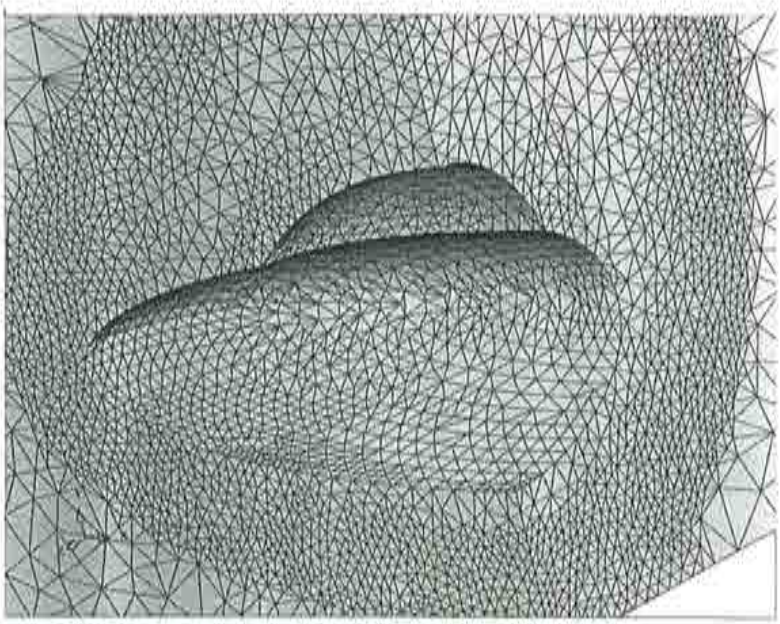
A Contribution to Adaptive Numerical Solution of Compressible Flow Problems

T.R. Fischer
E. Oñate



Monograph CIMNE Nº 34, October 1996

A Contribution to Adaptive Numerical Solution of Compressible Flow Problems

T.R. Fischer
E. Oñate

Monograph CIMNE Nº 34, October 1996

International Center for Numerical Methods in Engineering
Gran Capitán s/n, 08034 Barcelona, Spain

The cover designed by: Jordi Pallí

First published, October 1996

Edited by:
International Center for Numerical Methods in Engineering
C/ Gran Capitán, s/n
08034 Barcelona, Spain

ISBN: 84-87867-85-5
Deposito Legal: B-41390-96

Contents

Chapter 1

Introduction

Numerical methods for aerodynamic design and fluid flow prediction gained very much in significance about three decades ago in the late sixties and early seventies. At that time, the use of panel methods made it feasible to predict aerodynamic coefficients of even complex geometries with little computational effort. This was possible for both subsonic and supersonic flows. In the next decade, the use of potential methods and their extension to transonic flows by introducing non linear effects made it possible to predict fluid flows around jet aircraft which normally operate at transonic speeds. The last decade has advanced tremendously in the field of resolving more complete non linear equations governing compressible flow, such as the Euler and laminar Navier-Stokes equations, already yielding accurate solutions. Currently, some of the main fields of research is turbulence modeling, combustion and coupled problems, such as aeroelasticity. I will not enter a thorough description of each of these fields of interest, which the interested reader can obtain from the specialized literature.

Computational fluid dynamics (CFD) has therefore reached a mature state in several areas in the last decade. The algorithms for the solution of the Euler equations or even laminar Navier-Stokes equations have improved almost as drastically as the computer hardware. Next to the improvement of numerical schemes, such as multigrid, local time stepping or implicit time integration just to name a few, emphasis was also put on improving the effectiveness for a given accuracy. This lead to that, for instance, finite element and finite volume methods using unstructured meshes, error estimation and adaptive remeshing became competitive.

The possibilities and advantages of CFD now become apparent. The following is a non exhaustive list of advantages and drawbacks of computational methods versus experimental fluid dynamics (ie. wind tunnel or flight testing):

Advantages

- Faster and cost-efficient design and testing of aerodynamic behavior, especially in the initial design stages.
- During the initial design process more possibilities of combinations and creativity

for prototypes are feasible, whereas in the experimental design a decision for prototypes must be taken at a much earlier stage leading to a narrower collection of potential models.

- Postprocessing and visualization possibilities are greatly enhanced requiring less time and cost and provide greater detail.
- The lead time for development is greatly reduced. This is a direct consequence of the first two advantages where a redesign using numerical methods can sometimes even be automatic (ie. optimum shape design), whereas experimental methods require a rebuilding or modification of the experimental model.
- The simulation of some inaccessible flow situations are now possible using CFD (ie. aerospacial reentry simulation), which can not yet reproduced by experiments.
- Some experiments, especially those of large scale wind tunnel simulations, may have a limitation of the Reynolds number or wind speed, thus affecting scalability of the found results.
- The tendency of decreasing computer hardware cost while increasing computer efficiency and improving CFD algorithms provides a positive outlook for the reduction of computer simulation costs. Consequently CFD becomes more and more cost-effective versus experimental fluid dynamics.

Drawbacks

However, there remain important drawbacks of numerical simulation to date:

- Turbulence modeling of high Reynolds number complex flows still require excessive amounts of computational power.
- Some design phases need more confidence in the precision of the flow solver as can be assured to date.
- Mesh generation, especially in three dimensions, is still not at a mature phase to date, often requiring more time than the flow solution itself requiring large amounts of human capital and good knowledge of numerical models as well as aerodynamical expertise.

This monograph treats two of the main problems of today's problems related to computational fluid dynamics. One problem dealt with is the assured accuracy of a numerical model, at least for the problems which are presented. At any stage, the accurate simulation of the mathematical and physical phenomena is the declared aim. For example, error estimation and adaptive remeshing techniques are used as a way of ensuring higher accuracy at lower cost.

The other problem which is addressed is the vast amount of time spent in generating (or regenerating) a mesh in the design process. In particular, the formation of a set

of interconnections of nodes to form non overlapping elements in three dimensions can be very troublesome. For this, a meshless technique is proposed that may reduce some of the time spent for the generation of connectivities of points. This monograph is therefore an initiative of this recent field of investigations in fluid dynamics. Other researchers using points without the typical elements have obtained very promising results in other fields of research of numerical methods. The objective herein is to give an introduction of the so called finite point method using clouds of points for the solution of some compressible flow problems.

Finally, this monograph presents some important aspects of improving the cost effectiveness of a given scheme. It is known that there is a tradeoff between a general formulation and computational power. Hence, any competitive numerical algorithm must be optimized for a specific task as far as possible. From the general initial formulation, the objective is to exploit the given resources as far as possible. First, without the loss of generality, the given algorithm can be transformed in such a way to reduce the computational effort significantly by introducing specific procedures which are less expensive than general routines. Then, a computer program must be developed which should be portable to different platforms but optimized for each environment. Of course, the extreme of a fast algorithm is to devise each scheme directly for each system of processors individually, such as a vector processing environment and developing a specific data structure which suits each computer system individually. Recently, the trend for increasing computational power is going towards parallel computing systems in order to be able to meet the demands of CFD which also requires the rethinking of the existing structure of computer codes and the rewriting of existing routines.

1.1 Outline of the Monograph

The following chapter describes the physical properties of fluid flow by stating the governing equations for different types and ranges of fluid dynamics. A general taxonomy of the partial differential equations is given and their implications for the respective equation system and boundary conditions.

In chapter 3, the approximation process of the partial differential equations is treated in order to be able to numerically solve the fundamental equations. Time and spatial discretizations are dealt with. Traditional explicit time integration forms are used, whereas different types of spatial divisioning are presented. Both classical methods like finite element and finite volume schemes as well as new concepts of approximation, such as discretization based on points only, are introduced.

Chapter 4 extends the well known Taylor-Galerkin scheme to quadrilateral elements and a different type of artificial dissipation scheme is used to enhance the accuracy in zones where the velocity is small. Also, error estimation and adaptivity is used to enhance the effectivity of the algorithm and the three dimensional Taylor-Galerkin scheme is simplified to cover axisymmetric flows. Extensive test cases are presented to validate the proposed methodology for different categories in compressible flow and to estimate the accuracy of the method.

The finite point method is presented in chapter 5. The new type of discretization

using only points is applied to two dimensional compressible flow problems. Some of the test cases that are shown in chapter 4 are compared to the results of the new methodology. Also, difficulties and limitations of the proposed method are addressed.

The final chapter is dedicated towards more effectively exploiting the given resources. Thus, optimization of the algorithm, the computer implementation and parallel computing are discussed. Again examples are shown to underline the findings.

1.2 Acknowledgements

First of all, I would like to thank my wife very much for all her support and courage given to me during the course of this research work. I deeply appreciate her sacrifice during these years. I'm also indebted for the confidence and support given to me from my and her parents.

This monograph would not have been possible without the unselfish help and advice of many people. Most of all, I would like to gratefully thank Prof. Eugenio Oñate, the director of this work, who has always encouraged and supported me. Appreciation also goes to Prof. Sergio Idelsohn and Prof. Juan Miquel and Dr. Ramon Codina for their vital help and discussions.

I am also thankful to my friends and colleagues who actively took part in this work: Dr. Ralf Kreiner, Dr. Gabriel Bugeda, Dr. Marcelo Venere and Prof. Reinald Löhner, who not only supplied the mesh generation programs which are so vital in finite element simulations but also valuable advice. Also Dr. Fernando Quintana played an important role for the stimulation in the beginning of this research work. Tilmann Raible contributed to this work with his profound knowledge of LaTeX, which was used to write this work. I am indebted to both Maria Angeles Viciana and Rosa Olea for their help especially in the beginning of my stay in Barcelona.

My special thanks also goes to those people, who in one form or another have made this work easier: Dr. Diego Celentano and his wife Marcela Cruchaga, Orlando Soto and Adriana Blanco, Chris Morton, Ernesto Zangelmi, Uwe Schäfer, Francesc Bassas, Emmanuel Ilpide, Dr. Juan Carlos Cante, Eduardo Martel, Miguel de Riera, Eduardo Car, William Taylor, Oscar Fruitos, Alex Hanganu, Diego Cobo, Ramon Ribó, Mariano Vázquez and all the other members of CIMNE who I did not mention explicitly.

This research work has been partly sponsored by grants from CIRIT (Generalitat de Catalunya), the program of "Human Capital and Mobility" by the European Communities, the CESCA (Centre de Supercomputació de Catalunya) and CEPBA (Centre de Paral·lelització de Barcelona). Some support came from the HERMES contract, Ref. 08W01581, and the ECARP project. These contributions are gratefully acknowledged.

Chapter 2

Physical and Mathematical Background

2.1 Introduction

In this chapter, a basic classification and analysis of the governing equations of fluid flow is made. Generally, speaking the basic equations of flow can be cast into the category of partial differential equations (PDE), containing derivatives in time up to first order and space derivatives up to second order. In addition, the space derivatives can be a non linear function whereas the time derivatives remain linear. More complex flow situations are represented by coupled systems of equations rather than a single equation only.

