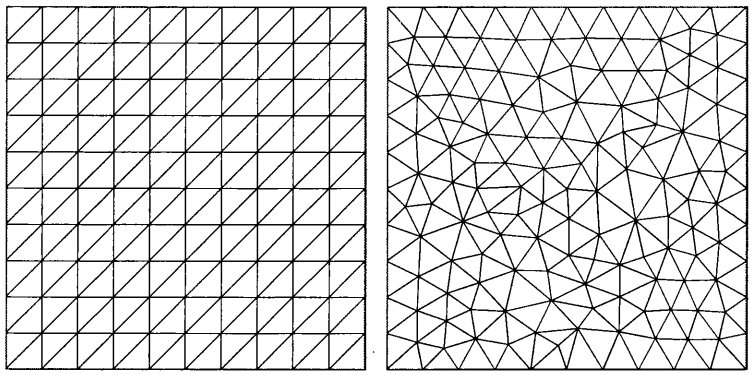
Figure 5.1: Steady-state heat conduction in a square plate. Geometry and isothermal boundary conditions

where w is the width, H is the height of the plate, T_{top} is the temperature at the top edge and T_{side} is the temperature at the other sides of the plate.

Two meshes were used in this study, these are shown in Figure 5.2. The first, Mesh A, is a uniform structured mesh with 200 triangular elements. Mesh B, is an unstructured mesh with 266 triangular elements. The initial temperature of the plate is assumed to be at 0°C, and the steady-state solution is obtained through time-stepping to a prescribed residual error tolerance of $\epsilon < 1 \times 10^{-09}$. Here ϵ is the error in the solution and can be calculated using [103]

$$\epsilon = \sum_{i=1}^{nnode} \left[\frac{|T^{n+1} - T^n|}{\Delta t} \right] \tag{5.3}$$

In this study, solutions were obtained from both using linear and quadratic element formulations on each mesh. For each formulation, three different methods were tested: LCG (explicit), LCG (implicit), and global Galerkin (explicit). In addition, for each of the three methods, computations were carried out for both consistent-mass and lumped-mass variations. This produced twelve different solution strategies for each mesh, giving an excellent examination of the performance of the proposed LCG schemes, with comparisons being made to the global Galerkin method.

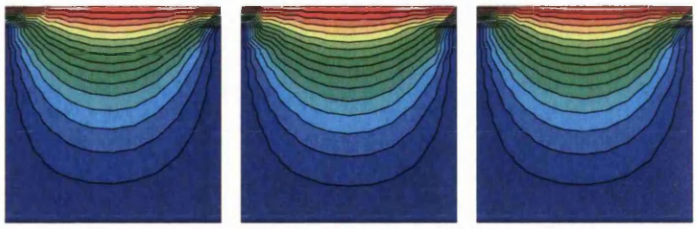


(a) Mesh A (b) Mesh B

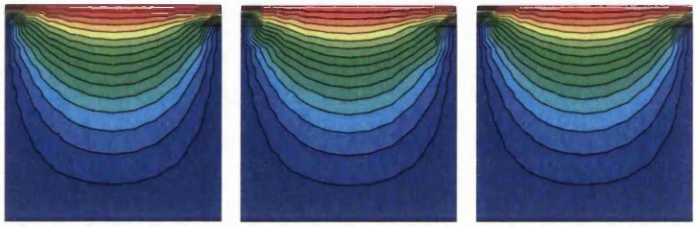
Figure 5.2: Steady-state heat conduction in a square plate. Details of the structured and unstructured meshes used

Results obtained from each method, on structured Mesh A, are shown in Figure 5.3 for linear elements, and in Figure 5.4 for quadratic elements. Figure 5.5 and Figure 5.6 give the computed solutions, using unstructured Mesh B, for linear and quadratic elements respectively. An initial glance at the computed results, shows that the methods are all generally in close agreement with each other. At this level of grid coarseness, an additional advantage of using quadratic elements, for all methods, is immediately seen. For each method, the temperature field obtained using quadratic elements is much smoother than the temperature field, obtained using same method, with linear elements.

Closer inspection of the results using linear elements on Mesh A (Figure 5.3) and Mesh B (Figure 5.5), show that the best solutions were given by the lumped mass versions of each scheme, with good symmetry along the mid-vertical centreline. This is confirmed by examining the graphs shown in Figure 5.7. Here, for each method, the temperatures - computed along the mid-horizontal and mid-vertical centre-lines of each mesh, have been plotted against the exact solution given by Equation (5.2). All lumped-mass methods show excellent accuracy for linear elements. The graphs of temperature along the mid-horizontal line, illustrate the high symmetry of the solutions computed by the lumped-mass methods. It was found in the calculations, that the performance of each method was reduced when a

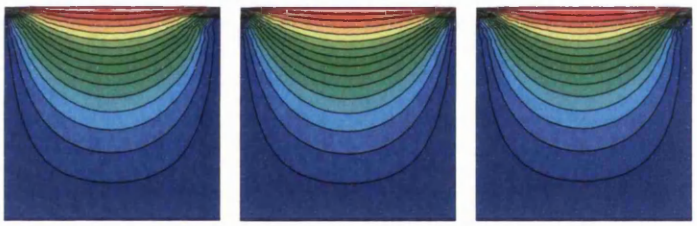


(a) Explicit LCG (consistent mass) (b) Implicit LCG (consistent mass) (c) Explicit Galerkin (consistent mass)

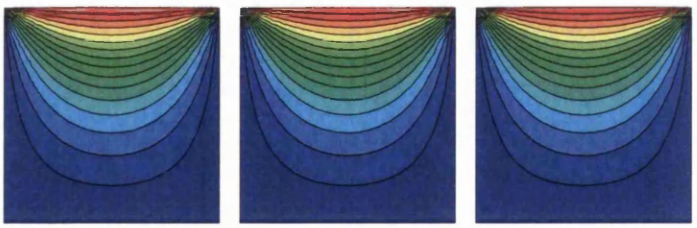


(d) Explicit LCG (lumped mass) (e) Implicit LCG (lumped mass) (f) Explicit Galerkin (lumped mass)

Figure 5.3: Steady-state heat conduction in a square plate. Temperature contours obtained for each scheme using Mesh A and linear elements.

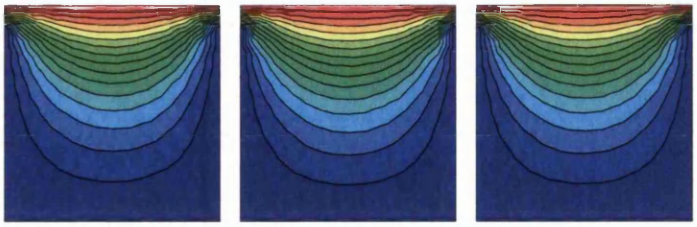


(a) Explicit LCG (consistent mass) (b) Implicit LCG (consistent mass) (c) Explicit Galerkin (consistent mass)

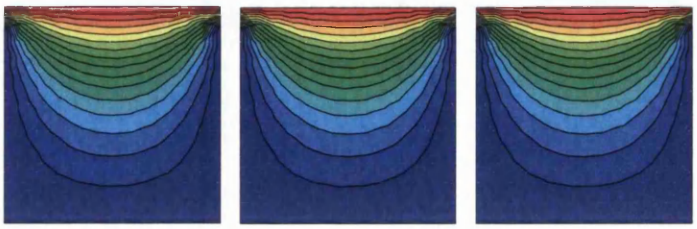


(d) Explicit LCG (lumped mass) (e) Implicit LCG (lumped mass) (f) Explicit Galerkin (lumped mass)

Figure 5.4: Steady-state heat conduction in a square plate. Temperature contours obtained for each scheme using Mesh A and quadratic elements.

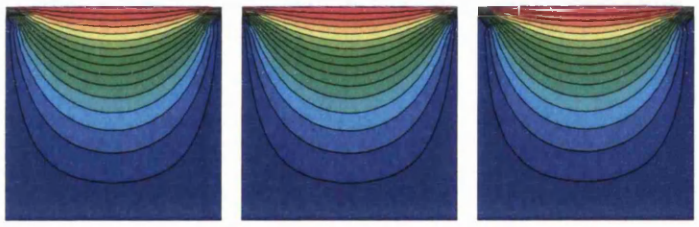


(a) Explicit LCG (consistent mass) (b) Implicit LCG (consistent mass) (c) Explicit Galerkin (consistent mass)

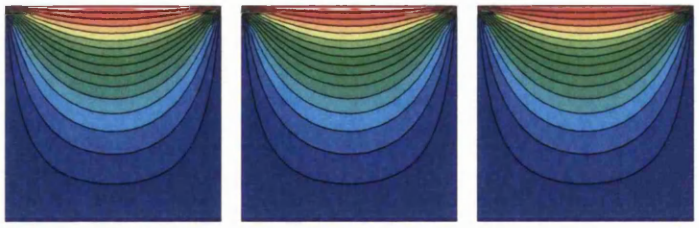


(d) Explicit LCG (lumped mass) (e) Implicit LCG (lumped mass) (f) Explicit Galerkin (lumped mass)

Figure 5.5: Steady-state heat conduction in a square plate. Temperature contours obtained for each scheme using Mesh B and linear elements.

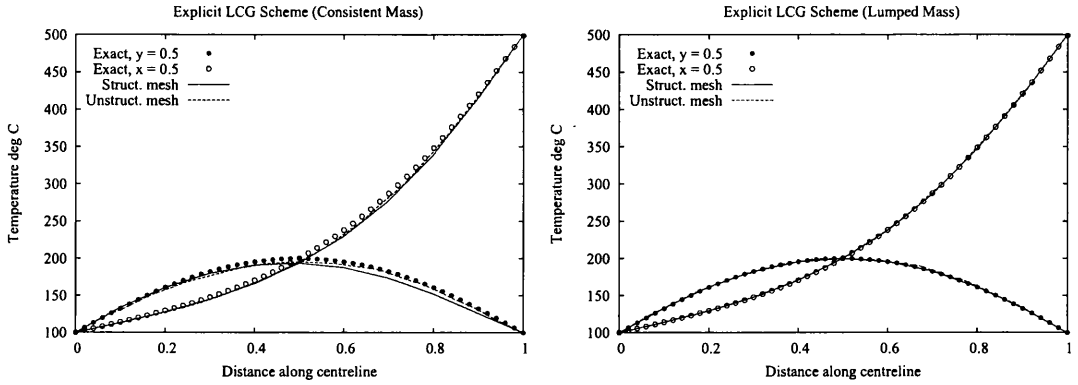


(a) Explicit LCG (consistent mass) (b) Implicit LCG (consistent mass) (c) Explicit Galerkin (consistent mass)



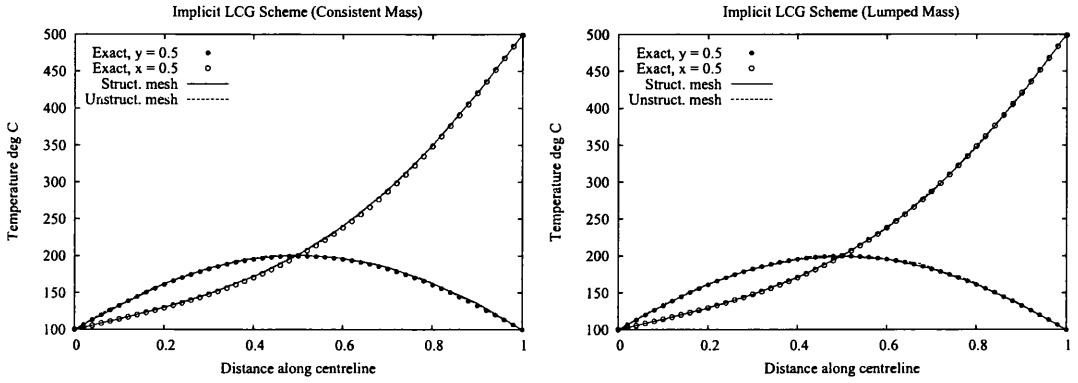
(d) Explicit LCG (lumped mass) (e) Implicit LCG (lumped mass) (f) Explicit Galerkin (lumped mass)

Figure 5.6: Steady-state heat conduction in a square plate. Temperature contours obtained for each scheme using Mesh B and quadratic elements.



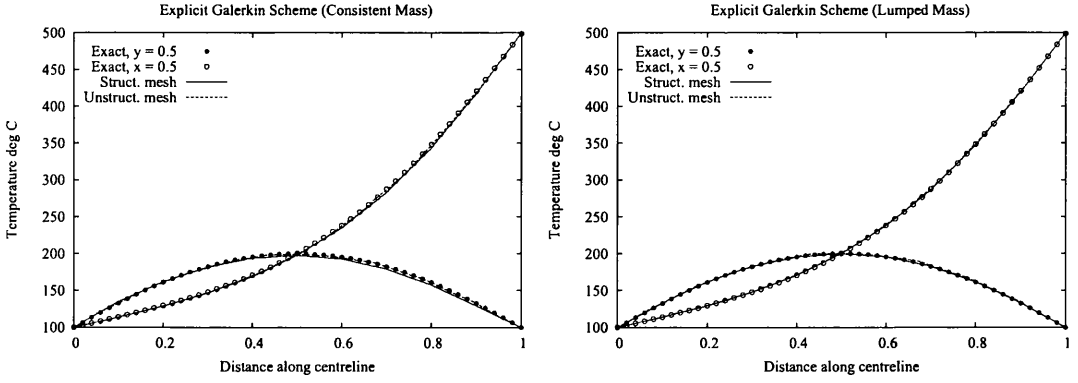
(a) Explicit LCG (consistent mass)

(b) Explicit LCG (lumped mass)



(c) Implicit LCG (consistent mass)

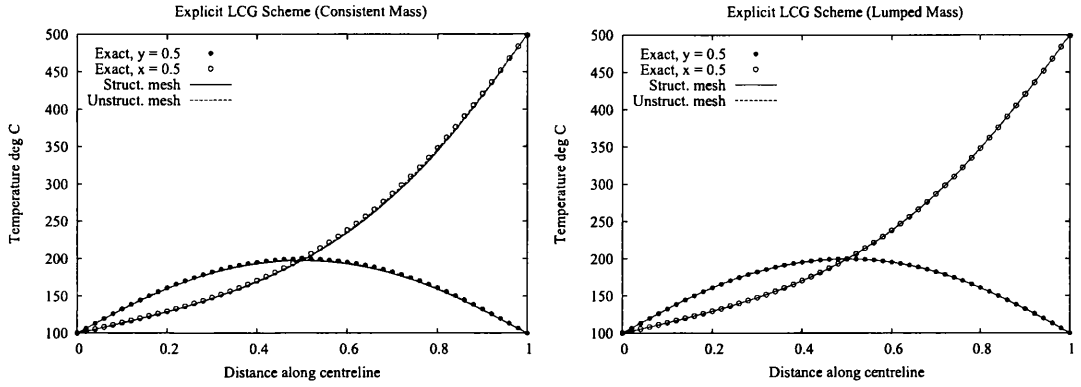
(d) Implicit LCG (lumped mass)



(e) Explicit Galerkin (consistent mass)

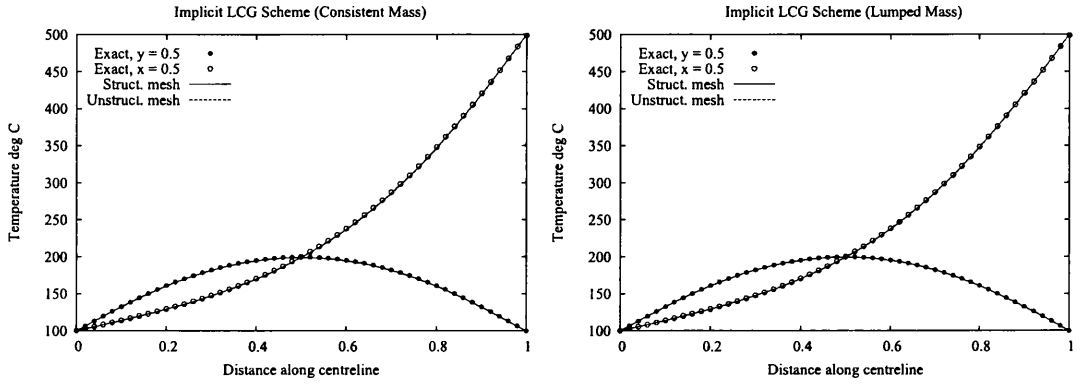
(f) Explicit Galerkin (lumped mass)

Figure 5.7: Steady-state heat conduction in a square plate. Comparison of temperature along centre-lines with exact solution for meshes A and B, using linear elements



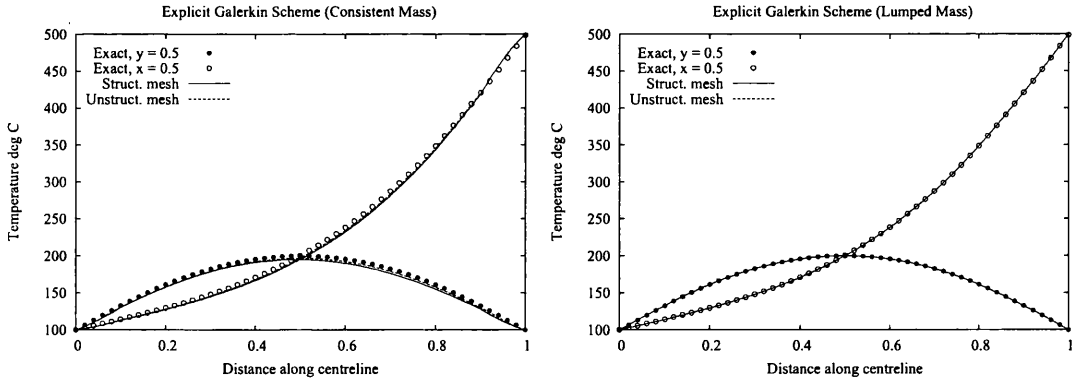
(a) Explicit LCG (consistent mass)

(b) Explicit LCG (lumped mass)



(c) Implicit LCG (consistent mass)

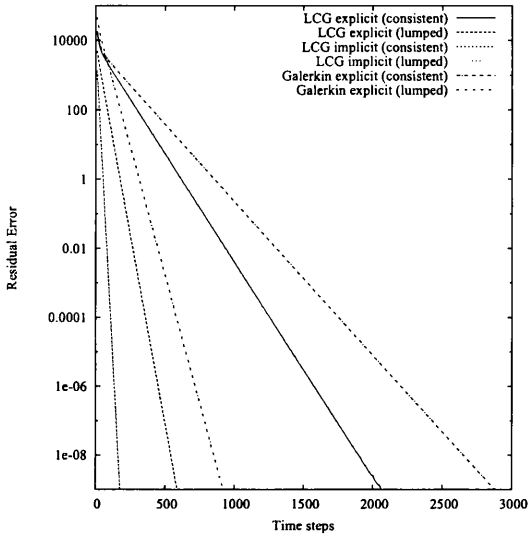
(d) Implicit LCG (lumped mass)



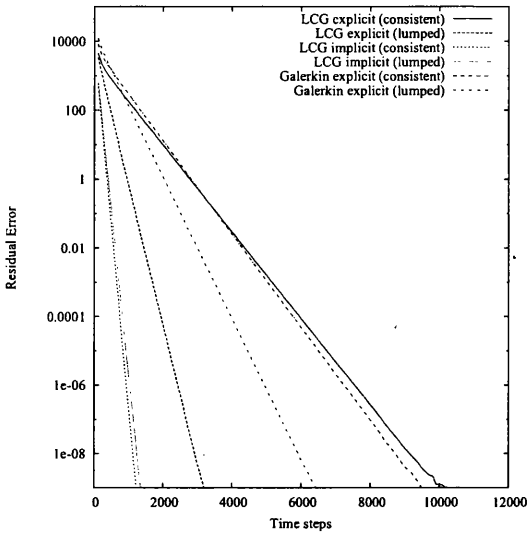
(e) Explicit Galerkin (consistent mass)

(f) Explicit Galerkin (lumped mass)

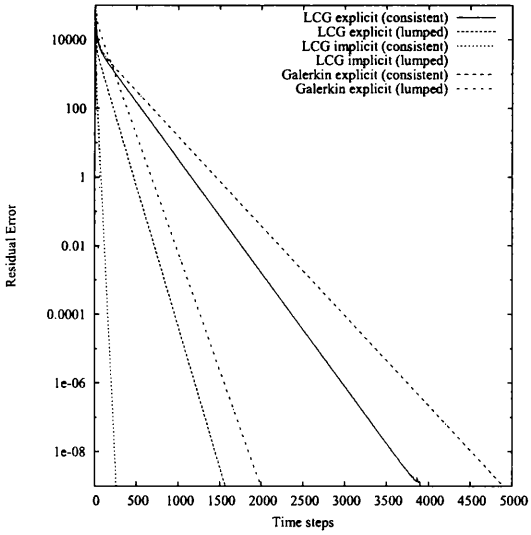
Figure 5.8: Steady-state heat conduction in a square plate. Comparison of temperature along centre-lines with exact solution for meshes A and B, using quadratic elements



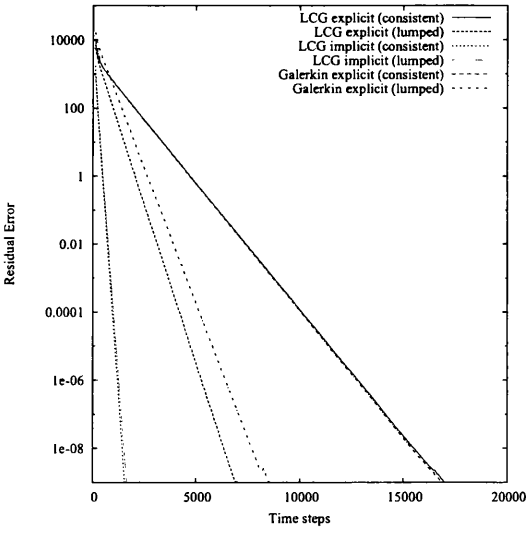
(a) Mesh A, linear elements



(b) Mesh A, quadratic elements



(c) Mesh B, linear elements



(d) Mesh B, quadratic elements

Figure 5.9: Steady-state heat conduction in a square plate. Convergence history of all schemes to a residual error of 1×10^{-09}

Table 5.1: Steady-state heat conduction in a square plate. Comparison of temperature calculated at the centre of the plate

Method (Mesh A)	Linear	Quadratic
Explicit LCG (consistent mass)	193.80°C	202.36°C
Explicit LCG (lumped mass)	200.00°C	200.00°C
Implicit LCG (consistent mass)	201.59°C	201.07°C
Implicit LCG (lumped mass)	201.69°C	200.74°C
Explicit global Galerkin (consistent mass)	197.77°C	204.98°C
Explicit global Galerkin (lumped mass)	200.00°C	200.00°C
Method (Mesh B)	Linear	Quadratic
Explicit LCG (consistent mass)	195.44°C	201.42°C
Explicit LCG (lumped mass)	200.28°C	200.03°C
Implicit LCG (consistent mass)	201.26°C	200.75°C
Implicit LCG (lumped mass)	201.28°C	200.49°C
Explicit global Galerkin (consistent mass)	199.29°C	203.42°C
Explicit global Galerkin (lumped mass)	200.65°C	199.51°C

Table 5.2: Steady-state heat conduction in a square plate. Comparison of CPU times for preprocessing and iterations

Method (Mesh A)	Linear	Quadratic
Explicit LCG (consistent mass)	0.64s	5.75s
Explicit LCG (lumped mass)	0.19s	1.70s
Implicit LCG (consistent mass)	0.11s	0.78s
Implicit LCG (lumped mass)	0.10s	0.81s
Explicit global Galerkin (consistent mass)	1.33s	51.20s
Explicit global Galerkin (lumped mass)	0.13s	1.30s
Method (Mesh B)	Linear	Quadratic
Explicit LCG (consistent mass)	1.17s	12.77s
Explicit LCG (lumped mass)	0.71s	4.81s
Implicit LCG (consistent mass)	0.17s	1.22s
Implicit LCG (lumped mass)	0.14s	1.28s
Explicit global Galerkin (consistent mass)	2.92s	154.67s
Explicit global Galerkin (lumped mass)	0.23s	1.91s

consistent mass matrix was used in the calculations. The consistent-mass results were not as accurate and symmetrical as their corresponding lumped-mass counterparts. Figure 5.7 shows that the implicit LCG method gave the most accurate and symmetric solution, out of the three methods, when a consistent mass was used with linear elements.

The solutions obtained for each method, using quadratic elements, are given in Figures 5.4 and 5.6, for meshes A and B respectively. Additionally, the temperatures computed along the mid-horizontal and mid-vertical centre-lines of each mesh have been plotted against the exact solution, and are shown in Figure 5.8. It can be seen quite clearly, that all lumped-mass versions of each method, all possess high symmetry, and are all in excellent agreement with the exact solution on both Meshes. For quadratic elements the explicit global Galerkin method, with consistent-mass, was found to be the least accurate on both meshes. On Mesh A, the contours produced by this version are unsymmetrical. Non-symmetry is also seen in the solutions obtained by using the consistent-mass version of the explicit LCG method, but to a much lesser extent. Here, there is a strong improvement in accuracy and symmetry of results obtained, when quadratic elements are used, compared to the results obtained with linear elements. The results obtained from using the consistent-mass version of the implicit LCG method are again the most accurate of all the consistent-mass versions, with a good symmetry along the mid-vertical centreline.

The temperature at the centre of the plate, is given by Equation (5.2) as 200.00°C. For convenience, Table (5.1) gives the computed temperatures obtained from the different versions, at the centre of the two meshes. For unstructured Mesh B, the recorded temperatures have all been interpolated. Table (5.2) gives the relative CPU times to converge to a residual error of 1×10^{-09}. This time includes pre-processing and iterations. CPU times are given for all six variations on both meshes A and B, using both linear and quadratic elements.

There are a number of fast iterative techniques available, for inverting large matrices [142]. However, in this work simple LU decomposition was used for both the implicit methods, and also for the consistent-mass version of the explicit global Galerkin method. One advantage of using the consistent form of the explicit LCG method, instead of the global Galerkin version, is that the LHS mass matrix can be inverted by hand and written

into the source code. This is reflected in much lower CPU costs, as shown in Table (5.2). As expected, both of the implicit LCG methods were the fastest. Using these methods, steady-state results were given in under a tenth of a second for both meshes with linear elements, and at around a second for both meshes with quadratic elements. The lumped-mass version of the explicit global Galerkin was relatively faster than the explicit LCG method with a lumped-mass. This is expected, as the LCG method solves the system of equations element by element and a flux calculation step is required at each time-step in the LCG method.

Figure 5.9 gives the complete convergence histories, to a residual error of 1×10^{-09}, for both lumped- and consistent-mass versions of each method. For Mesh A, the convergence histories are given in Figure 5.9(a), for linear elements, and in Figure 5.9(b), for quadratic elements. Figures 5.9(c) and 5.9(d) give the convergence history on Mesh B, for linear and quadratic elements respectively. The procedure adopted during the calculations, was to start with a large value for the time-step, and reduce incrementally until the method started converging. For each method the maximum allowable time-step for convergence was used.

During the computations, it was found, from using this procedure, that the lumped-mass explicit LCG method allowed a larger value of time-step, than the equivalent explicit global Galerkin method with lumped-mass. This was found to be true on both meshes, and for both linear and quadratic elements. On Mesh A, the largest time-step values for the explicit lumped-mass LCG method were 2.5×10^{-03} and 4.7×10^{-04} for linear and quadratic elements respectively. Whilst on the same mesh, the largest time-step values for the explicit lumped-mass global Galerkin method were 2.8×10^{-04} for linear elements, and 1.0×10^{-04} for quadratic elements. On Mesh B the time-steps were 9.5×10^{-04} (linear) and 2.1×10^{-04} (quadratic) for the explicit lumped-mass LCG method; and 1.3×10^{-04} (linear) and 7.5×10^{-05} (quadratic) for the explicit lumped-mass global Galerkin method. The advantage of the explicit lumped-mass LCG having a larger allowable time-step is seen in Figures 5.6(a) to Figures 5.6(d), as the explicit LCG method with lumped-mass converges to the desired level of tolerance at a faster rate than the explicit global Galerkin method with lumped-mass.

The convergence rates of the lumped-mass and consistent-mass implicit LCG methods were, for both meshes, very fast, and at almost similar rates to each other for both meshes and element-types. For linear elements the consistent-mass explicit LCG method was faster than the consistent-mass global Galerkin method on both meshes. However, the latter method is slightly faster on each mesh, when quadratic elements are used. Using quadratic elements instead of linear element on each mesh increased the number of iterations required for convergence, for each method. This is expected, since a lower stable time-step limit is imposed on each method, for each mesh, when quadratic elements are used.

5.2.2 Three-dimensional steady state heat conduction

The direct extension of the LCG spatial-discretisation to three-dimensions, is simple and quite straight forward. This section illustrates the application of the LCG method in three dimensions, and validates the diffusive flux calculation for linear tetrahedral elements. Both explicit and implicit LCG methods are considered here, with both a lumped-mass and consistent-mass matrix.

The 3D diffusive problem considered in this section, is the steady-state heat conduction within a cube. The problem statement is given by Figure 5.10. This is a direct extension of the 2D heat conduction problem defined by Figure 5.1, in the last section. As seen, the geometry has been extended in the third-dimension by 1m. The top, bottom and side boundaries are subjected to isothermal boundary conditions as in the 2D problem. To preserve the two-dimensionality of the problem, the two extra surfaces - appearing at the front and back, are subjected to a no-heat flux condition. Two unstructured 3D meshes were generated for this study, and are shown in Figure 5.11. Mesh C is relatively coarser of the two meshes used, and contains 892 linear tetrahedral finite elements. Mesh D is much finer, with 66534 linear tetrahedral finite elements. Both meshes were created from the *flite* software, developed by Hassan, Morgan and Weatherill [143] at Swansea University, and made available within the School of Engineering.

For each mesh, computations were carried out using both the explicit LCG, and

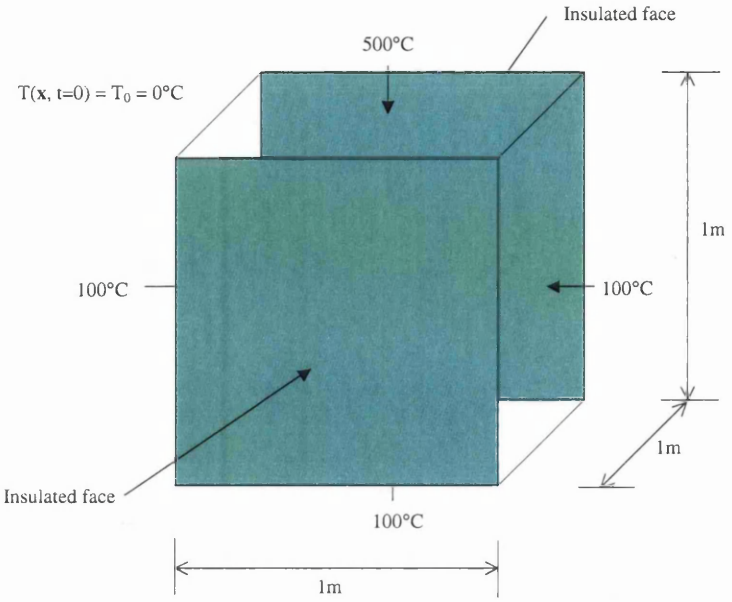
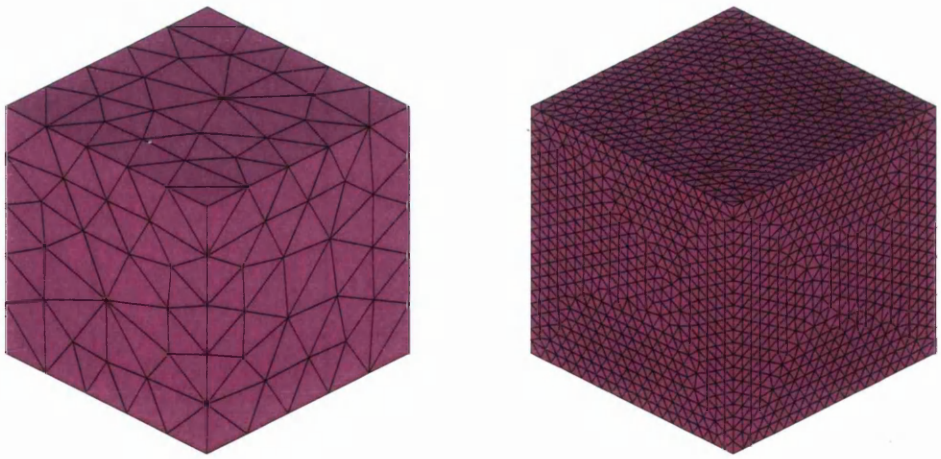
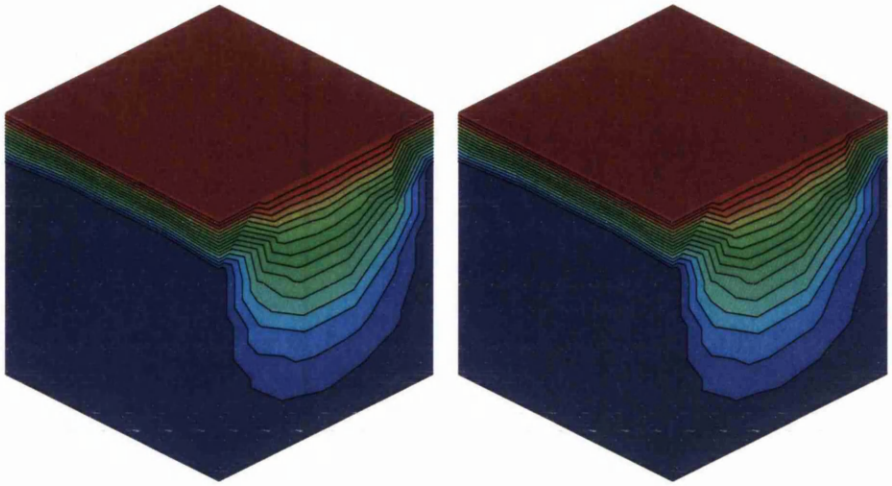


Figure 5.10: 3D Steady-state heat conduction in a square cube. Geometry and boundary conditions



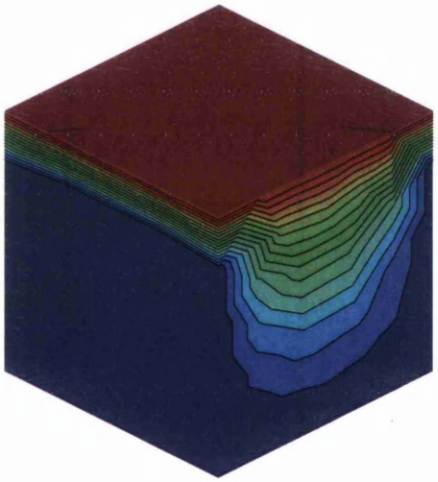
(a) Mesh C (b) Mesh D

Figure 5.11: 3D Steady-state heat conduction in a square cube. Unstructured meshes of linear tetrahedral finite element

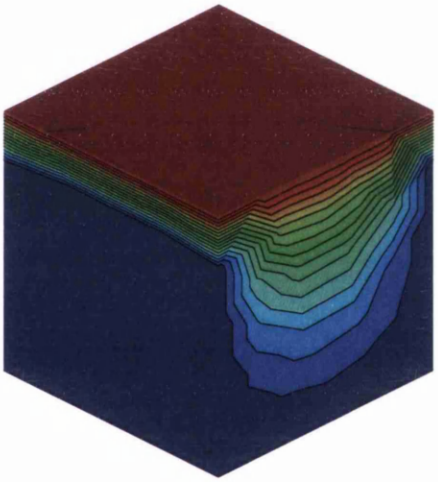


(a) Explicit LCG (consistent mass)

(b) Explicit LCG (lumped mass)

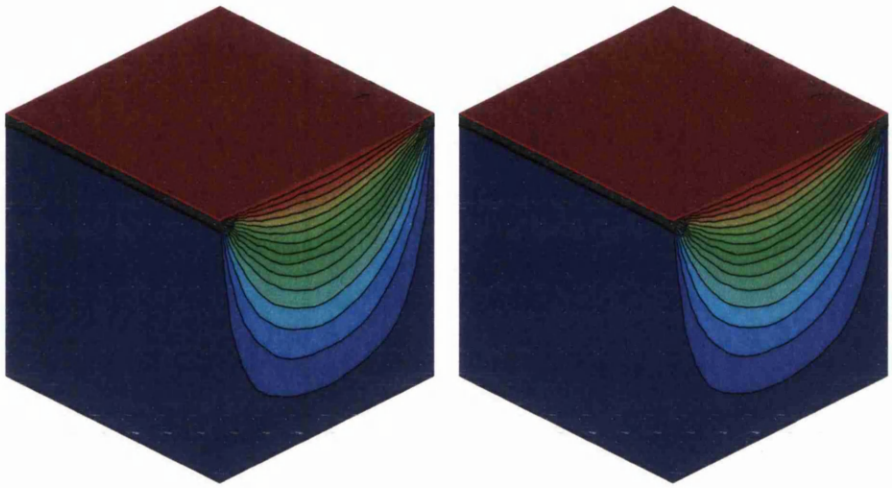


(c) Implicit LCG (consistent mass)



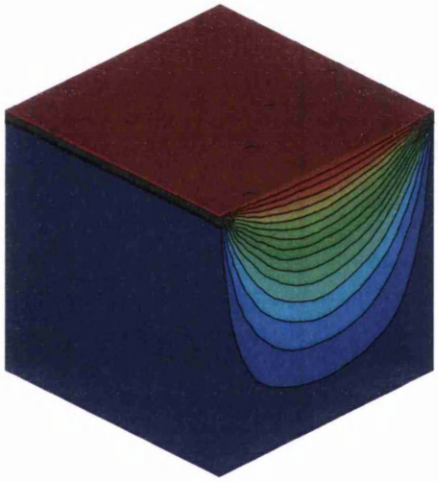
(d) Implicit LCG (lumped mass)

Figure 5.12: 3D Steady-state heat conduction in a square cube. Temperature contours obtained for each scheme using Mesh C.

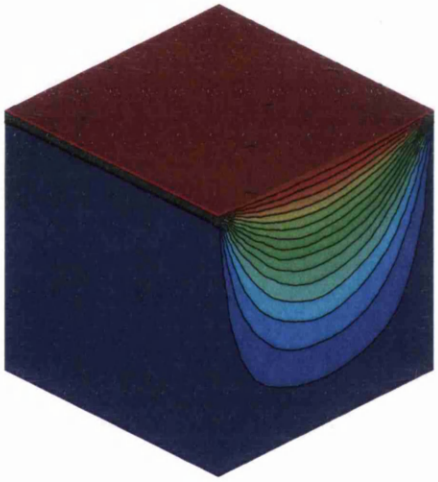


(a) Explicit LCG (consistent mass)

(b) Explicit LCG (lumped mass)

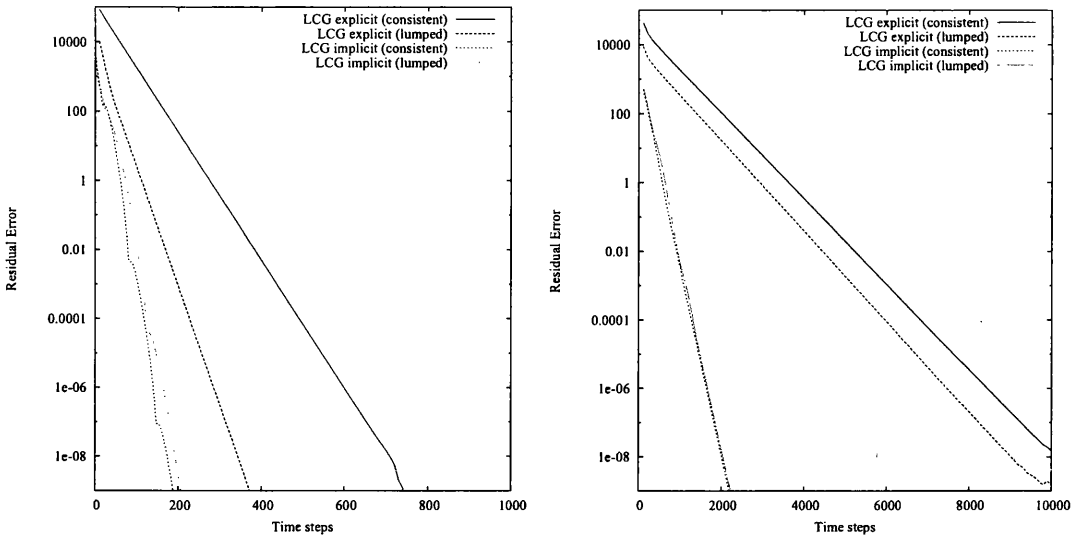


(c) Implicit LCG (consistent mass)



(d) Implicit LCG (lumped mass)

Figure 5.13: 3D Steady-state heat conduction in a square cube. Temperature contours obtained for each scheme using Mesh D.