The following gives a brief overview over notation, the basic mathematical equations and their properties which can describe certain physical situations of fluid dynamics. The definition of systems of conservation laws is used to introduce a short classification of PDEs with some simple examples and its implication on the boundary conditions. Finally, the system of Euler and Navier-Stokes equations is briefly described. It is not intended to give an exhaustive description of the following equations which can be obtained from different sources, ie. [1, 2, 3, 4, 5]. However, for a better understanding of this thesis it is necessary to include the next sections which form the basis of later chapters.

2.2 Notation

In order to avoid confusion with respect to the symbols appearing in this thesis a brief description of the notation is made.

Apart from the expressions explained in the text the following notation is used. Boldface characters denote vectors and matrices, whereas scalars are written in italics. Generally, matrices distinguish themselves from vectors by capital letters. Gradients are denoted by ∇ if they are expressed in compact form. Divergence is generally denoted

as $\frac{\partial(\cdot)_i}{\partial x_i}$ where x_i are the Cartesian coordinates sometimes also written as (x, y, z). The time domain is denoted by $\in [0, T]$ where $T > 0$. If a coordinate transformation occurs ξ, η, ζ are used in three space dimensions. Cylindrical coordinates are denoted by x, r, Θ in this thesis for three dimensions and x, r are used if the problem is axisymmetric.

Δt specifies the time increment, Δx or h is the element height. ϕ refers to an arbitrary function which is usually unknown or sought. Ω stands for the physical space or domain of interest and Γ refers to the boundary of Ω. Since i usually refers to nodal values we define the imaginary number $I = \sqrt{-1}$.

2.2.1 Definition of Boundary Conditions

For the forthcoming sections it is convenient to define the the following different types of boundary conditions which may be applied through this thesis. Let Dirichlet type boundary conditions be defined as [6]:

$$u = f \quad \text{on } \Gamma_D \tag{2.1}$$

Neumann boundary conditions:

$$\frac{\partial u}{\partial n} = g \quad \text{on } \Gamma_N \tag{2.2}$$

and mixed boundary conditions:

$$\alpha u + \beta \frac{\partial u}{\partial n} = h \quad \text{on } \Gamma_M \tag{2.3}$$

where n denotes the boundary normal direction outward, α and β are weighting constants and f, g, h are constants which are prescribed.

2.3 Conservation Laws

A physical system in fluid dynamics may be completely determined by the fundamental concept of conservation laws. This means that for instance mass, momentum and energy are conserved[1]. Thus, the concept of conservation is important for the description of the physics and also has its implication for the numerical simulation of flows. Conservation laws state that any quantity u in a domain Ω changes at a rate equal to its flux $f = f(u)$ through the boundary Γ of Ω:

$$\frac{\partial}{\partial t} \int_\Omega u d\Omega = - \int_\Gamma f_n d\Gamma \tag{2.4}$$

where subscript n denotes the outward normal on Γ. For simplicity only one space dimension is considered. In divergence form, eq. 2.4 can be rewritten as

[1] If shocks appear, another law is necessary for the correct description of the physics: The conservation or increase in entropy $\frac{\partial}{\partial t} \int \rho S dx \geq 0$, also called the entropy inequality or the second law of thermodynamics [1] where S is the entropy and ρ the density.

$$\frac{\partial u}{\partial t} + \frac{\partial f}{\partial x} = 0 \tag{2.5}$$

which is a partial differential equation governing problems, ie. transport for $f = Au$ and A =constant.

2.4 Partial Differential Equations

2.4.1 Types of Partial Differential Equations

It is important, especially for the specification of boundary conditions to give a taxonomy of the governing equations within the class of partial differential equations. For the scope of this thesis, first order and second order PDEs are dealt with. The method of characteristics can be used to classify PDEs. Characteristics are lines, surfaces or families of surfaces on which the solution remains constant. To be called hyperbolic, the characteristics should be real and distinct if more than one exists. If they are complex the PDE is called elliptic, otherwise parabolic.

First Order PDEs

Consider the following first order PDE:

$$a\frac{\partial \phi}{\partial x} + b\frac{\partial \phi}{\partial y} = c \tag{2.6}$$

where a, b, c are constants and ϕ is any unknown function.

A classification by characteristics determines the type of PDE of eq. 2.6. The only characteristic of eq. 2.6 is $(\frac{dy}{dx} = \frac{b}{a})$ real, therefore eq. 2.6 is by definition hyperbolic.

Second Order PDEs

A linear second order partial differential equation can be written in the following form:

$$A\frac{\partial^2 \phi}{\partial x^2} + B\frac{\partial^2 \phi}{\partial x \partial y} + C\frac{\partial^2 \phi}{\partial y^2} + D\frac{\partial \phi}{\partial x} + E\frac{\partial \phi}{\partial y} + F\phi + G = 0 \tag{2.7}$$

where A-G are constants. Then, three main categories of partial differential equations can be classified:

- Elliptic partial differential equations, if

$$B^2 - 4AC < 0 \tag{2.8}$$

- Parabolic partial differential equations, if

$$B^2 - 4AC = 0 \tag{2.9}$$

- Hyperbolic partial differential equations, if

$$B^2 - 4AC > 0 \tag{2.10}$$

As can be seen from above conditions, the classification depends only on the highest order derivatives of the PDE.

2.4.2 Examples

Transport problems in which the dissipation effects are neglected, are generally hyperbolic partial differential equations. Dissipation problems are usually governed parabolic partial differential equations. Dissipation comes, for instance, from the viscous properties of a fluid in motion and gain importance when velocity gradients increase, ie. near a wall. Steady state solutions such as inviscid, incompressible potential flow are governed by elliptic partial differential equations. These types of equations are not explicitly treated within this thesis.

2.5 Transport

Various computational techniques are applied to simple equations for which analytical solutions exist in order to compare strengths and weaknesses in the discretization process. The transport equation as a hyperbolic PDE is a useful test case for computational solutions to test the convective behavior a numerical algorithm. The linear transport equation which describes the propagation of disturbances (without source terms) for a function u is:

$$\frac{\partial u}{\partial t} + A\frac{\partial u}{\partial x} = 0 \quad \text{in } \Omega: \ 0 \leq x \leq L \tag{2.11}$$

where u is a function of x and t with the initial condition, ie. $u(x,0) = g(x)$, g being an arbitrary function of x, A is the known constant convective velocity, and let $t \in [0,T]$ and $T > 0$.

2.5.1 Interpretation by Characteristics

An interpretation by characteristics leads to the introduction of a transformation variable $\xi = x - At$ and $t = t$ so that u is now a function of ξ, t: $u(x,t) = u^*(\xi(x,t),t)$. The substitution in eq. 2.11 results in

$$\frac{\partial u^*}{\partial t} = 0 \tag{2.12}$$

so that u^* is constant in time. The problem of eq. 2.11 reduces to a simpler problem along the characteristics, eq. 2.12. The so called characteristics introduced earlier are defined for the 1D transport equation by

$$\frac{dx}{dt} = A = \text{constant} \tag{2.13}$$

Two features can be observed from eq. 2.13: u is constant along the characteristics and characteristic curves are straight lines. Thus, eq. 2.11 can be interpreted by a motion of u in x-direction without change in shape and at a constant speed A.

To see that the solution u^* is constant in time is also observed from the elementary solution of eq. 2.11 in the system ξ, t by expanding the analytical solution in Fourier series, each mode being:

$$\hat{u}^*(\xi, t) = a e^{I\frac{2\pi}{l}\xi} \tag{2.14}$$

where a is the amplitude, l is the wavelength and $I = \sqrt{-1}$.

2.5.2 Boundary Conditions for the Transport Equation

The method of characteristics serves as a basis for determining the correct boundary conditions to be applied. We know that information is transported along the characteristics. Hence, it is not possible to prescribe boundary conditions when characteristics are leaving the domain. Only when they enter the domain, boundary conditions can be prescribed. In case of incoming characteristics, Dirichlet type (Γ_D) boundary conditions are specified, for outgoing characteristics the specification of the boundary condition is left free.

This is equally true in case of the non linear Burgers equation ($A = A(u) = u$ of eq. 2.11), only that the characteristics must be determined at any time increment because they depend on the solution itself.

2.6 Diffusion Equation

An example of a parabolic PDE is the diffusion equation:

$$\frac{\partial u}{\partial t} - \kappa \frac{\partial^2 u}{\partial x^2} = 0 \quad \text{in } \Omega: \ 0 \le x \le L \ \text{ and } 0 \le t \le T \tag{2.15}$$

where κ is a diffusion constant. The interpretation by the definition of eq. 2.7 leads to that eq. 2.15 can be classified as parabolic because $C = B = 0$.

2.6.1 Boundary Conditions for the Diffusion Equation

The analytical solution is again expanded in a Fourier series. Then, a harmonic of this linear PDE, $\hat{u} = a e^{-\kappa\left(\frac{2\pi}{l}\right)^2 t} e^{I\frac{2\pi}{l}x}$, shows that any mode of the initial distribution, $a e^{I\frac{2\pi}{l}x}$, is damped in time by the term $e^{-\kappa\left(\frac{2\pi}{l}\right)^2 t}$ which affects all other points in the domain. Hence, the specification of an initial distribution $u(x, 0) = u_0(x)$ is necessary. At the contour of Ω any boundary condition is possible, ie. Dirichlet (Γ_D), Neumann (Γ_N) or mixed (Γ_M), see eqs. 2.1, 2.2 and 2.3.

2.7 Convection-Diffusion Equation

The transient convection-diffusion equation results from the addition of the diffusion term of eq. 2.15 to eq. 2.11:

$$\frac{\partial u}{\partial t} + A\frac{\partial u}{\partial x} - \kappa\frac{\partial^2 u}{\partial x^2} = 0 \quad \text{in } \Omega: \ 0 \leq x \leq L \ \text{ and } 0 \leq t \leq T, \ \ T > 0 \tag{2.16}$$

which is of parabolic type according to condition 2.9. The presence of the diffusion term guarantees that the solution of eq. 2.16 will be continuous.

As mentioned earlier, dissipation effects often occur in a narrow layer of the flow domain, such as the boundary layer. Indeed the steady state solution ($\frac{\partial u}{\partial t} = 0$) of the convection-diffusion equation can describe some phenomena occurring in a rapidly changing solution across the boundary layer [3, 6].

2.7.1 Boundary Conditions for the Convection-Diffusion Equation

A characteristic harmonic of eq. 2.16 contains both damping and transportation terms:

$$\tilde{u}(x,t) = ae^{-\kappa\left(\frac{2\pi}{\ell}\right)^2 t} e^{I\frac{2\pi}{\ell}(x-At)} \tag{2.17}$$

So, similar to the diffusion equation, the boundary conditions to be specified for the convection-diffusion equation can be Dirichlet, Neumann or mixed, see eqs. 2.1, 2.2 and 2.3. However, one should be consistent and specify only one condition for the complete problem, ie. Γ_D: $u(0,t) = u_0$ and $u(L,t) = u_L$.

In chapter 3, the convection-diffusion equation will be described in some more detail, together with numerical examples.

2.8 Euler Equations

The non linear n-dimensional Euler equations can be written in conservative form neglecting source terms as

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}_i}{\partial x_i} = 0 \quad i = 1, n \tag{2.18}$$

where $\mathbf{f}$ denotes the vector of convective fluxes which are a function of the vector of conservative variables $\mathbf{u}$ which will be detailed in forthcoming chapters.

These equations describe an inviscid flow system without heat conduction. Eqs. 2.18 are the simplification to the laminar Navier-Stokes equations without the viscous and heat conductions terms. It is a first order non linear partial differential equation describing isentropic flows in regions without discontinuities. The integral form of the eq. 2.18 makes it also possible to solve discontinuities such as shocks. The range of

application is wide, approximating high Reynolds number flows away from viscous regions such as the boundary layer. Often, the effects of viscosity are small so that the Euler equations are a valid model to approximate well the flow physics at a reasonable computational cost. Especially the description of many transonic flow constellations by the Euler equations enjoys great popularity.

2.8.1 Boundary Conditions for the Euler Equations

The Euler equations are a set of first order non linear hyperbolic PDEs. The fact that they are hyperbolic has its definite implications upon the specification of the boundary conditions as described in section 2.5. Applying a linearized characteristics analysis, we obtain the necessary specifications of the boundary conditions. Therefore incoming characteristics are specified as Dirichlet type (Γ_D) whereas outgoing characteristics imply that the boundary condition is left free.

However, in a multidimensional non linear context and depending on the flow conditions (ie. subsonic flow), the specification of the boundary conditions becomes more complicated. Not all characteristics for the characteristic equation system are incoming or outgoing at the same time. In addition, another boundary type, apart from inflow or outflow, is present: the impermeable wall. The wall boundary condition states that no materia is allowed to cross the boundary so that the velocity normal to the surface must be zero:

$$\mathbf{v}_n = 0 \tag{2.19}$$

where $\mathbf{v}_n$ is the velocity normal to the surface.

The detailed description can be found in [6, 7, 8], whereas the specific implementation of these boundary conditions are dealt with in section 4.4.1.

2.9 Navier-Stokes Equations

The widest form for describing continuum fluid flow can be obtained from the full system of Navier-Stokes equations, describing the conservation of mass, momentum and energy. In conservative form for Cartesian coordinates and in absence of source terms the set of N_D-dimensional laminar Navier-Stokes equations are:

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}_i}{\partial x_i} = \frac{\partial \mathbf{g}_i}{\partial x_i} \quad i = 1, N_D \tag{2.20}$$

where the viscous fluxes $\mathbf{g}$ have been introduced. The signification of these terms can again be found in forthcoming chapters or in literature [5, 6, 9].

The Navier-Stokes equations are second order PDEs because of the diffusion terms in the momentum and energy equations. Eqs. 2.20 are according to the above classification generally elliptic. However, most of the time, the viscous terms are nearly insignificant away from boundaries, that is away from gradients in velocities and the set of equations

becomes hyperbolic. Close to the boundary, for instance, the equations are mixed hyperbolic-parabolic due to the differences in magnitude of the viscous terms in stream wise direction and in crosswind direction. Generally speaking, eqs. 2.20 can be classified as parabolic-hyperbolic.

2.9.1 Boundary Conditions for the Navier-Stokes Equations

The application of the correct boundary conditions for the Navier-Stokes equations should be applied compatible with its classification by characteristics as well as to comply with the physical nature of the flow. However, considering that many different flow constellations are possible, this specification may not be unique. Numerically, the boundary conditions are applied similar to the Euler equations above for the inviscid limit or hyperbolic part of eqs. 2.20. The major difference is that the velocity at the wall $\mathbf{v}_W$ is set equal to zero. Two other types of boundary conditions for the wall may specified:

- adiabatic wall of Neumann type

$$q = -k\frac{\partial T}{\partial n} = 0 \tag{2.21}$$

  where the heat flux q across the wall is zero. In eq. 2.21 T is the temperature and k is a constant.

- isothermal wall of Dirichlet type

$$T = T_W \tag{2.22}$$

  where T_W is a specified wall temperature.

A more detailed description of the boundary conditions is shown is section 4.4.2.

2.10 Conclusion

The classification of the different partial differential equations is important for the correct modeling of the boundary conditions. Hence, each type of PDE requires a different approach for inflow, outflow and solid wall. The method of characteristics serves as a basis for determining the propagation properties of each equation. Hence, an interpretation of the flow features by characteristics can state which boundary conditions have to be applied. In more complex non linear flow, the boundary conditions have to be applied in a local manner, because the type of condition may not be known *a priori* as it results from the solution itself.

References

[1] Courant R., Friedrichs K.O. "Supersonic Flows and Shock Waves", Interscience Publishers, New York, 1948

[2] Schade H., Kunz E. "Strömungslehre", Walter de Gruyter, Berlin, 1980

[3] Schlichting H., "Teoría de la Capa Límite", Urmo, Bilbao, 1972

[4] Bronstein I.N., Semendjajew K.A. "Taschenbuch der Mathematik", Harri Deutsch, Thun, 1985

[5] Hirsch, C. "Numerical Computation of Internal and External Flows", Volume 1, Wiley, Chichester, 1989

[6] Peraire, J. "A Finite Element Method for Convective Dominated Flows". PhD Thesis at University College of Swansea, 1986

[7] Peiro, J. "A Finite Element Procedure for the Solution of the Euler Equations on Unstructured Meshes". PhD Thesis at University College of Swansea, 1989

[8] Usab W. J., Murman E., "Embedded Mesh Solutions of the Euler Equations Using a Multiple Grid Method", Advances in Computational Transonics (W.G. Habashi, Ed.), Pineridge Press (UK) 447-472, 1985.

[9] Zienkiewicz O.C., Taylor R.L. "The Finite Element Method" 4th Edition, Volume 2, Mc Graw Hill, 1991

Chapter 3

Numerical Modeling in 1D

3.1 Discretization Techniques

Many engineering fields such as fluid dynamics are already relying on computational techniques for the fast and cost effective resolution of their problems. Guided by mathematical equations, many physical problems can be approximated. However, exact solutions to complex problems can often not be solved analytically and other approaches must be taken. So, numerical modeling of mathematical equations has evolved in recent years because of the availability of modern computer technology which is providing large computational capacity for industry. Generally speaking, the idea behind numerical techniques is to divide continuum problems of a previously defined system into discrete subspaces for which a local equilibrium can be stated. Then, it is necessary to assemble the subspaces in order to obtain the approximation for the complete system. There is an increasing trend for industry to use numerical methods, so that different techniques have evolved to solve their problems.

Assuming that we are solving partial differential equations governed by time and space derivatives, we can divide the discretization process into time and space. In this chapter, there is a brief description of some classical methods for the time and space discretization, followed by the introduction of a young concept for the space discretization using clouds of points, which does not require the typical grid or mesh.

3.1.1 Finite Difference Method

As one of the first methods for space discretization, the main idea in the finite difference method is simple. In order to construct the derivative for a given point x, it makes use of the definition of the derivative of an arbitrary function, say ϕ [1]. Performing Taylor expansions around the point x, the different forms of difference approximations can be obtained, such as forward differences, backward difference and central differences.

Let Ω be divided into equal subspaces of h. So, around an interior point i, the node $i-1$ is to its left and node $i+1$ is to the right (Figure 3.1) for the typical finite difference notation. Then, for instance, a central difference approximation of the first

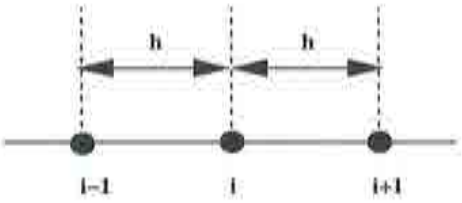


Figure 3.1 : Finite difference discretization around an interior node i within domain Ω in one dimension.

derivative around point i is defined as

$$\left(\frac{\partial \phi}{\partial x}\right)_i = \frac{\phi_{i+1} - \phi_{i-1}}{2h} + O(h^2) \tag{3.1}$$

and a central difference approximation of the second derivative around point i is defined as

$$\left(\frac{\partial^2 \phi}{\partial x^2}\right)_i = \frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{h^2} + O(h^2) \tag{3.2}$$

Another possibility for approximating the first derivative would be a forward difference like:

$$\left(\frac{\partial \phi}{\partial x}\right)_i = \frac{\phi_{i+1} - \phi_i}{h} + O(h) \tag{3.3}$$

The subscripts refer to values at the nodes and the terms $O(\cdot)$ denote the order of the truncation error.

A major drawback of the finite difference method is the need for structured meshes for multidimensional calculations, making it less practical for more complex calculations. On the other hand, it is very useful for comparisons with other methods and validation of some approaches because of its simplicity. In particular, it is possible to give an estimation of the order of accuracy.

Order of Approximation

The concept of the Taylor expansion used for the derivation of finite difference formulas is very important for estimating the truncation error and, hence, the order of approximation. The order of approximation gives an estimate of the power by which the truncation error tends toward zero when the mesh size h tends toward zero. Thus, equation 3.1 gives an approximation to the first derivative of ϕ with second order accuracy for equal mesh spacings.

From the definition of the order of approximation, the accuracy of the space discretization can be estimated. CFD solvers should generally be higher order accurate (at least second order) for the overall solution of "smooth" parts of the flow. However, in regions of discontinuities, such as shocks, it is accepted to switch to a first

order approximation, ie. eq. 3.3, because high order solutions can exhibit unphysical oscillations. In later sections these concepts will treated in more detail.

3.1.2 Finite Element Method

The finite element method (FEM) is probably the most popular and most successful numerical method for dealing with all types of numerical engineering problems. Also in the field of CFD it has become one of the popular methods of solution. The FEM has its main advantage in its general formulation, making it possible to use it in an elegant form with mathematical rigorousity. The subdivision of a flow field into non-overlapping elements, proper to this method, also provides an easy way to understand the physics of the computational process and a straightforward basis for the calculations.

The cost of generality, however, can be quite high such that the FEM needs modification to be competitive with more specialized methods in CFD as will be shown later on. The following outline describes several methods, especially concentrating on deriving methods for compressible flow.