(a) Mesh C

(b) Mesh D

Figure 5.14: 3D Steady-state heat conduction in a square cube. Convergence history to a residual error of 1×10^{-09}

Table 5.3: 3D Steady-state heat conduction in a square cube. Comparison of temperature calculated at the centre (0.5,0.5,0.5) of the cube

Method	Mesh C	Mesh D
Explicit LCG (consistent mass)	203.40°C	201.02°C
Explicit LCG (lumped mass)	200.35°C	199.89°C
Implicit LCG (consistent mass)	201.63°C	200.97°C
Implicit LCG (lumped mass)	201.04°C	200.63°C

Table 5.4: 3D Steady-state heat conduction in a square cube. Comparison of CPU times for preprocessing and iterations

Method	Mesh C	Mesh D
Explicit LCG (consistent mass)	0.98s	3047.83s
Explicit LCG (lumped mass)	0.50s	2200.81s
Implicit LCG (consistent mass)	0.30s	1639.84s
Implicit LCG (lumped mass)	0.30s	1624.84s

the implicit LCG methods. The effect of lumping was also considered for each method, and results are given using both a consistent and lumped mass matrix. Figure 5.12 gives the computed temperature distribution, for each scheme on Mesh C. As can be seen, all four solutions are in very good agreement with each other. Despite the coarseness of the mesh, the solutions obtained are physically meaningful, with the temperature distribution, in the plane perpendicular to the third-dimension, being the same as the 2D problem considered in the last section. This is more easily seen in Figure 5.13, where solutions, obtained for each method on Mesh D, are much smoother. The element-sizes in Mesh D are small enough not to influence the temperature contours significantly. It can be seen here, that there is little variation in temperature in the third dimension.

Table (5.3) gives the computed temperatures obtained from the different methods, interpolated at the geometric centre (0.5,0.5,0.5) of the two meshes. The explicit lumped-mass LCG scheme, was shown to be the most accurate method on both meshes. For Mesh C and Mesh D, accuracy was at 99.83% and 99.94% respectively.

The convergence histories obtained, for each scheme on the 3D meshes, are given in Figure 5.14(a) - for Mesh C, and Figure 5.14(b) - for Mesh D. Of all the methods, the lumped-mass and consistent-mass implicit LCG methods were the fastest, with very similar rates of convergence on each mesh. As with the 2D analysis of the last section, the maximum value of time-step allowed for convergence, was used in each scheme. The implicit lumped-mass LCG method allowed the highest maximum time-step, with values of 2.8×10^{-02} and 6.5×10^{-03} for Mesh C and Mesh D respectively. The scheme with the lowest allowable time-step was the consistent-mass explicit LCG scheme with values of 8.5×10^{-04} (Mesh

C) and 2.9×10^{-05} (Mesh D).

Table (5.4) gives the relative CPU times for preprocessing and iterations to a residual error of 1×10^{-09}. For the coarse mesh (Mesh C), solutions are obtained within a second of CPU time, even for the explicit LCG method with consistent mass. For Mesh D the CPU times are significantly longer. The quickest was the implicit lumped-mass LCG method with a CPU time of around 1624 seconds. The slowest was the consistent-mass explicit LCG scheme, with a CPU time of 3048 seconds. For this particular scheme, the error converged to 1×10^{-08} and *timed-out* at 12,000 iterations.

5.3 Diffusion with internal source terms

In this section, the analysis of solving diffusive problems is extended to include problems with internal sources. The analysis is limited to two-dimensional spaces. However, this is sufficient for illustrating the competency of the LCG schemes for solving diffusive problems when internal diffusive sources are present.

5.3.1 Two-dimensional steady-state heat conduction with heat source.

The final diffusion problem considered in this chapter, is heat conduction in a square plate. The problem definition is described in Figure 5.15. The domain is unit square, and the bottom and top extremities are insulated. The temperature, T, on the LHS is 0°C, and on the RHS, T is at 1°C. The thermal conductivity k, density ρ, and specific heat c_p are all assumed to be equal to unity in this problem. The domain is subjected to an internal heat source that produces negative heat generation within the plate's domain.

The governing equation for this problem, is of the type

$$\rho c_p \frac{\partial T}{\partial t} = -\frac{\partial F_i}{\partial x_i} + S_i \tag{5.4}$$

with flux

$$F_i = -k\frac{\partial T}{\partial x_i} \tag{5.5}$$

and source-term

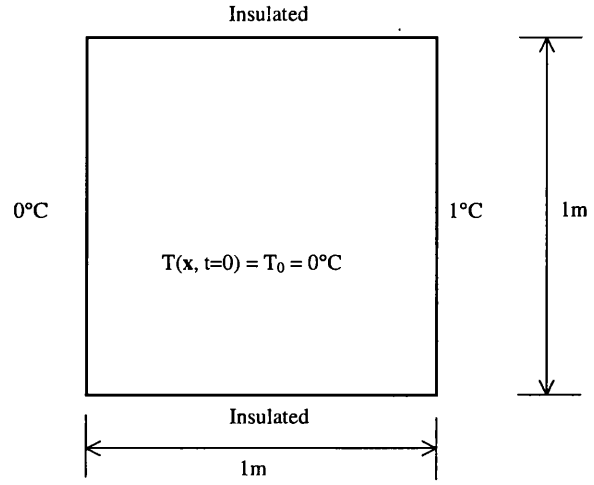


Figure 5.15: Steady-state heat conduction with heat source. Geometry and boundary conditions

$$S_i = -\delta_{1i} 12 k {x_i}^2 \tag{5.6}$$

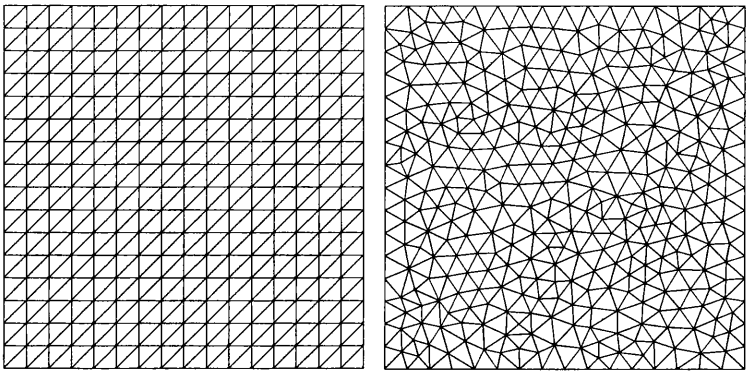
The Kronecker delta function, δ_{ij}, (equal to unity when $i = j$ and zero when $i \neq j$) is used in Equation (5.6), so that the source-term only acts in the x_1-direction. The inclusion of this internal heat source makes the temperature distribution along the x-direction non-linear. The exact solution to this problem is simply

$$T = {x_1}^4 \tag{5.7}$$

allowing a comparison of the accuracy of each scheme to be made.

Figure 5.16 shows the two meshes used for solving this problem. Mesh E is uniform in structure with 512 elements, and Mesh F is an unstructured grid of 674 elements. Initially, the plate temperature was 0°C, and a solution was found by time-stepping to steady-state conditions. For this problem steady-state conditions were assumed when the residual error ϵ, given by Equation (5.3), reached a tolerance of $\epsilon < 1 \times 10^{-10}$.

Both the explicit and implicit LCG schemes, and the explicit global Galerkin scheme, were investigated for this problem. As with previous examples in this chapter, the use of lumped- and consistent-masses were considered. Additionally, both linear and



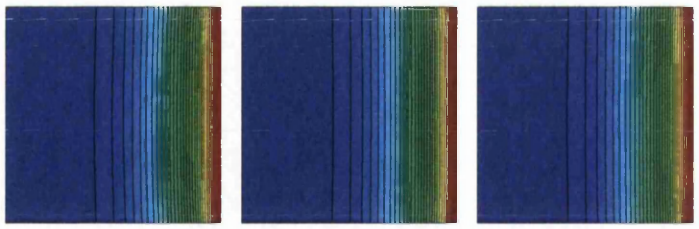
(a) Mesh E (b) Mesh F

Figure 5.16: Steady-state heat conduction with heat source. Triangular finite element meshes used

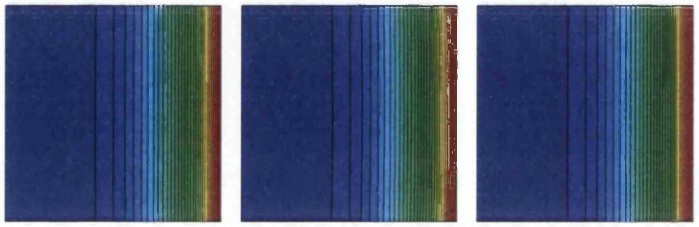
quadratic basis functions were used. This gave twelve solution methods to be considered on each mesh.

The computed temperatures for Mesh E, are shown in Figures 5.17 and 5.18, for linear and quadratic elements respectively. Figures 5.19 (linear) and 5.20 (quadratic) give the results on Mesh F. Additionally, Figures 5.21 (linear) and 5.22 (quadratic) give the computed temperatures, from both meshes, along the mid-horizontal and mid-vertical lines. As can be seen, the general agreement with the analytical solution is good, and all methods perform reasonably well on both meshes, using both linear and quadratic elements. The consistent mass versions of the explicit LCG and global Galerkin schemes, are a little over-diffusive when linear elements are used. However, the quality in solution is improved for these methods, when quadratic basis functions are used. The improvement in accuracy of these methods is more clearly seen, when examining Figures 5.21 and 5.22.

Figures 5.23(a) and 5.23(c) show the convergence history of the methods - using linear elements, on meshes E and F respectively. Figures 5.23(b) and 5.23(d) give the convergence histories on both meshes when quadratic elements are used. All graphs show a similar pattern. The fastest methods are again the implicit methods - both lumped and consistent versions. As for the linear diffusion problems considered in the last section, the maximum possible time-step for convergence was used. It was also found here, that the

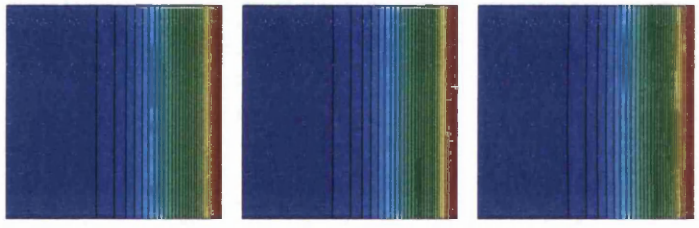


(a) Explicit LCG (consistent mass) (b) Implicit LCG (consistent mass) (c) Explicit Galerkin (consistent mass)

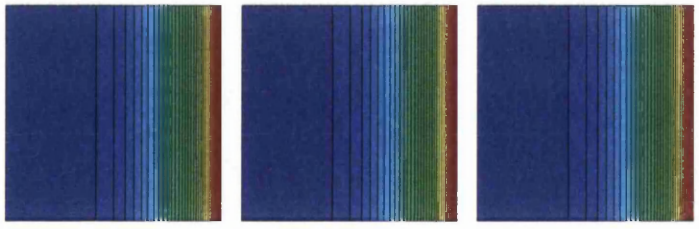


(d) Explicit LCG (lumped mass) (e) Implicit LCG (lumped mass) (f) Explicit Galerkin (lumped mass)

Figure 5.17: Steady-state heat conduction with heat source. Temperature contours obtained for each scheme using Mesh E and linear elements.

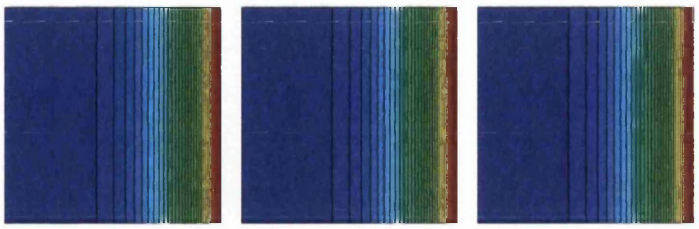


(a) Explicit LCG (consistent mass) (b) Implicit LCG (consistent mass) (c) Explicit Galerkin (consistent mass)

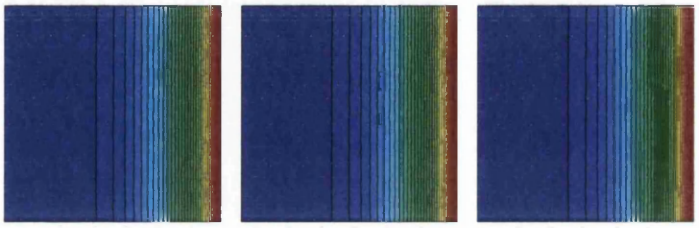


(d) Explicit LCG (lumped mass) (e) Implicit LCG (lumped mass) (f) Explicit Galerkin (lumped mass)

Figure 5.18: Steady-state heat conduction with heat source. Temperature contours obtained for each scheme using Mesh E and quadratic elements.

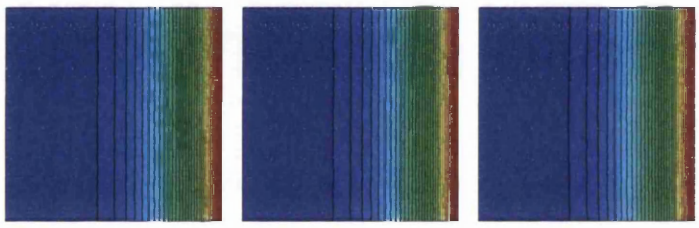


(a) Explicit LCG (consistent mass) (b) Implicit LCG (consistent mass) (c) Explicit Galerkin (consistent mass)

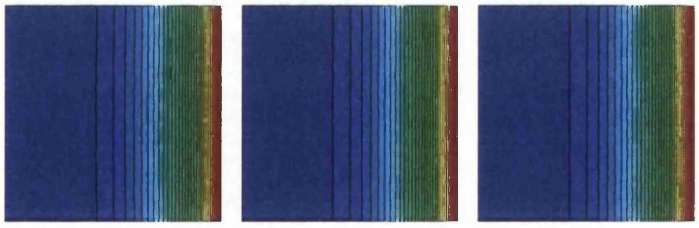


(d) Explicit LCG (lumped mass) (e) Implicit LCG (lumped mass) (f) Explicit Galerkin (lumped mass)

Figure 5.19: Steady-state heat conduction with heat source. Temperature contours obtained for each scheme using Mesh F and linear elements.

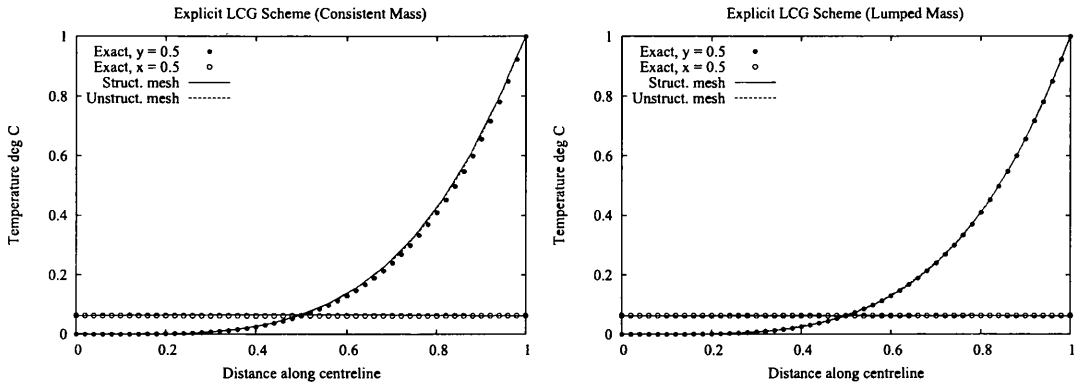


(a) Explicit LCG (consistent mass) (b) Implicit LCG (consistent mass) (c) Explicit Galerkin (consistent mass)



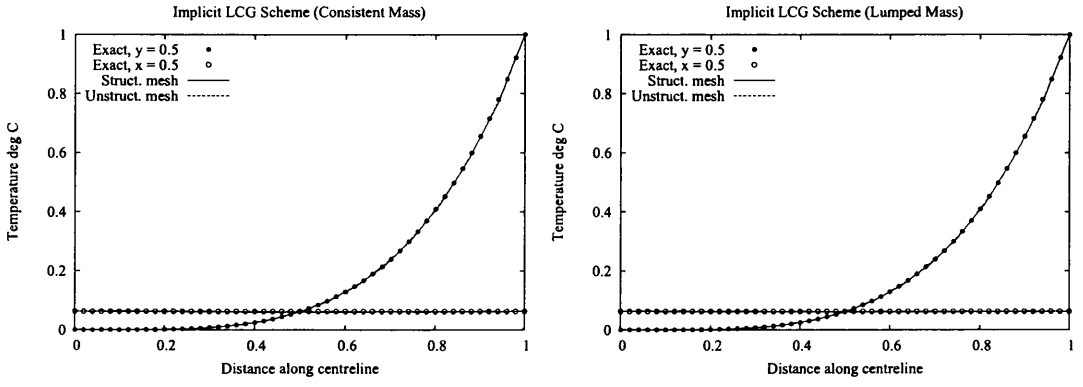
(d) Explicit LCG (lumped mass) (e) Implicit LCG (lumped mass) (f) Explicit Galerkin (lumped mass)

Figure 5.20: Steady-state heat conduction with heat source. Temperature contours obtained for each scheme using Mesh F and quadratic elements.



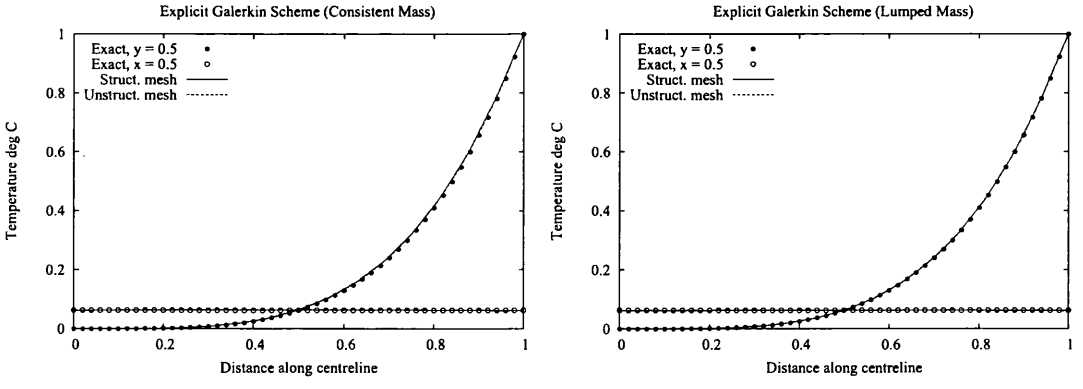
(a) Explicit LCG (consistent mass)

(b) Explicit LCG (lumped mass)



(c) Implicit LCG (consistent mass)

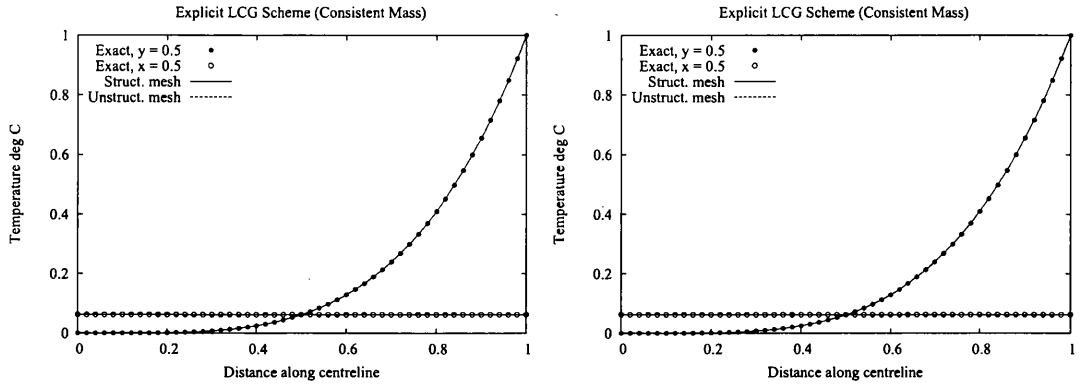
(d) Implicit LCG (lumped mass)



(e) Explicit Galerkin (consistent mass)

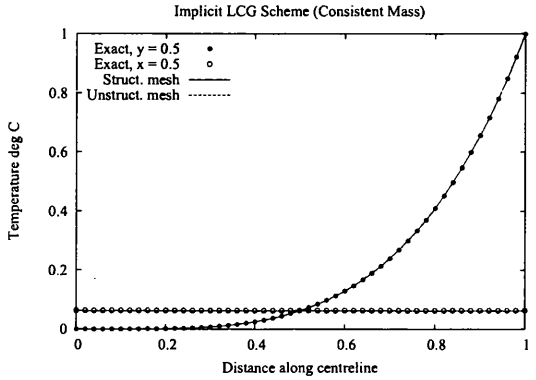
(f) Explicit Galerkin (lumped mass)

Figure 5.21: Steady-state heat conduction with heat source. Comparison of temperature along centre-lines with exact solution for meshes E and F, using linear elements

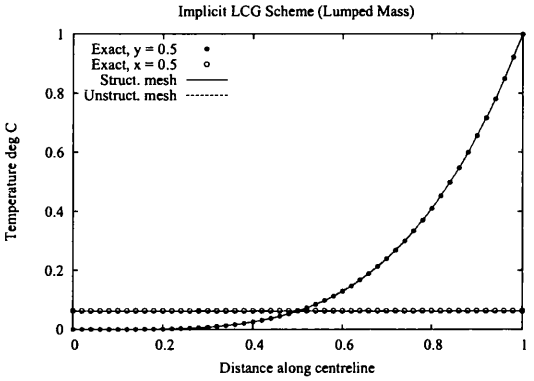


(a) Explicit LCG (consistent mass)

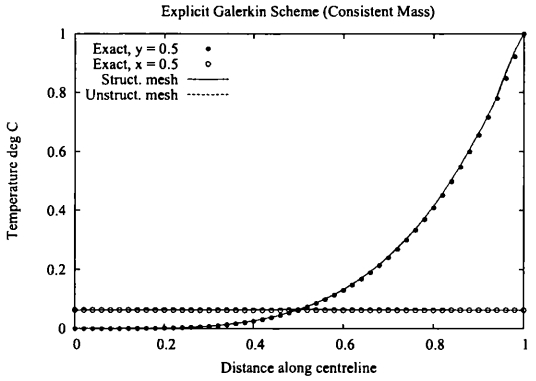
(b) Explicit LCG (lumped mass)



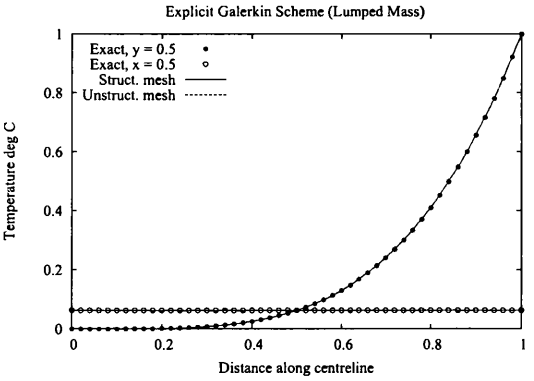
(c) Implicit LCG (consistent mass)



(d) Implicit LCG (lumped mass)

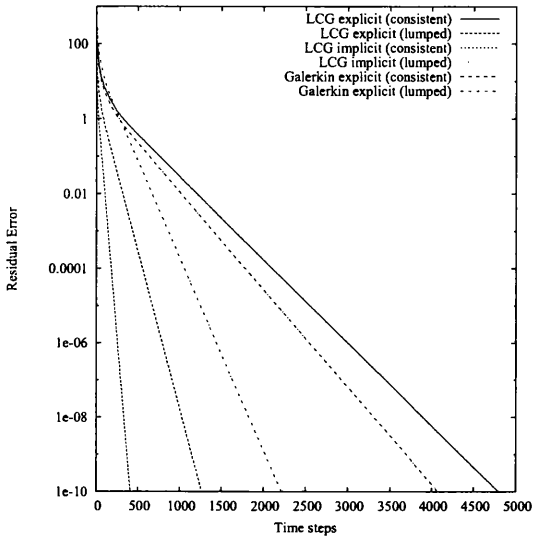


(e) Explicit Galerkin (consistent mass)

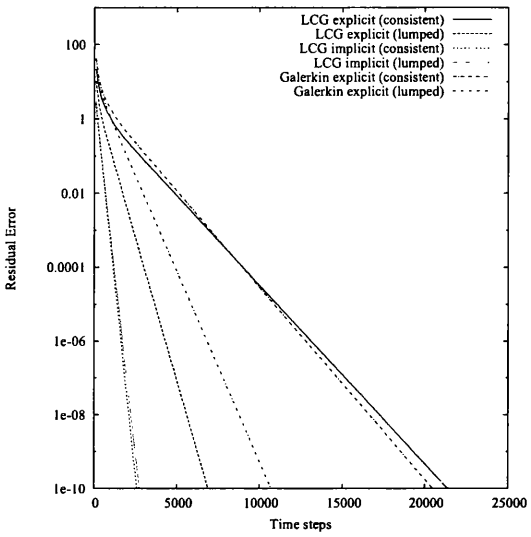


(f) Explicit Galerkin (lumped mass)

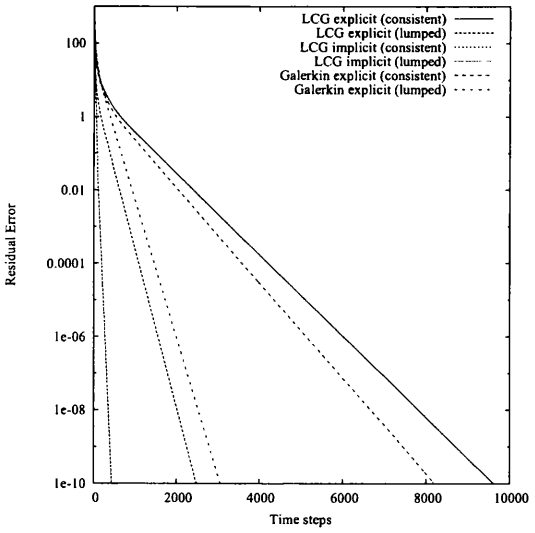
Figure 5.22: Steady-state heat conduction with heat source. Comparison of temperature along centre-lines with exact solution for meshes E and F, using quadratic elements



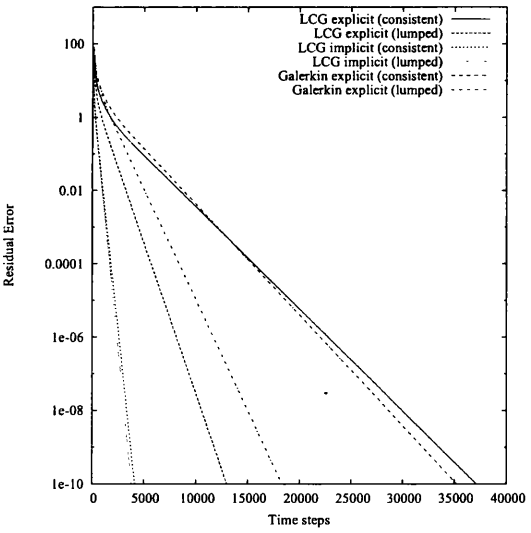
(a) Mesh E, linear elements



(b) Mesh E, quadratic elements



(c) Mesh F, linear elements



(d) Mesh F, quadratic elements

Figure 5.23: Steady-state heat conduction with heat source. Convergence history to a residual error of 1×10^{-10}

Table 5.5: 2D Steady-state heat conduction with heat source. Comparison of CPU times for preprocessing and iterations

Method (Mesh E)	Linear	Quadratic
Explicit LCG (consistent mass)	2.00s	30.63s
Explicit LCG (lumped mass)	0.50s	8.92s
Implicit LCG (consistent mass)	0.20s	3.89s
Implicit LCG (lumped mass)	0.20s	3.89s
Explicit global Galerkin (consistent mass)	5.78s	689.78s
Explicit global Galerkin (lumped mass)	0.33s	4.69s
Method (Mesh F)	Linear	Quadratic
Explicit LCG (consistent mass)	5.16s	100.95s
Explicit LCG (lumped mass)	1.22s	22.36s
Implicit LCG (consistent mass)	0.30s	8.06s
Implicit LCG (lumped mass)	0.28s	7.28s
Explicit global Galerkin (consistent mass)	27.80s	1977.34s
Explicit global Galerkin (lumped mass)	0.47s	10.99s

lumped mass version of the explicit LCG method allowed a higher value of time-step than the lumped mass version of the explicit global Galerkin method. The CPU times on each mesh, for all methods, using both linear and quadratic elements, are summarised in Table 5.5

5.4 Summary

To compare the performance of the LCG method with the standard global Galerkin method, a number of steady-state heat conduction problems, were considered in this chapter. Both implicit and explicit versions of the LCG scheme were tested, on both structured and unstructured grids. These examples gave an excellent validation of the diffusive portion of the numerical flux and LCG methods. It is the flux which allows an element by element solution of the discrete equations, and so it was important that these examples were carried out in-order to provide numerical evidence of its validity.

The application to three space-dimensions was illustrated in this chapter, by solv-

ing a steady-state heat conduction problem in 3D. Linear tetrahedral-elements were used. Using the example given, it was shown that both the diffusive-flux calculation and averaging procedure, were both valid in 3D. This was a key investigation, as it sets the foundations for further study of the LCG method in 3D, particularly in solving 3D fluid dynamics problems.

In the final sections of this chapter, it was shown that the LCG method could be applied to solve diffusive-problems, that included internal source terms in the governing equations. Excellent results were obtained for both explicit and implicit LCG methods. The proceeding chapter continues the analysis of the LCG method for scalar variables, by considering problems of convection-diffusion.

Chapter 6

The LCG Method for Convection-Diffusion Problems

6.1 Introduction

In Chapter 5, a validation of the proposed explicit and implicit LCG methods (discussed in Chapter 3) for purely diffusive problems, was presented for both linear and quadratic elements. Solutions were computed on both structured and unstructured grids, with results given for both consistent-mass and lumped-mass matrix alternatives. The nature of the problems studied allowed the chapter to solely focus on the validation of the diffusive edge-flux approximation and to investigate the benefits of using non-linear elements. In the last chapter for diffusion problems, it was shown that using quadratic-based shape functions instead of linear, increased the solution quality and accuracy on the same grid. Although good results were produced when using linear elements, accuracy in the calculation of edge-fluxes is improved by using high-order elements. This is because the gradients of the scalar variable, which form the edge-fluxes, are computable at the edge-nodes - a process which is not possible when using linear elements [1], where the gradient is constant throughout the element.

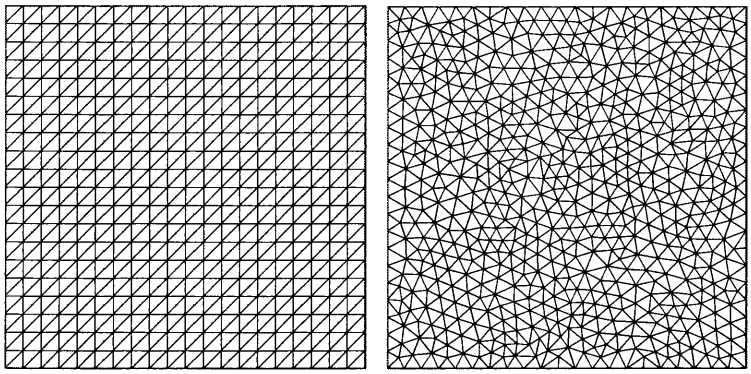
This chapter continues the validation of the LCG formulation with the extension being made to general convection-diffusion problems. To overcome the difficulties encoun-

tered in convection-diffusion problems when convective terms dominate, suitable stabilisation methods are incorporated into the LCG formulation. Proposed convection-stabilised LCG formulations were discussed in Chapter 3, where details were given on incorporating both the Streamline-Upwind Petrov-Galerkin (SUPG) method [28] and the *simple explicit* characteristic Galerkin (CG) [39] method. Although reference [1] presented a simple stabilised Petrov-Galerkin LCG method using linear elements, the method only applied convection-stabilisation in one-dimension.

This work presents the first examination of stabilised Petrov-Galerkin LCG methods in two dimensions - using the SUPG method, as well as the first application of the characteristic based stabilised LCG method. The latter method has the additional advantage of producing a stabilised LCG method that possesses second-order time accuracy. It was shown in Chapter 4, that a stream-wise influenced element-size gave better performance over a typical standard element-size calculation. The only drawback being that the need for updating is required every time-step, for a changing velocity field. For the simple convection-diffusion problems considered here, the velocity field is constant and element-sizes only need to be calculated once at the beginning of the program. Thus Equation (2.98) can be efficiently used here to optimise the SUPG stabilised LCG method.

This chapter validates both stabilised LCG methods, via a number of classic convection-diffusion problems. As with the previous chapter, results are presented for both explicit and implicit schemes, with and without a consistent mass matrix. In addition, results are given for both structured and unstructured grids, and comparisons are made with the standard global Galerkin method. Again, both linear and quadratic elements are employed in the examples. Before closing, this chapter gives a detailed investigation into the mesh convergence properties of each of the CG stabilised LCG methods.

A general analysis is given for a problem which includes source terms in the governing equation. Mesh convergence data is computed for employing both linear and quadratic basis functions.



(a) Mesh A (b) Mesh B

Figure 6.1: Stabilised LCG methods for convection-diffusion problems. Structured and unstructured triangular finite element meshes used.

6.2 Problems involving convection-diffusion transport

A number of classic convection-diffusion problems are considered in this section to examine the stabilised LCG schemes discussed in Chapter 3. The meshes used are shown in Figure 6.1. Mesh A is a structured mesh with 800 elements and Mesh B is an unstructured mesh with 1228 elements. In all the examples considered in this section, a steady-state solution is assumed when the residual error ϵ, reaches a tolerance of $\epsilon < 1 \times 10^{-09}$. The residual error was computed from [103] as

$$\epsilon = \sum_{i=1}^{nnode} \left[\frac{|\phi^{n+1} - \phi^{n}|}{\Delta t} \right] \tag{6.1}$$

6.2.1 Simple convection-diffusion transport in a square channel

The first example considered in this section is defined in Figure 6.2. It is a classic problem of simple convection-diffusion in a square channel, with Dirichlet-type boundary conditions applied at the inlet and exit. Here, the value of ϕ at the inlet is assumed to be zero, and ϕ is taken as equal to unity at the exit. No-flux boundary conditions are prescribed at the top and bottom channel walls. The convection velocity is in the x_1 direction and is assumed to be constant throughout the domain. The initial value of ϕ inside the domain

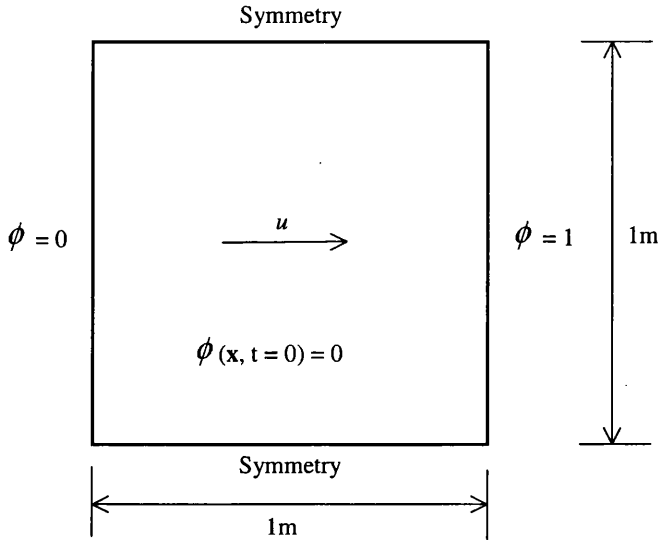


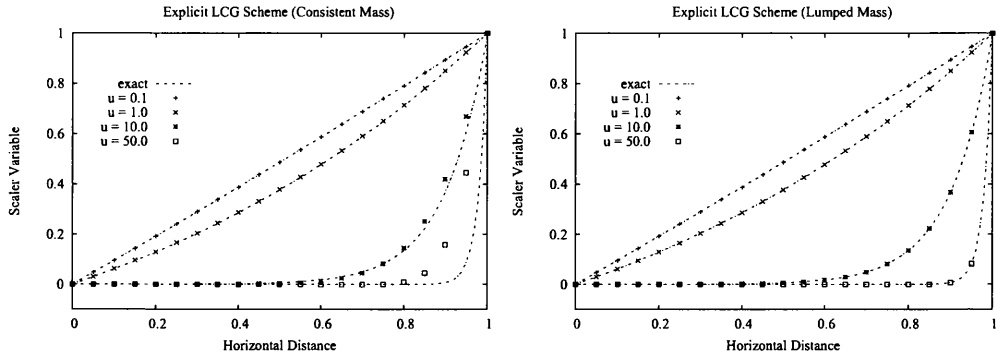
Figure 6.2: Convection-diffusion problem in a square domain. Problem geometry and boundary conditions

is zero everywhere, and the steady-state solution is obtained through time-stepping to the prescribed residual error tolerance. The solution at x_1, along any constant value of x_2, can be compared with the exact analytical one-dimensional solution, given by [28, 41] as:

$$\phi = \frac{1 - e^{\frac{u_1 x_1}{k}}}{1 - e^{\frac{u_1 L}{k}}} \tag{6.2}$$

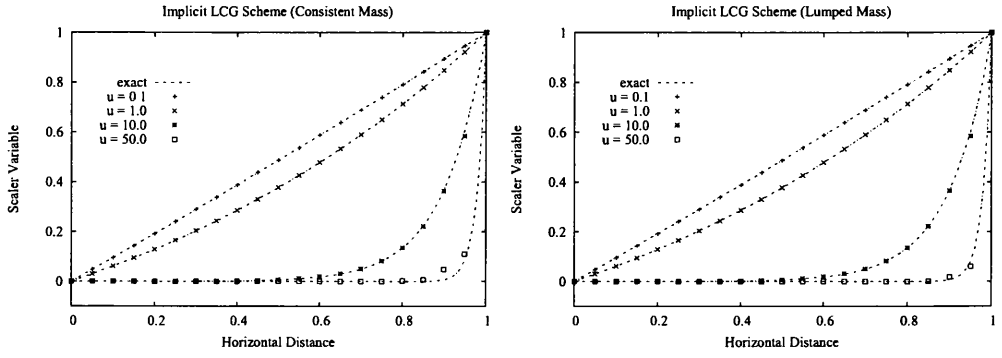
here, L represents the total length of the domain in the x_1 direction. For this problem, the SUPG stabilised LCG methods were used. Both explicit and implicit forms were tested against the explicit standard global Galerkin, and results were given for lumped and consistent mass matrices - using linear and quadratic elements, on each mesh.

The results for Mesh A are shown in Figures 6.3 and 6.4, for linear and quadratic elements respectively. The results for Mesh B are given in Figure 6.5 for linear elements and in Figure 6.6 for quadratic elements. In all figures, the computed *nodal* distribution of ϕ is plotted against the exact solution, given by Equation (6.2), for a number of different convection velocities including: $u = 0.1$, 1, 10, and 50. For the case of examining the nodal results on the unstructured grid Mesh B, the nodal solution is plotted for all nodes along x_1 that lie in the range $0.45 \leq x_2 \leq 0.55$.



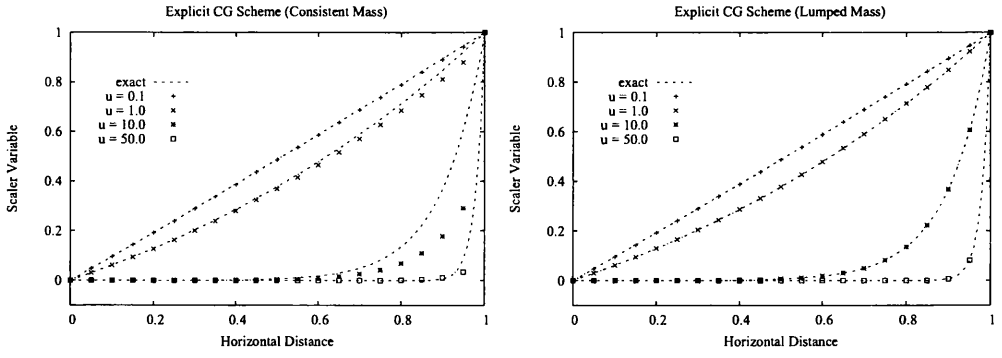
(a) Explicit LCG (consistent mass)

(b) Explicit LCG (lumped mass)



(c) Implicit LCG (consistent mass)

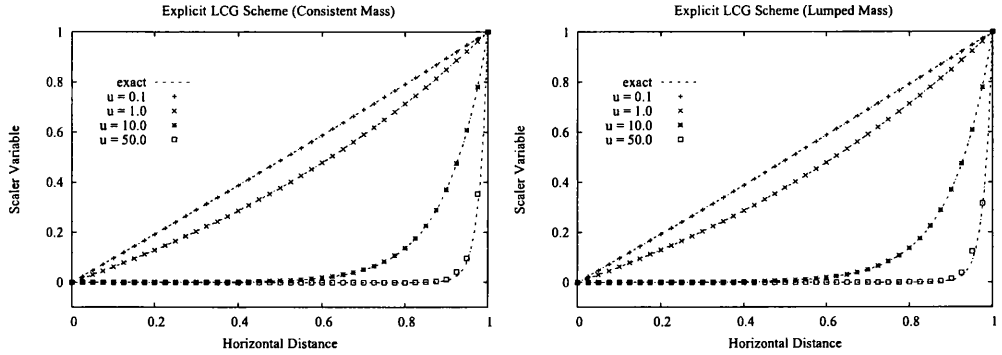
(d) Implicit LCG (lumped mass)



(e) Explicit Galerkin (consistent mass)

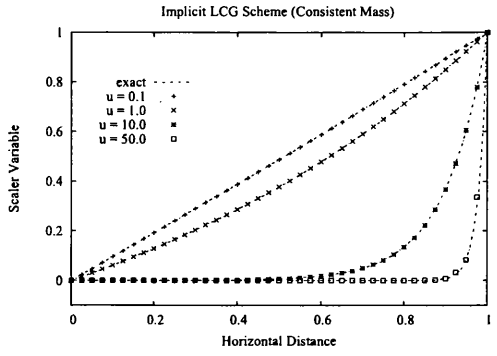
(f) Explicit Galerkin (lumped mass)

Figure 6.3: Convection-diffusion problem in a square channel. Nodal solutions of the scalar variable distribution, along the mid-horizontal line ($y = 0.5$) of Mesh A. Results given for SUPG stabilised LCG and global Galerkin methods, using linear basis functions.