Petrov-Galerkin Method

A very popular finite element method for CFD is the so called streamline upwind Petrov-Galerkin (SUPG) method successfully implemented by other authors, ie. [2, 3, 4]. The power of Petrov-Galerkin methods is the natural way by which balancing diffusion (or upwinding) can be incorporated into the numerical scheme.

Let us demonstrate this behavior using a classical example, for instance the one dimensional convection-diffusion equation 2.16 with the convective velocity A and a diffusion constant κ, but without source terms:

$$\frac{\partial u}{\partial t} + A\frac{\partial u}{\partial x} - \frac{\partial}{\partial x}\left(\kappa\frac{\partial u}{\partial x}\right) = 0 \quad \text{in } \Omega: \ 0 \leq x \leq L \ \text{ and } 0 \leq t \leq T \tag{3.4}$$

with $u = u(t,x)$ in Ω; $u(t,0) = u_0$ in Γ_0, $u(t,L) = u_L$ in Γ_L and $\Gamma = \Gamma_0 \cup \Gamma_L$.

At steady state ($\frac{\partial u}{\partial t} = 0$) equation 3.4 becomes:

$$A\frac{\partial u}{\partial x} - \frac{\partial}{\partial x}\left(\kappa\frac{\partial u}{\partial x}\right) = 0 \ \text{ in } \Omega: \ 0 \leq x \leq L \tag{3.5}$$

with $u = u(x)$.

Taking A and κ constant, the analytical solution of this first order homogeneous differential equation is obtained as:

$$u(x) = u_0 + (u_L - u_0)\frac{1 - e^{\frac{A}{\kappa}x}}{1 - e^{\frac{A}{\kappa}}} \tag{3.6}$$

With $u_0 = 1$ and $u_L = 0$, equation 3.6 reduces to

$$u(x) = 1 - \frac{1 - e^{\frac{A}{\kappa}x}}{1 - e^{\frac{A}{\kappa}}} \tag{3.7}$$

Eq. 3.5 is discretized in finite element manner using weighting functions W and replacing u within each element with standard Galerkin shape functions N as:

$$u = \sum_i N_i \bar{u}_i = \mathbf{N}\bar{\mathbf{u}} \quad \text{and} \quad \frac{\partial u}{\partial x} = \sum_i \frac{\partial N_i}{\partial x}\bar{u}_i = \frac{\partial \mathbf{N}}{\partial x}\bar{\mathbf{u}} \tag{3.8}$$

where $\bar{u}_i$ are the nodel values of unknown function and the summation extends across the nodes of the element. We obtain the weak formulation of eq. 3.5 with $(W(0) = W(L) = 0)$, omitting the overbar in what follows:

$$\int_\Omega \mathbf{W}^T A \frac{\partial \mathbf{N}}{\partial x}\mathbf{u}d\Omega - \int_\Omega \mathbf{W}^T (\frac{\partial}{\partial x}\kappa\frac{\partial \mathbf{N}}{\partial x})\mathbf{u}d\Omega = 0 \tag{3.9}$$

Integration by parts, using Green's theorem, the following system is reached:

$$\int_\Omega \mathbf{W}^T A \frac{\partial \mathbf{N}}{\partial x}\mathbf{u}d\Omega + \int_\Omega \frac{\partial \mathbf{W}^T}{\partial x}\kappa\frac{\partial \mathbf{N}}{\partial x}\mathbf{u}d\Omega = 0 \tag{3.10}$$

The integral across the boundary resulting from integration by parts $\int_\Gamma$ is equal to zero and therefore ignored. The original SUPG formulation which adds an original diffusion to eq. 3.5 and results in an additional sum over the element interiors, is not presented here [28]. Instead, the equivalent use of Petrov-Galerkin weighting functions is shown.

The weighting functions for the diffusive part of eq. 3.10 are replaced by standard Galerkin shape functions N, whereas the following Petrov-Galerkin type weighting functions are used only for the convective part:

$$W = N + \alpha\frac{h}{2}\frac{\partial N}{\partial x} \tag{3.11}$$

h is the element length and α is a parameter which will be defined later. The final equation system to be resolved is then

$$\int_\Omega \mathbf{N}^T A \frac{\partial \mathbf{N}}{\partial x}\mathbf{u}d\Omega + \int_\Omega \alpha\frac{h}{2}\frac{\partial \mathbf{N}^T}{\partial x}A\frac{\partial \mathbf{N}}{\partial x}\mathbf{u}d\Omega + \int_\Omega \frac{\partial \mathbf{N}^T}{\partial x}\kappa\frac{\partial \mathbf{N}}{\partial x}\mathbf{u}d\Omega = 0 \tag{3.12}$$

which is equivalent to the following finite difference equation system [5]:

$$[1 + Pe\,(\alpha + 1)]\,u_{i-1} - 2\,[1 + \alpha Pe]\,u_i + [1 + Pe\,(\alpha + 1)]\,u_{i+1} = 0 \tag{3.13}$$

where Pe is the dimensionless Peclet number, sometimes also called the local Reynolds number. The Peclet number indicates the relation between the convective velocity A and the diffusion κ together with the element height h and is defined as:

$$Pe = \frac{|A|h}{2\kappa} \tag{3.14}$$

Exact nodal solutions are obtained, if an optimal upwind parameter α_{opt} is used as the truncation error is zero [2, 3, 4, 5, 6, 11]:

$$\alpha_{opt} = coth(Pe) - \frac{1}{Pe} \tag{3.15}$$

Graphically, the result for the 1D convection-diffusion equation for $Pe = 1$ and $A < 0$ and the boundary conditions Γ_D: $u_0 = 1$, $u_L = 0$ are plotted in Figure 3.2.

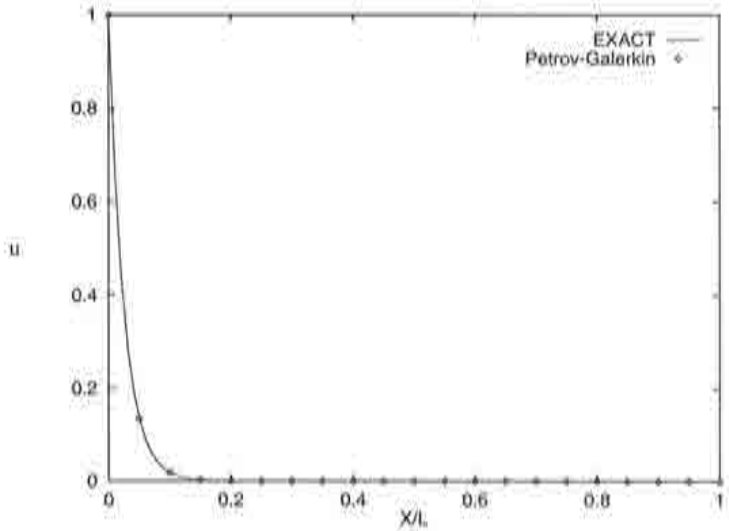


Figure 3.2 : Exact solution to the convection-diffusion equation for $Pe = 1$, $u_0 = 1$, $u_L = 0$ and $A < 0$ using a Petrov-Galerkin scheme for $\alpha = \alpha_{opt}$.

Taylor-Galerkin Method

In the previous section, it was shown that the Petrov-Galerkin scheme provides nodally exact results if α_{opt} is used for the Petrov-Galerkin weighting functions. Here, the derivation of the well known Taylor-Galerkin is described which serves as a basis for the finite point method explained in forthcoming sections. The advantage of the Taylor-Galerkin method is that the use of non symmetric weighting functions is avoided.

The Taylor-Galerkin scheme is an iterative process in time [12]. A test for time marching schemes is solving equation 3.4 by iterating until steady state is reached to approximate the exact result. This is usually done by expanding equation 3.4 in time using a Taylor series:

$$u^{n+1} = u^n + \sum_{i=1}^{\infty} \frac{\Delta t^i}{i!} \frac{\partial^i u}{\partial t^i} \tag{3.16}$$

where Δt is a small time increment. Expanding eq. 3.16 up to second order, we get

$$\Delta u = u^{n+1} - u^n = \Delta t \frac{\partial u^n}{\partial t} + \frac{\Delta t^2}{2} \frac{\partial^2 u^n}{\partial t^2} + O\left(\Delta t^3\right) \tag{3.17}$$

substituting eq. 3.4 in eq. 3.17 and neglecting derivatives of order three and higher (which can't be represented by linear elements) and the term $O(\Delta t^3)$

$$\Delta u \simeq \Delta t \left(-A\frac{\partial u}{\partial x} + \kappa \frac{\partial^2 u}{\partial x^2} \right) + \frac{\Delta t^2}{2} \left(A^2 \frac{\partial^2 u}{\partial x^2} \right) \tag{3.18}$$

Discretization of eq. 3.18 using Galerkin weighting functions $W = N$, eq. 3.8 to approximate u and dividing by Δt, we obtain omitting the overbar

$$\int_\Omega \mathbf{N}^T \mathbf{N} d\Omega \frac{\Delta \mathbf{u}}{\Delta t} = \mathbf{M}\frac{\Delta \mathbf{u}}{\Delta t} = \tag{3.19}$$

$$-\int_\Omega A\frac{\partial \mathbf{N}}{\partial x}\mathbf{u}d\Omega - \int_\Omega \frac{\partial \mathbf{N}^T}{\partial x}\kappa\frac{\partial \mathbf{N}}{\partial x}\mathbf{u}d\Omega - \frac{\Delta t}{2}\int_\Omega A^2 \frac{\partial \mathbf{N}^T}{\partial x}\frac{\partial \mathbf{N}}{\partial x}\mathbf{u}d\Omega \tag{3.20}$$

where $\mathbf{M}$ denotes the consistent mass matrix, usually approximated by a diagonalized form using the row-sum lumping technique. At steady state $\frac{\Delta \mathbf{u}}{\Delta t} = 0$, equivalence to the Petrov-Galerkin scheme can be shown by comparing eq. 3.12 to eq. 3.20, yielding

$$\alpha \frac{h}{2} A = \frac{\Delta t}{2} A^2 \tag{3.21}$$

This gives the following condition for Δt in order to obtain nodally exact results [6]

$$\Delta t_{opt} = \frac{h}{A}\alpha_{opt} \tag{3.22}$$

3.1.3 Stability

However, it is not yet known if Δt_{opt} of eq. 3.22 actually satisfies stability criteria. This section presents the stability requirements for the solution of the convection-diffusion eq. 3.4 using the explicit time marching Taylor-Galerkin scheme.

For any numerical scheme which approximates a PDE by discretization in space and time, a relationship between the mesh size h and the time increment Δt can be defined. The Courant number C determines this relation using the convective velocity $|A|$ by

$$C = |A|\frac{\Delta t}{h} \tag{3.23}$$

The objective of a numerical scheme is that the error with respect to the exact solution for a numerical iteration should not grow unbounded. This behavior is called stability. The error can be written as

$$\epsilon_i^n = \hat{u}_i^n - u_i^n \tag{3.24}$$

where the hat denotes the exact nodal solution. Superscript n refers to the time instant n and the subscript is the point in the mesh. The limits of C and therefore Δt are determined by the problem type, as well as by the time and spatial discretization method.