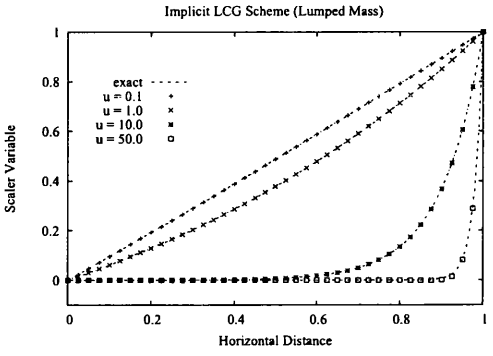


(a) Explicit LCG (consistent mass)

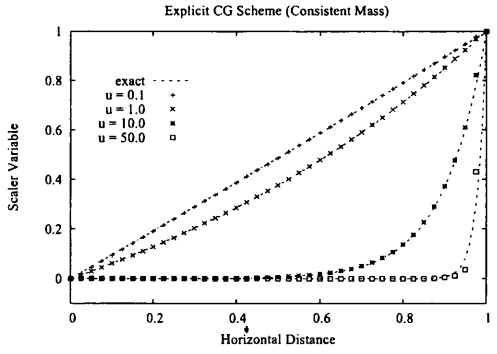
(b) Explicit LCG (lumped mass)



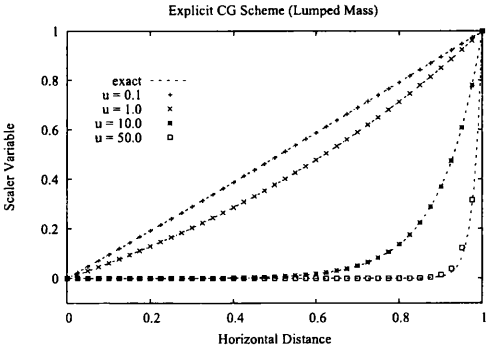
(c) Implicit LCG (consistent mass)



(d) Implicit LCG (lumped mass)

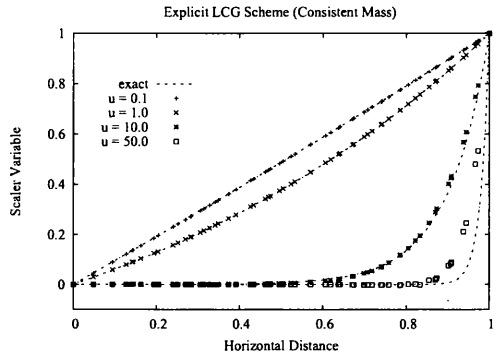


(e) Explicit Galerkin (consistent mass)

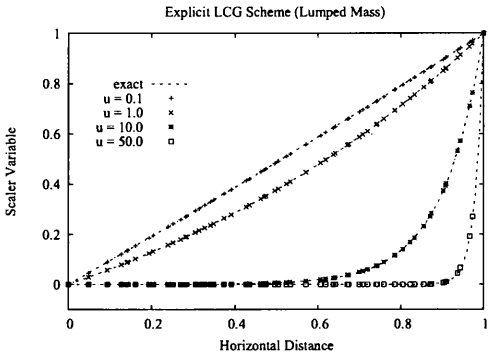


(f) Explicit Galerkin (lumped mass)

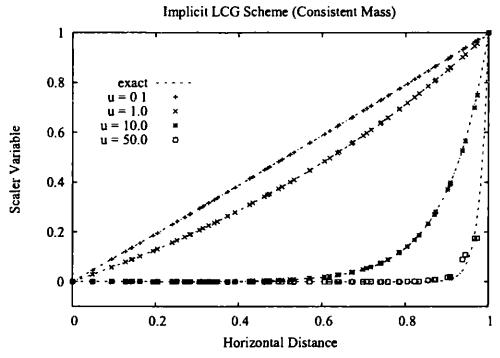
Figure 6.4: Convection-diffusion problem in a square channel. Nodal solutions of the scalar variable distribution, along the mid-horizontal line ($y = 0.5$) of Mesh A. Results given for SUPG stabilised LCG and global Galerkin methods, using quadratic basis functions.



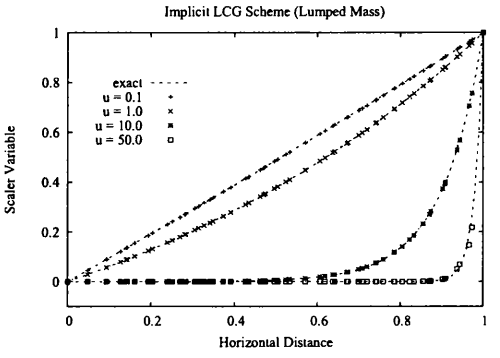
(a) Explicit LCG (consistent mass)



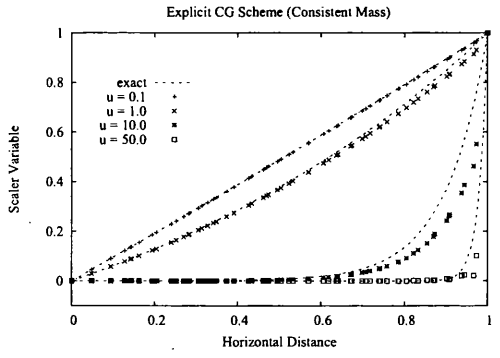
(b) Explicit LCG (lumped mass)



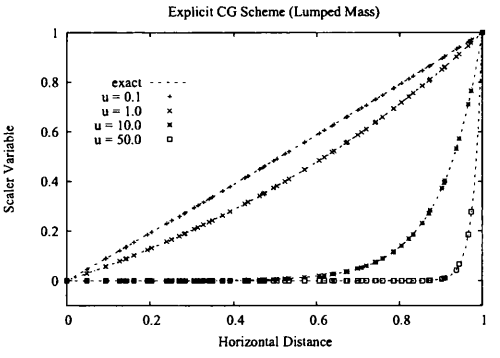
(c) Implicit LCG (consistent mass)



(d) Implicit LCG (lumped mass)

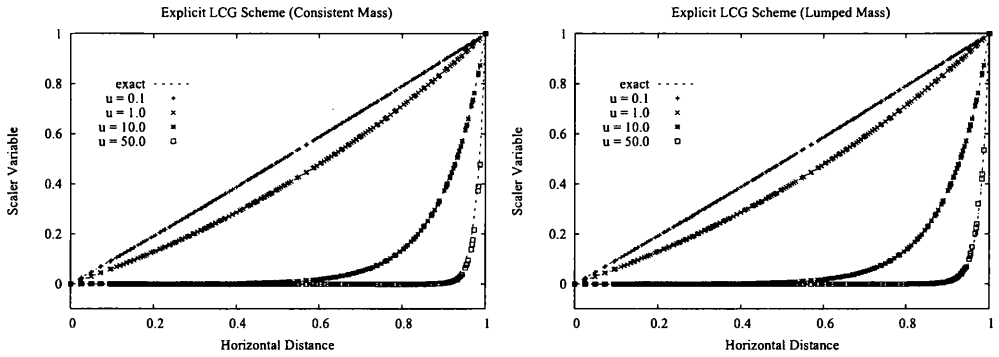


(e) Explicit Galerkin (consistent mass)



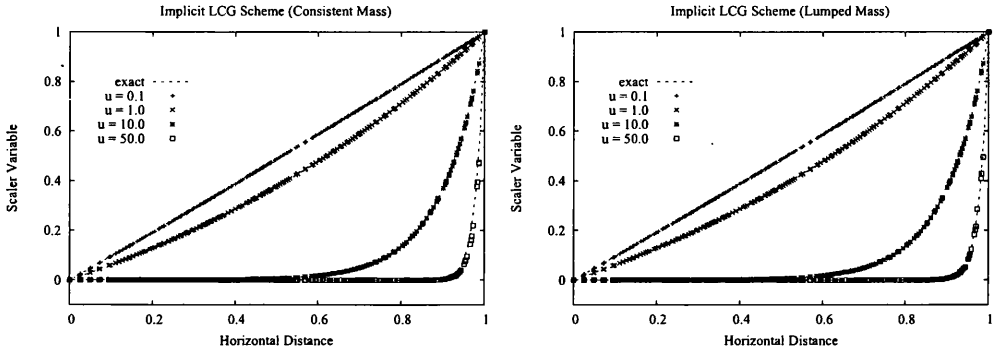
(f) Explicit Galerkin (lumped mass)

Figure 6.5: Convection-diffusion problem in a square channel. Nodal solutions of the scalar variable distribution, along x (for nodes in the range $0.45 \leq y \leq 0.55$) of Mesh B. Results given for SUPG stabilised LCG and global Galerkin methods, using linear basis functions.



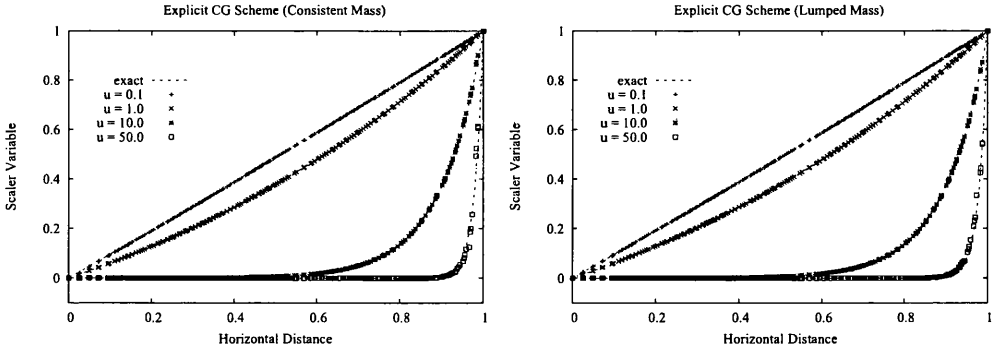
(a) Explicit LCG (consistent mass)

(b) Explicit LCG (lumped mass)



(c) Implicit LCG (consistent mass)

(d) Implicit LCG (lumped mass)



(e) Explicit Galerkin (consistent mass)

(f) Explicit Galerkin (lumped mass)

Figure 6.6: Convection-diffusion problem in a square channel. Nodal solutions of the scalar variable distribution, along x (for nodes in the range $0.45 \leq y \leq 0.55$) of Mesh B. Results given for SUPG stabilised LCG and global Galerkin methods, using quadratic basis functions.

In general, the implemented SUPG stabilisation was successful (at elevated Péclet numbers), in preventing oscillations from polluting all the results obtained. However, when using linear basis functions on meshes A and B, it was found that the consistent-mass version of each scheme did not give the same level of nodal accuracy as their respective lumped-mass counterparts. When a consistent-mass matrix was used, Figure 6.3 and Figure 6.5 show that the implicit LCG method performed the best. Both explicit consistent-mass versions of the global Galerkin and LCG method were considerably inaccurate at higher velocities. A significant improvement is made in the results, obtained for each scheme - using a consistent-mass, when quadratic basis functions are used. As can be seen in Figures 6.4 and 6.6, the improvement in quality is remarkable - even for the explicit schemes.

For lumped-mass versions of each scheme, results were excellent on both meshes A and B, whether using linear or quadratic elements. As with the purely diffusive results obtained in Chapter 5, both the lumped-mass versions of the explicit LCG and explicit global Galerkin gave identical results for Mesh A and Mesh B - thus providing further experimental proof of their equivalent nodal equations (discussed in Chapter 3).

6.2.2 2D convection-diffusion transport of discontinuous inlet data

The second convection-diffusion problem examined in this section is depicted in Figure 6.7. This problem is both well known and widely used for illustrating the qualities of various finite element schemes [41]. It is used here to further demonstrate that LCG method can be readily used with established stabilisation methods, without any detrimental effect on the solution.

As shown in Figure 6.7, the computational domain is a unit square and the flow is uni-directional and constant. However, the convection velocity direction is at an angle θ to the x_1 axis. Additionally, the inlet boundary data presents a discontinuity in the scalar variable, and natural downwind boundary conditions are placed at the outlet. Solutions to this problem are considered at angles of $\theta = 30^o$ and 45^o with $k = 1 \times 10^{-4}$ and $|U| = 1$. As with the first convection-diffusion example, the problem is solved on both structured and unstructured meshes, using Mesh A and Mesh B. However, quadratic basis functions are used throughout. For this problem, suitable stabilisation of convective terms is required

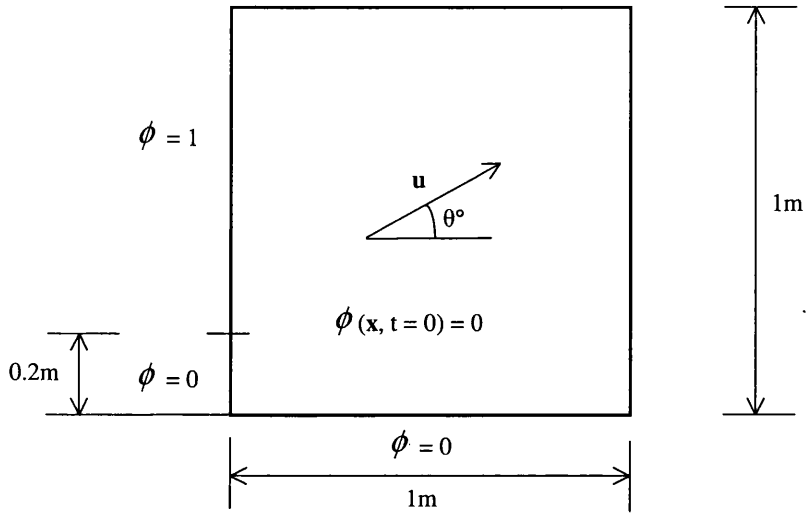


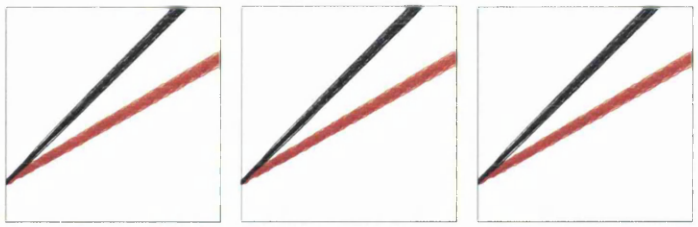
Figure 6.7: Convection of discontinuous inlet data skew to the mesh. Problem geometry and boundary conditions

to prevent oscillations polluting the computed solutions that are obtained from each of the six versions. Here, results are presented for both the LCG and global Galerkin methods - using the *simple explicit* characteristic Galerkin (CG) method of Löhner *et al.* [39] and the SUPG stabilisation method, introduced by Brooks [28] *et al.*.

The results computed for structured Mesh A, are given in Figures 6.8 and 6.9 for the CG and SUPG stabilised methods respectively. It can be seen that: for the case of $\theta = 45^o$, the quality of all twelve solutions are high and differ very little from one and other. This is primarily due to the element-orientation of Mesh A. Here, the discontinuity - present in the convected inlet data, travels along a natural discontinuity between elements on its path, and is strongly captured as the flow travels downstream.

At $\theta = 30^o$ however, the convected discontinuity travels through the elements and the computed contours at this angle of skew are not as sharp as the solutions computed at $\theta = 45^o$. For the CG stabilised methods shown in Figure 6.8, the consistent mass implicit LCG method gave the most diffused solution for $\theta = 30^o$. The other five versions give better quality solutions and are in close agreement with each other. The same case is found for the results in Figure 6.9 - computed using SUPG stabilisation. As can be seen for each

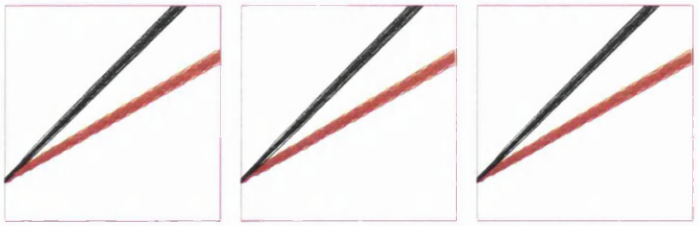
(a) Explicit LCG (consistent mass) (b) Implicit LCG (consistent mass) (c) Galerkin explicit (consistent mass)



(d) Explicit LCG (lumped mass) (e) Implicit LCG (lumped mass) (f) Galerkin explicit (lumped mass)

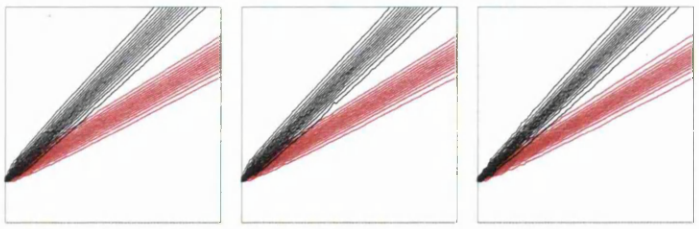
Figure 6.8: Characteristic Galerkin stabilised steady-state solutions using Mesh A for the 2D convection of discontinuous inlet data skew to mesh ($\theta = 30^o, 45^o$).

(a) Explicit LCG (consistent mass) (b) Implicit LCG (consistent mass) (c) Galerkin explicit (consistent mass)

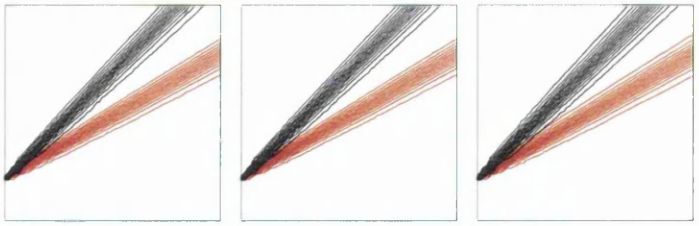


(d) Explicit LCG (lumped mass) (e) Implicit LCG (lumped mass) (f) Galerkin explicit (lumped mass)

Figure 6.9: SUPG stabilised steady-state solutions using Mesh A for the 2D convection of discontinuous inlet data skew to mesh ($\theta = 30^o, 45^o$).

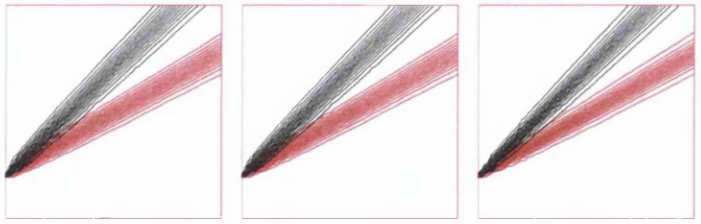


(a) Explicit LCG (consistent mass) (b) Implicit LCG (consistent mass) (c) Galerkin explicit (consistent mass)

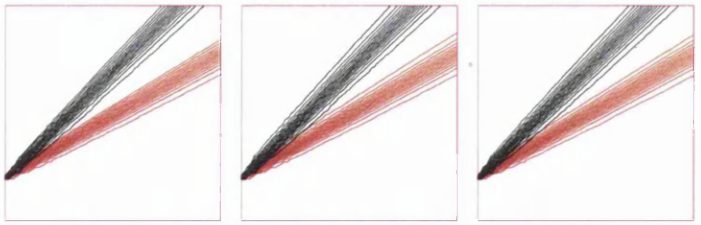


(d) Explicit LCG (lumped mass) (e) Implicit LCG (lumped mass) (f) Galerkin explicit (lumped mass)

Figure 6.10: Characteristic Galerkin stabilised steady-state solutions using Mesh B for the 2D convection of discontinuous inlet data skew to mesh ($\theta = 30^o, 45^o$).

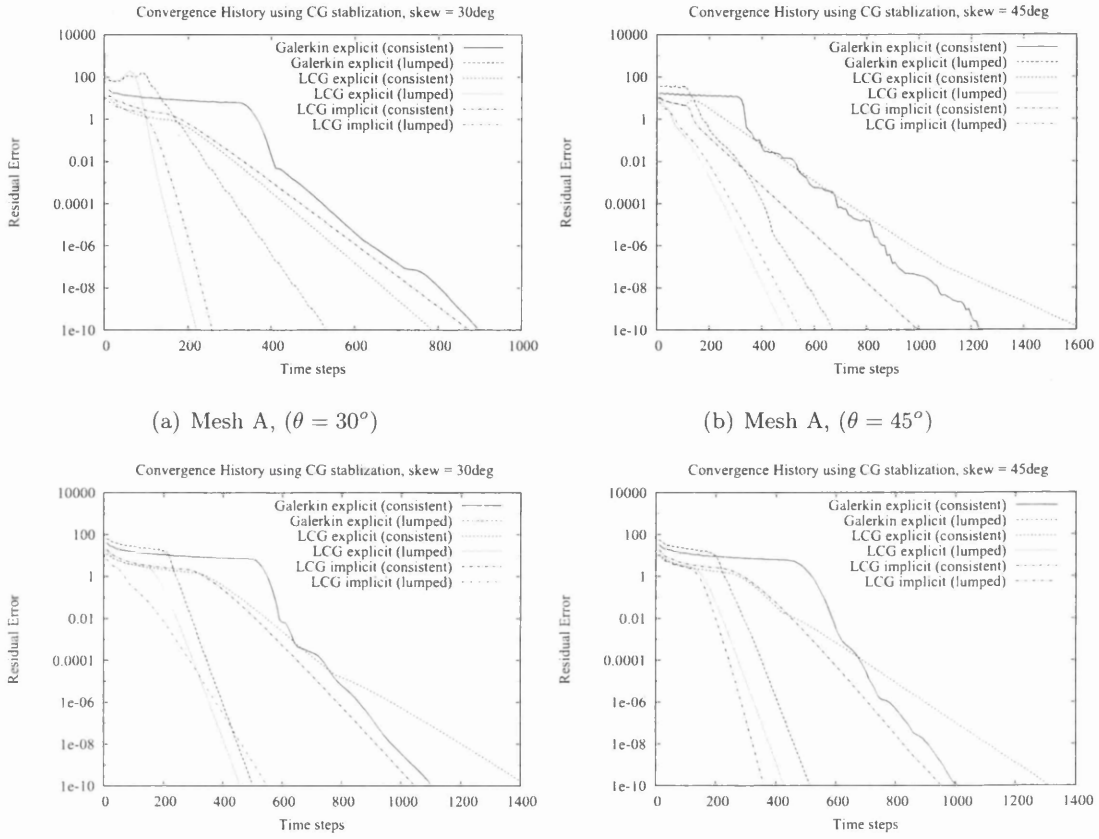


(a) Explicit LCG (consistent mass) (b) Implicit LCG (consistent mass) (c) Galerkin explicit (consistent mass)



(d) Explicit LCG (lumped mass) (e) Implicit LCG (lumped mass) (f) Galerkin explicit (lumped mass)

Figure 6.11: SUPG stabilised steady-state solutions using Mesh B for the 2D convection of discontinuous inlet data skew to mesh ($\theta = 30^o, 45^o$).



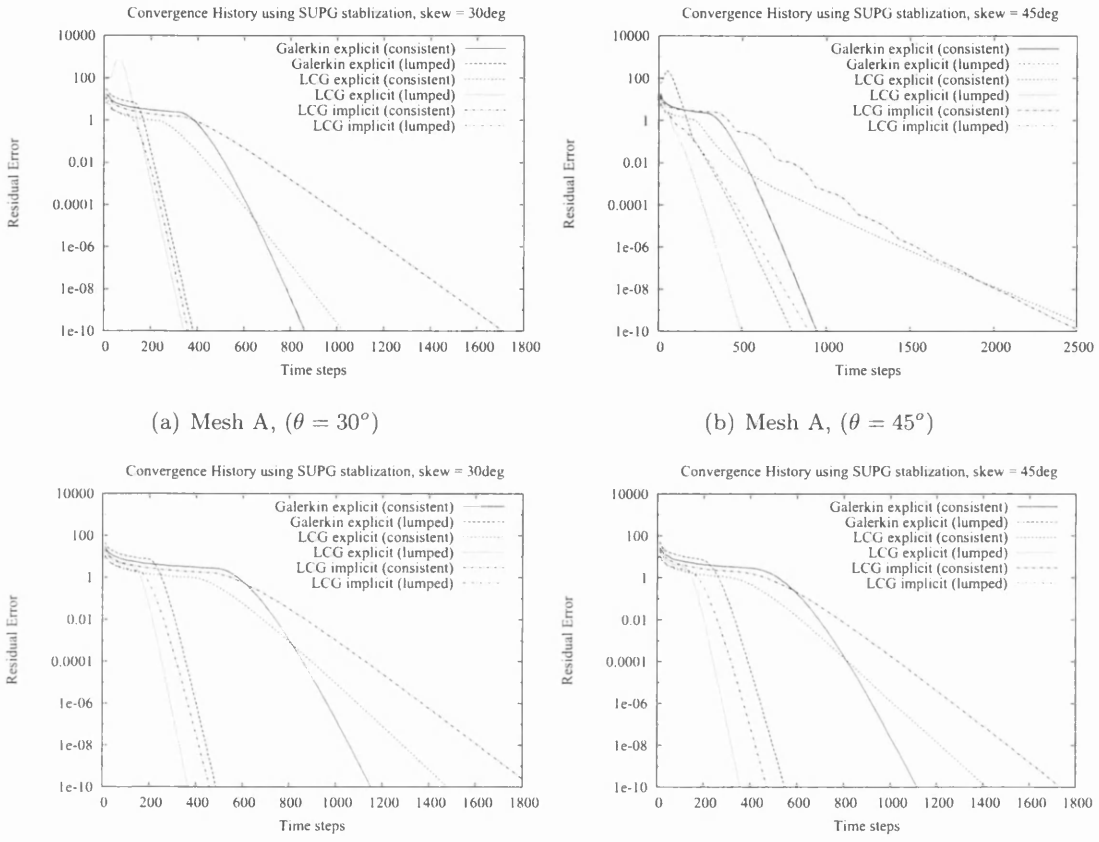
(a) Mesh A, ($\theta = 30^o$)

(b) Mesh A, ($\theta = 45^o$)

(c) Mesh B, ($\theta = 30^o$)

(d) Mesh B, ($\theta = 45^o$)

Figure 6.12: 2D convection of discontinuous inlet data skew to mesh. Steady-state convergence history to a residual error of 1×10^{-10}, using characteristic Galerkin stabilisation and quadratic elements



(a) Mesh A, ($\theta = 30^o$)

(b) Mesh A, ($\theta = 45^o$)

(c) Mesh B, ($\theta = 30^o$)

(d) Mesh B, ($\theta = 45^o$)

Figure 6.13: 2D convection of discontinuous inlet data skew to mesh. Steady-state convergence history to a residual error of 1×10^{-10}, using SUPG stabilisation and quadratic elements

Table 6.1: 2D convection of discontinuous inlet data skew to mesh. Comparison of CPU times for preprocessing and iterations to a residual error of 1×10^{-10}, using Characteristic Galerkin stabilisation and quadratic elements

Method	Mesh A, $\theta = 30^o$	Mesh A, $\theta = 45^o$
Explicit LCG (consistent mass)	2.73s	5.81s
Explicit LCG (lumped mass)	0.77s	1.66s
Implicit LCG (consistent mass)	2.80s	3.13s
Implicit LCG (lumped mass)	0.86s	1.70s
Explicit global Galerkin (consistent mass)	121.77s	120.33s
Explicit global Galerkin (lumped mass)	0.88s	1.12s
Method	Mesh B, $\theta = 30^o$	Mesh B, $\theta = 45^o$
Explicit LCG (consistent mass)	7.78s	7.19s
Explicit LCG (lumped mass)	2.41s	2.25s
Implicit LCG (consistent mass)	5.16s	4.70s
Implicit LCG (lumped mass)	1.97s	1.78s
Explicit global Galerkin (consistent mass)	376.83s	360.89s
Explicit global Galerkin (lumped mass)	1.31s	1.36s

Table 6.2: 2D convection of discontinuous inlet data skew to mesh. Comparison of CPU times for preprocessing and iterations to a residual error of 1×10^{-10}, using SUPG stabilisation and quadratic elements

Method	Mesh A, $\theta = 30^o$	Mesh A, $\theta = 45^o$
Explicit LCG (consistent mass)	5.63s	12.39s
Explicit LCG (lumped mass)	1.92s	2.75s
Implicit LCG (consistent mass)	8.73s	13.23s
Implicit LCG (lumped mass)	1.91s	4.67s
Explicit global Galerkin (consistent mass)	120.12s	127.01s
Explicit global Galerkin (lumped mass)	1.45s	3.05s
Method	Mesh B, $\theta = 30^o$	Mesh B, $\theta = 45^o$
Explicit LCG (consistent mass)	10.39s	10.02s
Explicit LCG (lumped mass)	2.50s	2.45s
Implicit LCG (consistent mass)	11.83s	11.23s
Implicit LCG (lumped mass)	2.92s	3.05s
Explicit global Galerkin (consistent mass)	389.14s	381.61s
Explicit global Galerkin (lumped mass)	2.06s	2.31s

method, the results obtained using SUPG stabilisation are similar to the corresponding results obtained using the characteristic Galerkin method (Figure 6.8).

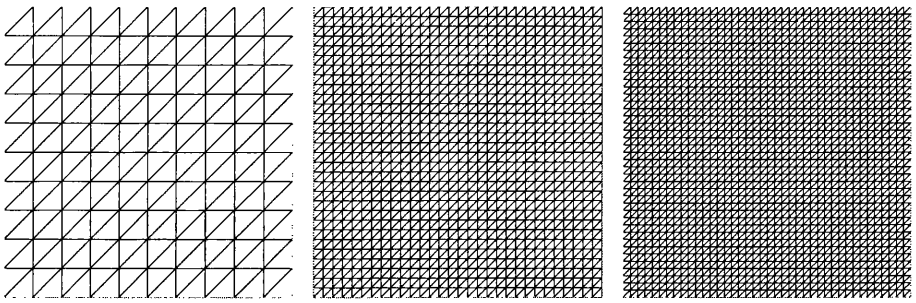
The solutions obtained on Mesh B, are given in Figure 6.10 for CG stabilisation and Figure 6.11 for SUPG stabilisation. Generally the results on this mesh are more diffused, but considering the unstructured nature of the mesh, good non-oscillatory results are obtained. It was also found on Mesh B, that the results obtained at $\theta = 30^o$ and $\theta = 45^o$, for each method using the CG stabilisation (Figure 6.10), are both very similar to the corresponding result using the SUPG stabilisation, shown in Figure 6.11. Closer inspection shows that all the lumped mass variations gave slightly less diffused contours.

The convergence histories for each method were recorded for both types of stabilisation. In Figure 6.12, the convergence histories are given using CG stabilisation, for both skew angles on both Mesh A and Mesh B. Figure 6.13 gives the corresponding convergence histories recorded, using SUPG stabilisation. Generally the lumped-mass versions gave the fastest convergence rates. It should be noted that the explicit lumped-mass LCG method converged faster than the equivalent explicit lumped-mass global Galerkin method. This is similar to what was found in Chapter 5 for pure diffusive problems and is due to the explicit lumped-mass LCG method permitting a higher maximum time-step. The various CPU times recorded for each method, using Characteristic Galerkin stabilisation, are given in Table 6.1. The recorded CPU times for each method using using SUPG stabilisation are given in Table 6.2. In general, the lumped-mass versions of each scheme give the fastest computations.

6.3 Mesh convergence properties of the explicit and implicit CG based LCG methods

This section gives an analysis of the mesh convergence properties of the explicit and implicit CG based LCG methods. To examine the properties of each LCG scheme, the solution of the general convection-diffusion equation with a source term is solved.

The solutions computed on a number of different meshes, are used to obtain a graph of solution error as a function of mesh element-size. From this graph, the order of



(a) Mesh C (b) Mesh D (c) Mesh E

Figure 6.14: L_2 rates of mesh convergence of the LCG methods, at various Péclet numbers for the general convection-diffusion equation with source term. Structured triangular finite element meshes used.

each particular scheme is easily obtained. The effect of mass lumping is taken into account by giving results for each method with and without a consistent-mass matrix. Additionally, both linear and quadratic elements are investigated, and all results are compared with the benchmark data of Donea *et al.* [144].

6.3.1 Analysis for linear and quadratic triangular elements

The order of mesh convergence describes the behavior of the solution error E as a function of mesh element-size h. Abanto *et al.* [145] give the error as

$$E = Ch^q + H.O.T. \tag{6.3}$$

If high-order terms are neglected, it is clear that by plotting a curve of Log(E) as a function of Log(h), the mesh convergence rate, q, can be obtained from the slope of the curve.

In this convergence study, both the explicit and implicit CG based LCG methods are investigated for the general convection-diffusion equation with source term. The governing equation for this type of problem is given by Equation (3.57). For each scheme the L_2 error for various Péclet numbers, is computed on a number of structured meshes and plotted against h. The analysis considers the use of both linear and quadratic triangular elements. To determine the effect on the convergence characteristics of each scheme by

the lumping procedure discussed in Chapter 2 (see Figure (2.1), both lumped-mass and consistent-mass versions are investigated.

A similar problem definition to the one described in the previous section by Figure 6.2, is used. The domain is a unit square channel, with Dirichlet-type boundary conditions of $\phi = 0$ and $\phi = 1$ applied at the inlet and exit respectively. On the top and bottom channel walls, a no-flux boundary condition is prescribed. Only the velocity in the x_1 direction was varied to obtain the desired Péclet number, with k equal to unity and u_2 equal to zero being assumed over the whole domain.

In addition to the above, the following source term is used

$$S_i = \delta_{1i}(4u_i{x_i}^3 - 12k{x_i}^2) \tag{6.4}$$

where the Kronecker delta function ensures that the source-term only acts only in the x_1-direction. The inclusion of the internal source gives an exact solution to this problem of

$$\phi(\mathbf{x}) = {x_1}^4 \tag{6.5}$$

This solution is always the same, regardless of the value of Péclet number used. The source term, for the LCG methods, is applied to the RHS forcing vector of the elemental equation sets. The error E along the mid-horizontal line was calculated by

$$E^2 = \int_{x_1=0}^{L} (\tilde{\phi} - \phi)^2 dx_1 \tag{6.6}$$

The above integral was evaluated numerically using the trapezium rule. Sufficient sampling points were created along the mid-horizontal line until the calculated value of E converged to an accuracy of 4 decimal places.

Structured Mesh A, shown in Figure 6.1 for the previous examples, was used in this study. Additionally three other structured meshes, of similar structure are employed. These are shown in Figure 6.14. Mesh C has 200 elements, Mesh D has 1800 elements, and Mesh E is the finest with 3200 elements. An initial value of $\phi = 0$ is used everywhere inside the domain, and the steady-state solution is obtained through time-stepping until

the prescribed residual error tolerance defined in Equation (6.1), reaches a tolerance of $\epsilon < 1 \times 10^{-10}$.

The study considers the use of linear triangular elements first, for different values of Péclet number, Pe. Figure 6.15 shows the graphs of convergence rates computed for each of the explicit and implicit LCG methods, using linear basis functions. The values of Pe selected are: 0, 10, 25, and 50. The results at $Pe = 0$, correspond to the standard LCG procedure (Equation 3.7). In addition to Figure 6.15, the convergence rates for each method is summarised in Table 6.3, for each Péclet number. For the $Pe = 0$ case, each scheme in Figure 6.15 is shown to have second-order accuracy. For $Pe > 0$, all CG based LCG schemes have convergence rates of 2.00 or higher. Table 6.3 also gives the mesh convergence rates for using SUPG stabilisation computed by Donea *et al.* [144]. As can be seen, second-order accuracy is obtained for the $Pe = 0$ case, however the rate of convergence reduces to first-order as the problem becomes convection dominated.

Computations performed using quadratic elements, for the LCG methods, are discussed next. Figure 6.16 shows the convergence rates, computed for the explicit and implicit LCG methods with quadratic basis functions. For comparison, the same values of Pe used for linear elements are chosen and Table 6.4 summarises the convergence rates at each Péclet number, for each variation. As can be seen, both consistent-mass versions of the explicit and the implicit LCG method gave a third-order convergence rate for the purely diffusive case of $Pe = 0$, with $q \approx 3.02$ and 3.03 respectively. This is consistent with the rate of $q = 3.00$, obtained by Donea *et al.* [144] for quadratic elements using the global Galerkin method. As the Péclet number increases, the convergence rate decreases. At $Pe = 50$ the mesh convergence rates are $q = 2.20$ and $q = 2.31$ for the consistent-mass explicit LCG method and the consistent-mass implicit LCG method respectively. However both these rates were higher than the mesh convergence rate of $q = 2.15$, which is given by Donea. A lumping procedure, frequently used for quadratic triangular elements [21, 42, 88], was discussed in Chapter 2. It was revealed that lumping the quadratic element mass matrix in this way, had an adverse effect on the mesh convergence rate for both the explicit and the implicit CG based LCG schemes. The convergence rate dropped from third-order to second order for the $Pe = 0$ case. As the problem became convection dominated, the computed

Table 6.3: Calculated L_2 rates of mesh convergence for $Pe = 0$, $Pe = 10$, $Pe = 25$ and $Pe = 50$, using linear triangular elements

Method	$Pe = 0$	$Pe = 10$	$Pe = 25$	$Pe = 50$
Explicit LCG (consistent mass)	2.19	2.16	2.17	2.11
Explicit LCG (lumped mass)	2.07	2.06	2.05	2.01
Implicit LCG (consistent mass)	2.25	2.17	2.16	2.12
Implicit LCG (lumped mass)	2.18	2.17	2.16	2.14
Donea *et al.* (SUPG) [144]	2.00	0.95	0.94	0.93

Table 6.4: Calculated L_2 rates of mesh convergence for $Pe = 0$, $Pe = 10$, $Pe = 25$ and $Pe = 50$, using quadratic triangular elements

Method	$Pe = 0$	$Pe = 10$	$Pe = 25$	$Pe = 50$
Explicit LCG (consistent mass)	3.02	2.85	2.22	2.20
Explicit LCG (lumped mass)	2.09	1.84	1.55	0.99
Implicit LCG (consistent mass)	3.03	2.81	2.79	2.31
Implicit LCG (lumped mass)	2.08	1.83	1.64	1.20
Donea *et al.* (SUPG) [144]	3.00	2.22	2.16	2.15

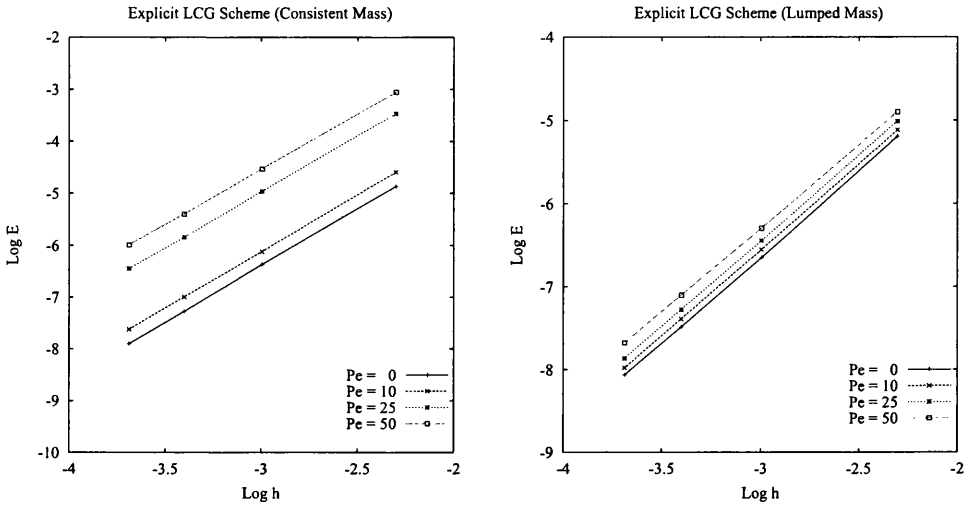
rates reduced further, becoming first-order convergent for $Pe = 50$.

6.4 Summary

In this Chapter, SUPG stabilised LCG methods and characteristic based LCG methods were validated. Both explicit and implicit approaches, were shown to give good performance for solving general convection-diffusion problems. Results using the global Galerkin methods were also shown, to provide a comparison in some of the problems.

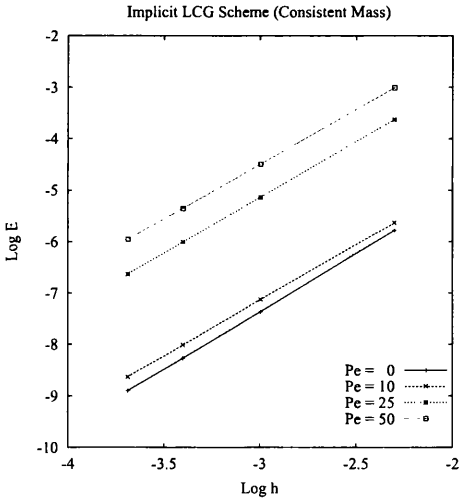
During the analysis, the robustness of the explicit and implicit LCG methods have been illustrated in 2D, on both linear and quadratic triangular elements. For these methods, the effect of mass lumping was also considered, with results being given for using both a lumped-mass and a consistent-mass matrix.

Comparisons of the LCG method clearly showed that the solutions obtained are

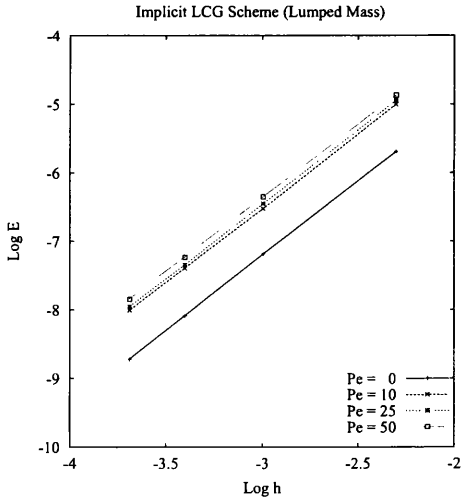


(a) LCG explicit (consistent mass)

(b) LCG explicit (lumped mass)

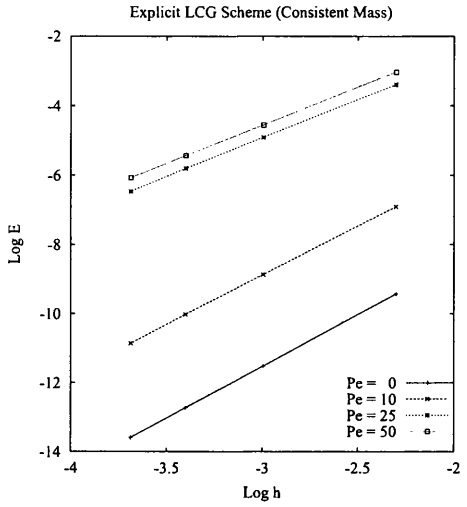


(c) LCG implicit (consistent mass)

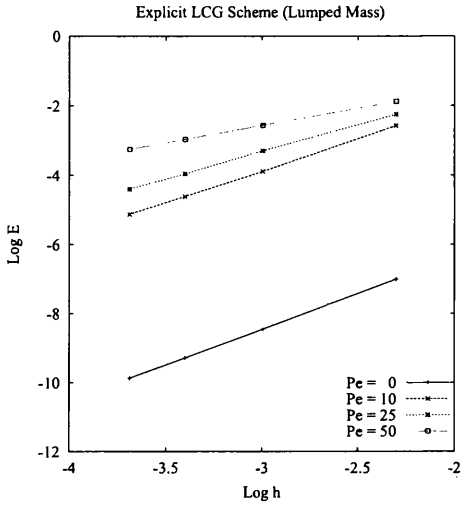


(d) LCG implicit (lumped mass)

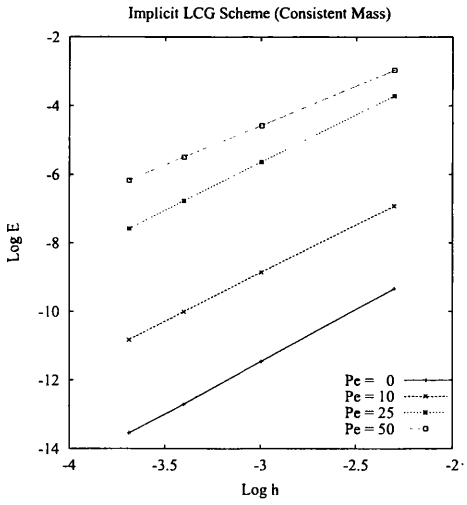
Figure 6.15: L_2 rates of mesh convergence at various Péclet numbers, for the general convection-diffusion equation with source term. Rates given for the LCG methods using characteristic based stabilisation and linear triangular elements



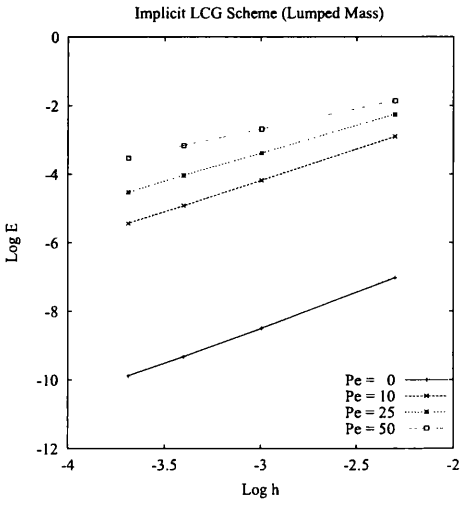
(a) LCG explicit (consistent mass)



(b) LCG explicit (lumped mass)



(c) LCG implicit (consistent mass)



(d) LCG implicit (lumped mass)

Figure 6.16: L_2 rates of mesh convergence at various Péclet numbers, for the general convection-diffusion equation with source term. Rates given for the LCG methods using characteristic based stabilisation and quadratic triangular elements

at least as good as global Galerkin methods. Additionally, the LCG method has the further advantage of solving smaller equation sets and provides element-wise and global conservation.

Mesh convergence rates for the explicit and implicit LCG methods were computed. When linear elements are used, the characteristic based LCG schemes gave second-order convergence. For consistent-mass LCG methods, the mesh convergence rate was shown to increase for each scheme when quadratic elements were used. However, using the lumping approximation described in Figure 2.1 for quadratic elements, was shown to reduce mesh convergence.

Chapter 7

The LCG-CBS Scheme for Incompressible Flow Problems

7.1 Introduction

Chapter 6 showed that the characteristic based LCG methods gave excellent performance for convection dominated transport problems. This chapter extends the evaluation of characteristic based schemes, to solve problems of incompressible flow.

In this chapter, the characteristic based split, locally conservative Galerkin (CBS-LCG) method is evaluated using a number of known incompressible flow problems. The first problem solved is the flow inside a lid-driven cavity [23, 90, 95, 96, 97, 98, 108, 114, 146, 147, 148]. Both Stokes flow and Navier-Stokes flows are considered. Grids of linear and quadratic elements are employed, and results are compared to benchmark data [146]. The effect of using a lumped edge flux is also investigated.

The next problem considered in this chapter, is the solution of flow past a circular cylinder at low values of Reynolds number. The results of physical experiments for this classic problem are well known and have been reported by many authors [149, 150, 151, 129, 152]. Additionally, this problem is often used in numerical methods - as a benchmark problem, and detailed computed results have been given for analysis [108, 153, 154, 155, 156, 157]. In this example, test cases are performed, using the LCG-CBS scheme, for a

range of Reynolds numbers between $Re = 1$ and $Re = 40$.

In the final section, problems of laminar incompressible-flow through two-dimensional channels are solved using the LCG-CBS scheme. The first problem considered, is the development of simple Poiseuille-flow through a rectangular channel. This problem is employed to clearly demonstrate that mass conservation is satisfied for open systems, when the linear and quadratic CBS-LCG schemes are used. The second channel-flow problem considered in this section, is the flow over a downstream-facing step in a two-dimensional channel. This type of channel flow problem has been investigated extensively in the literature, both experimentally [158, 159, 160], and as a benchmark test for evaluating accuracy of numerical algorithms [23, 57, 92, 96, 97, 105, 113, 114, 147]. The problem involves regions of flow separation and subsequent reattachment, as well as a recirculating flow behind the step. This mixture of flow regimes make it a tough benchmark problem for validating numerical schemes. The step-design is based on an original experiment conducted by Denham and Patrick [158], and thus allows a quantitative comparison to be made with the available experimental data. Results are given for structured and unstructured grids.

7.2 Incompressible flow inside a lid-driven square cavity

The first problem considered in this chapter is the incompressible flow of an isothermal fluid in a lid-driven cavity. The problem is investigated for a range of Reynolds numbers including the case of Stokes flow ($Re = 0$). Qualitative results are given using both linear and quadratic elements, additionally comparisons of the velocity profile are made with reliable benchmark data [23, 146].

Incompressible flow inside a lid-driven cavity is a well known benchmark problem. It has been studied by many authors and is frequently used in validating new CFD source-codes [23, 96, 90, 114]. The most notable of these works is the detailed investigation of Ghia *et.al* [146] which is often used as the standard to provide benchmark data for this problem.

There are many descriptions of the cavity problem given within the literature. The most popular type is studied here to allow comparisons to be made with the standard benchmark data of Ghia and co-workers [146]. The cavity geometry and boundary con-

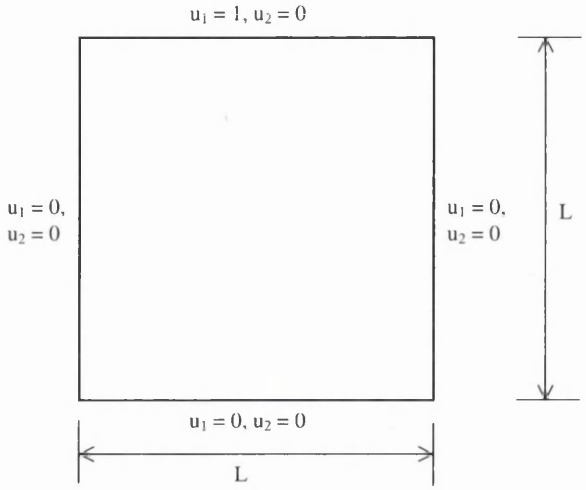
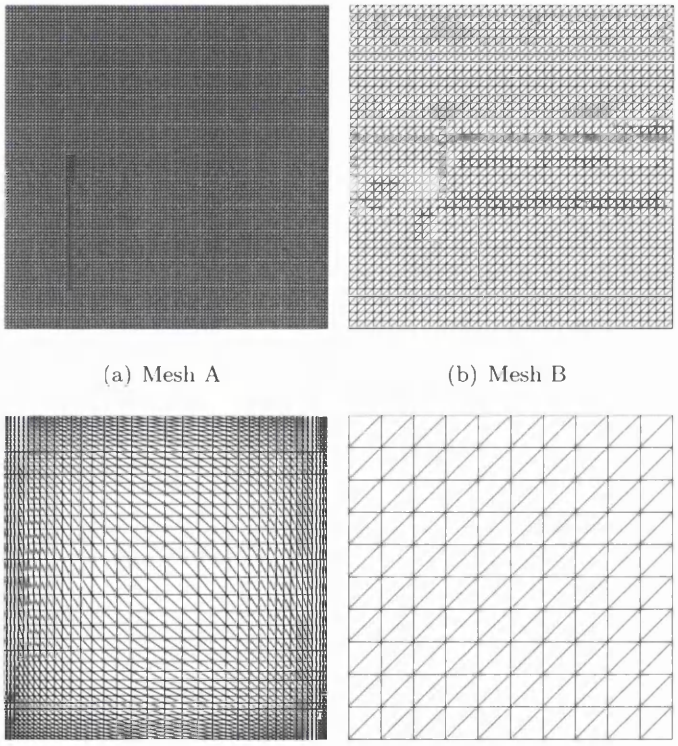


Figure 7.1: Flow in a square cavity. Geometry and boundary conditions



(a) Mesh A (b) Mesh B

(c) Mesh C (d) Mesh D

Figure 7.2: Flow in a square cavity. Structured meshes of linear and quadratic elements, used in the computations.

ditions are shown in Figure 7.1. It is unit square in size and sealed so that no fluid may enter or leave its boundaries. No-slip boundary conditions are applied to the cavity walls and the flow of the fluid inside the cavity is generated by the motion of the top surface, which travels in the horizontal direction. As the cavity is non-leaky the top two corners of the cavity are considered to be part of the walls instead of the lid and the velocity is zero at these points. This is often referred to as ramp conditions in the literature [23]. The singularities at the top two corners make this problem difficult to solve.

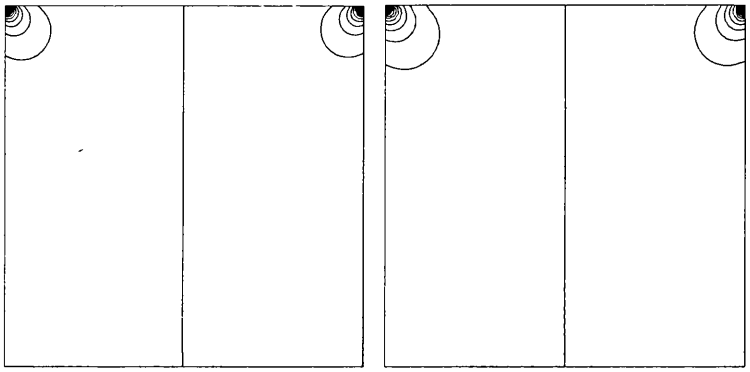
Solutions have been obtained for the diffusive case of Stokes flow, as well as for the range of Reynolds numbers: $10 \leq Re \leq 3200$. This gives an assessment of the performance of the LCG-CBS schemes for a full range of problems from purely diffusive to convection dominated. For this problem type, Re is based on both the lid-velocity and its length.

A number of linear and quadratic structured meshes were used in this study and are shown in Figure 7.2. Mesh A is a uniform structured mesh of 20000 linear elements and it has a nodal resolution of 10201 nodes. Mesh B is also a uniform structured mesh, but consists of 3200 quadratic elements and has 6561 nodes. Mesh C is a non-uniform grid, with 1521 linear elements and 2888 nodes. Mesh D is the coarsest of the grids used, it has only 200 quadratic elements and 441 nodes.

The case of Stokes flow is considered first, with all meshes being used in the study. Nithiarasu [96] notes that many artificial compressibility (AC) schemes have difficulties in solving viscous flows. The AC-CBS scheme developed for the global Galerkin method [23, 96, 97, 98, 76], suffers from no such problems. However, since the CBS-LCG method used here also employs an AC parameter (see Chapter 3) it is important to show that this scheme is also compatible for Stokes flow.

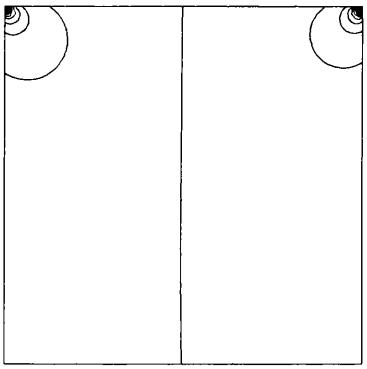
In order to model Stokes flow: the convection and characteristic terms in Step 1 (Equation 3.82) and the characteristic term in Step 3 (Equation 3.90) were switched off, also Re was set to unity. Step 2 was unaltered - retaining second-order pressure stabilisation. The above changes give a matrix free fractional time-stepping scheme, similar in structure to that described by Nithiarasu for incompressible solid mechanics [76]

The results for Stokes flow are shown in Figures 7.3 to 7.7. Figure 7.3 gives the pressure contours computed on each mesh. The qualitative results are all of good quality,

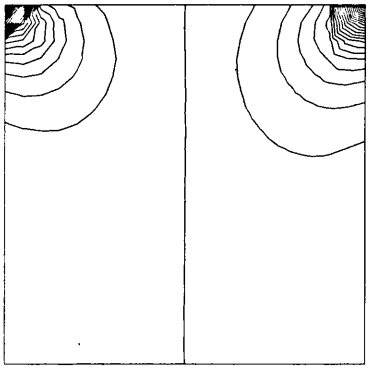


(a) Mesh A

(b) Mesh B

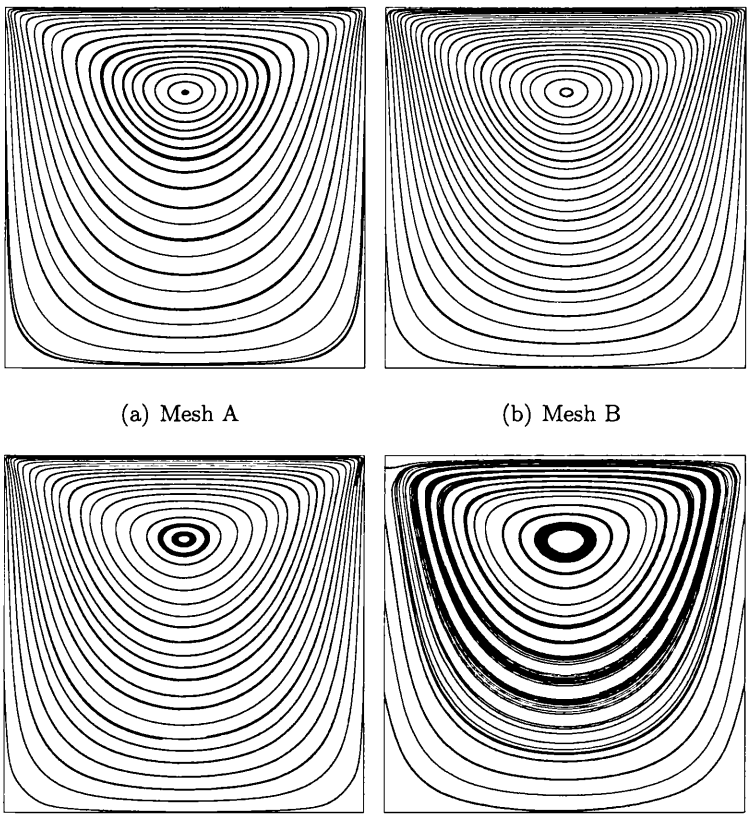


(c) Mesh C



(d) Mesh D

Figure 7.3: Stokes flow in a square cavity. Comparison of computed pressure contours using meshes A-D for Stokes-flow



(a) Mesh A (b) Mesh B

(c) Mesh C (d) Mesh D

Figure 7.4: Stokes flow in a square cavity. Comparison of stream-traces of computed flow using meshes A-D for Stokes-flow

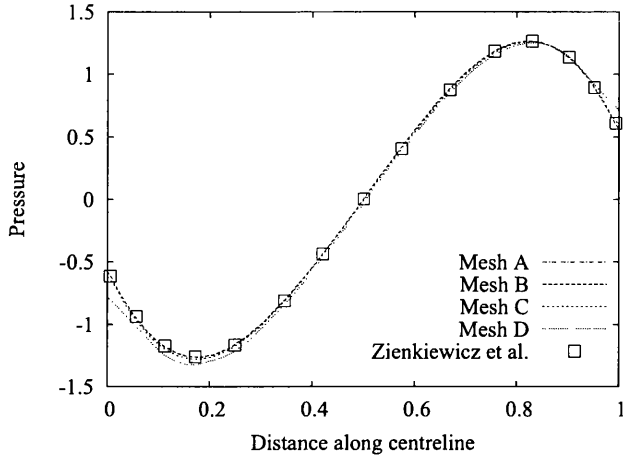


Figure 7.5: Stokes flow in a square cavity. Comparison of pressure distribution along mid-horizontal line.

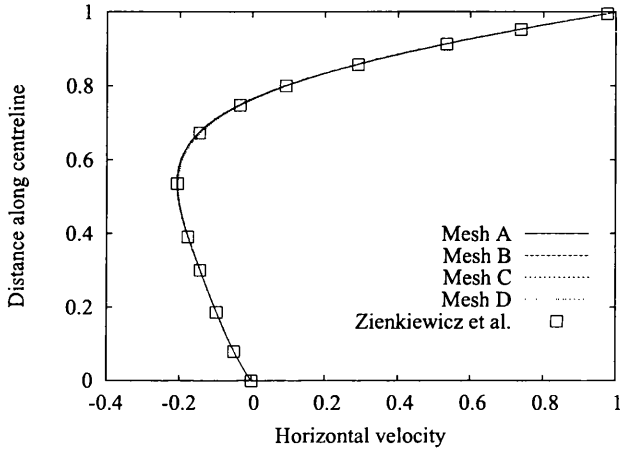


Figure 7.6: Stokes flow in a square cavity. Comparison of the horizontal velocity component along mid-vertical line.

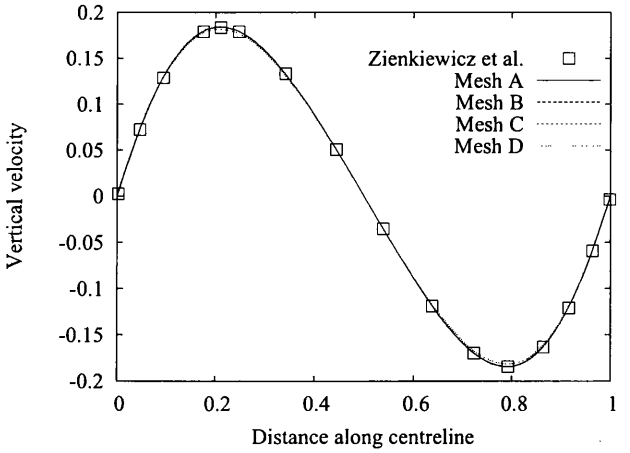


Figure 7.7: Stokes flow in a square cavity. Comparison of the vertical velocity component along mid-horizontal line.

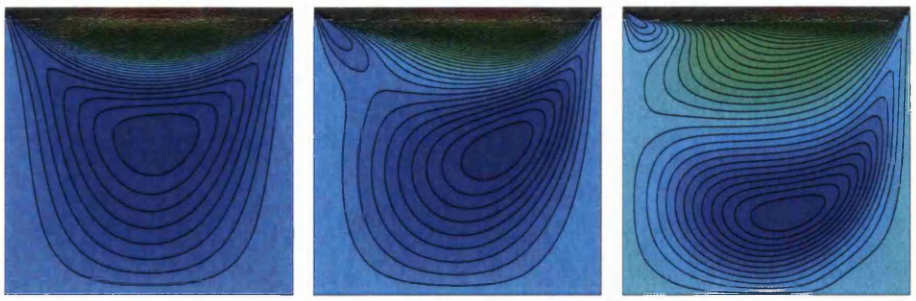
with no non-physical oscillations appearing in the solutions. All solutions are in excellent agreement and the negative and positive singularities in the top left and right corners, are predicted. Additionally, the contour at zero pressure value coincides with the mid-vertical centreline - even for the coarse grid (Mesh D).

To provide a comparison of the accuracy of the pressure solution, the pressure along the mid-height, horizontal-line has been plotted in Figure 7.5 for each mesh. Additionally the benchmark solution of Zienkiewicz *et al.* [23] has also be plotted. As seen there is strong agreement with the solutions obtained on Meshes A, B, and C. Comparing with the benchmark solution, Meshes A and B give the most accurate results. The pressure results on Mesh D are the least accurate. This is due to the grid coarseness.

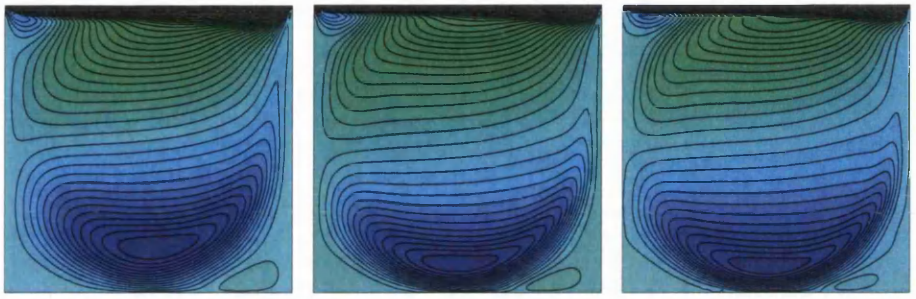
The stream-traces of the velocity field computed for each mesh, are shown in Figure 7.4. Here, Meshes A - C are seen to give similar flow patterns. To compare the velocities computed on each mesh, the vertical and horizontal velocity components have been plotted on the horizontal and vertical centrelines respectively in Figures 7.6 and 7.7. Again the results show that the results on Meshes A, B, and C are in excellent agreement with each other. Additionally, comparisons made with the benchmark solution of Zienkiewicz *et al.* [23] show that the results on these meshes are also highly accurate. For the velocity components, the solution on Mesh D is very good.

The lid-driven cavity problem is now modelled at different values of Reynolds number. Here, the selection: Re = 10, 100, 400, 1000, 2000, 3200 is used. Both Mesh A and Mesh B are employed here to illustrate the performance of the CBS-LCG scheme, for both linear and quadratic elements.

The qualitative results for the selected Reynolds numbers, are given in Figures 7.8 to 7.15. The first pair, show the contours of the horizontal velocity component at different values of Re. Fifty contours were used in the plot and solutions using linear elements are shown in Figure 7.8. The corresponding results for quadratic elements are shown in Figure 7.9. All the results are excellent, and there is good agreement between the solutions obtained using linear elements and the solutions obtained using quadratic elements. Figures 7.10 and 7.11 give the computed vertical velocity fields, on Mesh A and Mesh B respectively. Again,

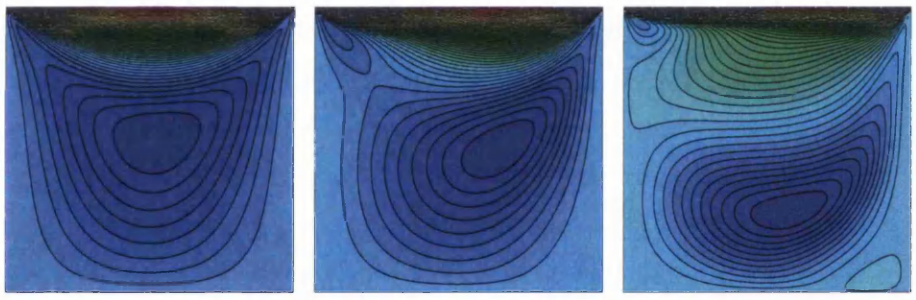


(a) u_1 velocity, $Re = 10$ (b) u_1 velocity, $Re = 100$ (c) u_1 velocity, $Re = 400$

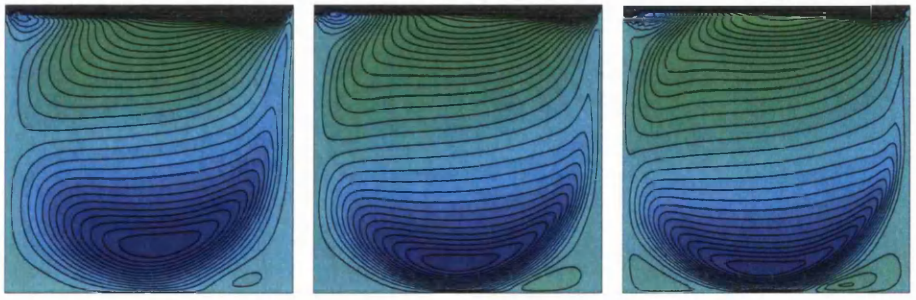


(d) u_1 velocity, $Re = 1000$ (e) u_1 velocity, $Re = 2000$ (f) u_1 velocity, $Re = 3200$

Figure 7.8: Flow in a square cavity. Contours of horizontal velocity components at different values of Re, using linear elements and Mesh A

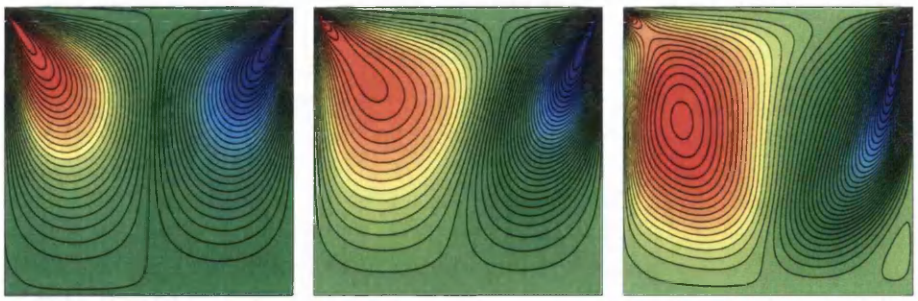


(a) u_1 velocity, $Re = 10$ (b) u_1 velocity, $Re = 100$ (c) u_1 velocity, $Re = 400$

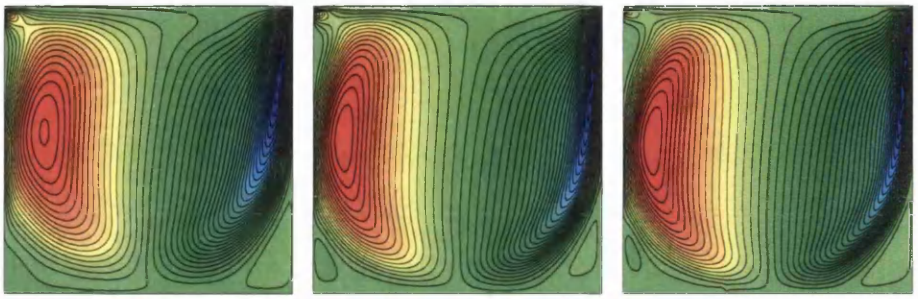


(d) u_1 velocity, $Re = 1000$ (e) u_1 velocity, $Re = 2000$ (f) u_1 velocity, $Re = 3200$

Figure 7.9: Flow in a square cavity. Contours of horizontal velocity components at different values of Re, using quadratic elements and Mesh B

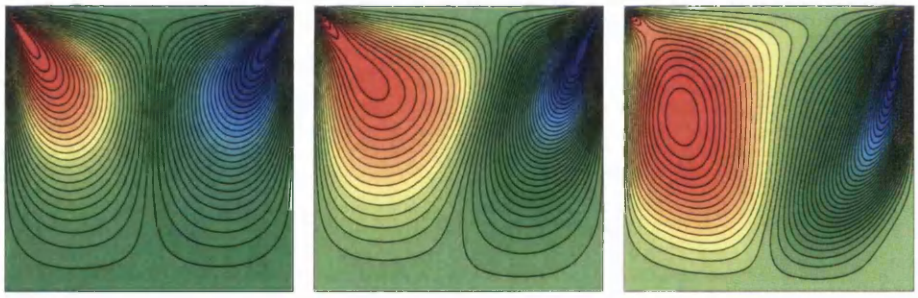


(a) u_2 velocity, $Re = 10$ (b) u_2 velocity, $Re = 100$ (c) u_2 velocity, $Re = 400$

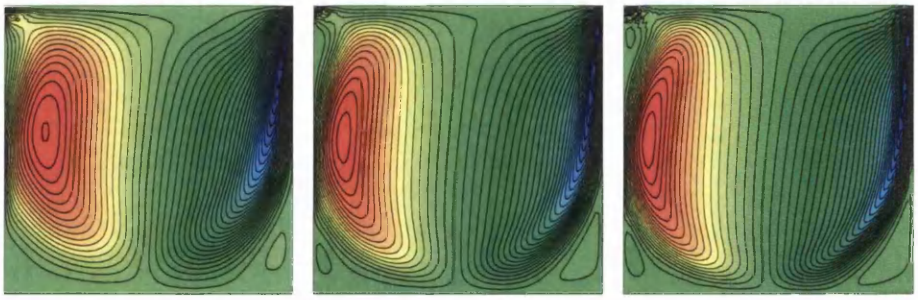


(d) u_2 velocity, $Re = 1000$ (e) u_2 velocity, $Re = 2000$ (f) u_2 velocity, $Re = 3200$

Figure 7.10: Flow in a square cavity. Contours of vertical velocity components at different values of Re, using linear elements and Mesh A

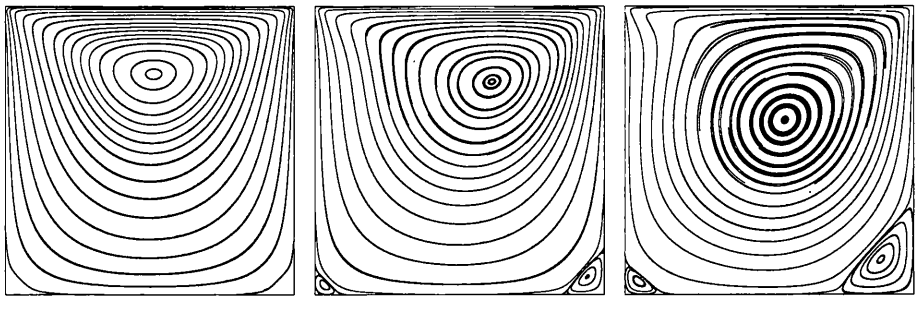


(a) u_2 velocity, $Re = 10$ (b) u_2 velocity, $Re = 100$ (c) u_2 velocity, $Re = 400$

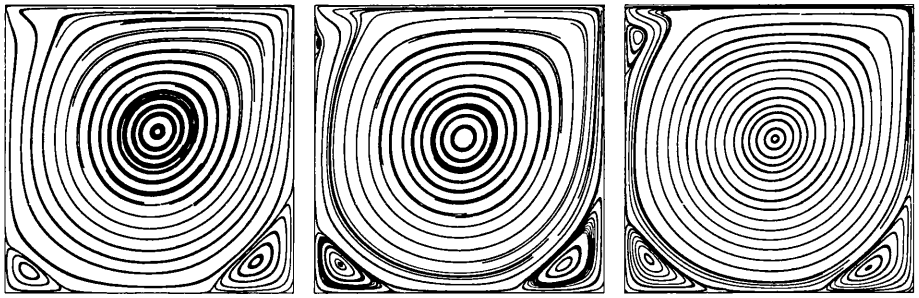


(d) u_2 velocity, $Re = 1000$ (e) u_2 velocity, $Re = 2000$ (f) u_2 velocity, $Re = 3200$

Figure 7.11: Flow in a square cavity. Contours of vertical velocity components at different values of Re, using quadratic elements and Mesh B

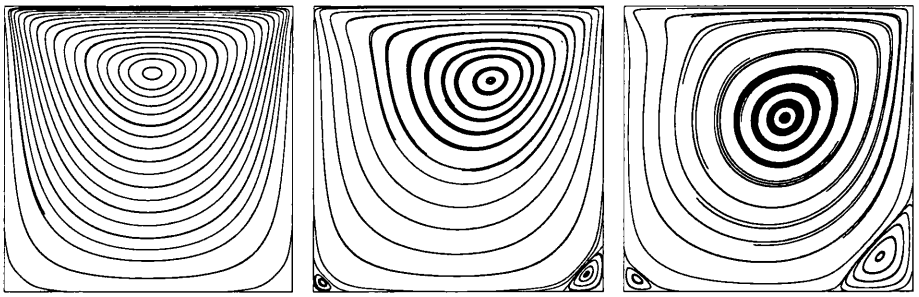


(a) $Re = 10$ (b) $Re = 100$ (c) $Re = 400$

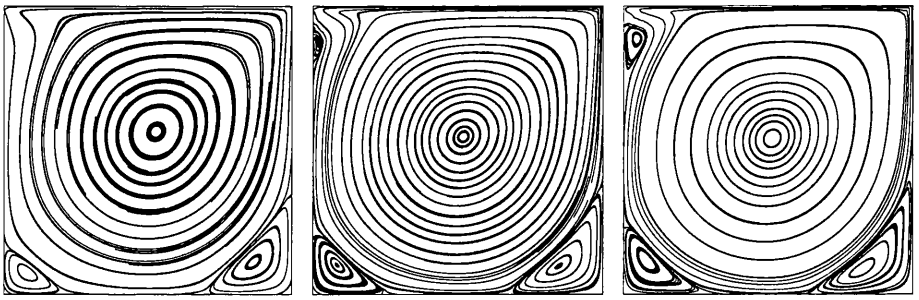


(d) $Re = 1000$ (e) $Re = 2000$ (f) $Re = 3200$

Figure 7.12: Flow in a square cavity. Stream-traces at different values of Re, using linear elements and Mesh A

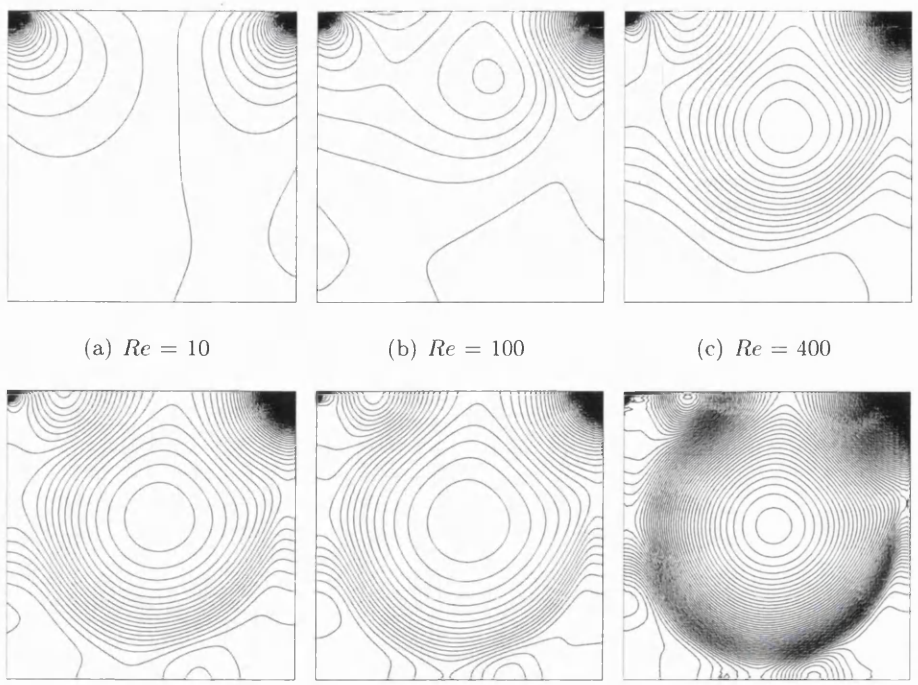


(a) $Re = 10$ (b) $Re = 100$ (c) $Re = 400$



(d) $Re = 1000$ (e) $Re = 2000$ (f) $Re = 3200$

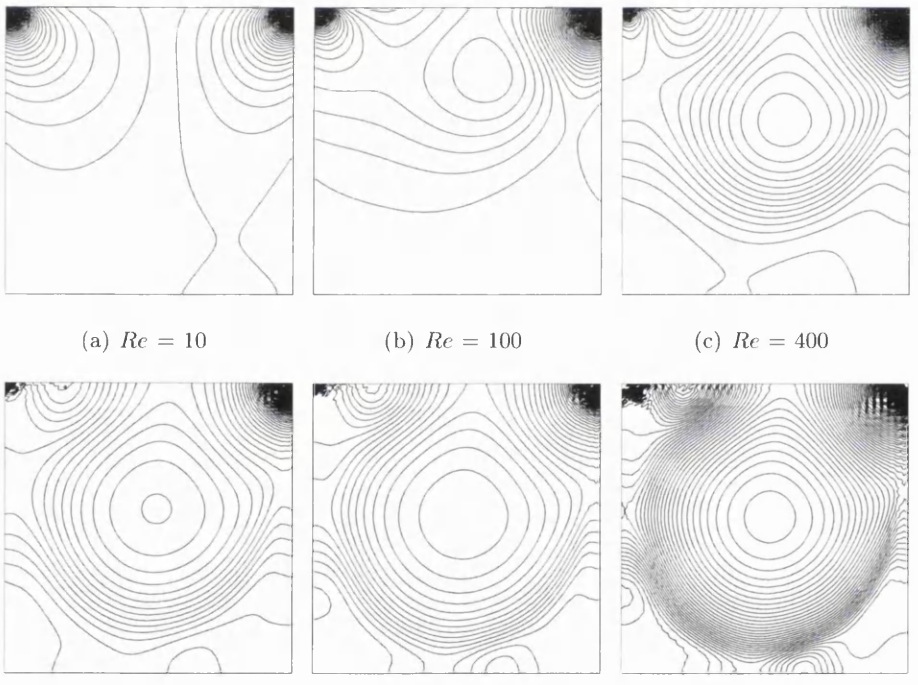
Figure 7.13: Flow in a square cavity. Stream-traces at different values of Re, using quadratic elements and Mesh B



(a) $Re = 10$ (b) $Re = 100$ (c) $Re = 400$

(d) $Re = 1000$ (e) $Re = 2000$ (f) $Re = 3200$

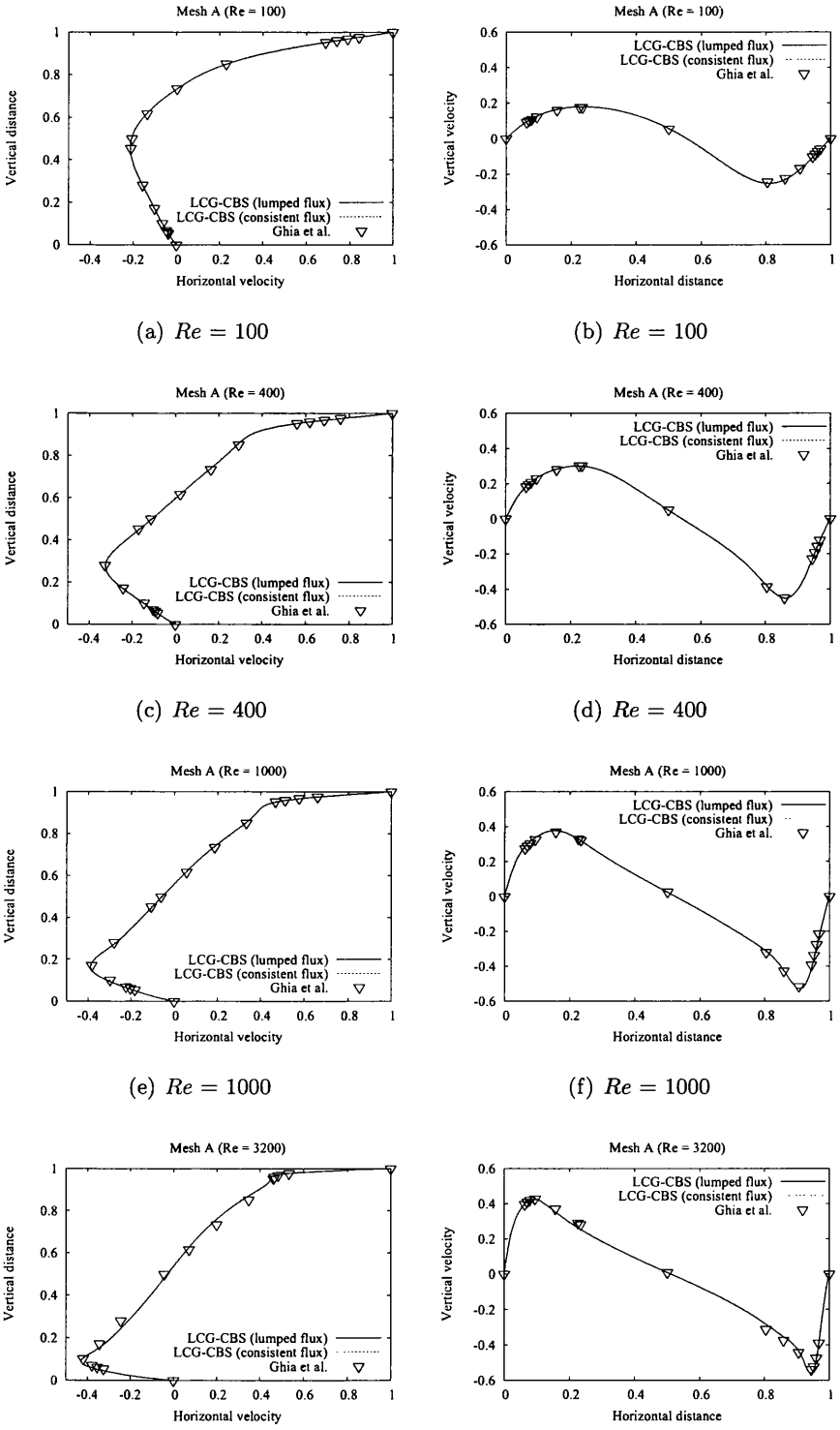
Figure 7.14: Flow in a square cavity. Computed pressure contours at different values of Re, using linear elements and Mesh A