So, every algorithm has its own stability properties. Depending on the formulation of the algorithm, there exist three general categories:

1. A limit upon the time step Δt or C, leading to a "conditionally stable" scheme, ie. explicit schemes,
2. no limit upon Δt or C, forming a "unconditionally stable" scheme, usually achieved by implicit schemes and
3. "unconditionally unstable" schemes leading to unstable results no matter how Δt is chosen.

Hence, the objective is to find the limit of C for the explicit Taylor-Galerkin scheme.

A possibility to determine the stability properties of a scheme can be obtained from a Von Neumann stability analysis using a Fourier decomposition which only gives a necessary condition, but works well in practice. The error ϵ^n at time n, defined by eq. 3.24, which also satisfies the discretized equation system can then be expanded by a Fourier series, for which the amplitude of the error of any mode is introduced as E^n. Stability requires that E^n must not grow in time. Or, the modulus of the so called amplification factor $|G| = \left|\frac{E^{n+1}}{E^n}\right| \leq 1$. Since the error satisfies the same discretized equation we can also write

$$|G| = \left|\frac{u^{n+1}}{u^n}\right| \leq 1 \quad \text{or} \quad |G|^2 \leq 1 \tag{3.25}$$

More details can be found in [1] [1]. The discretized equation system of eq. 3.20 for an interior node i on a mesh of equal mesh spacing $h = const. > 0$ (Figure 3.1) can be written in finite difference form as:

$$u_i^{n+1} = u_i^n - \left[\frac{C}{2}\left(u_{i+1} - u_{i-1}\right) + \left(\frac{C}{2Pe} + \frac{C^2}{2}\right)\left(u_{i+1} - 2u_i + u_{i-1}\right)\right]^n \tag{3.26}$$

where superscript n indicates the time instant. The analytical solution is expanded by a Fourier series where any mode is $\hat{u}(x,t) = ae^{-(\xi+I\omega)t}e^{Ikx}$ where a is the amplitude, ξ is the damping, ω is the frequency, k any real number and $I = \sqrt{-1}$. Then, we obtain for any interior node i and time instant n the harmonic

$$\hat{u}_i^n = ae^{-(\xi^h+I\omega^h)n\Delta t}e^{Iikh} \tag{3.27}$$

which is evaluated at $(x_i, t^n) = (ih, n\Delta t)$. $h = |x_i - x_{i-1}| = |x_{i+1} - x_i|$ is the mesh size, the algorithmic damping and algorithmic frequency are ξ^h and ω^h, respectively. Inserting eq. 3.27 into eq. 3.25, the exponential term containing the damping ξ and

[1] If more than one equation exists, the amplification factor becomes a matrix: $\mathbf{G}$. The Von Neumann necessary but not sufficient stability criterion requires that the spectral radius of the amplification matrix $\rho(\mathbf{G})$ meets $\rho(\mathbf{G}) \leq 1$.

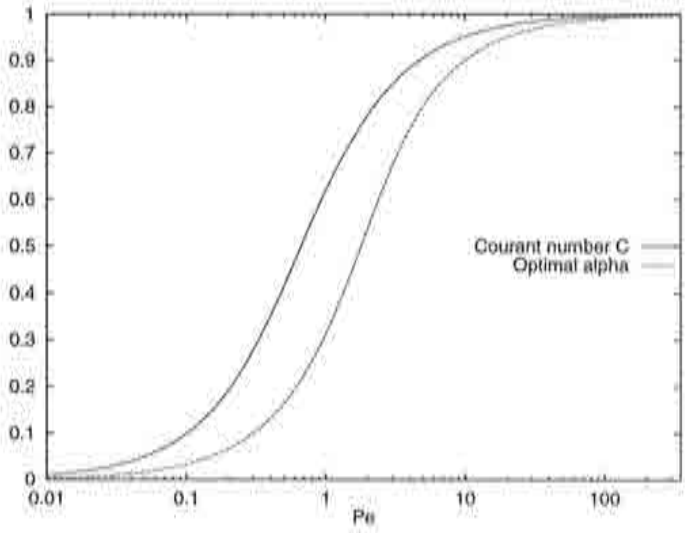


Figure 3.3 : Plot for the limits of α_{opt} and the Courant number C. Note that always $\alpha_{opt} < C$. The abscissa is in logarithmic scale and only plots Pe ranging from 0.01 to 350 for clarity.

frequency ω cancels by division and we can write the modulus of the amplification factor for any mode after introducing trigonometric functions and simplification

$$|G|^2 = \left|\frac{\hat{u}_i^{n+1}}{\hat{u}_i^n}\right|^2 = \left[1 + \left(\frac{C}{Pe} + C^2\right)(cos(p) - 1) - ICsin(p)\right]^2 \leq 1 \tag{3.28}$$

where $p = kh$. The square of real and imaginary parts for above inequality, leads to the following inequality after simplification:

$$\left(\frac{C}{Pe} + C^2\right)^2 (cos(p) - 1) + 2\left(\frac{C}{Pe} + C^2\right) - C^2\,(cos(p) + 1) \geq 0 \tag{3.29}$$

which is satisfied under two conditions: if $C^2 \leq \frac{C}{Pe} + C^2$, which is always satisfied for $C > 0$ by definition, and if

$$\frac{C}{Pe} + C^2 \leq 1 \tag{3.30}$$

Within the interval $[-\frac{1}{2Pe} - \sqrt{\frac{1}{4Pe^2} + 1}, -\frac{1}{2Pe} + \sqrt{\frac{1}{4Pe^2} + 1}]$, the values for the Peclet number comply with the inequality and hence a condition for the Courant number C is

$$C \leq -\frac{1}{2Pe} + \sqrt{\frac{1}{4Pe^2} + 1} \tag{3.31}$$

The expression for the critical value of Δt_{crit} becomes:

$$\Delta t_{crit} = C\frac{h}{A} \tag{3.32}$$

It can be graphically shown (see Figure 3.3) that $\alpha_{opt} < C$ for $Pe > 0$ (for clarity, a limited range of Pe is displayed in logarithmic scale) and therefore,

$$\Delta t_{opt} < \Delta t_{crit} \tag{3.33}$$

which satisfies stability criteria. The optimal time increment Δt_{opt} is given by eq. 3.22 for the Taylor-Galerkin scheme in order to obtain nodally exact results. Figure 3.4 shows the exact nodal values computed using a Taylor-Galerkin two-step scheme with Δt_{opt} and $Pe = 1$ [16, 6].

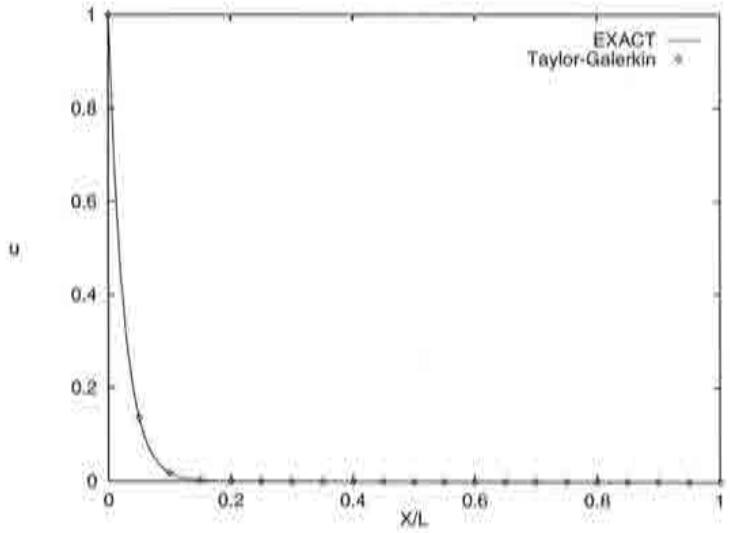


Figure 3.4 : Exact solution to the convection-diffusion equation for $Pe = 1$, $u_0 = 1$, $u_L = 0$ and $A < 0$ using a Taylor-Galerkin scheme.

For the inviscid limit $Pe \to \infty$, stability requires that $C \leq 1$, which will also apply for the non linear Euler equations using a similar definition for C.

Remark

- A stability criteria was demonstrated for solving the convection-diffusion equation using a Taylor-Galerkin scheme for interior nodes only, but not for nodes on the boundary. Thus, this method is not rigorous, but practical considerations make it effective. Generally, the influence of the boundary and the boundary conditions is not that important for the stability of the scheme, because the unstable behavior of the algorithm is induced away from boundaries [4].

- Two concepts, consistency and convergence, are not exposed here in detail. Consistency refers to the fact that the discretized equation should tend towards the differential equation if Δt and h tend to zero, or in other words the truncation error tends to zero for $\Delta t,\ h \to 0$. This analysis is done by expanding the discrete values u_i^n in a Taylor series and replacing them in eq. 3.4. Convergence relates the

computed solution to the exact solution of the differential equation. Consistence and stability are necessary and sufficient conditions for convergence to exist [1].

3.2 Finite Point Method

3.2.1 Introduction

In the following, a very recent concept of discretization is described. The idea is to achieve the computer modeling of flow problems using an approximation based on "clouds of points" which does not require the definition of a mesh. The so called Finite Point Method (FPM) [13] is presented showing some examples for the solution of the 1D convection-diffusion equation in this chapter and 2D compressible flows in forthcoming chapters.

In previous sections, the finite element method (FEM) has been presented. Using unstructured meshes in more than one dimension, the main advantage is its ability to deal with complicated multidimensional domains in a simple manner while maintaining a local character in the approximation. For instance, the weak form for the finite element formulation of eq. 2.18 is:

$$\int_\Omega W \frac{\partial \mathbf{u}}{\partial t} d\Omega + \int_\Omega W \frac{\partial \mathbf{f}_i}{\partial x_i} d\Omega = 0 \quad i = 1, N_D \tag{3.34}$$

where W are weighting functions and N_D is the number of space dimensions of the problem.

Another widely used methodology in CFD is the finite volume method (FVM). Using eq. 2.4, the integral form of conservation laws for a discrete control volume volume Ω_V using finite volumes can be written for eq. 2.18:

$$\int_{\Omega_V} \frac{\partial \mathbf{u}}{\partial t} d\Omega_V + \int_\Gamma \sum_{i=1}^{N_D} (\mathbf{f}_i n_i)\, d\Gamma = 0 \tag{3.35}$$

where the spatial integration extends across each boundary Γ of a local control volume Ω_V, $\mathbf{f}_i$ is the flux of $\mathbf{u}$ in direction x_i and n_i are the Cartesian components of the outward normal on Γ.