(a) $Re = 10$ (b) $Re = 100$ (c) $Re = 400$

(d) $Re = 1000$ (e) $Re = 2000$ (f) $Re = 3200$

Figure 7.15: Flow in a square cavity. Computed pressure contours at different values of Re, using quadratic elements and Mesh B



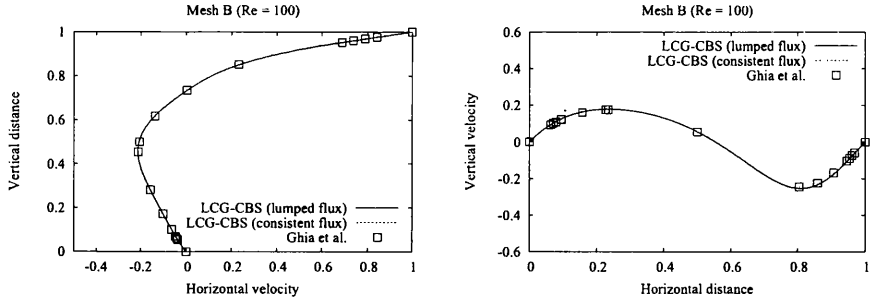
(a) $Re = 100$ (b) $Re = 100$

(c) $Re = 400$ (d) $Re = 400$

(e) $Re = 1000$ (f) $Re = 1000$

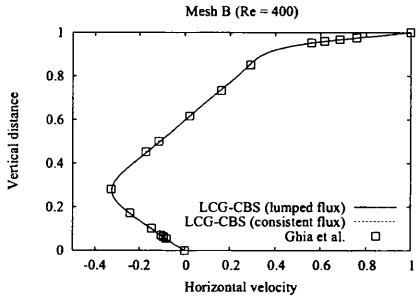
(g) $Re = 3200$ (h) $Re = 3200$

Figure 7.16: Flow in a square cavity. Comparison of the velocity components along the centrelines, with the benchmark data of Ghia *et al.* Results shown at different values of Re, using linear elements and Mesh A

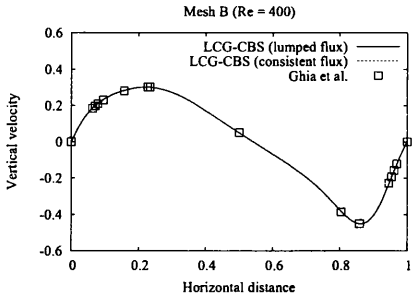


(a) $Re = 100$

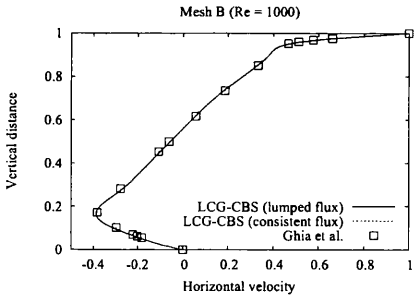
(b) $Re = 100$



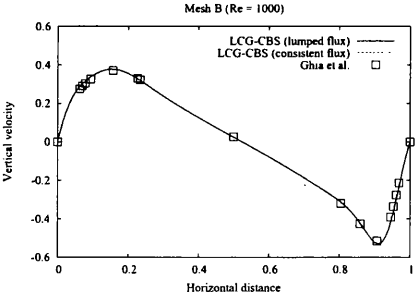
(c) $Re = 400$



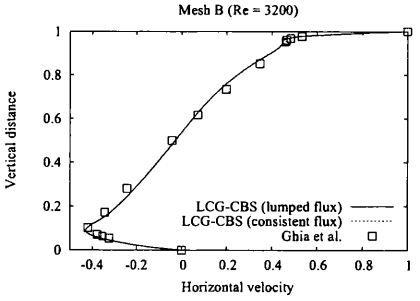
(d) $Re = 400$



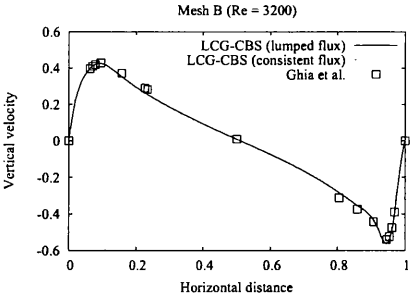
(e) $Re = 1000$



(f) $Re = 1000$

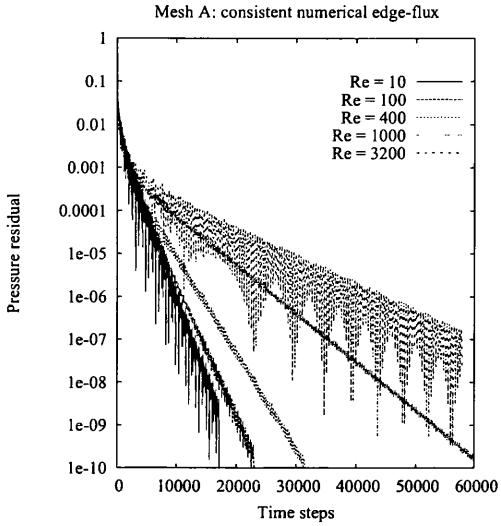


(g) $Re = 3200$

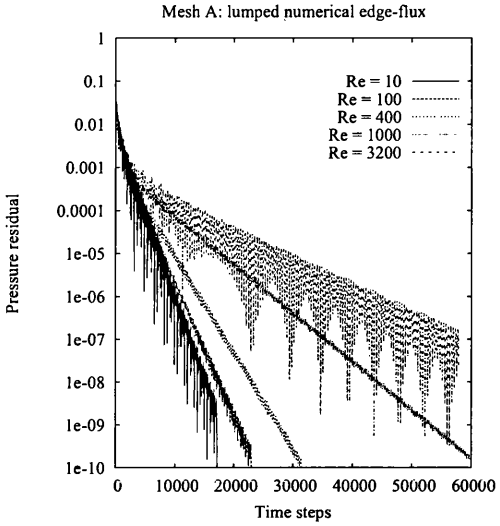


(h) $Re = 3200$

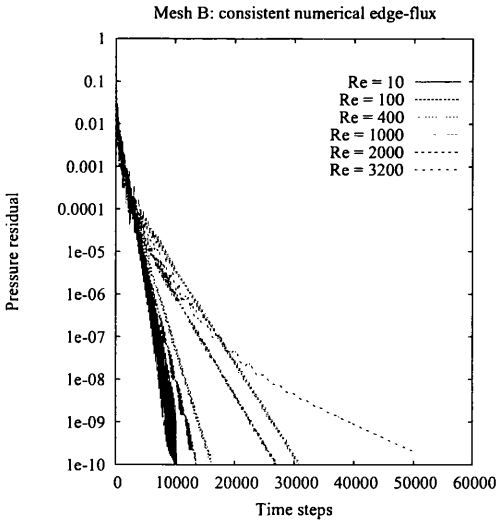
Figure 7.17: Flow in a square cavity. Comparison of the velocity components along the centrelines, with the benchmark data of Ghia *et al.* Results shown at different values of Re, using quadratic elements and Mesh B



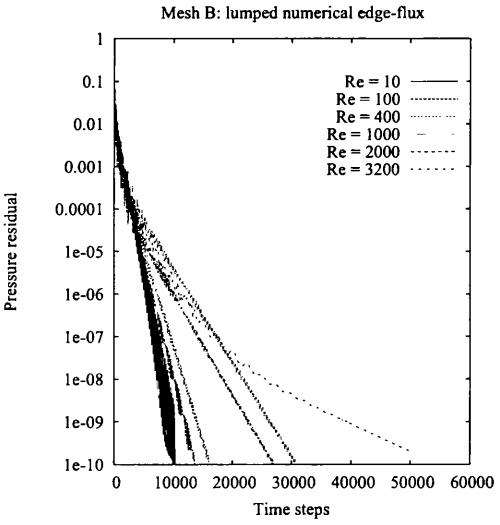
(a) Mesh A - consistent edge flux



(b) Mesh A - lumped edge flux



(c) Mesh B - consistent edge flux



(d) Mesh B - lumped edge flux

Figure 7.18: Flow in a square cavity. Convergence history of the pressure residual on each mesh, for the LCG-CBS schemes. Results given for lumped and consistent forms of the numerical edge-flux

Table 7.1: Flow in a square cavity. Comparison of CPU times for preprocessing and iterations. Results given for lumped and consistent forms of the numerical edge-flux.

Mesh A	Lumped-flux	Consistent-flux
$Re = 10$	1941.97s	2030.53s
$Re = 100$	2376.41s	2519.19s
$Re = 400$	3388.38s	3401.52s
$Re = 1000$	6489.52s	6615.44s
$Re = 2000$	7231.64s	7443.48s
$Re = 3200$	7547.97s	7759.12s
Mesh B	Lumped-flux	Consistent-flux
$Re = 10$	507.84s	571.25s
$Re = 100$	616.61s	1035.79s
$Re = 400$	757.23s	1180.95s
$Re = 1000$	1420.80s	2404.45s
$Re = 2000$	2122.17s	3211.40s
$Re = 3200$	2627.33s	3795.53s

the results on both meshes are in excellent agreement with each other.

The stream-traces at different values of Re, are given in Figure 7.12 for Mesh A, and Figure 7.13 for Mesh B. Both meshes also give good agreement here. For $Re = 100$ secondary vortices are shown at the bottom corners. As Re increases, these vortices grow larger. At $Re \geq 2000$, the flow is shown to have separated near the top left-hand corner. Here, a third vortex appears.

The pressure fields are shown in Figures 7.14 and 7.15. Generally, good results are given by both meshes for the range of Reynolds numbers considered. At $Re = 3200$ though, some minor oscillations occur at the top left corner. However, these oscillations are concentrated to just within the locality of the singularity and do not pollute the entire solution.

For a quantitative evaluation of the performance of the CBS-LCG schemes, the velocity components along the mid centrelines are compared with the benchmark solution by Ghia *et al.* [146]. The velocity distributions at various Reynolds numbers are given in Figures 7.16 and 7.17 for Mesh A and Mesh B respectively. At lower Re, the comparison with Ghia *et al.* for both Mesh A and Mesh B is excellent. At $Re = 3200$, a small amount

of deviation occurs.

During the computations, the effect of lumping the numerical edge-flux was considered. Implementing a lumped flux approximation, was discussed in Chapter 3. However its effect - bad or good, was not clear. Here, in Figures 7.16 and 7.17, it can be seen that lumping the edge-flux has a negligible effect on the computed solution. Furthermore, in Figure 7.18, the convergence histories - recorded on each mesh for both flux approximations, show that using a lumped edge-flux does not alter the convergence profiles for the the same input data.

Using a lumped edge-flux, does not therefore have any adverse effect on the solution procedure. It does however, have the advantage of requiring less computation. For linear elements the number of computations needed for the edge-flux are reduced from four to two and for quadratic elements the computations are reduced from nine to three. As shown in Table 7.1, using a lumped edge flux reduces calculation costs. The reduced cost, which for longer calculations, can lead to significant savings in CPU time.

7.3 Flow past a circular cylinder at low Reynolds numbers

In this section, the flow past a circular cylinder is solved for an incompressible and viscous fluid, at low values of Reynolds numbers. Here, the Reynolds number is based on the free-stream inlet velocity, U_∞, and the diameter of the cylinder, D. The results of physical experiments for this classic problem are well known, and have been reported by many authors [149, 150, 151, 129, 152]. Additionally, this problem is often used in numerical methods as a benchmark problem, and detailed computed results have been given for analysis.

In this example, test cases are performed using the LCG-CBS scheme for Reynolds numbers in the range: $1 \leq Re \leq 40$. Within this range, a steady flow occurs that is symmetrical with respect to the horizontal axis passing through the centre of the circle. At around $Re = 6$, the flow separates from the cylinder surface, and two symmetrical eddies are formed behind the cylinder. These eddies become progressively larger with further increases of Re, until they become unstable at around $Re = 40$. For $Re > 40$, periodic shedding of

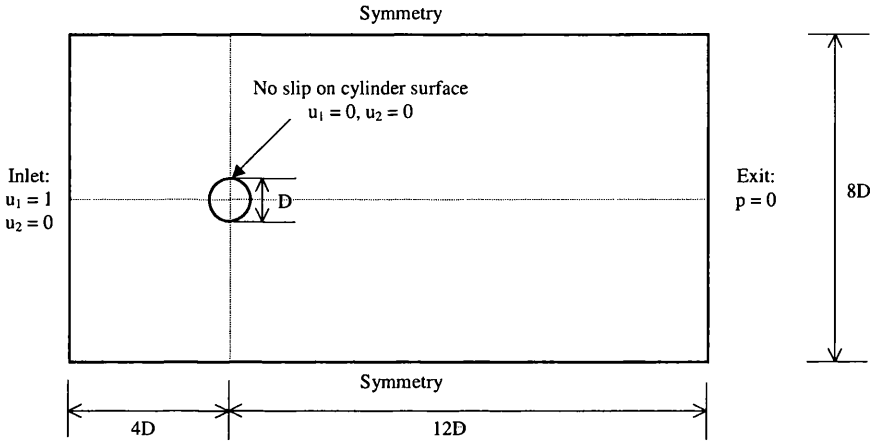
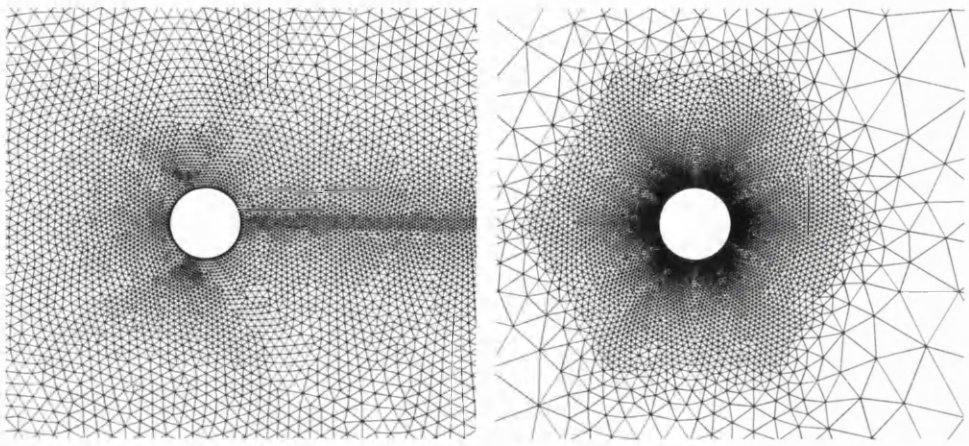


Figure 7.19: Steady flow past a circular cylinder. Geometry and boundary conditions

vortices occurs, and the flow becomes unsteady. A transient LCG-CBS scheme for solving unsteady flows, is tested for such vortex shedding problems in Chapter 8.

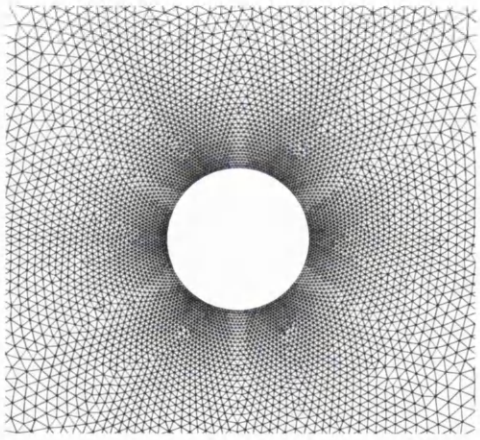
Figure 7.19 gives the problem statement and geometry. The computational domain is $16D$ in length and $8D$ in width. The centre of the cylinder is located at a distance of $4D$ from the inlet, along the centre line. As standard with incompressible subsonic flows, two boundaries conditions are required at the inlet and one boundary condition at the exit [132]. At the inlet boundary, the horizontal and vertical velocity components are prescribed as unity and zero respectively. At the exit, the pressure is set to zero. Slip conditions are imposed at the top and bottom extremities. The only solid surface in the domain is that of the cylinder and no-slip conditions are prescribed on its surface. At $t = 0$, the initial conditions everywhere are: horizontal velocity is unity, and the vertical velocity and pressure are both zero.

Two different unstructured meshes were used in this study. Details of each mesh, in the vicinity of the cylinder, are given in Figure 7.20. Mesh A (Figure 7.20(a)) is an unstructured grid of 20,960 elements. Although Mesh A has good mesh resolution throughout the domain, the mesh-size has been further refined in areas around the cylinder and in the wake region. Mesh B is shown in Figure 7.20(b). It is also an unstructured grid, but has only 12,223 elements. This mesh is coarse throughout the domain, apart from a small



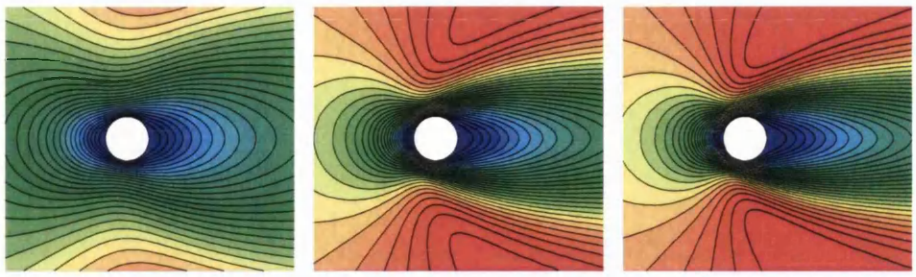
(a) Mesh A

(b) Mesh B

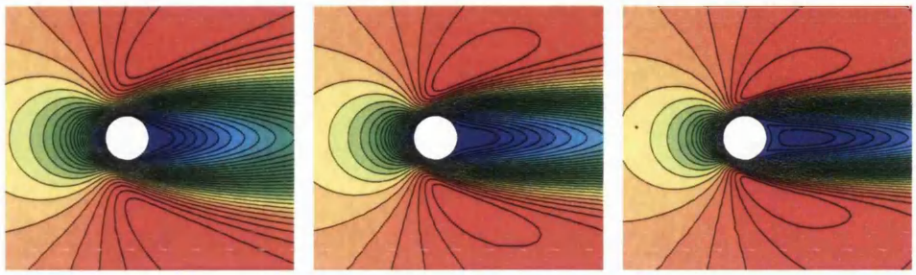


(c) Close up detail of Mesh B around cylinder

Figure 7.20: Steady flow past a circular cylinder. Close up detail of the elements in the vicinity of the cylinder on each of the unstructured meshes used.



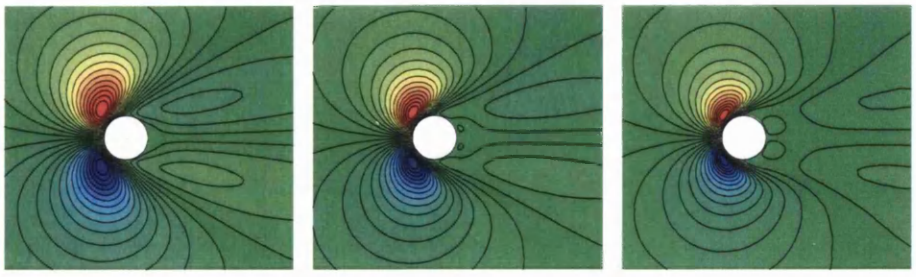
(a) u_1 velocity, $Re = 1$ (b) u_1 velocity, $Re = 5$ (c) u_1 velocity, $Re = 7$



(d) u_1 velocity, $Re = 10$ (e) u_1 velocity, $Re = 20$ (f) u_1 velocity, $Re = 40$

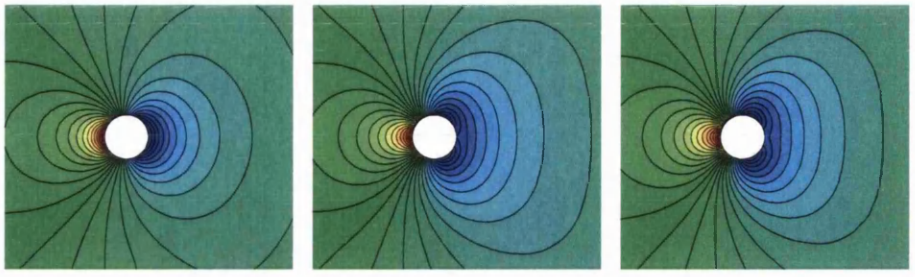
Figure 7.21: Steady flow past a circular cylinder. Contours of the horizontal velocity component at different values of Re, using Mesh A

(a) u_2 velocity, $Re = 1$ (b) u_2 velocity, $Re = 5$ (c) u_2 velocity, $Re = 7$

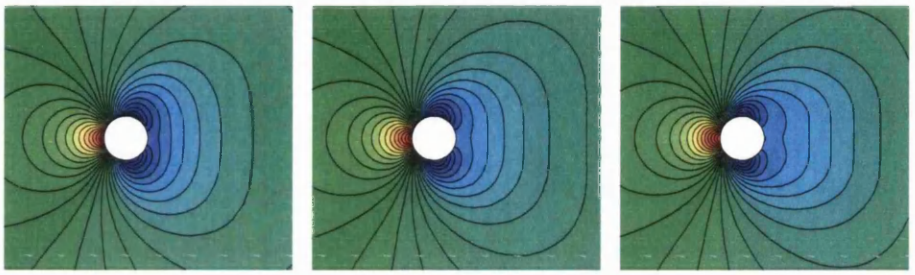


(d) u_2 velocity, $Re = 10$ (e) u_2 velocity, $Re = 20$ (f) u_2 velocity, $Re = 40$

Figure 7.22: Steady flow past a circular cylinder. Contours of the vertical velocity component at different values of Re, using Mesh A

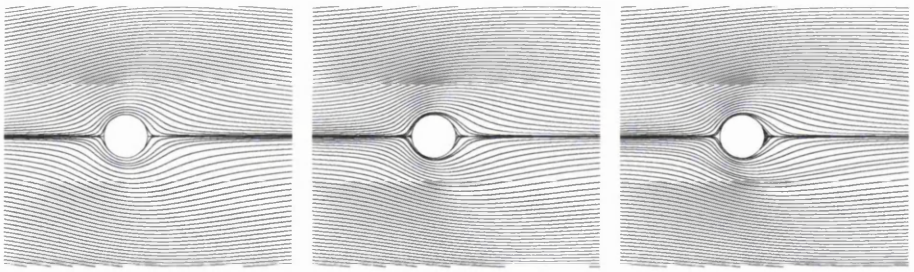


(a) $Re = 1$ (b) $Re = 5$ (c) $Re = 7$

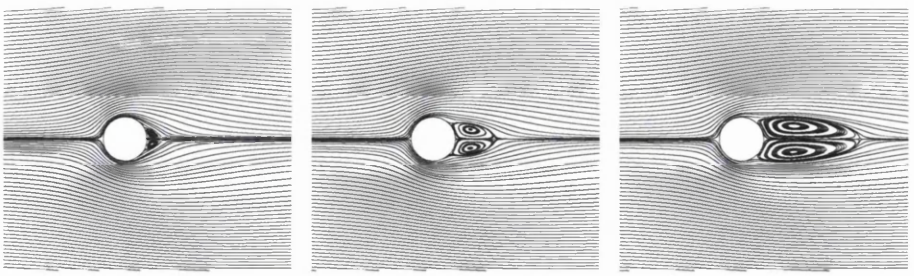


(d) $Re = 10$ (e) $Re = 20$ (f) $Re = 40$

Figure 7.23: Steady flow past a circular cylinder. Computed pressure contours at different values of Re, using Mesh A

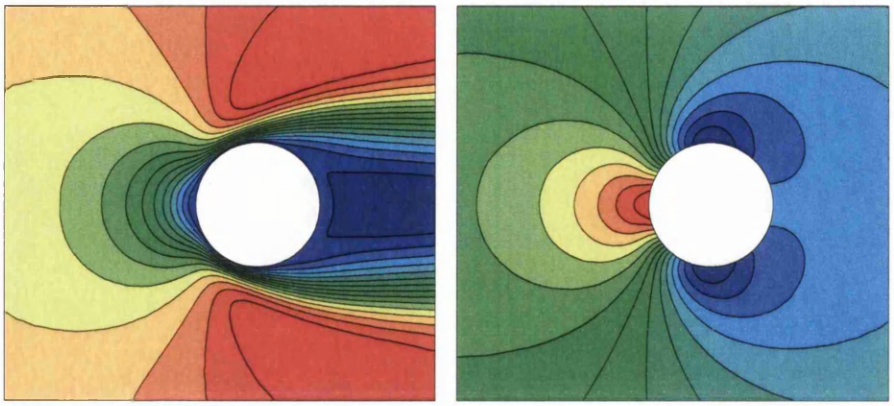


(a) $Re = 1$ (b) $Re = 5$ (c) $Re = 7$



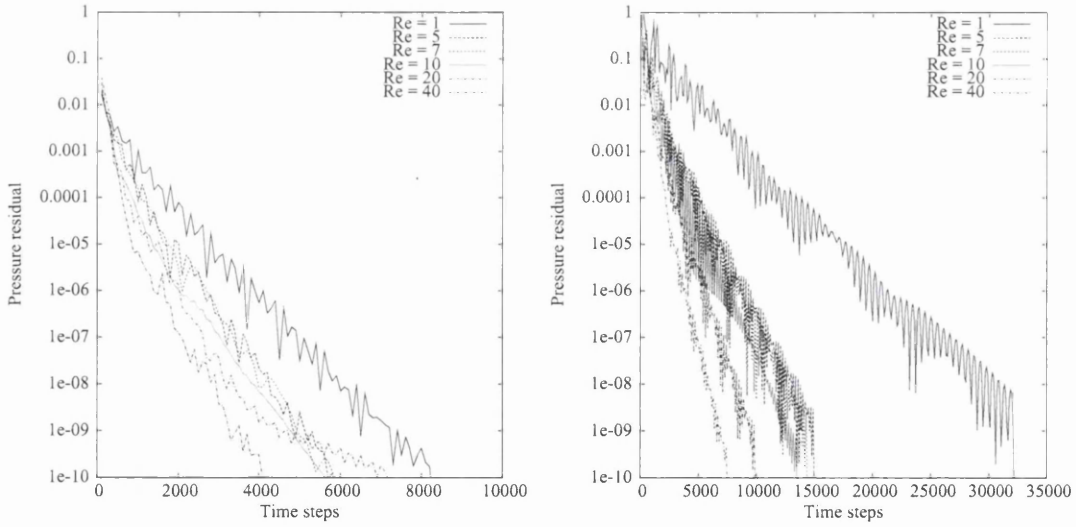
(d) $Re = 10$ (e) $Re = 20$ (f) $Re = 40$

Figure 7.24: Steady flow past a circular cylinder. Plotted stream-traces at different values of Re, using Mesh A



(a) u_1 Velocity (b) Pressure

Figure 7.25: Steady flow past a circular cylinder. Close up detail of the computed velocity and pressure contours on Mesh B, around the vicinity of the cylinder, at $Re = 40$.



(a) Mesh A (b) Mesh B

Figure 7.26: Steady flow past a circular cylinder. Convergence history of the pressure residual for the LCG-CBS scheme on both Mesh A and Mesh B

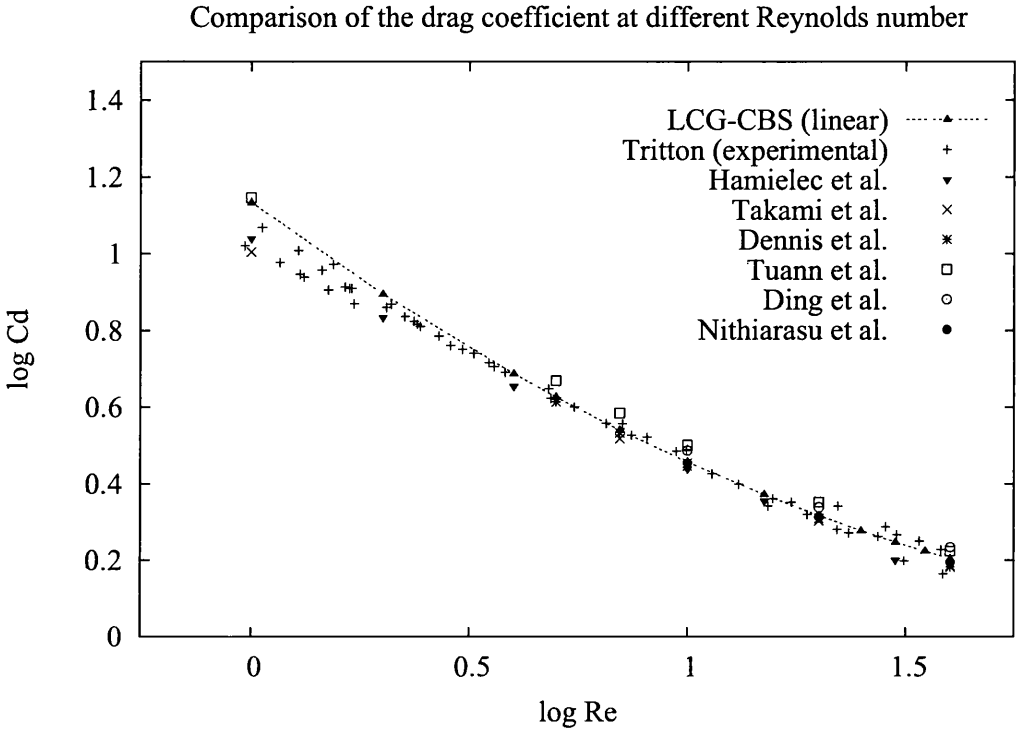


Figure 7.27: Steady flow past a circular cylinder at low Reynolds numbers. A Comparison of the computed drag coefficient at different values of *Re* using the LCG-CBS scheme on Mesh B, with numerical results and experimental data obtained in the literature.

Table 7.2: Laminar steady flow past a circular cylinder; CPU times for preprocessing and iterations for the pressure residual to reach a tolerance of 1 x 10^{-10}

Reynolds number	Mesh A	Mesh B
Re = 1	847.97s	2008.44s
Re = 5	891.44s	1656.30s
Re = 7	1096.75s	1585.95s
Re = 10	1055.44s	1490.47s
Re = 20	797.36s	1087.55s
Re = 40	1259.04s	841.51s

hexagonal sub-region that surrounds the cylinder boundary. The vast majority of elements are concentrated within this sub-region, where the element-size is greatly reduced to being considerably fine at the cylinder boundary (Figure 7.20(c)). Mesh B allows more accuracy in quantitatively computing the pressure and stresses at the cylinder surface, but at a cost of loss in contour quality outside the hexagonal sub-region.

The performance of the LCG-CBS scheme for solving this problem, is first made by examining the quality and symmetry of the solutions obtained on the unstructured meshes. Figures 7.21 through to 7.24, show the contours of the velocity components, pressure and stream-traces, at different Reynolds numbers on Mesh A. As can be seen, the computed solutions are excellent, and have good symmetrical properties with respect to the centreline of the domain. For comparison, the solution computed on Mesh B for the case of $Re = 40$, is shown in Figure 7.25. As can be seen, the results computed on this mesh are also highly symmetric and smooth.

Figure 7.23 gives the computed pressure for different values of Reynolds number, on Mesh A. As shown, the LCG-CBS scheme gives excellent pressure stabilisation, with the computed pressure contours being highly smooth and non-oscillatory. Closer examination of the stream-traces given in Figure 7.24 for Mesh A, show the flow is still attached to the cylinder at $Re = 5$. This is in good agreement with experimental data [149]. At $Re > 5$ the flow separates from the cylinder before reaching the rear stagnation point and a pair of eddies start to form in the wake behind the cylinder. These eddies become progressively larger as the Reynolds number is increased further. For $Re > 40$, the flow in the wake becomes

Table 7.3: Steady flow past a circular cylinder at low Reynolds numbers. Tabulated values of the computed drag coefficient (C_d) using the present LCG-CBS scheme on Mesh B and published results obtained from the literature for various Reynolds numbers up to $Re = 40$

Scheme	*Re* 1.0 C_d	*Re* 2.0 C_d	*Re* 4.0 C_d	*Re* 5.0 C_d	*Re* 7.0 C_d	*Re* 10.0 C_d
Hamielec *et al.* [153]	10.970	6.830	4.520	-	-	2.750
Takami *et al.* [154]	10.283	6.637	4.437	-	3.291	2.754
Dennis *et al.* [155]	-	-	-	4.116	3.421	2.846
Tuann *et al.* [156]	14.013	-	-	4.661	3.849	3.177
Ding *et al.* [157]	-	-	-	-	-	3.070
Nithiarasu *et al.*[108]	-	-	-	-	-	2.850
LCG-CBS (linear)	**13.604**	**7.854**	**4.870**	**4.236**	**3.477**	**2.870**
Scheme	*Re* 15.0 C_d	*Re* 20.0 C_d	*Re* 25.0 C_d	*Re* 30.0 C_d	*Re* 35.0 C_d	*Re* 40.0 C_d
Hamielec *et al.* [153]	2.270	-	-	1.588	-	-
Takami *et al.* [154]	2.266	2.003	-	1.717	-	1.536
Dennis *et al.* [155]	-	2.045	-	-	-	1.522
Tuann *et al.* [156]	-	2.253	-	-	-	1.675
Ding *et al.* [157]	-	2.180	-	-	-	1.713
Nithiarasu *et al.* [108]	-	2.060	-	-	-	1.564
LCG-CBS (linear)	**2.356**	**2.076**	**1.896**	**1.770**	**1.675**	**1.603**

unstable and a steady-state solution could not be obtained. A steady-state solution was considered when the pressure residual, given by Equation (3.97), reached a tolerance of 1×10^{-10}. In Figure 7.26, the convergence histories of the pressure residual are given on each mesh for different values of Reynolds number. Additionally the recorded CPU times are given in Table 7.2.

To quantitatively examine the accuracy of the LCG-CBS scheme, the coefficient of drag at different levels of Re has been computed for the cylinder using Mesh B. Table 7.3 gives the computed drag coefficient, c_d. As can been seen, comparisons of the computed c_d have been made with other numerical results for this problem [108, 153, 154, 155, 156, 157]. Furthermore, for convenience, the tabulated results have been plotted graphically with experimental data published by Tritton [149, 150], and are shown in Figure 7.27. Generally, the results are in excellent agrement with the other reported results.

7.4 Two-dimensional channel flow

In this section, problems of laminar incompressible-flow through two-dimensional channels are solved using the LCG-CBS scheme. The first problem considered is the development of Poiseuille flow through a rectangular channel. Although relatively simple, this problem is excellent in illustrating the schemes ability in modeling boundary layer flow - caused by the retardation of a viscous fluid next to (no-slip) solid walls. Additionally, if the channel is of sufficient length, then the fully-developed velocity profile can be compared with the analytical solution of Poiseuille. Also, the conservation of mass can be easily confirmed, by comparing the volume flow-rate at the exit with the inlet volume flow-rate.

The second channel flow problem considered in this section, is the flow over a downstream-facing step in a two-dimensional channel. This type of channel flow problem has been investigated extensively in the literature, both experimentally [158, 159, 160], and numerically [147, 92, 113, 114, 96, 97, 105, 57]. The problem involves regions of flow separation and subsequent reattachment, as well as a recirculating flow behind the step. This mixture of flow regimes make it a tough benchmark problem for validating numerical schemes. The expansion ratio chosen is 2:3, this seems to be the most popular choice within

the revelent literature, and experimental data is available for the case of $Re = 229$ [158]. Again a qualitative and quantitative assessment of the LCG-CBS scheme is given.

7.4.1 Flow through a two-dimensional rectangular channel

The first test-case considers the flow in a straight channel between two flat parallel walls. This classical problem is well known in fluid mechanics studies [150, 151, 129, 152, 161]. As the fluid passes into the channel from the inlet, it is immediately retarded in the vicinity of each wall, becoming zero at the wall-surfaces. As a result of this, a momentum boundary layer is formed on the walls. The boundary layer thickness grows as the fluid passes further downstream. To conserve volume-flow, a decrease of flow near the walls is compensated by a corresponding increase along the axis of the channel. Eventually at some point downstream, the two boundary layers merge into each other, and the velocity profile asymptotically transforms into the parabolic distribution of Poiseuille flow.

Steady-state solutions were solved by the LCG-CBS scheme, using both linear and quadratic elements. The Reynolds number for this problem is 100. The problem geometry and boundary conditions are given in Figure 7.28. The channel is $10L \times 1L$, where L is the width of the channel. For this problem type, the Reynolds number is based on both the inlet velocity and the channel width. At the inlet section, non-dimensional values of unity and zero were prescribed for the horizontal and vertical velocity components respectively. No-slip conditions are applied on both parallel walls. At the exit, a constant pressure value ($p = 0$) is applied. In order for the latter condition to be valid along the exit boundary, a sufficiently long domain is used. This is to allow the flow to reach a fully developed state well before the exit boundary, so that the pressure varies only in the horizontal direction [103].

Figure 7.29 gives a closeup view of the structured and unstructured meshes, used in the computations. Mesh A is a structured grid of 32000 linear elements. Mesh B is unstructured, and much coarser, with only 6720 linear elements. Finally, Mesh C is a structured grid of 8000 quadratic elements. Due to its higher-order elements, Mesh C offers the same nodal resolution as Mesh A.

The computed horizontal velocity contours, are shown in Figure 7.30. As can be

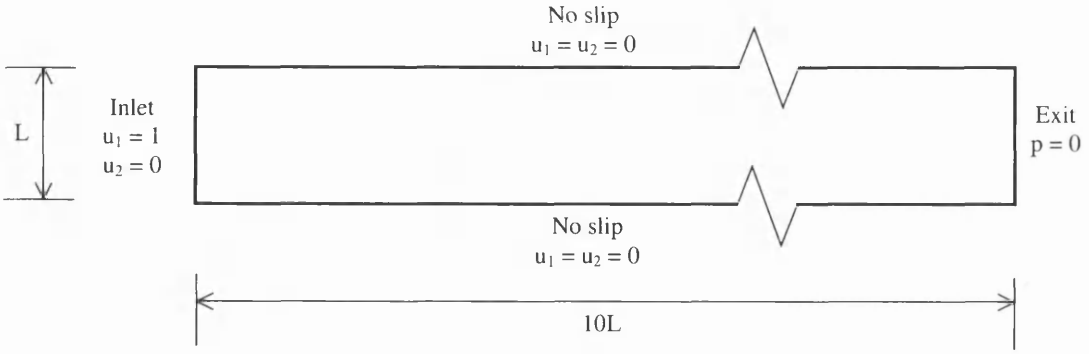
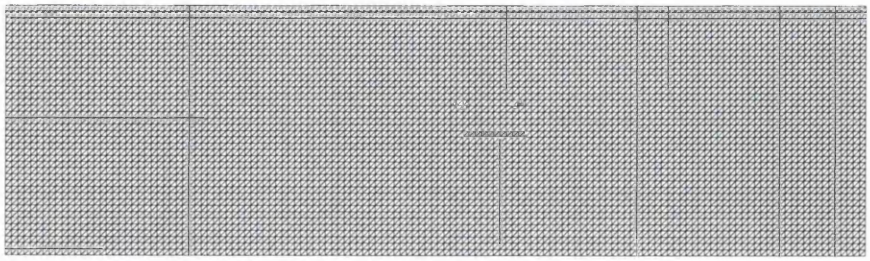
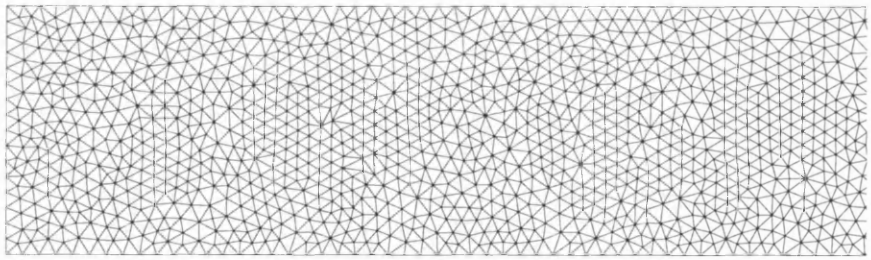


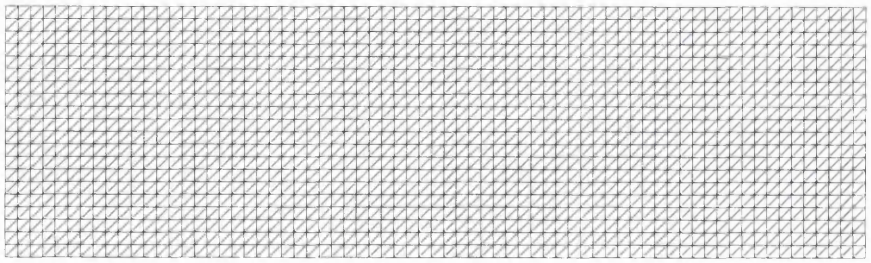
Figure 7.28: Laminar flow through a two-dimensional rectangular channel. Geometry and boundary conditions



(a) Mesh A



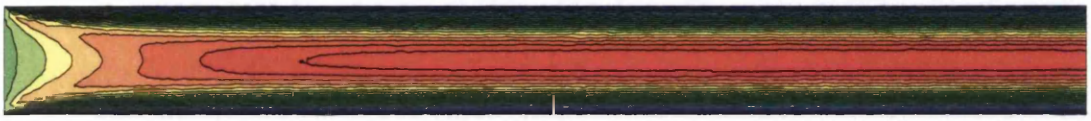
(b) Mesh B



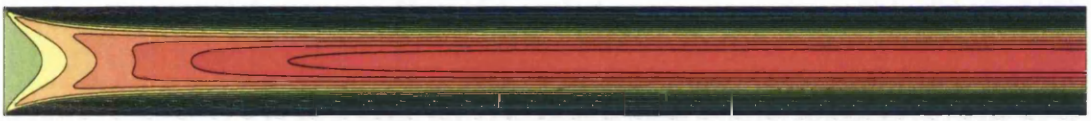
(c) Mesh C

Figure 7.29: Laminar flow through a two-dimensional rectangular channel. Closeup views of the structured and unstructured meshes, used in the computations.

(a) Mesh A



(b) Mesh B



(c) Mesh C

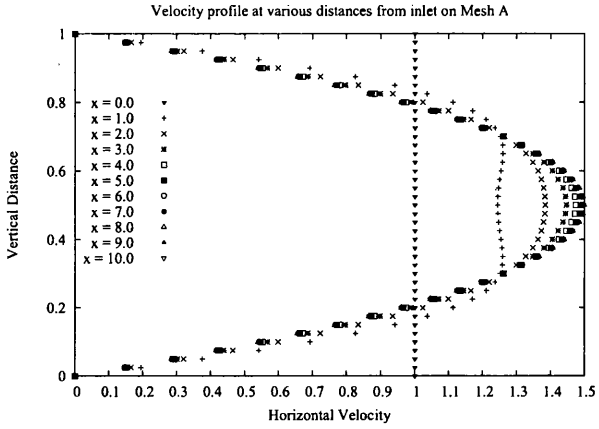
Figure 7.30: Laminar flow through a two-dimensional rectangular channel. Computed horizontal velocity contours.

(a) Mesh A

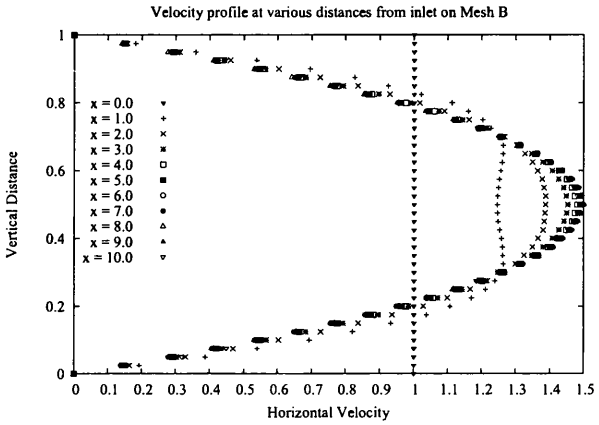
(b) Mesh B

(c) Mesh C

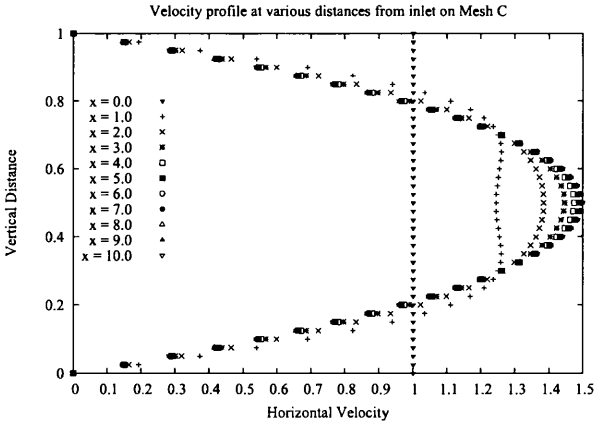
Figure 7.31: Laminar flow through a two-dimensional rectangular channel. Computed pressure contour distribution along the channel.



(a) Mesh A



(b) Mesh B



(c) Mesh C

Figure 7.32: Laminar flow through a two-dimensional rectangular channel. Comparison of velocity profiles, at various distances along the channel, for each mesh.

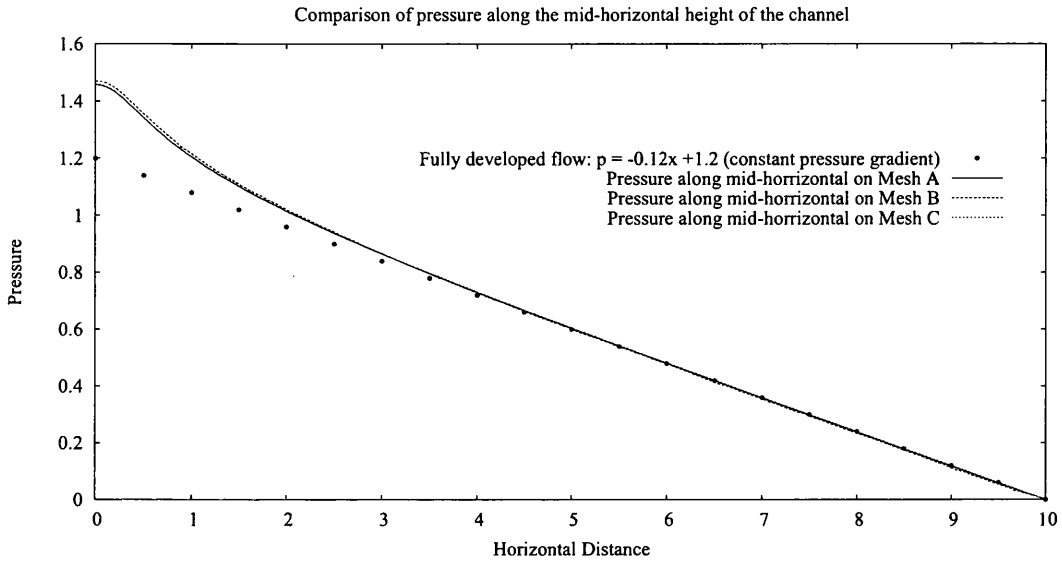


Figure 7.33: Laminar flow through a two-dimensional rectangular channel.

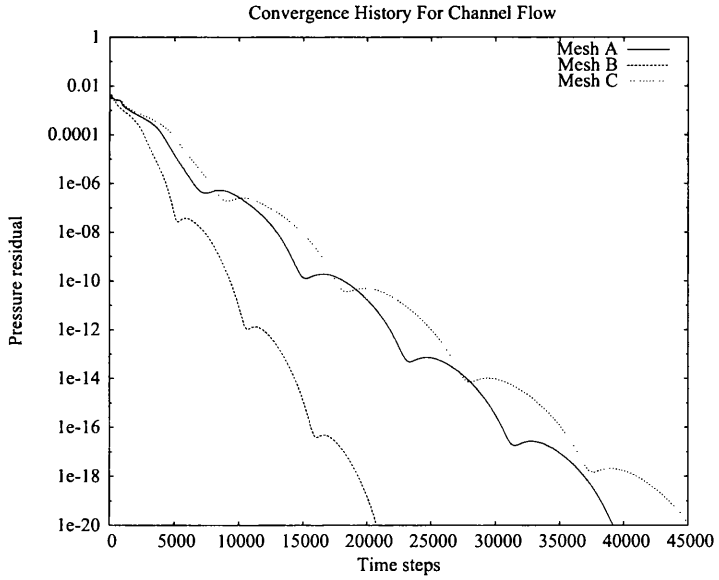


Figure 7.34: Laminar flow through a two-dimensional rectangular channel. Convergence history of the pressure residual to a tolerance of 1 x 10^{-20}, for the LCG-CBS scheme on each mesh

seen, the results on all meshes are in excellent agreement with each other. The results from Meshes A and C are almost identical to each other. The results obtained on the structured grids give the best quality, with high symmetry and smoothness. Mesh B, despite being relatively coarse and unstructured, still gives a good quality solution. Figure 7.31 gives the computed contours of pressure for each mesh. Again, the solutions obtained on Mesh A and Mesh C, are almost identical. The results on the unstructured mesh, Mesh B, are non-oscillatory, but are not as smooth as the structured meshes. The use of a finer unstructured mesh would give a much smoother pressure field.

A detailed analysis of the computed results of each mesh is carried out by examining the velocity profiles at various sections along the channel. Additionally, a comparison is made of the pressure distribution, computed on each mesh along the channel, with fluid mechanics theory. Figure 7.32 gives the the velocity profiles for each mesh, at $1 \times L$ intervals along the channel, from the inlet to the exit. As can clearly be seen, the solutions given by each of the three meshes, agree excellently with each other.

Initially at the start of the boundary layer formation, the flow's velocity profile is characterized by two peaks of velocity near the walls, and a lower velocity at the centre. This is explained by Panton [151], as being caused by the streamlines following a curved path as the fluid enters the channel. The pressure on the outside of a curved streamline is higher than the pressure on the inside of the streamline. From Bernoulli's equation, a low pressure implies high velocity and vice versa, giving the horned profile seen. This effect can last between $1 - 2$ channel widths downstream until the streamlines become parallel. At the exit boundary, the fully developed profile is parabolic in nature with a maximum velocity of $u_{max} = 1.500$ at the centre. Noting that the area of a parabola is two-thirds that of enclosing rectangle, it can be seen that the average velocity, u_{avg}, at the exit is unity, thus demonstrating that the proposed LCG-CBS scheme satisfies mass conservation for open systems.

In Figure 7.33, the pressure computed along the mid-horizontal has been plotted for each mesh. The computed pressures obtained on the linear and quadratic meshes are identical. The results obtained using Mesh B differ slightly to the results of Meshes A and C, near the inlet section. However, the solution is in excellent agreement with the structured

Table 7.4: Laminar flow through a two-dimensional rectangular channel. CPU times for preprocessing and iterations for the pressure residual to reach a tolerance of 1 x 10^{-20}

Mesh	CPU time
Mesh A	9911.02s
Mesh B	1038.17s
Mesh C	6719.09s

meshes after a distance of $2.5L$, with only tiny deviations - due to its coarse unstructured nature. Where the pressure gradient becomes linear, indicates the point at which a fully developed stage has been reached. The developing or entry length, l_e, i.e. the distance from the inlet for the flow to become fully developed, depends on the problem's Reynolds number. Schlichting [152] gives an approximate relation for the non-dimensional developing length as $l_e = 0.04L \times Re$, which gives $l_e = 4L$. A derived relation given by Massy [129], is more conservative and accurate with a relation of $l_e = 0.057L \times Re$, which for $Re = 100$, gives a l_e at a distance of 5.7L. For a fully developed flow, the non-dimensional pressure gradient is given as [161]

$$\frac{dp}{dx_1} = -\left(\frac{12 \times u_{avg}}{L^2 \times Re}\right) \tag{7.1}$$

For the convenience of the reader, the solution to Equation (7.1), for $Re = 100$, has been plotted in Figure 7.33. It is clear from examining Figure 7.33, that the computed pressures appear to become identical to the analytical solution, at $x_1 \geq 5.5L$, indicating a fully developed flow has been reached at this point. This is consistent with the relation of Massy [129].

Figure 7.34 gives the convergence histories for the pressure residual - computed on each mesh using Equation 3.97, to steady-state. In this example, iterations were carried out until an error tolerance of 1×10^{-20} was reached. The associated CPU times for each mesh are given in Table 7.4. It can be seen that the quadratic mesh - Mesh C, despite having the same number of nodes as Mesh A, takes a lower number of iterations and CPU time to reach the tolerance level.

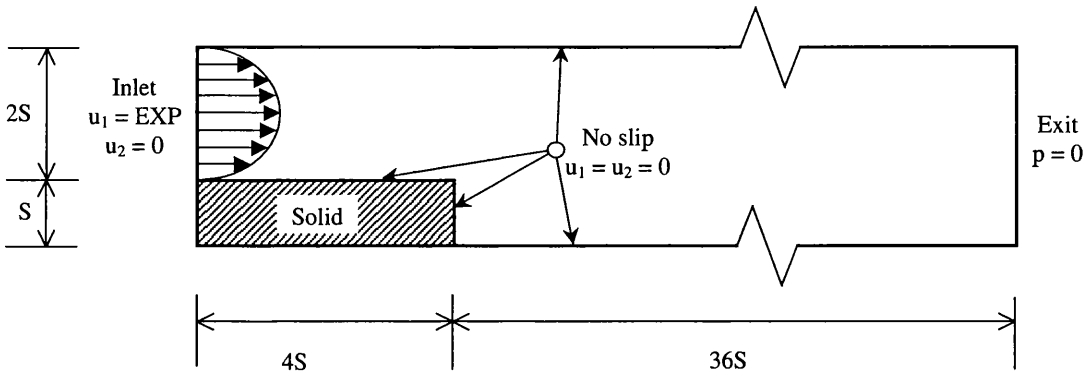


Figure 7.35: Laminar flow over a downstream-facing step in a channel, $Re = 229$. Problem definition and geometry

7.4.2 Laminar flow over a downstream-facing step in a channel

The next problem considered, is laminar flow though a channel with a sudden single-plane expansion. This problem is commonly referred to as flow over a backward facing step, due to an abrupt step that expands the channel-width, at some point along the channel in the stream-wise direction. This problem has been investigated by many authors, both experimentally [158, 159, 160], and as a benchmark test for testing numerical algorithms [147, 92, 113, 114, 96, 97, 105, 57]. Unlike the previous problem of flow in a straight rectangular channel, this problem exhibits a complex flow regime. The sudden change in geometry causes flow separation at the upstream step corner and subsequent flow reattachment on the channel-wall, further downstream. The flow separation is caused by an adverse pressure gradient, and gives rise to a vortex formation immediately behind the step face.

The specific problem modelled in this study is the flow past a downstream facing step, with an expansion ratio of 2:3. The problem definition and geometry is given in Figure 7.35. The step design is based on an original experiment conducted by Denham and Patrick [158], and thus allows a quantitative comparison to be made with the available experimental data. The inlet is situated at a distance $4S$ upstream of the step, where S is the step height as shown in Figure 7.35. The inlet section is twice as high as the step. The channel extends from the step a further distance of $36S$ downstream, giving the total length of the channel

to be $40S$. The Reynolds number for this case is $Re = 229$, and is based on the step height.

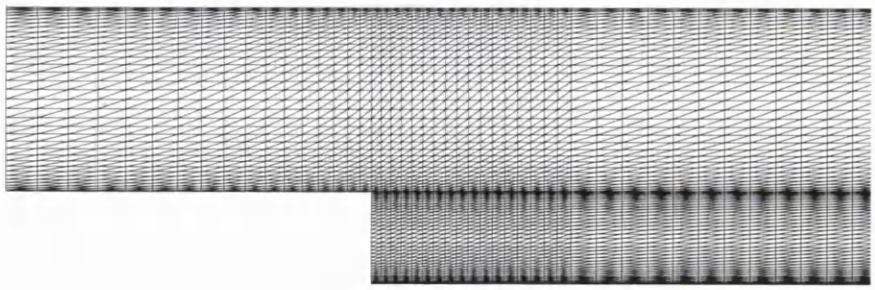
Apart form the inlet and exit boundaries, all other boundaries are solid walls, in which a no-slip boundary condition is applied. At the inlet, a velocity profile constructed from velocity measurements recorded by Denham and Patrick [158], was used for the horizontal velocity component. To allow the recorded velocities to be superimposed on an arbitrary mesh, a sixth-order polynomial curve was created to fit the experimental data:

$$u_1 = 0.6624y^6 - 7.5547y^5 + 33.900y^4 - 75.283y^3 + 83.368y^2 - 37.793y + 2.6959 \qquad (7.2)$$

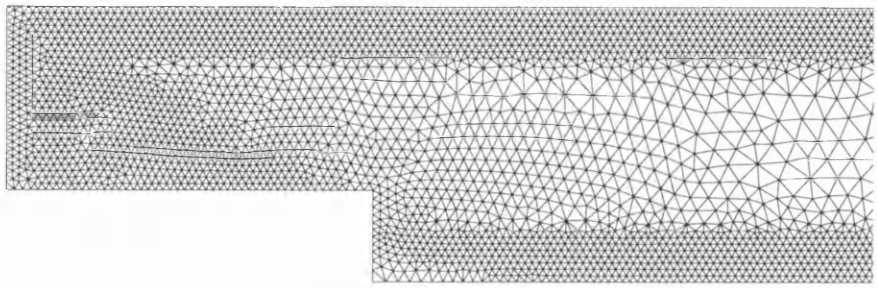
It was important to match the experimental data as closely as possible, as even the smallest differences here, will be amplified further down stream. However, even with curve of this order, it was not possible to match the profile exactly with the recorded values, and some inherent inaccuracy will remain. At all times, a vertical velocity of zero was assumed at the inlet. At the exit boundary, a value of zero was prescribed for the pressure. The length of the channel is sufficient for the disturbance, created by the recirculation zone in the vicinity of the step, to be stabilised by the time the flow reaches the exit, making this exit boundary condition valid. All computations are started with initial conditions of u_1 equal to unity and u_2 and p equal to zero, at all points inside the domain.

A number of structured and unstructured meshes were used in this study. Close-up images of each mesh, in the vicinity of the step, are shown in Figure 7.36. Mesh A is a non-uniform structured mesh of 8092 linear elements. Meshes B and C are unstructured grids of 22257 and 47359 linear elements respectively. Mesh B is refined near the boundaries, but becomes very coarse away from the walls. Mesh C offers better resolution throughout the domain with the elements being rather uniform in the channel section downstream of the step. Mesh D is uniformly structured throughout, and is made up of only 116 quadratic elements. The aspect ratio of all the elements in this grid is 1:2, with the smallest side of the element being equal to the step height. Although too coarse to give an accurate solution, it was interesting to see how much detail can be picked up from a mesh of this resolution, when quadratic elements are used.

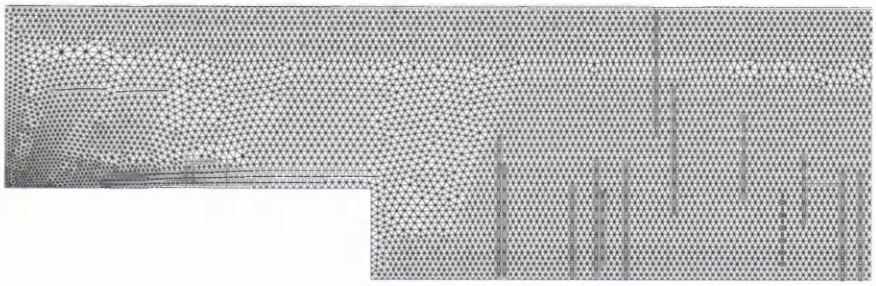
Figures 7.37 and 7.38 give the computed solutions, for the horizontal velocity and



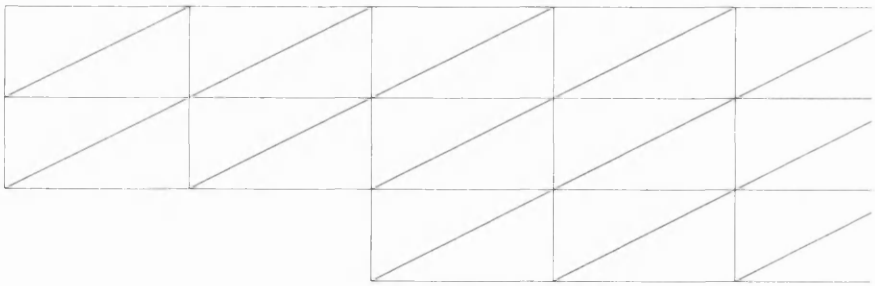
(a) Mesh A



(b) Mesh B

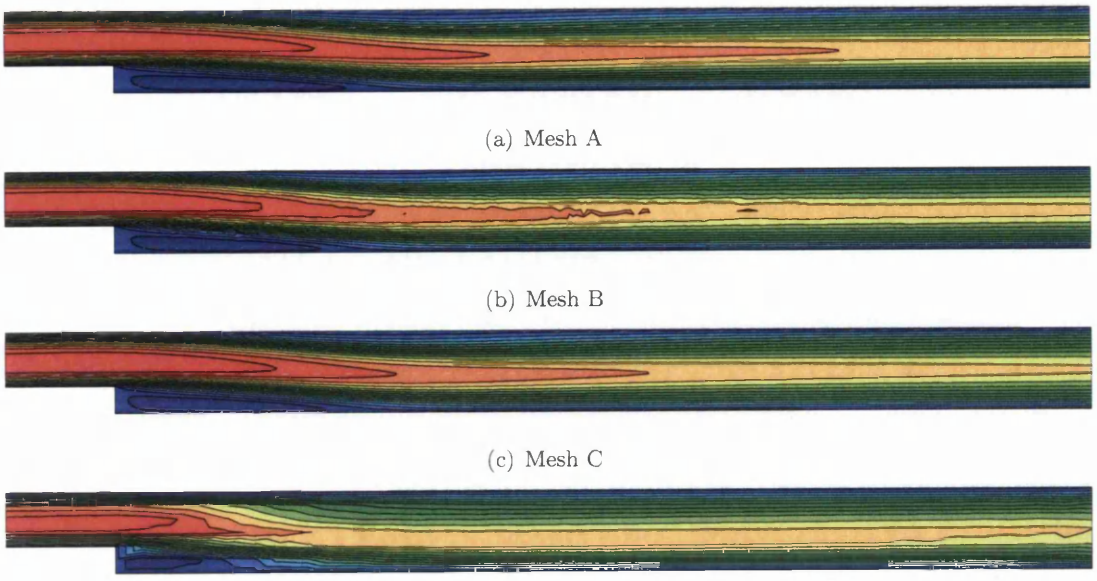


(c) Mesh C



(d) Mesh D

Figure 7.36: Laminar flow over a downstream-facing step in a channel, $Re = 229$. Structured and unstructured meshes used in the computations.



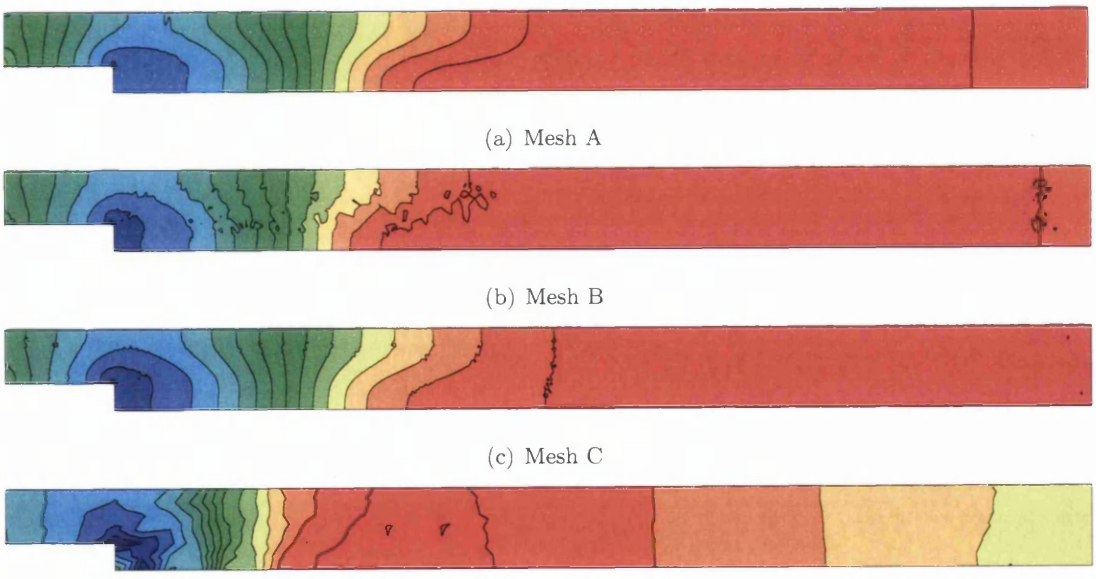
(a) Mesh A

(b) Mesh B

(c) Mesh C

(d) Mesh D

Figure 7.37: Laminar flow over a downstream-facing step in a channel, $Re = 229$. Computed contours of the horizontal velocity component.



(a) Mesh A

(b) Mesh B

(c) Mesh C

(d) Mesh D

Figure 7.38: Laminar flow over a downstream-facing step in a channel, $Re = 229$. Computed pressure contours obtained for each mesh.

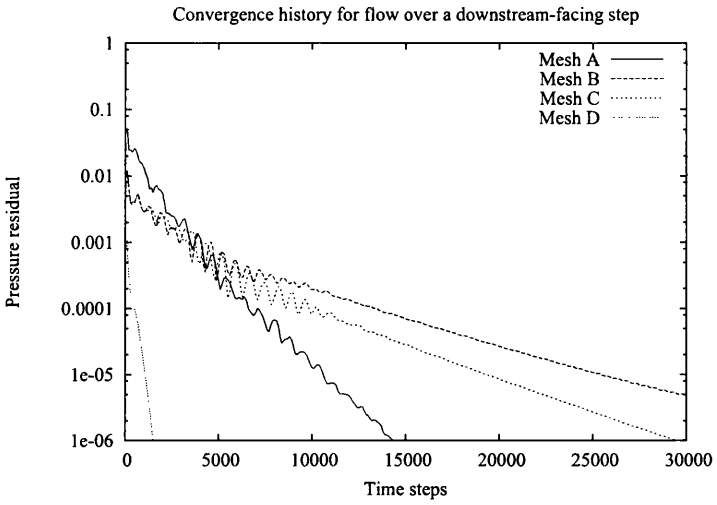
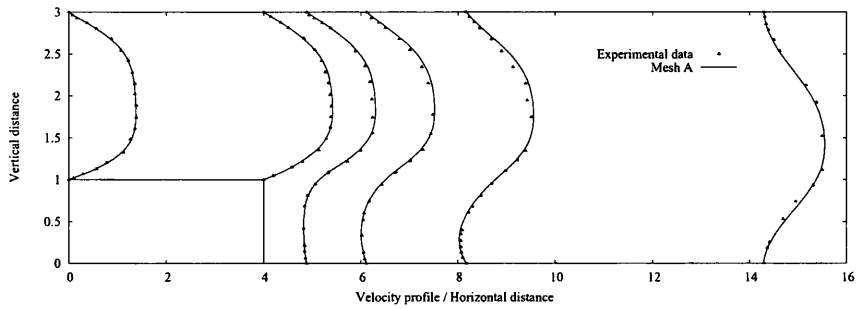


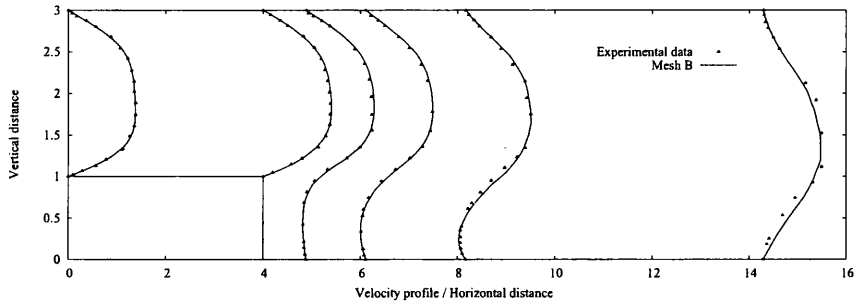
Figure 7.39: Laminar flow over a downstream-facing step in a channel, $Re = 229$. Comparison of the pressure residual convergence histories, on each mesh.

pressure field, on each mesh. As can be seen in Figure 7.37, a similar flow pattern is obtained on each of the meshes. All solutions show the flow separation and subsequent reattachment downstream. It is seen in Figure 7.38 that the best solutions were computed by Mesh A and Mesh C, with the former mesh producing the smoothest pressure contours. A good solution was also computed on Mesh B, however the contours produced are not as smooth. The pressure field is also marked with minor oscillations in regions where the element-size is large. Good results, were also obtained from Mesh D. Despite having a size of only 3×40 elements, the major features of this problem were predicted. The flow is shown to separate at the step corner and a recirculating flow is captured in the vicinity behind the step. The pressure singularity is also captured at the step corner.

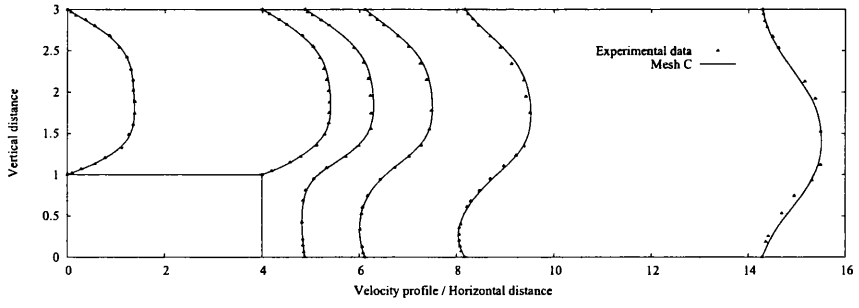
Figure 7.39 gives the pressure residual convergence, for each mesh. A steady-state tolerance of 1×10^{-6}, was used in order for the time-stepping to terminate. The fastest solution retrieved, was in under 2000 iterations by Mesh D. Mesh A was the next fastest, with a converged solution obtained after 15000 time-steps. Both unstructured meshes had the slowest convergence rates, with Mesh C taking twice as long as Mesh A to reach the same level of convergence, and Mesh B even longer.



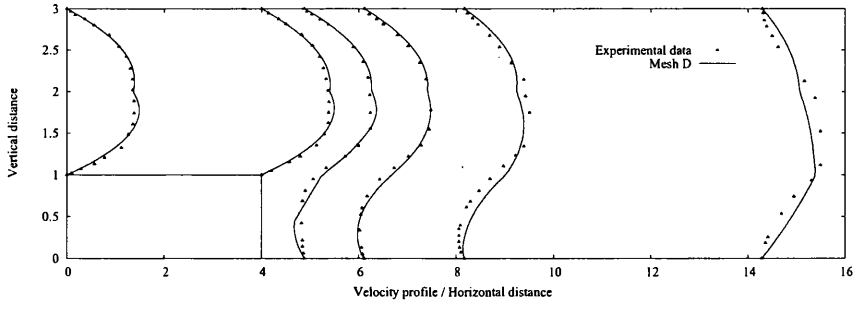
(a) Mesh A



(b) Mesh B



(c) Mesh C



(d) Mesh D

Figure 7.40: Laminar flow over a downstream-facing step in a channel, $Re = 229$. Comparison of computed velocity profiles with experimental data.

To quantify the accuracy of the LCG-CBS scheme for solving this problem, the velocity profiles have been computed at appropriate sections of the geometry, and compared with the experimental data of Denham and Patrick [158]. Figure 7.40 gives the velocity profiles computed on each mesh. As expected the linear structured grid (Mesh A) gave the best accuracy. The two unstructured meshes also gave comparable accuracy, with only slight differences further downstream. As already mentioned, good performance was found from Mesh D. It can be seen here, despite having oversized elements, the reverse flow behind the step is captured. Clearly, the use of quadratic elements is advantageous for this problem type - where the flow is parabolic. It is shown here, that the linear and quadratic element versions of the LCG-CBS schemes give good performance for open systems, with stable computed pressure and mass conservation across open boundaries being satisfied.

7.5 Summary

The accuracy of CBS-LCG scheme has been evaluated in this chapter, for a number of well known incompressible flow problems. The method has been shown to be robust on both structured and unstructured grids.

Flow in a square cavity was solved first and results showed good performance for this type of problem. Here, the use of a lumped numerical edge-flux was investigated for linear and quadratic basis functions and shown to be efficient.

The benchmark case of flow past a cylinder was tested next on unstructured grids. It also gave excellent accuracy. Quantitative data on the drag coefficient was shown to be highly accurate.

The last section considering the CBS-LCG scheme, was to solve channel flow problems. Accurate results were obtained here, for the development of Poiseuille flow and the flow past a backward facing step. The computed results on linear and quadratic elements compared well to both analytical and experimental data. In conclusion, it has been shown that the CBS-LCG scheme is a robust approach for solving incompressible flow problems.

Chapter 8

Extension of the LCG Method for Solving Transient Problems

8.1 Introduction

This chapter presents an evaluation of characteristic based LCG methods for solving transient problems. Two different schemes are investigated. The first method is considered in Section 8.2. It is the explicit characteristic LCG scheme for hyperbolic (purely-convective) scalar transport. Here the LCG method is used to solve 2D pure convection of a product-cosine hill in a circulating flow [23, 38, 41]. In the absence of natural diffusion, the governing equation type is hyperbolic and describes a propagation phenomenon. The characteristic based LCG method used gives a LCG equivalent to the Lax-Wendroff method [40].

Section 8.3 presents an evaluation of the characteristic based split, locally conservative Galerkin (CBS-LCG) method for solving unsteady flow problems. In Chapter 2, a dual time-stepping procedure [23, 76, 96, 97, 98, 113, 114, 115] was implemented for the CBS-LCG scheme. This procedure essentially transforms an unsteady flow calculation into a series of several instantaneous steady-states in real-time. A pseudo time-step is used to iterate the solution within each instantaneous steady-state until the solution converges to a desired level of tolerance. In order to recover the true transient solution, a real time

term was added to the momentum equation of the CBS temporal scheme, before being spatially discretised by the LCG scheme. By incorporating dual time-stepping, the error in the transient solution, which is introduced by both lumping the mass matrix and using the artificial compressibility parameter, is considerably reduced. This chapter evaluates the performance of the dual time-stepping approach in the CBS-LCG method for unsteady flows. The problem considered, is unsteady flow past a circular cylinder at $Re = 100$. At this Reynolds number, a complex and time-dependent flow pattern is found with periodic vortex shedding occurring behind the cylinder.

8.2 The rotating cone problem

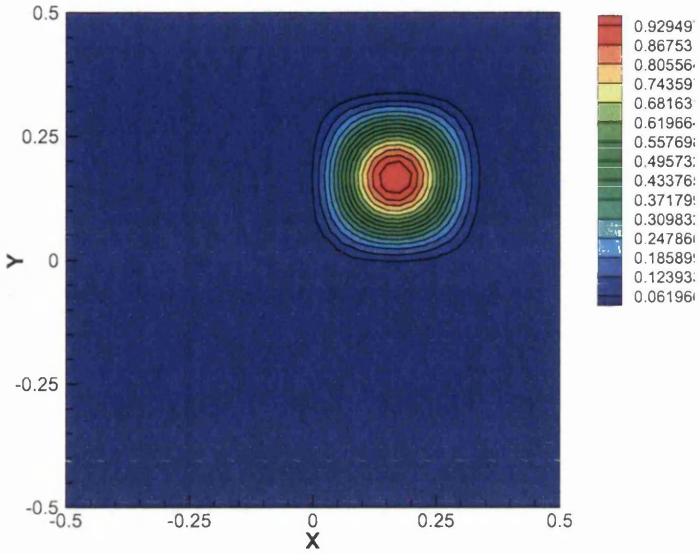
In Chapter 3, an explicit characteristic based LCG method was described for general convection-diffusion problems. The same methodologies also apply to pure-hyperbolic problems. In this case, there is no need to evaluate a diffusive edge-flux component as the problem is purely convective. Thus the edge-flux is always evaluated at the edge-nodes, even for linear elements.

In this section, the explicit characteristic based LCG method is evaluated for solving two-dimensional problems of pure convection. The governing equation for this problem is still described by Equation (3.1). However, for the purely convective transport of scalar variable ϕ, the flux term, F_i, is given as

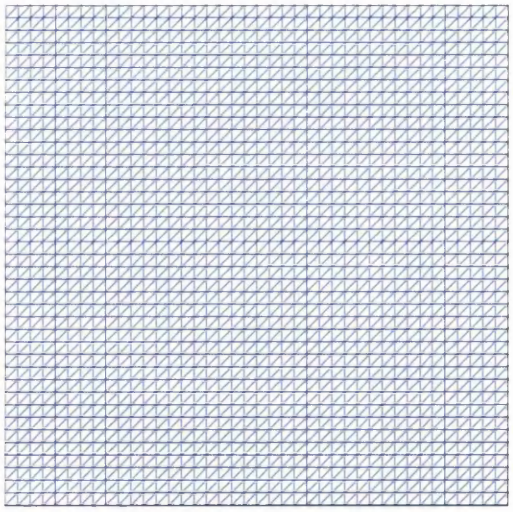
$$F_i = (u_i \phi) \tag{8.1}$$

where u_i are the velocity components.

The specific problem studied here, is the convection of a product-cosine hill in a purely rotational velocity-field. This classical problem is frequently used to test two-dimensional convection schemes [23, 38, 41]. Here, the problem statement described by Donea *et al* [41] is used. The problem domain is unit square with the origin located at the geometric centre. The initial data for this problem is



(a) Initial configuration



(b) 40×40 triangular finite element mesh

Figure 8.1: Convection of a cosine hill in a pure rotation field. Details of the problem statement and structured mesh used.

$$\begin{aligned}\phi(\mathbf{x},0) &= \frac{1}{4}\left(1+cos\pi X_1\right)\left(1+cos\pi X_2\right) \quad if \quad {X_1}^2+{X_2}^2 \leq 1 \\ \phi(\mathbf{x},0) &= 0 \quad if \quad {X_1}^2+{X_2}^2 > 1 \end{aligned} \tag{8.2}$$

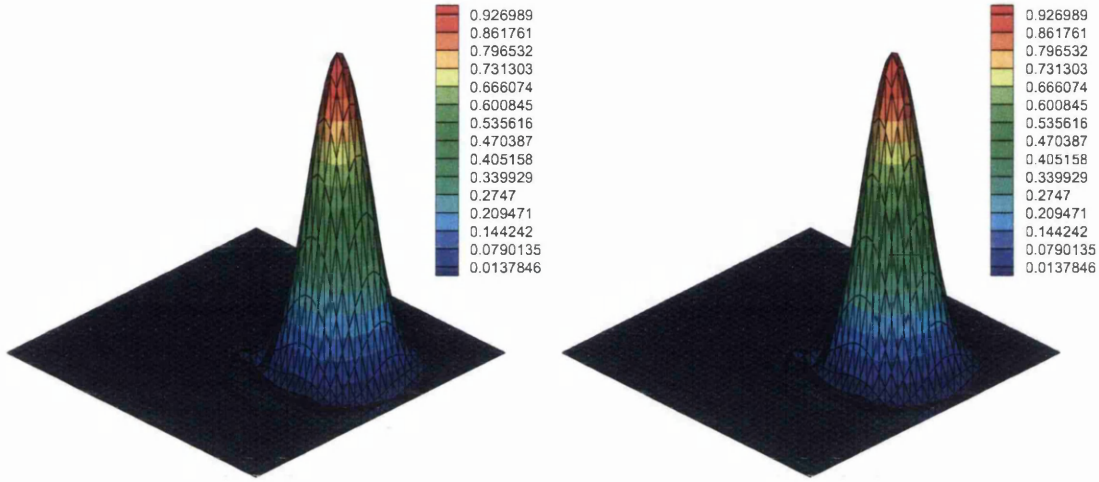
where $\mathbf{X} = (\mathbf{x}-\mathbf{x}_0)/\sigma$.

The initial position of the centre and the radius of the cosine hill are $\mathbf{x}_0$ and σ, respectively. In the examples they are chosen as $\mathbf{x}_0 = (\frac{1}{6}, \frac{1}{6})$ and $\sigma = 0.2$. The convection field is a purely rotational with unit angular velocity, thus $\mathbf{u}_0 = (-x_2, x_1)$. On the boundary Γ_ϕ, $\phi = 0$ for all t. A uniform 40×40 structured mesh of 3200 elements is used in the calculations. The original configuration and mesh used are shown in Figure 8.1.

As mentioned, the explicit characteristic based LCG method was used for this problem. A lumped-mass matrix was employed, giving a LCG equivalent of the Lax-Wendroff method. A time-step of $\triangle t = 2\pi/200$ was used and solutions were computed at: $t = 50$, 100, 150, and 200. The results for each time interval are given in Figures 8.2, 8.3, 8.4, and 8.5 respectively. For comparison, results obtained using the equivalent global Galerkin method are also given for each time interval. Additionally, the maximum and minimum values of ϕ computed, have been given in Table 8.1. As can be seen the global Galerkin and LCG results are identical at each time interval. This is quantitatively confirmed in Figure 8.6. Here the computed scalar variable along $(x, 1/6)$ and $(1/6, y)$ have been plotted for each method. As shown the LCG and global Galerkin methods give identical solutions.