Both methods divide the total domain into a finite number of subdomains (or elements) and integrate across the volume. Thus, a conservative discretization is limited by geometrical regularity of non overlapping elements, ie. positive volume or a limited aspect ratio between elements, angles, etc. Even though this poses no difficulties for 2D situations, the lack of efficient 3D mesh generators makes the solution of 3D problems a difficult task.

It is widely acknowledged that efficient 3D mesh generation remains one of the big challenges in finite element and finite volume computations. Even very complex problems in CFD, such as some 3D solutions of the Navier-Stokes equations, can be very accurately resolved if acceptable 3D meshes and sufficient computational power

are available. However, the generation of 3D meshes, despite its recent advances is still a bottle neck and it can consume far more time and effort than the numerical solution itself. Very often, the design engineer needs to modify an existing mesh by adding or removing nodes in certain mesh regions, which usually requires a remeshing process. Since a lot of manpower is dedicated to the task of mesh generation, a large time and cost reduction in the design process can be achieved by optimizing the discretization process.

Different authors have recently investigated the possibility of deriving numerical methods without using meshes. Nayroles *et al* [7] proposed a technique, calling it diffuse element method (DEM), where only some nodes and a boundary description is necessary to formulate the Galerkin equations. The interpolating functions are polynomials fitted to the nodel values by a least squares approximation. Although no finite element mesh is explicitly required in this method, still some kind of "auxiliary grid" is needed in order to compute numerically the integral expressions deriving from the Galerkin approach. This requirement may prelude the successful extension of the DEM to 3D problems.

More recently, Belytschko *et al* [8] have proposed an extension of the DEM which they call the element-free Galerkin (EFG) method. In that work, generalized moving least squares interpolants typically exploited in curve and surface fitting are used to define the local approximation. This provides additional terms in the derivatives of the unknowns field omitted by Nayroles *et al* [7]. In addition, a regular cell structure is chosen as the "auxiliary grid" to compute the integrals by means of a higher order quadrature. Finally, Lagrange multipliers are used to enforce the essential boundary conditions. The same approach has been further generalized by Liu *et al* [9] by introducing concepts from wavelet theory.

The use of "clouds of points" to define local approximations is not new and it has enjoyed some popularity among finite difference (FD) practitioners to derive generalized FD schemes for arbitrary irregular grids. Here typically the concept of a "star" of nodes is introduced to derive FD approximations by means of a local Taylor expansion using the information by the number and position of nodes contained in each star. These ideas have been successfully applied in fluid mechanics under the name of Smooth Particle Hydrodynamics Method. A recent extension of these concepts to the solution of high speed flow problems has recently been attempted by Batina [10].

In the following, a general methodology for the numerical solution of the convection-diffusion equation in 1D using a finite set of arbitrary points is described. The proposed approach incorporates the main features of generalized finite difference schemes and other more recent point data based procedures such as the DEM and the EFG [13]. The accuracy and applicability of this method is shown in some examples of application in 1D. The theoretical basis of the method in the context of the solution of viscous and inviscid flows are described in some detail in chapter 5 with some typical examples of 2D compressible flows.

3.2.2 General Formulation

Let us describe a local interpolating domain Ω_i, also called "clouds of points", by a set of n points as seen in Figure 3.5. Point i is the central node of Ω_i and points j are the other points forming part of Ω_i. Hence subscript i and j refer to these points in the following sections.

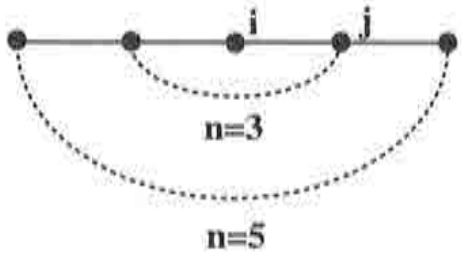


Figure 3.5 : Local interpolating domain Ω_i in one dimension

With the help of these points a sought function $u(x)$ can be approximated by a new function $\hat{u}(x)$

$$u(x) \simeq \hat{u}(x) \tag{3.36}$$

Performing a polynomial expansion of order m the function $\hat{u}(x)$ can be described within Ω_i as

$$\hat{u}(x) = a_1 + a_2 x + a_3 x^2 + ... + a_m x^{m-1} \tag{3.37}$$

where a_m are polynomial coefficients. In vectorial form we can write

$$\hat{u}(x) = \mathbf{p}^T(x)\mathbf{a} \tag{3.38}$$

where the base functions $\mathbf{p}$ are introduced. The vector $\mathbf{a}$ contains the polynomial coefficients a_m. For the linear and quadratic case, they become

$$\mathbf{p}^T = [1, x] \text{ for } m = 2 \tag{3.39}$$

$$\mathbf{p}^T = [1, x, x^2] \text{ for } m = 3 \tag{3.40}$$

in one dimension [13].

The above approximation can now be sampled at n points within Ω_i where the values u_i are known. They are defined as

$$u_i = u(x_i) \tag{3.41}$$

For the local interpolating domain, the following expression is obtained where the unknown polynomial coefficients $\mathbf{a}$ are sought.

$$\mathbf{u} = \begin{Bmatrix} \mathbf{p}_1^T \\ \mathbf{p}_2^T \\ \vdots \\ \mathbf{p}_n^T \end{Bmatrix} \mathbf{a} = \mathbf{C}\mathbf{a} \tag{3.42}$$

where $\mathbf{p}_j^T = \mathbf{p}^T(x_j)$. The vector $\mathbf{u}$ contains the nodal values u_i and $\mathbf{C}$ is a nxm matrix.

Since the unknown polynomial coefficients $\mathbf{a}$ are on the right hand side, it is obvious that they can not be calculated unless $\mathbf{C}$ is square and can be inverted. Another possibility to obtain the coefficients $\mathbf{a}$ is to perform a least squares fit which minimizes the distances between $\hat{u}(x)$ and u. For a solution to exist, a necessary condition is that $\text{rank}(\mathbf{C}) = m$, which means that

$$n \geq m \tag{3.43}$$

is necessary at least. This becomes evident, for instance, if we construct a linear function ($m = 2$) in each Ω_i, at least 2 points ($n = 2$) are needed to reproduce it. The condition $n \geq m$ is not sufficient, however, as will be explained in chapter 5.

In the following, let us assume that the rank of $\mathbf{C}$ is equal to m, and that because of careful selection of nodes, $n \geq m$ is sufficient. Then, it is possible to distinguish two cases. One is $n = m$, which means that this is the minimum number of nodes in each interpolation domain, and the other case: $n > m$ for which $\mathbf{C}$ is overdetermined.

Finite Element Interpretation

Consider a linear one dimensional finite element. Let the local interpolating domain be the nodes of that element and let the order of approximation be equal to that of the finite element. In this particular case $n = m = 2$ and by inverting $\mathbf{C}$ of eq. 3.42 and substituting into eq. 3.38, we obtain

$$\hat{u}(x) = \mathbf{p}^T\mathbf{C}^{-1}\mathbf{u} \tag{3.44}$$

A closer look at the vector $\mathbf{p}^T\mathbf{C}^{-1}$ reveals that it is composed of the standard finite element shape functions $\mathbf{N}$ [14] for a linear one dimensional finite element

$$\mathbf{p}^T\mathbf{C}^{-1} = \mathbf{N}^T \tag{3.45}$$

The equivalence to finite differences and linear finite elements is shown later in this section. For an adequate choice of the basis functions, the shape functions of other (also multidimensional) elements can also be recovered, however, the global scheme may not be equivalent to the proposed scheme anymore, because overlapping interpolation domains can arise.

3.2.3 Least squares interpolation

Increasing the number of nodes in Ω_i to $n > m$, matrix $\mathbf{C}$ is overdetermined and we cannot directly invert $\mathbf{C}$ anymore. However, a function $\hat{u}(x)$ can be defined by minimizing the square distances of the nodal unknowns u_i to $\hat{u}(x)$. The sum of the square distances can be cast into a function J

$$J = \sum_{j=1}^{n} (u_j - \hat{u}(x_j))^2 \tag{3.46}$$

Replacing $\hat{u}(x_j)$ by eq. 3.38, the definition for J can be written as a function of $\mathbf{a}$

$$J(\mathbf{a}) = \sum_{j=1}^{n} (u_j - \mathbf{p}_j^T \mathbf{a})^2 \tag{3.47}$$

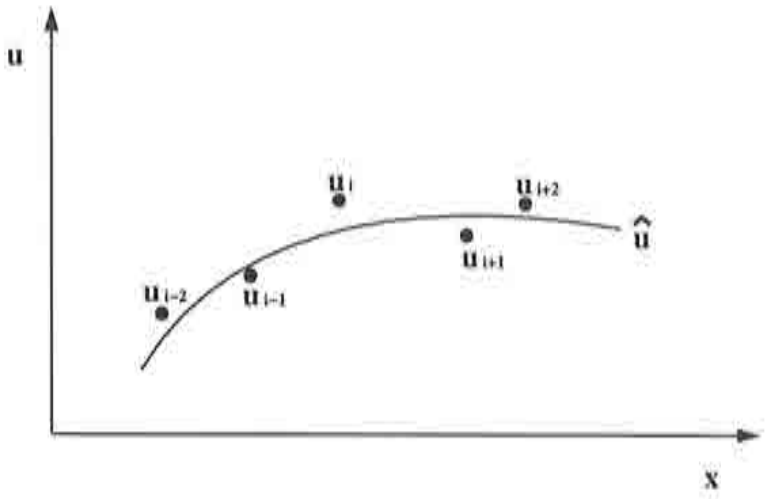


Figure 3.6 : Nodal unknowns u and the interpolated function $\hat{u}$.