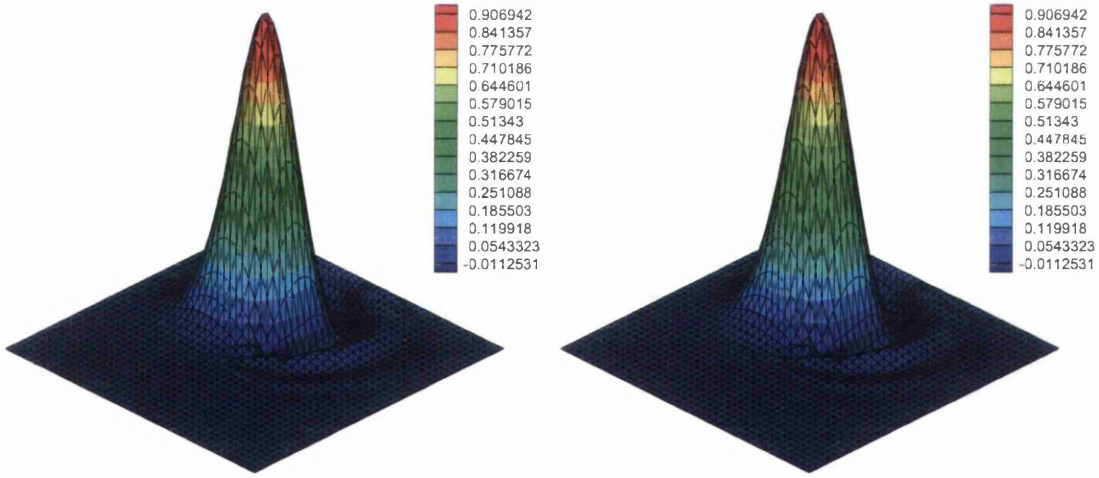
8.3 Unsteady flow past a circular cylinder

In Chapter 7, the problem of viscous steady-flow past a circular cylinder was solved for an incompressible fluid. Solutions were obtained for Reynolds numbers in the range: $1.0 \leq Re \leq 40.0$. Since all flows within this flow regime possessed a steady-state solution, the fully explicit LCG-CBS scheme (with an artificial compressibility parameter) was suitable for its investigation. It was shown that excellent performance and solution accuracy was achieved, when the flow is steady.



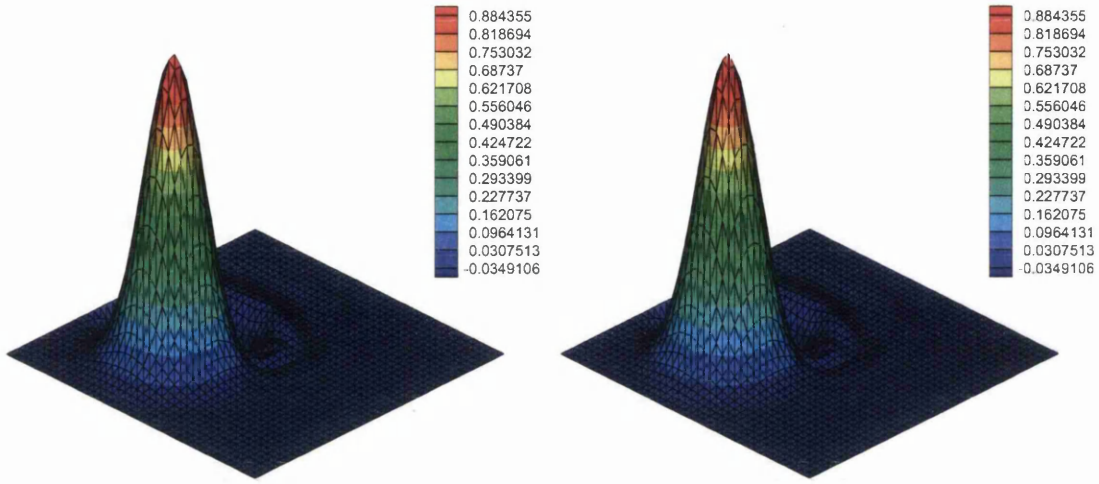
(a) LCG (b) global Galerkin

Figure 8.2: Convection of a cosine hill in a pure rotation field. Comparison of computed solutions for the characteristic based LCG and global Galerkin methods at $t = 50$



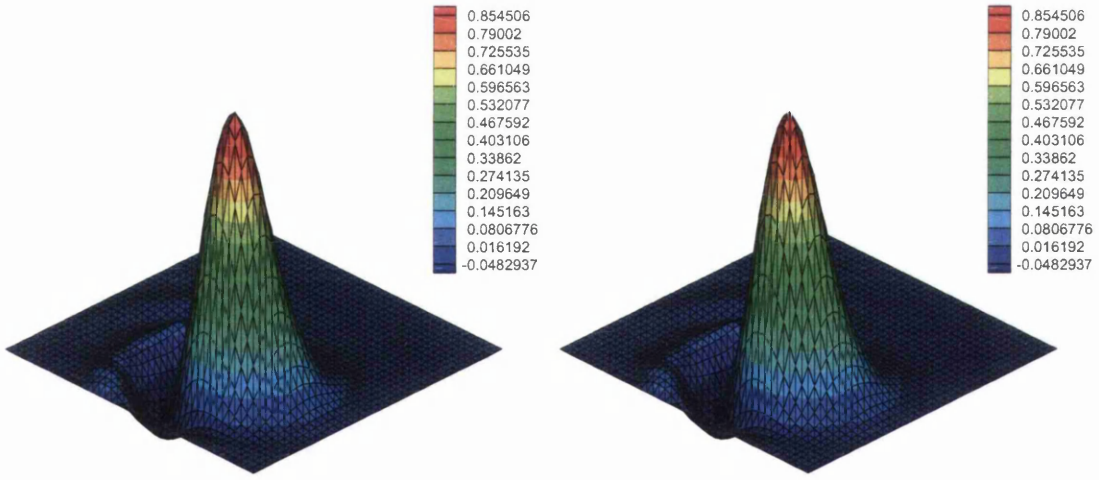
(a) LCG (b) global Galerkin

Figure 8.3: Convection of a cosine hill in a pure rotation field. Comparison of computed solutions for the characteristic based LCG and global Galerkin methods at $t = 100$



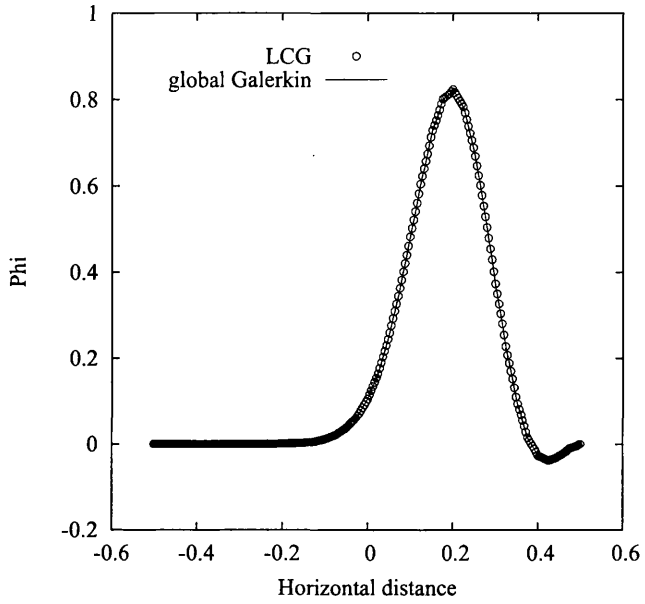
(a) LCG (b) global Galerkin

Figure 8.4: Convection of a cosine hill in a pure rotation field. Comparison of computed solutions for the characteristic based LCG and global Galerkin methods at $t = 150$

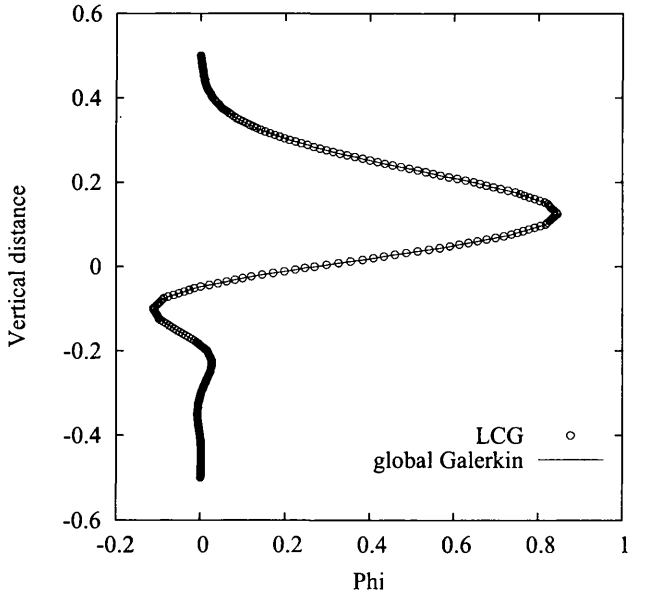


(a) LCG (b) global Galerkin

Figure 8.5: Convection of a cosine hill in a pure rotation field. Comparison of computed solutions for the characteristic based LCG and global Galerkin methods at $t = 200$



(a) Plot of computed scaler field ϕ along $-1/2 \leq x \leq 1/2$ at $y = 1/6$, for the LCG and global Galerkin methods



(b) Plot of computed scaler field ϕ along $-1/2 \leq y \leq 1/2$ at $x = 1/6$, for the LCG and global Galerkin methods

Figure 8.6: Convection of a cosine hill in a pure rotation field. Comparisons of the computed scaler field ϕ with the exact solution, for the explicit lumped-mass, LCG and global Galerkin methods

Table 8.1: Convection of a cosine hill in a pure rotation field. Comparison of computed maximum and minimum values of ϕ at various time intervals, for both LCG and global Galerkin methods.

t	$\phi_{max/min}$	LCG	global Galerkin
$t = 50$	ϕ_{max}	0.992218	0.992218
$t = 50$	ϕ_{min}	-0.051444	-0.051444
$t = 100$	ϕ_{max}	0.972528	0.972528
$t = 100$	ϕ_{min}	-0.076839	-0.076839
$t = 150$	ϕ_{max}	0.950017	0.950017
$t = 150$	ϕ_{min}	-0.100572	-0.100572
$t = 200$	ϕ_{max}	0.918992	0.918992
$t = 200$	ϕ_{min}	-0.112779	-0.112779

In this section the analysis is extended for unsteady flows with $Re > 40$. As already discussed in Chapter 3, the LCG-CBS scheme in its fully explicit nature is only suitable for steady-state solutions. However, the true transient solution is recovered here by using a dual time-stepping method [23, 76, 96, 97, 98, 113, 114, 115], which is incorporated into the LCG-CBS scheme.

The ability of the fully explicit LCG-CBS scheme (with dual time-stepping) to simulate transient flow is illustrated here by computing the vortex shedding in the wake of flow past a circular cylinder at $Re = 100$. This has been a popular test case for validating the transient part of numerical schemes [23, 96, 97, 108, 114, 148]. The problem definition is standard and Figure 8.7 gives the problem statement and geometry. The computational domain is $16D$ in length and $8D$ in width, the centre of the cylinder is located at a distance of $4D$ from the inlet, along the centre line. As before, for the steady flow problems, the Reynolds number is based on the free-stream inlet velocity, U_∞, and the diameter of the cylinder, D. At the inlet boundary, the horizontal and vertical velocity components are respectively prescribed as unity and zero. At the exit, the pressure is also set to zero. The top and bottom extremities are treated as slip walls. The only solid surface in the domain is that of the cylinder, here no-slip conditions are prescribed. At $t = 0$, the initial conditions everywhere are: horizontal velocity is unity, and the vertical velocity and pressure are both

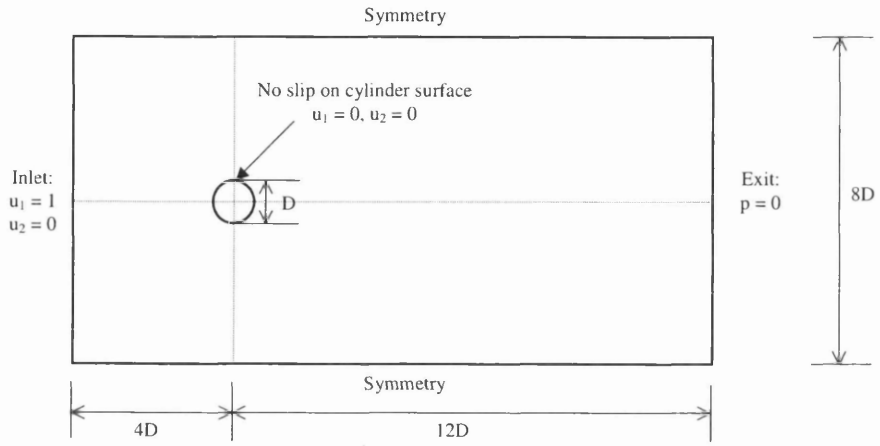


Figure 8.7: Unsteady flow past a circular cylinder at $Re = 100$. Problem domain and boundary conditions

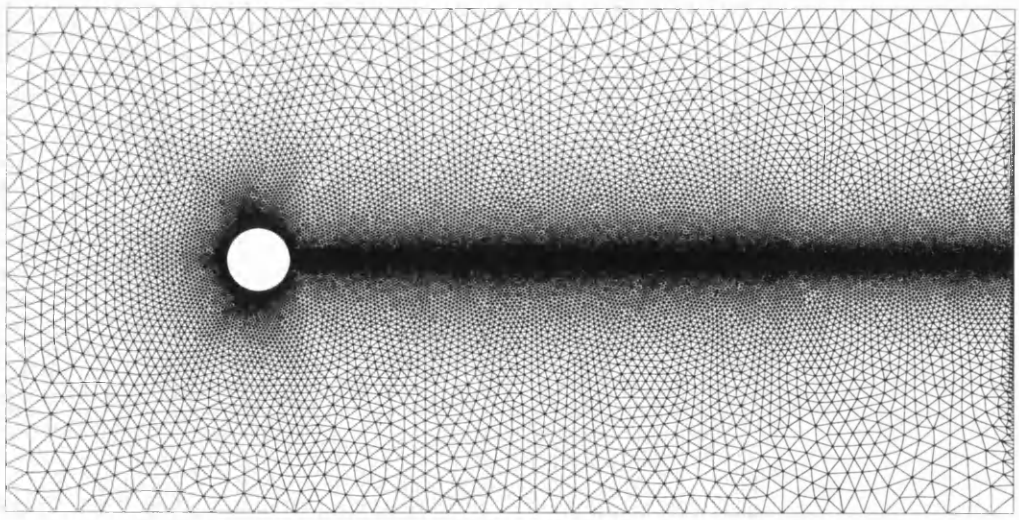
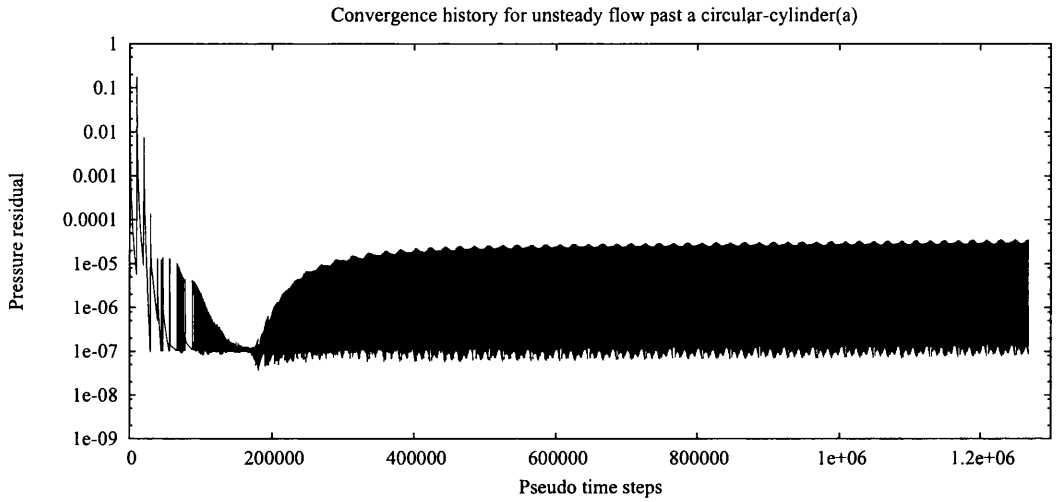
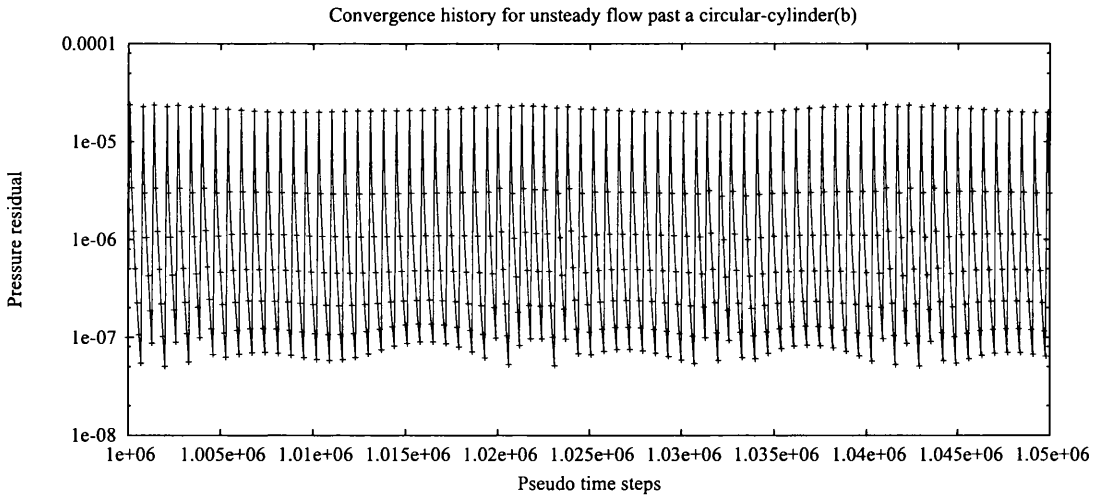


Figure 8.8: Unsteady flow past a circular cylinder at $Re = 100$. Unstructured meshes used in the computations.

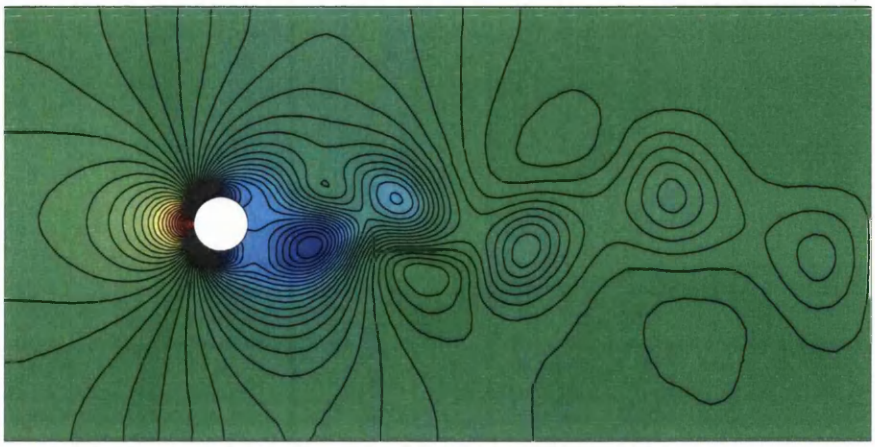


(a) Full pseudo time-step history

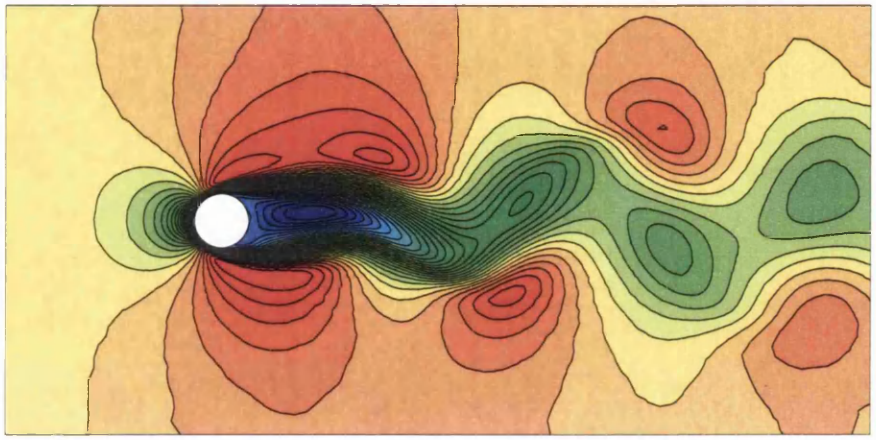


(b) Zoom of pseudo time-step history at a periodic state

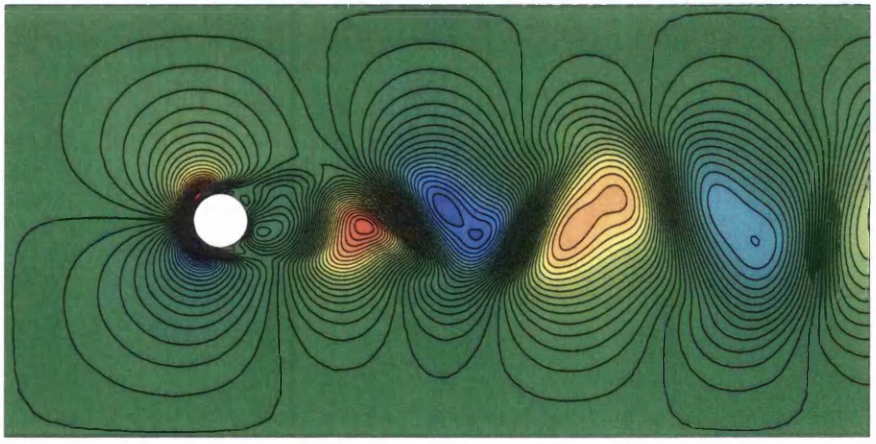
Figure 8.9: Unsteady flow past a circular cylinder at $Re = 100$. Convergence history of the pressure residual as a function of the cumulative pseudo time-step number



(a) Pressure contours

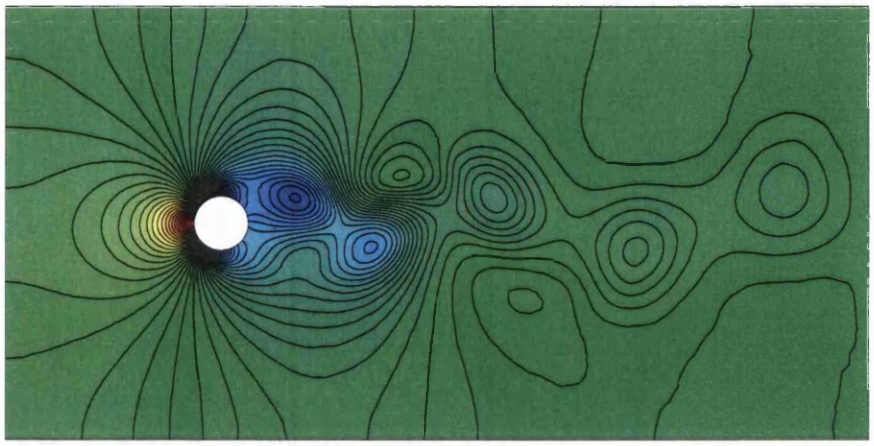


(b) u_1 velocity contours

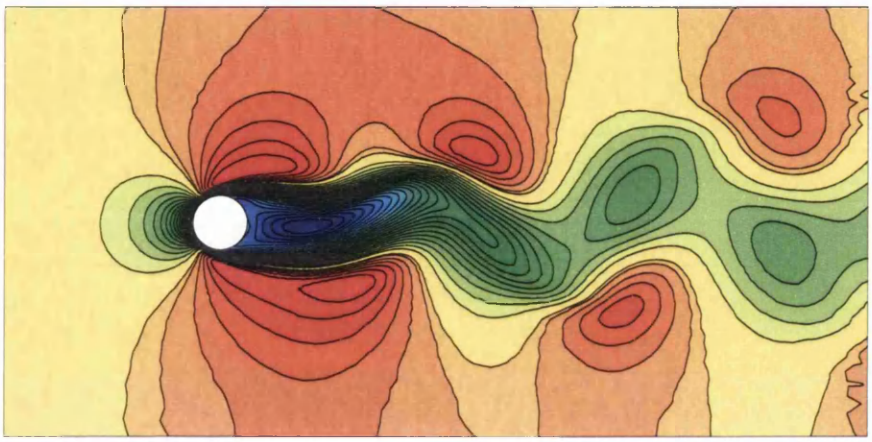


(c) u_2 velocity contours

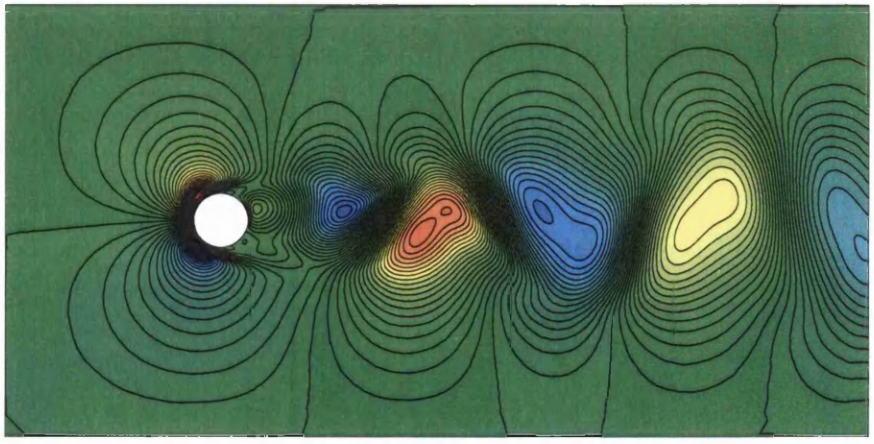
Figure 8.10: Unsteady flow past a circular cylinder at $Re = 100$. Computed solution at a non-dimensional real time of 150



(a) Pressure contours

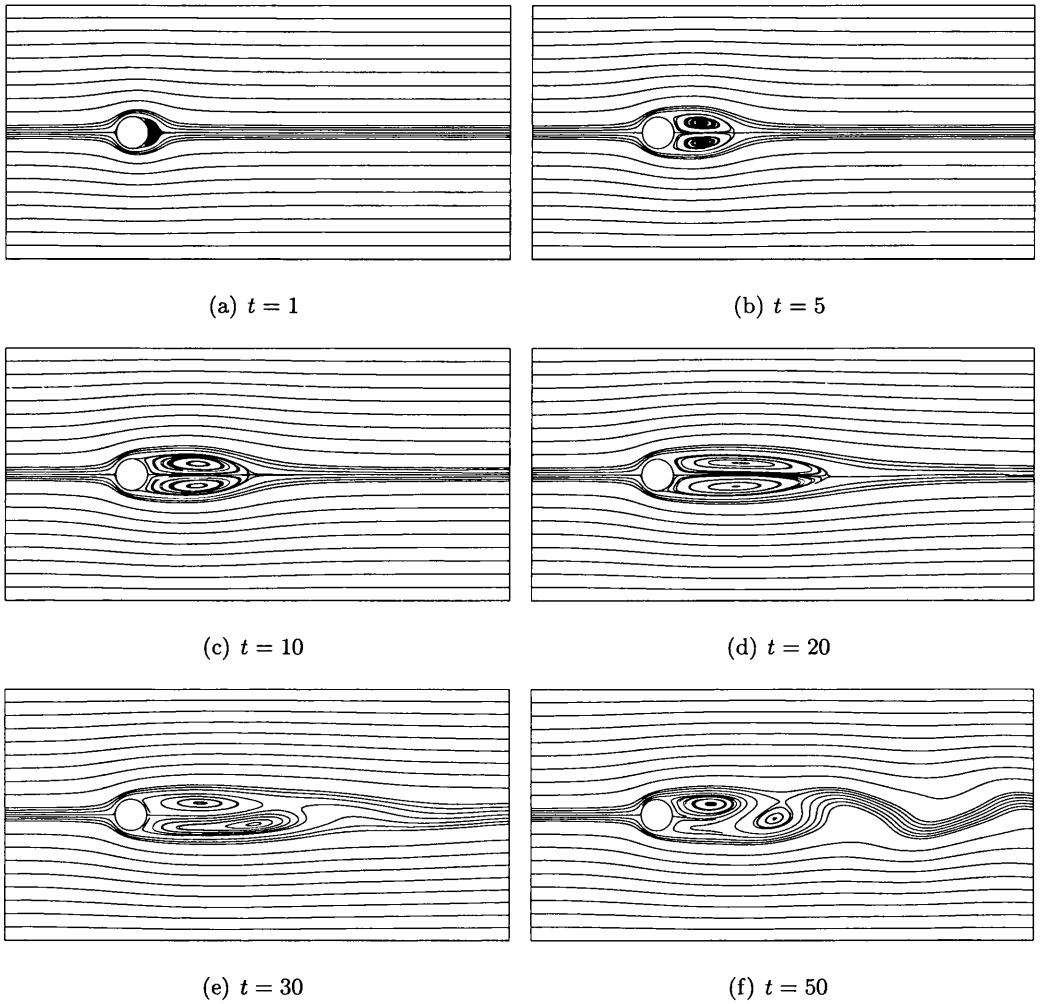


(b) u_1 velocity contours



(c) u_2 velocity contours

Figure 8.11: Unsteady flow past a circular cylinder at $Re = 100$. Computed solution at a non-dimensional real time of 200

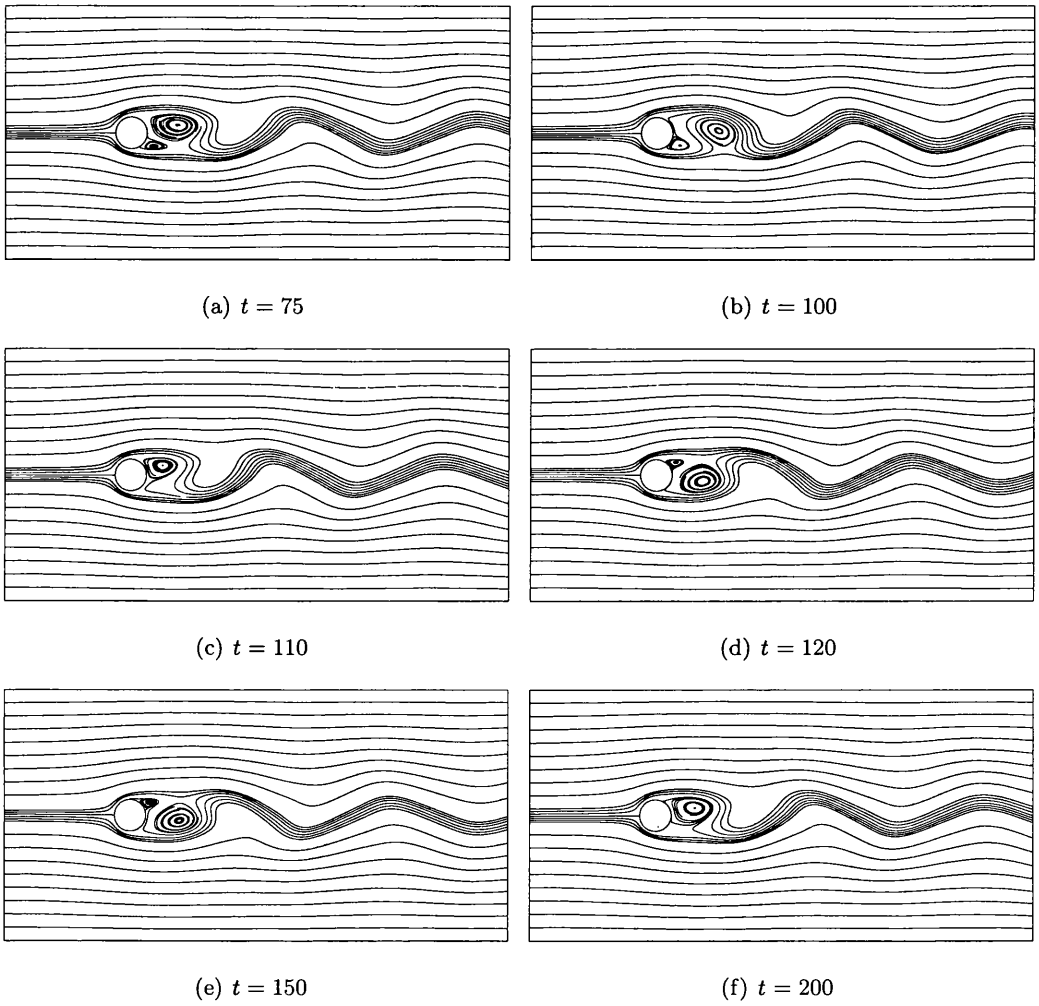


(a) $t = 1$ (b) $t = 5$

(c) $t = 10$ (d) $t = 20$

(e) $t = 30$ (f) $t = 50$

Figure 8.12: Unsteady flow past a circular cylinder at $Re = 100$. Flow stream-traces at different values of real-time for $1 \leq t \leq 50$.

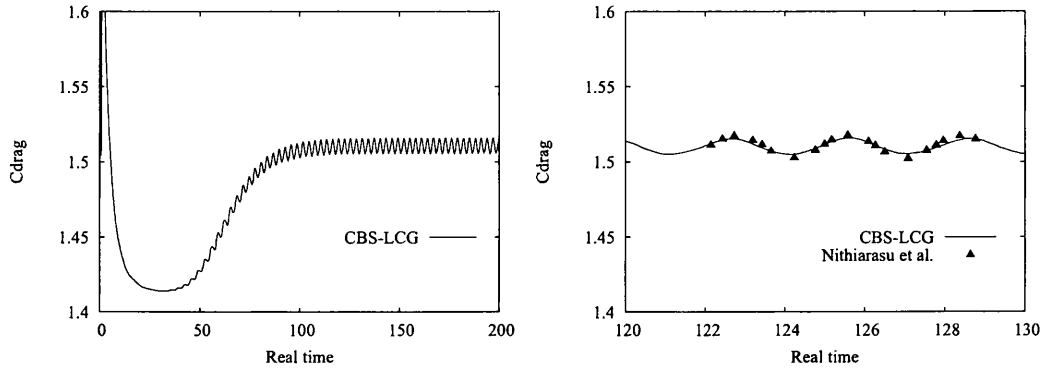


(a) $t = 75$ (b) $t = 100$

(c) $t = 110$ (d) $t = 120$

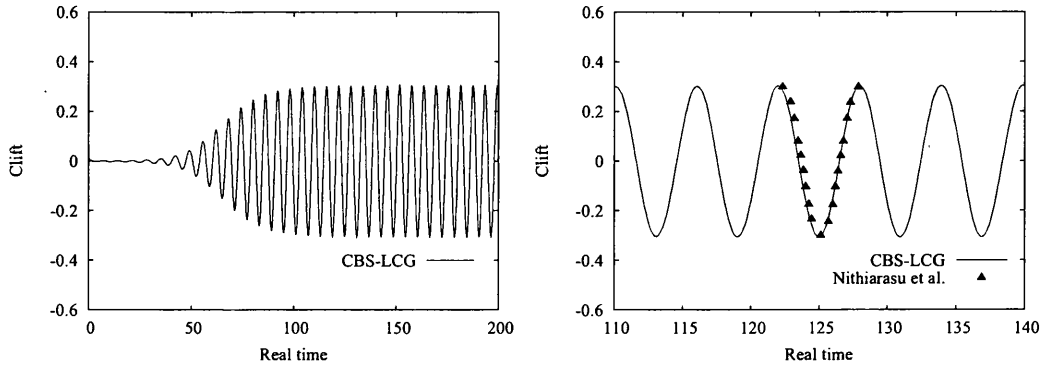
(e) $t = 150$ (f) $t = 200$

Figure 8.13: Unsteady flow past a circular cylinder at $Re = 100$. Flow stream-traces at different values of real-time for $75 \leq t \leq 200$.



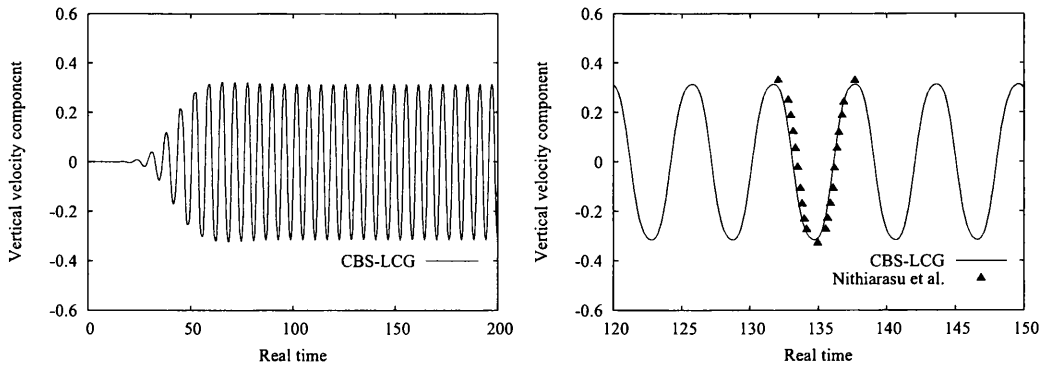
(a) Full history of drag coefficient c_d

(b) Comparison of c_d with ref. [108]



(c) Full history of lift coefficient c_l

(d) Comparison of c_l with ref. [108]



(e) Vertical velocity component u_2 at central exit

(f) Comparison of u_2 at exit, with ref. [108]

Figure 8.14: Unsteady flow past a circular cylinder at $Re = 100$. Computed coefficients of the drag and lift, and computed vertical velocity component at central exit point. All plotted as a function of the non-dimensional real time

set equal to zero.

A single unstructured mesh was used in this study and it is shown in Figure 8.8. It has 46,433 elements and 23,452 nodes. This mesh has been designed using previous knowledge on the formation of the unsteady wake and subsequent vortex shedding [129, 152, 151]. Thus the elements are very refined within this area. A fixed homogeneously fine mesh is expensive and so non-smoothness of contours in coarser areas of the domain, may be expected. The real time-step size chosen for this problem is 0.1, and simulations were carried out for a real non-dimensional time of 200. For each physical real time step the pressure residual, given by Equation (3.97), reached a tolerance of 1×10^{-7}. The maximum number of iterations allowed is 10,000 and the lowest is 100. Figure 8.9(a) provides a full simulation analysis of the convergence histories for each real time step on the mesh. As can be seen, once the initial transient stage has passed, the simulation settles down to an almost periodic convergence pattern. This can be seen more clearly in Figure 8.9(b). Here, a criss-cross mark has been employed to mark every 100 pseudo time-steps. After the initial transient it takes roughly 600-700 pseudo time-steps to reach this tolerance, for every real time-step.

The qualitative results are shown in Figures 8.10 and 8.11. Here, the contours of pressure and horizontal and vertical velocities are shown, for the real non-dimensional times of 150 and 200 respectively. All results are of high quality with no non-physical oscillations. A description of the flow pattern, at various values of real-time, is given by plotting stream-traces. The stream-traces are shown in Figure 8.12 for $1 \leq t \leq 50$ and Figure 8.13 for $75 \leq t \leq 200$.

A quantitative analysis of the results was also conducted and are shown in Figure 8.14. Here, the real-time history of both the lift and drag coefficients are given, along with the variation of the vertical velocity component at the central exit point. Generally, all the results shown for the CBS-LCG scheme are in good agreement with the results of Nithiarasu *et al.* [108] - obtained using a global Galerkin matrix-free CBS scheme for the non-conservation form of the incompressible flow equations.

Figures 8.14 (a) and (b) show the coefficient of drag, c_d, computed over the cylinder surface as a function of time. The full temporal history of c_d for $1 \leq t \leq 200$ is given in

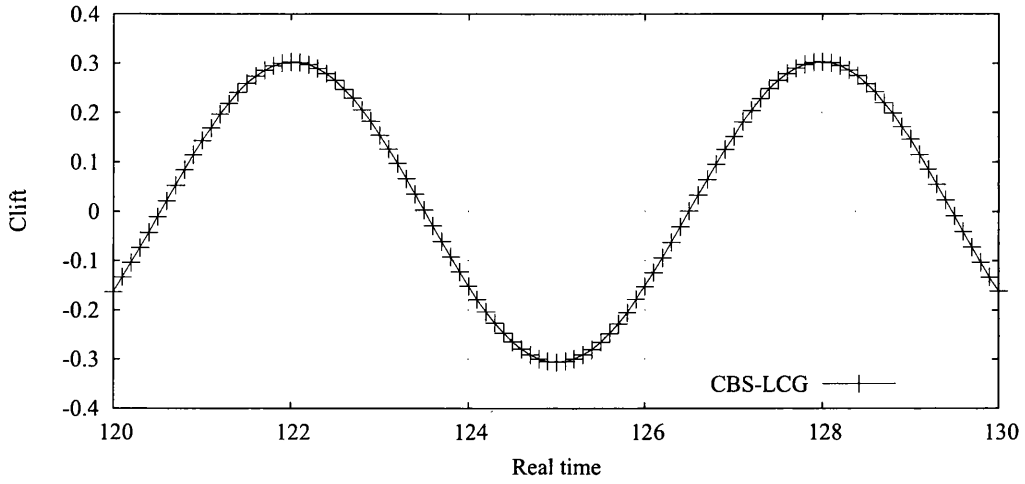


Figure 8.15: Unsteady flow past a circular cylinder at $Re = 100$. Plot of c_l in the steady-periodic region at $120 \leq t \leq 130$.

Figure 8.14(a). Figure 8.14(b) shows a zoom of the c_d for $120 \leq t \leq 130$. Here, the results of Nithiarasu *et al.* [108] have been included for comparison. As seen, despite using a coarser mesh for the CBS-LCG scheme, the difference is quite small between the two results. Details of the coefficient of Lift, c_l, are shown in Figures 8.14 (c) and (d). The former gives the full real-time history and the latter gives a comparison with Nithiarasu *et al.* [108], at $110 \leq t \leq 140$. As seen the agreement here is excellent. The final set, Figures 8.14 (e) and (f), give the real time history of the vertical velocity component - computed at the centre-point on the exit boundary. In Figure 8.14 (f), the comparison with Nithiarasu *et al.* [108] is small and may be due to a coarser mesh distribution being employed downstream of the cylinder at the exit in the present study.

After the initial transient, each variable develops a periodic variation. This is due to the periodic shedding of vortices from behind the cylinder. The nondimensional frequency of this oscillation is defined by Panton [151] as the *Strouhal number* St. In Figure 8.15, the lift coefficient is plotted in the steady-periodic region at $120 \leq t \leq 130$. Taking the two adjacent peaks in Figure 8.15), gives a nondimensional wavelength of 6. This in turn gives $St = 0.1666$, which agrees excellently with other numerical methods [114, 148].

8.4 Summary

This chapter firstly evaluated the explicit characteristic based LCG method, for transient hyperbolic problems. The rotating cosine-cone problem was used to demonstrate the LCG method was suitable for this application. Additionally, comparisons made with the global Galerkin method show that the two methods give identical solutions, when a lumped-mass matrix is used. The results of this small investigation are important for further applications, particularly in computational electro-magnetics, where an element-wise solution strategy would be cost effective.

It has also been shown in this Chapter that the matrix-free CBS-LCG scheme, based on the artificial compressibility method, has been successful in simulating unsteady, incompressible flows past a circular cylinder. The scheme was based on the non-conservation form of the equations, and a dual time-stepping approach was employed to recover the transient solution. Results were pressure stable and in good agreement with other schemes. The method has been proved to be robust in dealing with unsteady laminar incompressible flows.

Chapter 9

Conclusions and Future Work

9.1 General conclusions

A locally conservative Galerkin (LCG) procedure has been presented in this thesis. The basic concept introduced by Nithiarasu [1], has been thoroughly investigated and as a result, a family of LCG schemes have been developed. These schemes have been shown to be successful in solving a wide range of physical problems, from simple scalar transport to more complex problems such as the transient solution of unsteady Navier-Stokes flow.

The methodology presents an appealing alternative to the global Galerkin based methods, and finds a compromise between continuous and discontinuous Galerkin methods. Using a LCG approach, a continuous solution is obtained, but the solution procedure is element-wise. Additional benefits of the LCG method include both explicit element-wise and global conservation.

As practitioners of the discontinuous Galerkin are aware, the flexibility of solving the discrete equations elementally comes at a price of establishing continuity between neighbouring elements. In this work it is shown that a relatively cheaper numerical-flux procedure can be used in the LCG schemes to obtain accurate results.

Due to the nature of the research effort in this thesis, new codes were written to test all the LCG schemes. However, the procedures developed here can be incorporated into existing codes without complete revision. This is an additional benefit which makes

the method more appealing to industry.

Another important issue was the ease in which established global Galerkin schemes for convection-dominated problems could be incorporated into the LCG framework. This thesis gave details on the successful implementation of SUPG and characteristic based LCG methods. Benchmark numerical problems were solved to illustrated the LCG methods' accuracy.

The major objective of this thesis was to finally produce a LCG method for solving incompressible Navier-Stokes equations. The fully explicit characteristic based split (CBS) algorithm, provided the temporal template in which to spatially discretise using the LCG scheme. The resulting CBS-LCG method which uses artificial compressibility, was successfully evaluated for a number of steady-state problems and was shown to give good results. For unsteady flow the CBS-LCG scheme was adapted to incorporate dual time-stepping. It was shown to be successfully in giving the real transient solution.

9.2 Further work

The pioneering work presented in this thesis is only the beginning for research and development of the LCG method. This work has provided the foundations for numerous further investigations, all of which will hopefully give an even greater insight on this novel and interesting approach. A number of possible research routes are suggested:

1. Discontinuous Galerkin methods enjoy both element $h-$ and/or polynomial $p-$ refinement and/or a combination of both. It was shown in this thesis, that the LCG method also enjoys the refinement of element-size to obtain a more accurate and smoother solution. It was also shown that a increase in accuracy could be gained without refining h by using higher-order elements, *viz* using quadratic elements instead of linear elements. In the examples shown however, $p-$refinement was made to all elements in the domain. The LCG method supports an element-wise solution. It therefore is possible to exploit this approach by using combination of $h-$ and $p-$ refinement to obtain accuracy where it is needed and save CPU costs where accuracy and solution smoothness is not as important. It is proposed here, that work should

be carried out to investigate the use of mixed-order elements, within the same domain for the purpose of grid-adaptivity.

2. The numerical edge-fluxes discussed in this thesis are relatively cheap. This is one of the benefits of the LCG approach. It should not be ruled out however, investigating other methods of performing the edge-flux calculation. A number of possible routes could be taken here, especially for the diffusive terms. Though, if the original philosophy of the LCG approach is to be adhered to, then cost-effectiveness should be a priority.

3. It was shown that the LCG method could be applied to three-dimensions for diffusive problems. The 3D edge-flux calculation was evaluated and shown to be accurate. It is suggested that this should be extended to 3D convection-diffusion problems with the intension of extending the approach further, to solve 3D fluid dynamics problems.

4. The characteristic based LCG method was also evaluated for pure convection problems. The results obtained were found to be identical to the global Galerkin method. Time restrictions prevented further research into using LCG methods for solving hyperbolic problems. It is suggested here, that further research on incorporating established high-order temporal schemes into the LCG framework should be carried out. This area of research may be more favourable to areas of computational electromagnetics, where improving the phase and minimising dissipation is crucial for transient electro-magnetic scattering solutions.

5. Finally, in this thesis the LCG method was combined with the CBS temporal scheme, to successfully solve fluid mechanics problems. The CBS-LCG method developed, solved the non-conservation form of the incompressible Navier-Stokes equations. Two lines of possible research stem from this work: Firstly it is suggested that a CBS-LCG method should be developed for the conservation form of the incompressible flow equations. Secondly, artificial compressibility was employed in the CBS-LCG scheme developed in this thesis. It is reasonable to presume that the CBS-LCG method could potentially be applied to solve compressible flow problems.

Bibliography

[1] Nithiarasu P. A simple locally conservative galerkin (LCG) finite element method for transient conservation equations. *Numerical Heat Transfer - Part B Fundamentals*, 2004.

[2] Thomas CG. and Nithiarasu P. A locally conservative galerkin (LCG) method using quadratic elements. 13^{th} *International Congress on Finite Element for Flow Problems. Proceedings of FEF05 IACM Special Interest Conference supported by ECCOMAS, University of Wales Swansea.*, 4-6 April 2005.

[3] Thomas CG. and Nithiarasu P. A locally conservative galerkin (LCG) method for convection-diffusion and navier-stokes equations. *Proceedings of a Joint Conference of the Association for Computational Mechanics in Engineering (UK) and the Irish Society for Scientific and Engineering Computation, Cecil G. Armstrong (Ed.), Queen's University Belfast.*, 19-20 April 2006.

[4] Nithiarasu P. and Thomas CG. A element-wise, locally conservative galerkin (LCG) method for diffusion and convection-diffusion problems. *International Journal for Numerical Methods in Engineering*, In preparation.

[5] Thomas CG. and Nithiarasu P. A locally conservative galerkin (LCG) method for solving incompressible flow problems, using the AC-CBS temporal scheme. *Int. J. Num. Meth. Fluids*, In preparation.

[6] Zienkiewicz OC. Origins, milestones and directions of the finite element method -

personal view. *Handbook of Numerical Analysis, P.G. Ciarlet and J.L Lions (eds), Elsevier*, (4):7–67, 1996.

[7] Oden JT. Finite elements: an introduction. *Handbook of Numerical Analysis, P.G. Ciarlet and J.L Lions (eds), Elsevier*, (2):3–16, 1996.

[8] Gupta KK. and Meek JL. A brief history of the beginning of the finite element method. *Handbook of Numerical Analysis, P.G. Ciarlet and J.L Lions (eds), Elsevier*, (39):3761–3774, 1996.

[9] Felippa CA. A historical outline of matrix structural analysis: a play in three acts. *Comput. Struct.*, 79:1313 – 1324, 2001.

[10] Argyris JH. Energy theorems and structural analysis. *Aircraft Eng.*, 26:Oct – Nov, 1954.

[11] Argyris JH. Energy theorems and structural analysis. *Aircraft Eng.*, 27:April – May, 1955.

[12] Turner MJ. Clough RW. Martin HC. and Topp LJ. Stiffness and deflection analysis. *J. Aero. Sci*, 23:805 – 823, 1956.

[13] Turner MJ. The direct stiffness method of structural analysis. *Structural and Materials Panel Paper, AGARD Meeting, Aachen, Germany*, 1956.

[14] Clough RW. The finite element in plane stress analysis. *Proc. 2nf ASCE Conf. on Electronic Computation, Pittsburgh PA, September*, 1960.

[15] Zienkiewicz OC. and Cheung YK. Finite elements in the solution of field problems. *Engineer*, (200):507–510, 1965.

[16] Zienkiewicz OC. with Cheung YK. *The Finite Element Method in Continuum and Structural Mechanics.* McGraw Hill, 1967.

[17] Galerkin BG. Sterzhni i plastiny. ryady v nekotorykh voprosakh uprogogo ravnovesiya sterzhnei i plastin (rods and plates. series occuring in various questions concerning

the elastic equilibrium of rods and plates). *Vestnik Inzhenerov (engineers Bulletin)*, 19:897 – 908, 1915.

[18] Bubnov IG. The comments of referee on reviewing the paper by professor S.P. Timoshenko, 'on the stability of elastic systems' was awarded by the D.I. Zhuravskii. *Sbornik St Petersburg Inta Inzhenerov Putei Soobshch (Symposium of the Institute of Communication Engineers)1913,81,33-36, See also* S.G. *Mikhlin Variational methods in mathematical physics, translated from Russian by* T. *Boddington; editorial introduction translated by editor* L.I.G *Chambers. Pergamon, 1964.*

[19] Strang G. and Fix GJ. *An analysis of the finite element method.* Prentice Hall, 1973.

[20] Zienkiewicz OC. and Morgan K. *Finite Elements and Approximation.* Wiley, 1st edition, 1982.

[21] Zienkiewicz OC. Taylor RL. and Zhu J. *The Finite Element Method, The basis, (Volume 1).* Elsevier Butterworth Heinemann, London, 6th edition, 2005.

[22] Zienkiewicz OC. Taylor RL. and Zhu J. *The Finite Element Method, Solid Mechanics, (Volume 2).* Elsevier Butterworth Heinemann, London, 6th edition, 2005.

[23] Zienkiewicz OC. Taylor RL. and Nithiarasu P. *The finite element method for fluid Dynamics, (Volume 3).* Elsevier Butterworth Heinemann, London, 6th edition, 2005.

[24] Fletcher CA. *Computational Techniques for Fluid Dynamics, Vol 1 Fundementals and General Techniques.* Springer-Verlag, Berlin, 1988.

[25] Christie I. Griffiths DF. Mithchell AR. and Zienkiewicz OC. Finite element methods for second order differential equations with significant first derivatives. *Int J. Numer. Methods Eng.*, (10):1389–1396, 1976.

[26] Heinrich JC. Huyakorn PS. Zienkiewicz OC. and Mithchell AR. An "upwind" finite element scheme for two-dimensional convective transport equation. *Int J. Numer. Methods Eng.*, (11):134–143, 1977.

[27] Brooks AN. *A Petrov-Galerkin finite element formulation for convection dominated flows.* PhD thesis, California Inst. of Technology, Pasadena, California, 1981.

[28] Brooks AN. and Hughes TJR. Streamline upwind/petrov-galerkin formulations for convection dominated flows with particular empasis on th incompressible navier-stokes equations. *Comp. Meth. Appl. Mech. Eng.,*, 32:199 – 259, 1982.

[29] Hughes TJR. and Mallet M. A new finite element formulation for computational fluid dynamics: Iii. the generalized streamline operator for multidimensional advective-diffusive systems. *Comput. Methods Appl. Mech. Engrg*, 58:305–328, 1986.

[30] Hughes TJR. Franca LP. and Hulbert GM. A new finite element formulation for computational fluid dynamics: Viii. the galerkin/least-squares method for advective-diffusive systems. *Comput. Methods Appl. Mech. Engrg*, 73:173–189, 1989.

[31] Oñate E. Derivation of stabilized equations for numerical soluion of advective-diffusive transport and fluid flow problems. *Comput. Methods Appl. Mech. Engrg*, 151:233 – 65, 1998.

[32] Hughes TJR. Multiscale phenomena: Green's functions, the dirichlet-to-neumann formulation, subgrid scale models, bubbles and the origins of stabilized methods. *Comput. Methods Appl. Mech. Engrg*, 127:387–401, 1995.

[33] Codina R. A comparison of some finite element methods for solving the diffusion-convection-reaction equation. *Computer Methods in Applied Mechanics and Engineering*, 156:185 – 210, 1998.

[34] Codina R. On stablized finite element methods for linear systems of convection-diffusion-reaction equations. *Computer Methods in Applied Mechanics and Engineering*, 188:61 – 82, 2000.

[35] Codina R. Stablized finite element approximation of transient incompressible flows using orthogonal subscales. *Computer Methods in Applied Mechanics and Engineering*, 191:4295 – 4321, 2002.

[36] Huerta A. and Donea J. Time-accurate solution of stablized convection-diffusion-reaction equations: I -time and space discretization. *Comput. Methods Appl. Mech. Engrg. Anal.*, (18):565–573, 2002.

[37] Brezzi F. Franca LP. Russo A. Further considerations on residual-free bubbles for advective-diffusive equations. *Comput. Methods Appl. Mech. Engrg*, 166:25–33, 1998.

[38] Donea J. A taylor-galerkin method for convective transport problems. *Int J. Numer. Methods Eng.*, (48):25–43, 1984.

[39] Löhner R. Morgan K. and Zienkiewicz OC. The solution of non-linear hyperbolic equation systems by the finite element method. *Int. J. Numer. Methods Fluids*, (4):1043–1063, 1984.

[40] Lax PD. and Wendroff B. System of conservation laws. *Communications on Pure and Applied Mathematics*, 13:217 – 237, 1960.

[41] Donea J. and Huerta A. *Finite Element Methods for Flow Problems.* Wiley, Chichester, 2003.

[42] Zienkiewicz OC. and Codina R. A general algorithm for compressible flow - part i. the split characteristic-based scheme. *Int. J. Num. Meth. Fluids*, (20):869–885, 1995.

[43] Zienkiewicz OC. Nithiarasu P. Codina R. Vazquez M. and Ortiz P. The charateristic based split (CBS) procedure: An efficient and accurate algorithm for fluid mechanics problems. *Int. J. Num. Meth. Fluids*, 31:359 – 392, 1999.

[44] Nithiarasu P. Codina R. and Zienkiewicz OC. The characteristic based split scheme - a unified approach to fluid dynamics. *Int. J. Num. Meth. Eng.*, 66:1514 – 1546, 2006.

[45] Chung TJ. *Computational Fluid Dynamics.* Cambridge, 2002.

[46] Reed WH. and Hill TR. Triangular mesh methods for the neutron transport equation. *Technical report LA-UR-73-479, Los Alamos Scientic Laboratory,*, 1973.

[47] La Saint and Raviart. On a finite element method for solving the neutron transport equation. *Mathematical Aspects of Finite Elements in Partial Differntial Equations.*

Etd. Carl de Boor. Procedings of a Symposium Conducted by the Mathematic Research Center, Uni. Wisconsin-Madison, April 1-3, 1974.

[48] Cockburn B. Karniadakis G.E. and Shu C-W. (Eds.). *Lecture Notes in Computational Science and Engineering. The Development of Discontinuous Galerkin Methods. Discontinuous Galerkin Method. Theory, Computation and Applications.* Springer, Berlin, 2000.

[49] Chavent G. and Salzano G. A finite element method for the 1d water flooding problem with gravity. *Journal of Computational Physics.*, (45):307, 1982.

[50] Chavent G. and Cockburn B. The local projection p^0p^1 - discontinuous galerkin finite element method for scalar conservation laws. *RAIRO Modél. Math. Anal.Numer.*, (23):565–592, 1989.

[51] Cockburn B. and Shu C-W. Tvb runge-kutta local projection discontinuous galerkin finite element method for scalar conservation laws ii: General framework. *Math. Comp.*, (52):411–435, 1989.

[52] Cockburn B. Lin SY. and Shu C-W. Tvb runge-kutta local projection discontinuous galerkin finite element method for conservation laws iii: One dimensional systems. *Journal of Computational Physics*, (84):90–113, 1989.

[53] Cockburn B. Hou S. and Shu C-W. Tvb runge-kutta local projection discontinuous galerkin finite element method for conservation laws iv: The multidimensional case. *Math. Comp.*, (54):545–581, 1990.

[54] Cockburn B. and Shu C-W. The runge-kutta local projection p^1 discontinuous galerkin finite element method for scalar conservation laws. *RAIRO Modél. Math. Anal.Numer.*, (25):337–361, 1991.

[55] Cockburn B. and Shu C-W. The runge-kutta discontinuous galerkin finite element method for conservation laws v: Multidimensional systems. *Journal of Computational Physics*, (141):199–224, 1998.

[56] Cockburn B. Disconuous galerkin methods for convection dominated problems. high-order methods for computational science and engineering, vol.9. *Springer-Verlag, Berlin*, 1999.

[57] Cockburn B. Kanschat G. and Schötzau D. The local discontinuous galerkin method for linearized incompressible fluid flow: a review. *Computers and Fluids.*, (34):491–506, 2005.

[58] Richter GR. The discontinuous galerkin method with diffusion. *Mathematics of Computation*, (58,198):631–643, 1992.

[59] Dawson CN. Godunov-mixed methods for advection-diffusion equations in one space dimension. *SIAM J. Numer. Anal.*, (28):1282–1309, 1991.

[60] Dawson CN. Godunov-mixed methods for advection-diffusion equations in multidimensions. *SIAM J. Numer. Anal.*, (30):1315–1332, 1993.

[61] Baumann CE. and Oden JT. A discontinuous hp finite element method for convection-diffusion problems. *omp. Meth. Appl. Mech. Engg.*, (175):311 –341, 1999.

[62] Oden JT. Babuska I. and Baumann CE. A discontinuous *hp* finite elemnt method for diffusion problems. *Journal of Comp. Physics*, (146), 1998.

[63] Baumann CE. and Oden JT. A discontinuous hp finite element method for the euler and navier-stokes problems. *Int. J. Numer. Methods Fluids.*, (31):79–95, 1999.

[64] Oden JT. and Baumann CE. A conservative dgm for convection-diffusion and navier-stokes problems. *In Discontinuous Galerkin Methods: Theory, Computation and Applications, Berlin. Springer-Verlag*, pages 179–194, 2000.

[65] Bassi and Rebay. high-order accurate discontinuous finite element method for the numerical solution of the compressible navier-stokes equations. *J. Comp. Phy.*, 131:267–279, 1997.

[66] Cockburn B. and Shu C-W. The local discontinuous galerkin finite element method for convection-diffusion systems. *JSIAM J. Numer. Anal.*, (35):2440–2463, 1998.

[67] Castillo P. An optimal error estimate for the local discontinuous galerkin method. *In Discontinuous Galerkin Methods: Theory, Computation and Applications, Springer-Verlag, Berlin.*, pages 285–290, 2000.

[68] Castillo P. Cockburn B. Perugia I. and Schötzau D. An a prior error analysis of the local discontinuous galerkin method for elliptic problems. *SIAM J. Numer. Anal.*, (38):1676–1706, 2000.

[69] Castillo P. Cockburn B. Schötzau D. and Schwab C. Optimal a priori error estimates for the *hp*-version of the local discontinuous galerkin method for convection-diffusion problems. *Math Comp.*, (71):455–478, 2001.

[70] Cockburn B. Kanschat G. Schötzau D. and Schwab C. Local discontinuous galerkin methods for the stokes system. *SIAM J. Numer. Anal.*, (40):319–343, 2002.

[71] Cockburn B. Kanschat G. and Schötzau D. The local discontinuous galerkin method for the oseen equations. *Math Comp.*, (73):569–593, 2004.

[72] Cockburn B. Kanschat G. and Schötzau D. A locally conservative LDG method for the incompressible navier-stokes equations. *Math Comp.*, (74):1067–1095, 2005.

[73] Cockburn B. Kanschat G. and Schötzau D. The local discontinuous galerkin methods for linear incompressible flow: A review, computer and fluids (special issue: Residual based methods and discontinuous galerkin schemes). *Computers and Fluids*, (34):491–506, 2005.

[74] Zienkiewicz OC. Taylor RL. Sherwin SJ. and Peiro J. On discontinuous galerkin methods. *Int. J. Num. Meth. Engg.*, (58):1119 – 1148, 2003.

[75] Hughes TJR. Scovazzi G. Bochev PB. and Buffa A. A multiscale discontinuous galerkin method with the computational structure of a continuous galerkin method. *Computer Methods in Applied Mechanics and Engineering*, 195:2761 – 2787, 2006.

[76] Nithiarasu P. A matrix free fractional step method for static and dynamic incompressible solid mechanics. *Int. J. Comp. Meth. Eng. Sci and Mech.*, 7:369 – 380, 2006.

[77] Ewing RE. and Russell TF. Multistep galerkin methods along charcteristics for convection-diffusion problems. *Advances in Computation Methods for PDEs, IMACS, Rutgers University, Brunswick, NJ. Eds. R. Vichnevetsky and R Stepleman*, (IV):28–36, 1981.

[78] Douglas Jr J. and Russell TF. Numerical methods for convection dominated diffusion problems based on combining the method of characteristics with finite element or finite difference procedures. *SIAM J. Numer. Anal.*, (19):871–885, 1982.

[79] Bercovier M. Pironneau O. Harbani Y. Levine E. Characteristics and finite element methods applied to equations of fluids. *The Mathematics of Finite Elements and Applications, Whiteman J. Edt., Academic Press, London*, V:471–478, 1982.

[80] Bercovier M. Pironneau O. Sastri V. Finite elements and characteristics for some parabolic-hyperbolic problems. *Appl. Math. Modelling*, 7:89–96, 1983.

[81] Pironneau O. Liou J. and Tezduyar T. Characteristic galerkin and galerkin least squares space-time formulations for the advection-diffusion equation with time dependent domain. *Comput. Methods Appl. Mech. Engrg*, 100:117 – 141, 1992.

[82] Drikakis D. Govatsos PA. and Papantonis DE. A characteristic-based method for incompressible flows. *Int. J. Numer. Methods Fluids*, (19):667–685, 1994.

[83] Kaazempur-Mofrad MR. and Ethier CR. An efficient characteristic galerkin scheme for the advection equation in 3-d. *Comput. Methods Appl. Mech. Engrg*, 191:5345 – 5363, 2002.

[84] Kaazempur-Mofrad MR. Minev PD. and Ethier CR. An characteristic/finite element algorithm for time-dependent 3-d advection dominated transport using unstructured grids. *Comput. Methods Appl. Mech. Engrg*, 192:1281 – 1298, 2003.

[85] Drikakis D. and Rider W. *High-resolution methods for incompressible and low speed flows.* Springer, 2004.

[86] Shapiro E. and Drikakis D. Non-conservative and conservative formulations of characteristics-based numerical reconstructions for incompressible flows. *Int J. Numer. Methods Eng.*, (66):1466–1482, 2006.

[87] Zienkiewicz OC. and Codina R. Search for a general fluid mechanics algorithm. *In D.A. Caughey and M.M. Hafez, editors, frontiers of Computational Fluid Dynamics. Wiley*, pages 101–113, 1995.

[88] Zienkiewicz OC. Morgan K. Sai BVKS. Codina R. and Vazquez M. A general algorithm for compressible flow - part ii. tests on the explicit form. *Int. J. Num. Meth. Fluids*, 20:887–913, 1995.

[89] Chorin AJ. Numerical solution of navier-stokes equations. *Math Comput*, (22):745–762, 1968.

[90] Codina R. Vazquez M. and Zienkiewicz OC. General algoritm for compressible and incompressible flows, part iii. a semi-implicit form. *Int. J. Num. Meth. Fluids*, 27:13 – 32, 1998.

[91] Nithiarasu P. Zienkiewicz OC. Sai BVKS. Morgan K. Codina R. and Vazquez M. Shock capturing viscosities for the general fluid mechanics algorithm. *Int. J. Num. Meth. Fluids*, 28:1325 – 1353, 1998.

[92] Nithiarasu P. and Zienkiewicz OC. Adaptive mesh generation for fluid mechanics problems. *Int J. Numer. Methods Eng.*, 47:629 – 662, 2000.

[93] Nithiarasu P. and Zienkiewicz OC. On stabiliaztion of the cbs algorithm: Internal and external time steps. *Int J. Numer. Methods Eng.*, 48:875 – 880, 2000.

[94] Zienkiewicz OC. and Nithiarasu P. The characteristic-based-scheme (cbs) algorithm, stability and boundary conditions. *Arch. Mech. Warszawa*, 52:857–887, 2000.

[95] Nithiarasu P. On boundary conditions of the cbs algorithm for fluid dynamicss. *Int J. Numer. Methods Eng.*, 54:523 – 536, 2002.

[96] Nithiarasu P. An efficient artificial compressibility (ac) scheme based on the charcteristic based split (cbs) method for incompressible flows. *Int. J. Num. Meth. Fluids*, 56:1815 – 1845, 2003.

[97] Nithiarasu P. Mathur JS. Weatherill NP. and Morgan K. Three dimensional incompressible flow calculations using the characteristic based split (cbs) scheme. *Int. J. Num. Meth. Fluids*, 44:1207 – 1229, 2004.

[98] Nithiarasu P. and Liu C-B. Steady and unsteady flow calculations in a double driven cavity using the artifical compressibility (ac)- based charcteristic based split (cbs) scheme. *Int. J. Num. Meth. Fluids*, 63(3):380 – 397, 2005.

[99] Thomas CG. and Nithiarasu P. Influences of element size and variable smoothing on inviscid compressible flow solution. *International Journal of Numerical Methods in Heat and Fluid Flow*, (15):420–428, 2005.

[100] Zienkiewicz OC. and Ortiz P. A split-characteristic based finite element model for the shallow water equations. *Int. J. Num. Meth. Fluids*, 20:1061–1080, 1995.

[101] Massarotti N. Nithiarasu P. Zienkiewicz OC. The characteristic based split (cbs) algorithm for incompressible flow problems with heat transfer. *International Journal of Numerical Methods for Heat and Fluid Flow*, 8:969–990, 1998.

[102] Massarotti N. Nithiarasu P. Zienkiewicz OC. Natural convection in a porous medium - fluid interface problems a finite element analysis by using the characteristic based split (cbs) algorithm. *International Journal of Numerical Methods for Heat and Fluid Flow*, 11:473–490, 2001.

[103] Lewis R.W. Nithiarasu P. Seetharamu K. *Fundamentals of the Finite Element Method for Heat and Fluid Flow.* Wiley, 2004.

[104] Nithiarasu P. Massarotti N. and Mathur JS. Forced convection heat transfer from solder balls on a printed circuit board using the charcteristic based split (cbs) schemes. *International Journal of Numerical Methods in Heat and Fluid Flow*, (15):73–95, 2005.

[105] Nithiarasu P. and Liu C-B. An artificial compressibility based charcteristic based split (cbs) schemes for steady and unsteady turbulent incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, 195:2961 – 2982, 2006.

[106] Nithiarasu P. A fully explicit characteristic based split (cbs) scheme for viscoelastic flow calculations. *Int. J. Num. Meth. Eng.*, 60:949 – 978, 2004.

[107] Thomas CG. and Nithiarasu P. The characteristic based split (cbs) approach for inviscid compressible flow problems. effect of element size in the streamline direction. 4^{th} *European Congress on Computational Methods in Applied Sciences and Engineering. Proceedings of ECCOMAS 2004 - Vol. 1, P. Neittaanmäki, T. Rossi, K. Majava, and O. Pironneau (Eds.), Jyväskylä, Finland, 24-28 July*, 2004.

[108] Nithiarasu P. and Zienkiewicz OC. Analysis of an explicit and matrix free fractional step method for incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, 195:5537 – 5551, 2006.

[109] Ladyshenskaya OA. *The Mathematical Theory of Viscous Incompressible Flow.* Gordon and Breach, New York, 2^{nd} edition, 1969.

[110] Babuška I. The finite element method with lagrange multipliers. *Numerische Mathematik*, 20:179–192, 1973.

[111] Brezzi F. On the existence, uniqueness and approximation of saddle-point problems arising from lagrange multipliers. *RAIRO-Analyse Numerique-Numerical Analysis*, 8(R2):129–151, 1974.

[112] Chorin AJ. A numerical method for solving incompressible viscous problems. *J. Comput Phys*, (2):12–26, 1967.

[113] Malan AG. Lewis RW. and Nithiarasu P. An improved unsteady, unstructured, artificial compressibility, finite volume scheme for viscous incompressible flows: Part i. theory and implementation. *Int. J. Num. Meth. Engineering*, 54:695–714, 2002.

[114] Malan AG. Lewis RW. and Nithiarasu P. An improved unsteady, unstructured, artificial compressibility, finite volume scheme for viscous incompressible flows: Part ii. application. *Int. J. Num. Meth. Engineering*, 54:715–729, 2002.

[115] Gaitonde AL. A dual-time method for two-dimensional unsteady incompressible flow calculations. *Int. J. Num. Meth. Engineering*, 41:1153–1166, 1998.

[116] Douglas Jr J. Dupont T. and Wheeler MF. A Galerkin procedure for approximating the flux on the boundary for elliptic and parabolic boundary value problems. *Revue Francaise d'Automatique, Informatique et Recherche Opérationnelle*, (2):47, 1974.

[117] Carey GF. Derivative calculation from finite element solutions. *Computer Methods in Applied Mechanics and Engineering*, 35:1, 1982.

[118] Babuška I. and Miller A. The post-processing approack in the finite element method. calculation of displacements, stresses and other higher derivatives of the displacements. *Int. J. Num. Meth. Eng.*, 20:1085, 1984.

[119] Carey GF. Chow SS. and Seager MK. Approximate boundary flux calculations. *Computer Methods in Applied Mechanics and Engineering*, 50:107, 1985.

[120] Mizukami A. A mixed finite element method for boundary flux computation. *Comput. Methods Appl. Mech. Engrg*, 57:239, 1986.

[121] Barrett JW. and Elliott CM. Total flux estimates for a finite element approximation of ellptic equations. *I. M. A. J. Num. Anal.*, 7:129, 1987.

[122] Gresho PM. Lee RL. Sani RL. Maslanik MK. and Eaton BE. The consistent galerkin FEM for computing derived boundary quantities in thermal and/or fluids prolems. *Int. J. Num. Meth. Fluids*, 7:371–394, 1987.

[123] Hughes TJR. Franca LP. Harari I. Mallet M. Shakib F. and Spelce TE. Finite element method for high-speed flows: Consistent calculation of boundary flux. *AIAA 25TH Aerospace Sciences Meeting, Reno, Nevada*, AIAA-87-0556, 1987.

[124] Oshima M. Hughes TJR. and Jansen K. Consistent finite element calculation of boundary and internal fluxes. *International Journal of Computational Fluid Dynamics*, (9):227, 1998.

[125] Hughes TJR. Engel G. Mazzei L. and Larson MG. The continuous galerkin method is locally conservative. *Journal of Computational Physics*, 2000.

[126] Hughes TJR. Wells GN. Conservation properties for the galerkin and stabilised forms of the advection-diffusion and incompressible navier-stokes equaions. *Computer Methods in Applied Mechanics and Engineering*, 194:1141 – 1159, 2005.

[127] Morgan K. Peraire J. Peiro J. and Zienkiewicz OC. Adaptive remeshing applied to the solution of a shock interaction problem on a cylindrical leading edge. *Computational Methods in Aeronautical Fluid Dynamics, P. Stow (ed), Clarenden Press, Oxford*, pages 327 – 344, 1990.

[128] Shakib F. *Finite element analysis of the compressible Euler and Navier-Stokes equations.* PhD thesis, Stanford University, 1988.

[129] Massey IG. *Mechanics of Fluids.* Chapman and Hall, 6^{th} edition, 1989.

[130] Currie IG. *Fundamental Mechanics of Fluids.* McGraw-Hill, 1974.

[131] Anderson Jr JD. *Computational Fluid Dynamics.* McGraw Hill, 1^{st} edition, 1995.

[132] Hirsch C. *Numerical Computation of Internal and External Flows, Volume 1 Fundamentals of Numerical Discretization.* Wiley, 1995.

[133] Codina R. A finite element formultion for the numerical solution of the convection-diffusion equation. *Technical Report 14, International Center for Numerical Methods in Engineering (CIMNE), Barcelona.*, (14).

[134] Lebeau GJ. Ray SE. Aliabadi SK. and Tezduyar TE. Supg finite element computation of compressible flows with the entropy and conservation variable formulations. *Computer Methods in Applied Mechanics and Engineering*, 104:397–422, 1993.

[135] Catabriga L. and Coutinho ALGA. Implicit supg solution of euler equations using edge-based data structures. *Computer Methods in Applied Mechanics and Engineering*, (191):3477–3490, 2002.

[136] Codina R. and Zienkiewicz OC. Cbs versus gls stabilisation of the incompressible navier-stokes equations and the role of the time step as stabilisation parameter. *Comm. Num. Meth. Engg.*, 18:99 – 112, 2002.

[137] Zienkiewicz OC. and Zhu JZ. The superconvergence patch recovery and a posteriori error estimation in the finite element method, part i: A general superconvergent recovery technique. *Int. J. Num. Meth. Eng.*, 33:1331–1364, 1992.

[138] Hirsch C. *Numerical Computation of Internal and External Flows, Volume 2 Computational Methods for Inviscid and Viscous Flows.* Wiley, 1995.

[139] Pulliam TH. and Barton JT. Euler computations of agard working group 07 airfoil test cases. aiaa 23rd aerospace sciences meeting, jan 14-17, 1985/reno, nevada. *AIAA-85-0018*, 23, 1985.

[140] Çengel YA. *Introduction to Thermodynamics and Heat Transfer.* McGraw Hill Publishers, 1997.

[141] Holman JP. *Heat Transfer.* McGraw Hill Publishers, Singapore, 1989.

[142] Jennings A. and McKeown JJ. *Matrix Computation.* Wiley, 1992.

[143] Hassan O. Morgan K. and Weatherill NP. Flite system version 2. user manual. *Computational Dynamics Research Ltd.*, 1999.

[144] Donea J. Belytschko T. and Smolinski P. Ia generalized galerkin method for steady convection-diffusion problems with application to quadratic shape functions. *Computer Methods in Applied Mechanics and Engineering*, (48):25–43, 1985.

[145] Abanto J. Pelletier D. Gadon A. Trepanier J-Y. and Reggio M. Verification of some commercial cfd codes on atypical cfd problems. aiaa aerospace sciences meeting, 10 - 13 january 2005, reno, nevada. *AIAA-2005-0682*, 2005.

[146] Ghia U. Ghia KN. and Shin CT. High-re solutions for incompressible flow using the navier-stokes equations and a multigrid method. *Journal of Computational Physics*, 48:387 – 411, 1982.

[147] Mack ANF. An element level zero-divergence finite element approach. *Int. J. Num. Meth. Fluids*, 19:795 – 813, 1994.

[148] deSampaio PAB. Lyra PRM. Morgan K. and Weatherill NP. Petrov galerkin solutions of the incompressible navier-stokes equation in primitive variables with adaptive remeshing. *Computer Methods in Applied Mechanics and Engineering*, (106):143–178, 1992.

[149] Tritton DJ. Experiments on the flow past a circular cylinder. *Journal of Fluid Mechanics*, 6:547 – 567, 1959.

[150] Tritton DJ. *Physical Fluid Dynamics (2^{nd}) Edition.* Oxford Science Publications, 2002.

[151] Panton RC. *Incompressible Flow.* Wiley, New York, 1984.

[152] Schlichting H. *Boundary Layer Theory (6^{th}) Edition.* McGraw-HillPublications, 1968.

[153] Hamielec AE. and Raal JD. Numerical studies of viscous flow around circular cylinders. *The physics of fluids*, 12, 1:11 – 17, 1969.

[154] Takami H. and Keller HB. Steady two-dimensional viscous flow of an incompressible fluid past a circular cylinder. *Physics of Fluids*, 12:51, 1969.

[155] Dennis SCR. and Chang GZ. Numerical solutions for steady flow past a circular cylinder at reynolds numbers upto 100. *Journal of Fluid Mechanics*, 42:471, 1970.

[156] Tuann SY. and Olson MD. Numerical studies of the flow around a circular cylinder by a finite element method. *Computers and Fluids*, 6:219, 1978.

[157] Ding H. Shu C. Yeo KS. and Xu D. Simulation of incompressible viscous flow past a circular cylinder by hybrid fd scheme and meshless least square-based finite difference

method. *Computer Methods in Applied Mechanics and Engineering*, 193:727 – 744, 2004.

[158] Denham MK. and Patrick MA. Laminar flow over a downstream-facing step in a two-dimensional flow chanel. *Trans. Instn. Chem. Engrs.*, 52:361 – 367, 1974.

[159] Armaly BF. Durst F. Pereira JCF. Schönung B. Experimental and theoretical investigation f backward-facing step flow. *Journal of fluid mechanics*, 127:473 – 496, 1983.

[160] Barton IE. The entrance effect of laminar flow over a backward-facing step geometry. *Int. J. Num. Meth. Fluids*, 25:633 – 644, 1997.

[161] Rouse H. and Howe JW. *Basic Mechanics of Fluids.* Wiley, 1953.

Appendix A

Postprocessing

A.1 Coefficient of pressure

The local coefficient of pressure, C_p, is defined as

$$C_p = \frac{p - p_\infty}{\frac{1}{2}\rho_\infty {u_\infty}^2} \tag{A.1}$$

where p is the pressure, ρ is the density, u is the velocity, and a subscript ∞ indicates a free stream (inlet) value. The denominator $\left(\frac{1}{2}\rho_\infty {u_\infty}^2\right)$ is the dynamic pressure [129].

A.2 Coefficient of drag

Figure A.1 shows fluid flowing past a solid body. The resultant force, F, on the body, due to the fluid flowing past it, can be resolved into two components. The force acting

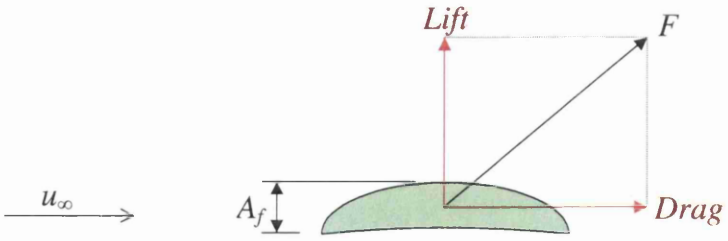


Figure A.1: Definition sketch for lift and drag forces on a solid body, as fluid flows past it.

in the horizontal direction is the drag force, and the force acting in the vertical direction is the lift force.

Generally the drag force is characterized by the coefficient of drag, C_d, which is given by

$$C_d = \frac{D}{A_f \frac{1}{2} \rho_\infty {u_\infty}^2} \tag{A.2}$$

Here D is the total drag force exerted on body by the fluid flowing past it, and A_f is the frontal area of body in the flow direction. The total drag force is the sum of both the pressure (form) drag and friction drag in the direction of the flow i.e.

$$D = Dp + Df \tag{A.3}$$

For a two dimensional problem, the solid wall boundary may be a line or a curve and approximated by an edge of an element. In this case Dp and Df may be calculated as follows [103]: The force due to pressure for each element edge along the solid boundary, is approximated as the average pressure of all the element's nodes that are on the boundary, multiplied by the length of the element's edge that lies on the boundary. This pressure force is multiplied by the direction cosine in the flow direction, to obtain the local pressure drag contribution for that element. Integration of all the element contributions along the solid boundary gives the drag force due to pressure Dp. The viscous drag force Df is calculated by integrating the viscous traction in the flow direction along the solid boundary. Adding the both integrals gives the total drag along the solid surface edge as

$$D = \int_{A_s} \left[\left(-p + \tau_{11} \right) n_1 + \left(\tau_{12} \right) n_2 \right] dA_s \tag{A.4}$$

where the components n_1 and n_2 make up the surface normal $\mathbf{n}$ along the solid boundary.

Author's Publications

1. Craig George Thomas and Perumal Nithiarasu, *The Characteristic Based Split (CBS) Approach for Inviscid Compressible Flow Problems. Effect of Element Size in the Streamline Direction*, 4^{th} European Congress on Computational Methods in Applied Sciences and Engineering. Proceedings of ECCOMAS 2004 - Vol. 1, P. Neittaanmäki, T. Rossi, K. Majava, and O. Pironneau (Eds.), Jyväskylä, Finland, 24-28 July 2004.

2. C. G. Thomas and P. Nithiarasu, *Influences of element size and variable smoothing on inviscid compressible flow solution*, International Journal of Numerical Methods for Heat and Fluid Flow. Vol. 15, No. 5, 2005, pp 420-428.

3. Craig G. Thomas and Perumal Nithiarasu, *A locally Conservative Galerkin (LCG) Method Using Quadratic Elements*, 13^{th} International Congress on Finite Element for Flow Problems. Proceedings of FEF05 IACM Special Interest Conference supported by ECCOMAS, University of Wales Swansea, 4-6 April 2005.

4. C. G. Thomas and and P. Nithiarasu, *A locally Conservative Galerkin (LCG) Method for Convection-Diffusion and Navier-Stokes Equations*, Proceedings of a Joint Conference of the Association for Computational Mechanics in Engineering (UK) and the Irish Society for Scientific and Engineering Computation, Cecil G. Armstrong (Ed.), Queen's University Belfast, 19-20 April 2006.

5. C. G. Thomas and and P. Nithiarasu, *A Element-wise, locally Conservative Galerkin (*LCG*) Method for Diffusion and Convection-Diffusion Problems*, International Journal for Numerical Methods in Engineering, To be submitted 2006.

6. C. G. Thomas and and P. Nithiarasu, *A locally Conservative Galerkin (*LCG*) Method for Solving Incompressible flow problems, using the* CBS *temporal scheme*, International Journal for Numerical Methods in Fluids, In preparation.