Minimization of J with respect to $\mathbf{a}$, means that the derivative of J must be zero:

$$\frac{\partial J}{\partial \mathbf{a}} = 0 \tag{3.48}$$

The derivative of J with respect to $\mathbf{a}$ is from eq. 3.47

$$\frac{\partial J}{\partial \mathbf{a}} = \begin{Bmatrix} \frac{\partial J}{\partial \mathbf{a}_1} \\ \frac{\partial J}{\partial \mathbf{a}_2} \\ \vdots \\ \frac{\partial J}{\partial \mathbf{a}_m} \end{Bmatrix} = \sum_{j=1}^{n} 2 \left(u_j - \mathbf{p}_j^T \mathbf{a} \right) \begin{Bmatrix} 1 \\ x \\ \vdots \\ x^{m-1} \end{Bmatrix} = \sum_{j=1}^{n} 2 \left(u_j - \mathbf{p}_j^T \mathbf{a} \right) \mathbf{p} \tag{3.49}$$

or, since $(u_j - \mathbf{p}_j^T \mathbf{a})$ is a scalar variable, we can rewrite eq. 3.49 and incorporate the minimization condition of eq. 3.48

$$\frac{\partial J}{\partial \mathbf{a}} = \sum_{j=1}^{n} 2\left(\mathbf{p}u_j - \mathbf{p}\mathbf{p}_j^T\mathbf{a}\right) = 0 \tag{3.50}$$

where the factor 2 disappears by division. Using eq. 3.42 we simply obtain

$$\mathbf{C}^T\mathbf{C}\mathbf{a} = \mathbf{C}^T\mathbf{u} \tag{3.51}$$

or for the unknown polynomial coefficients

$$\mathbf{a} = \mathbf{A}^{-1}\mathbf{B}\mathbf{u} \tag{3.52}$$

for which the following definition for $\mathbf{A}$ and $\mathbf{B}$ have been used

$$\mathbf{A} = (\mathbf{C}^T\mathbf{C}) \text{ and} \tag{3.53}$$

$$\mathbf{B} = \mathbf{C}^T \tag{3.54}$$

The new matrix of "shape functions" for each interpolating domain Ω_i are now obtained as

$$\mathbf{N}^T = \mathbf{p}^T\mathbf{A}^{-1}\mathbf{B} \tag{3.55}$$

This means that an interpolated curve $\hat{u}(x)$ is generated from some point values u_j $(j = 1, \cdots, n)$ in each cloud as shown in Figure 3.6. Note, as opposed to standard finite elements the fitted curve does not necessarily pass through the nodal unknowns u_j.

Remark

In the present approach the danger of a singular matrix $\mathbf{A}$ is avoided by appropriately selecting the points in the interpolation domain Ω_i, as mentioned earlier. In addition, the careful selection of points reduces both computational cost and memory, because only those points actually contributing to the solution in a certain region are selected. A more detailed description of the selection of points is found in the chapter 5 dealing with the two dimensional finite point method.

Recently, Batina [10] has used a similar type of least squares fit for fluxes and stresses in compressible flow analysis. However, he avoids the direct inversion of matrix $\mathbf{A}$ by doing a QR decomposition, thus circumventing the difficulty of $\mathbf{A}$ being singular and solves the following equation system instead [10]:

$$\mathbf{C}\mathbf{a} = \mathbf{u} \tag{3.56}$$

Now, performing a QR decomposition of matrix $\mathbf{C}$, which is not square, with the condition $\mathbf{Q}^T\mathbf{Q} = I$, we obtain

$$\mathbf{QRa} = \mathbf{u} \tag{3.57}$$

Multiplication of both sides by $\mathbf{Q}^T$, we get an equation system for which the sought coefficients $\mathbf{a}$ can be calculated:

$$\mathbf{Ra} = \mathbf{Q}^T\mathbf{u} \tag{3.58}$$

The author has also tested this approach suggested in [10], but has found that it is computationally more expensive. The multiplication of $\mathbf{Q}^T\mathbf{u}$ and the backsubstitution has to be performed every time the derivative of u is calculated. This takes approximately $2 * n * m$ operations, compared to $m*(m+1)$ operations of eq. 3.52, which is more than twice as much because $n > m$. In addition, the memory demands are higher as the storage of $\mathbf{Q}^T$ and $\mathbf{R}$ is required, whereas directly inverting matrix $\mathbf{A}$ requires the storage of $\frac{m*(m-1)}{2} + m$ which is of the order of $\frac{m*m}{2}$ since $\mathbf{A}$ is symmetric.

3.2.4 Weighted Least Squares Approach

A drawback of the interpolation procedure so far presented is that equal weight is given to all the points in Ω_i. This can rapidly cause a deterioration of the approximation [15]. A remedy can be the introduction of weighting functions, such as a Gauss function, which will be described next.

Following the least squares approach from above, we can directly include any weighting functions $w(x_j)$ in eq. 3.47:

$$J = \sum_{j=1}^{n} w(x_j)(u_j - \hat{u}(x_j))^2 = \sum_{j=1}^{n} w(x_j)(u_j - \mathbf{p}^T\mathbf{a})^2 \tag{3.59}$$

Again minimizing J with respect to $\mathbf{a}$, we obtain the same equation for the unknown polynomials

$$\mathbf{a} = \mathbf{A}^{-1}\mathbf{Bu} \tag{3.60}$$

but the matrices $\mathbf{A}$ and $\mathbf{B}$ are now composed of

$$\mathbf{A} = (\mathbf{C}^T\mathbf{WC}) \text{ and} \tag{3.61}$$

$$\mathbf{B} = \mathbf{C}^T\mathbf{W} \tag{3.62}$$

$\mathbf{W}$ is a diagonal matrix containing the weights $w(x_j)$ at each point in Ω_i.

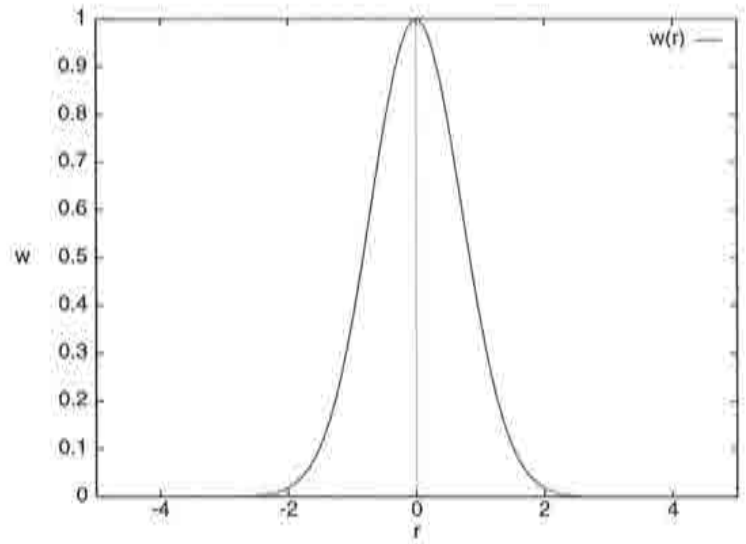


Figure 3.7 : Gauss weighting function; $w(r_j)$ quickly decreases as r_j increases (for $\lambda_i c = 1$).

Introduction of Weighting Functions

It is now of interest to see if the results can be improved by the use of weighting functions w_j within Ω_i. The idea is to give additional weight to points close to the central point and reduce it for points farther away. Within each Ω_i we define w_j as a function of the distance of each point j to the central point i:

$$w_j = w(r_j), \text{ and } r_j = |x_j - x_i| \tag{3.63}$$

A possible choice for weighting could be a Gauss distribution which was also used in all following calculations:

$$w(r_j) = e^{-\left(\frac{r_j}{\lambda_i c}\right)^p} \tag{3.64}$$

where λ_i is a characteristic length in each cloud Ω_i. c and p are user defined constants to adjust the sensitivity of the weighting function. Usually, $c = 1$ and $p = 2$ are chosen. Figure 3.7 displays $w(r_j)$ graphically for $\lambda_i c = 1$. Further information on the FPM can be found in [13, 15].

3.2.5 1D Convection-Diffusion Equation

Let us now apply the theoretical background to the previous test problem, the linear 1D convection-diffusion equation, and compare its results to known solutions from the FEM. Consider again the 1D convection-diffusion equation 3.4:

$$\frac{\partial u}{\partial t} + A\frac{\partial u}{\partial x} - \frac{\partial}{\partial x}\left(\kappa\frac{\partial u}{\partial x}\right) = 0 \quad \text{in } \Omega \tag{3.65}$$

with the same boundary and initial conditions as in eq. 3.4.

Again we are only interested in the steady state solution, which is given analytically by eq 3.7:

$$u(x) = 1 - \frac{1 - e^{\frac{A}{\kappa}x}}{1 - e^{\frac{A}{\kappa}}} \tag{3.66}$$

A discretization in space must be performed next. First, known and proven finite element methods will be presented, and then the finite point method proposed will be described.

Finite Element Solution

As shown earlier, the exact solution to this problem can be nodally reproduced by the finite element method using the Petrov-Galerkin method for all ranges of the Peclet number Pe using α_{opt}, see Figure 3.2. The so called Taylor-Galerkin approach can also be used to obtain exact nodal values for this problem as shown earlier in Figure 3.4, if

$$\Delta t = \Delta t_{opt} = \frac{h}{A}\alpha_{opt} \tag{3.67}$$

is chosen [6].

The Finite Point Method (FPM)

Let us now analyze the finite point method in the context of the 1D convection-diffusion equation. Integrating eq. 3.4 in time, performing a Taylor expansion of eq. 3.16 up to second order leads to:

$$u^{n+1} = u^n + \Delta t \frac{\partial u}{\partial t} + \frac{\Delta t^2}{2}\frac{\partial^2 u}{\partial t^2} + O(\Delta t^3) \tag{3.68}$$

inserting eq. 3.4 into eq. 3.68 and omitting third order derivatives, we obtain:

$$u^{n+1} = u^n - \Delta t \left[\left(A\frac{\partial u}{\partial x} - \frac{\partial}{\partial x}(\kappa \frac{\partial u}{\partial x}) \right) - \frac{A^2 \Delta t}{2}\frac{\partial^2 u}{\partial x^2} \right] \tag{3.69}$$

The discretization of the computational domain is performed locally using arbitrary points, without the need for fixed connectivities in a conventional mesh. Performing a least squares approximation in the vicinity of a point using using n points, we obtain an estimation of the necessary spatial derivatives $\frac{\partial u}{\partial x}$ and $\frac{\partial^2 u}{\partial x^2}$. Substituting these derivatives into eq. 3.69 leads to a system of equations from which the unknown point values u_j can be found for each time increment.

As explained earlier, the unknown functions $u(x)$ and its derivatives may be expanded as follows in a given cloud, for the linear case (in what follows we assume $u(x) = \hat{u}(x)$ for clarity):

$$u(x) = a_1 + a_2 x = \mathbf{p}^T \mathbf{a} \tag{3.70}$$

$$\frac{\partial u}{\partial x} = a_2 \tag{3.71}$$

and for the quadratic case:

$$u(x) = a_1 + a_2 x + a_3 x^2 = \mathbf{p}^T \mathbf{a} \tag{3.72}$$

$$\frac{\partial u}{\partial x} = a_2 + 2a_3 x \tag{3.73}$$

$$\frac{\partial^2 u}{\partial x^2} = 2a_3 \tag{3.74}$$

For the linear case, it is not possible to directly compute the necessary second order derivatives. This can be overcome by performing an accumulation of differences at the central point and the rest of the points j within the cloud. Hence,

$$\frac{\partial^2 u}{\partial x^2} = \sum_{j=2}^{n} \frac{(u_j - u_i)}{h^2} \tag{3.75}$$

with point i being the central point.

It can be shown that this scheme, for equally spaced points ($n = 3$ and $m = 2, 3$), is equivalent to a central-difference approximation which in turn is equivalent to FEM with Galerkin weighting functions [16].

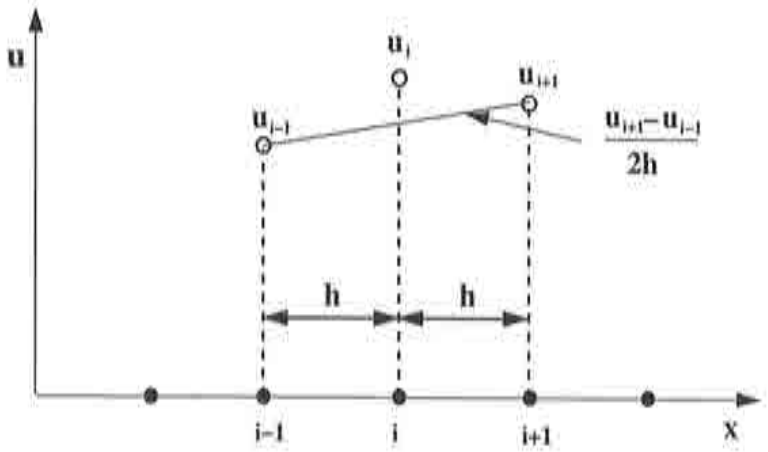


Figure 3.8 : Equally spaced points and their domain of influence for $n = 3$. Note the derivative is equivalent to a central difference approximation.

Linear case

The following shows this for the linear case (m=2). Three points ($i-1$,i,$i+1$) are considered with the coordinates (x_{i-1},x_i,x_{i+1}) and at equal distance h from each other as shown in Figure 3.8. Let their unknown values be u_{i-1},u_i,u_{i+1}, respectively. Then, $\mathbf{A}$ from equation 3.53 can be calculated as

$$\mathbf{A} = \begin{bmatrix} n & \sum_{j=1}^{n} x_j \\ \sum_{j=1}^{n} x_j & \sum_{j=1}^{n} x_j^2 \end{bmatrix} = \begin{bmatrix} 3 & 3x_i \\ 3x_i & 3x_i^2 + 2h^2 \end{bmatrix} \tag{3.76}$$

Inversion of $\mathbf{A}$ leads to

$$\mathbf{A}^{-1} = \begin{bmatrix} \frac{1}{3} + \frac{x_i^2}{2h^2} & -\frac{x_i}{2h^2} \\ -\frac{x_i}{2h^2} & \frac{1}{2h^2} \end{bmatrix} \tag{3.77}$$

Eq. 3.52 gives the polynomial coefficients a_1 and a_2 as:

$$\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \mathbf{A}^{-1}\mathbf{B}\mathbf{u} \tag{3.78}$$

The first derivatives in the linear case are constant. Using eq. 3.71, 3.77 and 3.78 the derivative of u at point i can be calculated from:

$$\frac{\partial u}{\partial x} = a_2 = (\frac{x_i}{2h^2} - \frac{x_i}{2h^2})u_i + (\frac{x_i}{2h^2} - \frac{x_i - h}{2h^2})u_{i-1} + (\frac{x_i}{2h^2} - \frac{x_i + h}{2h^2})u_{i+1} \tag{3.79}$$

Simplification gives

$$\frac{\partial u}{\partial x} = \frac{u_{i+1} - u_{i-1}}{2h} \tag{3.80}$$

which is exactly a central difference approximation. The second differences $\frac{\partial^2 u}{\partial x^2}$ are taken as accumulated differences at the central node i: $\sum_{j=2}^{n} \frac{u_j - u_i}{h^2}$ for equally spaced elements, leading to

$$\frac{\partial^2 u}{\partial x^2} \simeq \frac{u_{i-1} - u_i}{h^2} + \frac{u_{i+1} - u_i}{h^2} = \frac{u_{i-1} - 2u_y + u_{i+1}}{h^2} \tag{3.81}$$

which also is a central difference approximation.

Quadratic Case

By analogy, we can derive similar statements for the quadratic case ($m = 3$). Eq. 3.73 and eq. 3.74 are the definition using quadratic approximation for the first and second derivative within a cloud of points Ω_i, respectively. So, the first derivative of a function $u(x)$ at a point x_i is defined as

$$\frac{\partial u}{\partial x}(x_i) = a_2 + 2a_3 x_i \tag{3.82}$$

and the second derivative is constant within each interpolating domain Ω_i

$$\frac{\partial^2 u}{\partial x^2} = 2a_3 \tag{3.83}$$

After assembling and inverting $\mathbf{A}$ for $m = 3$, the unknown coefficients can be calculated from eq. 3.60

$$\begin{Bmatrix} a_1 \\ a_2 \\ a_3 \end{Bmatrix} = \mathbf{A}^{-1}\mathbf{B}\mathbf{u} \tag{3.84}$$

and we obtain for the desired derivatives of eq. 3.82 and eq. 3.83

$$\frac{\partial u}{\partial x}(x_i) = \frac{u_{i+1} - u_{i-1}}{2h} \tag{3.85}$$

and

$$\frac{\partial^2 u}{\partial x^2} = \frac{u_{i-1} - 2u_i + u_{i+1}}{h^2} \tag{3.86}$$

The detailed description of above equalities is presented in Appendix A.1.

Use of Weighting Functions

Consider again three points $(i-1, i, i+1)$ with the coordinates (x_{i-1}, x_i, x_{i+1}) at equal distance h from each other as shown in Figure 3.8 and linear weighting functions ($m = 2$) . Their associated weights are w,1,w and their unknown values are u_{i-1}, u_i, u_{i+1}, respectively. Then, $\mathbf{A}$ from equation 3.61 can be calculated as

$$\mathbf{A} = \begin{bmatrix} 1 + 2w & x_i + 2wx_i \\ x_i + 2wx_i & x_i^2 + 2w\left(x_i^2 + h^2\right) \end{bmatrix} \tag{3.87}$$

Inversion of $\mathbf{A}$ leads to

$$\mathbf{A}^{-1} = \frac{1}{2h^2 w\,(1+2w)} \begin{bmatrix} x_i^2 + 2w\left(x_i^2 + h^2\right) & -x_i - 2wx_i \\ -x_i - 2wx_i & 1 + 2w \end{bmatrix} \tag{3.88}$$

From eq. 3.62 we obtain the weighted matrix $\mathbf{B}$ and then the polynomial coefficients a_1 and a_2 as:

$$\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \mathbf{A}^{-1}\mathbf{B}\mathbf{u} \tag{3.89}$$

Eq. 3.79 can now be written as:

$$\begin{aligned} \frac{\partial u}{\partial x} = a_2 = \frac{1}{2h^2 w\,(1+2w)} &\left[\left(u_i + wu_{i-1} + wu_{i+1}\right)\left(x_i + 2wx_i\right) + \right. \\ &\left.\left(x_i u_i + wu_{i-1}\left(x_i - h\right) + wu_{i+1}\left(x_i + h\right)\right)\left(1 + 2w\right)\right] \end{aligned} \tag{3.90}$$

Simplification gives

$$\frac{\partial u}{\partial x} = \frac{u_{i+1} - u_{i-1}}{2h} \tag{3.91}$$

which is a central difference approximation. The second differences $\frac{\partial^2 u}{\partial x^2}$ are taken as weighted accumulated differences at the central node i, which leads to

$$\frac{\partial^2 u}{\partial x^2} \simeq \frac{w(u_{i-1} - u_i) + w(u_{i+1} - u_i)}{\hat{w}h^2} \tag{3.92}$$

where $\hat{w} = \frac{1}{n-1}\sum_{j=2}^{n} w_j$, which is an average weight across the cloud. For this problem ($n = 3$, $m = 2$ and equally spaced elements) $\hat{w} = w$ and further simplifying leads to

$$\frac{\partial^2 u}{\partial x^2} \simeq \frac{u_{i-1} - 2u_i + u_{i+1}}{h^2} \tag{3.93}$$

which again is the standard central difference approximation.

Order of Approximation

For any numerical scheme, the order of approximation must be estimated. The time integration of the proposed algorithm is second order accurate as seen from eq. 3.68. The estimation of the spatial accuracy is not trivial since different types of cloud constellations are possible.

3 Points per Cloud

For $n = 3$, equivalence to the central finite difference approximation was shown. So, the spatial accuracy is of second order for smooth meshes.

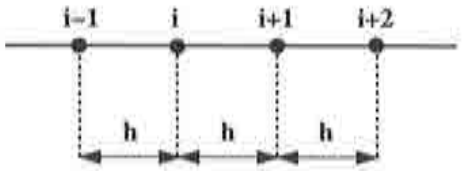


Figure 3.9 : 4 equally spaced points and their domain of influence for $n = 4$.

Clouds with more than 3 Points and without Weighting Functions

For $n > 3$, using no weighting functions, the accuracy deteriorates, because the discretization reduces to first order. Referring to Figure 3.9, eq. 3.79 now becomes for $n = 4$ and $m = 2$:

$$\frac{\partial u}{\partial x} = a_2 = \frac{1}{h}\left(-3u_{i-1} - u_i + u_{i+1} + 3u_{i+2}\right) \tag{3.94}$$

Second order accuracy would require

$$\frac{\partial u}{\partial x} = a_2 = \frac{1}{h}\left(au_{i-1} + bu_i + cu_{i+1} + du_{i+2}\right) + O(h^2) \tag{3.95}$$

together with the conditions [1]:

$$a + b + c + d = 0 \tag{3.96}$$

and

$$a + c + 4d = 0 \tag{3.97}$$

The second condition 3.97 is not satisfied as $a + c + 4d = -3 + 1 + 3 = 1$. Hence, the approximation is only first order accurate in space for the algorithm with no or constant weighting. The detailed description for obtaining eq. 3.94 can be found in Appendix A.2.1.

Clouds with more than 3 Points using Weighting Functions

The incorporation of Gauss weighting functions can retain second order accuracy as shown again for 4 noded clouds ($n = 4$) and $m = 2$. The first order derivative is defined as

$$\frac{\partial u}{\partial x} = a_2 \simeq \frac{1}{2h}\left(u_{i+1} - u_{i-1}\right) - \frac{w_2}{hw_1\left(1 + 2w_1\right)}\left(u_{i+2} - u_i\right) \tag{3.98}$$

where w_1 is the weight assigned to points $i-1$ and $i+1$, w_2 is the weight assigned to node $i+2$. Assuming that $w_2 \ll w_1 \leq 1$, then second order accuracy is obtained. The detailed proof is shown in Appendix A.2.2. For $m = 3$, similar conclusions are drawn, as the equalities are obtained in analogy to those of $m = 2$.

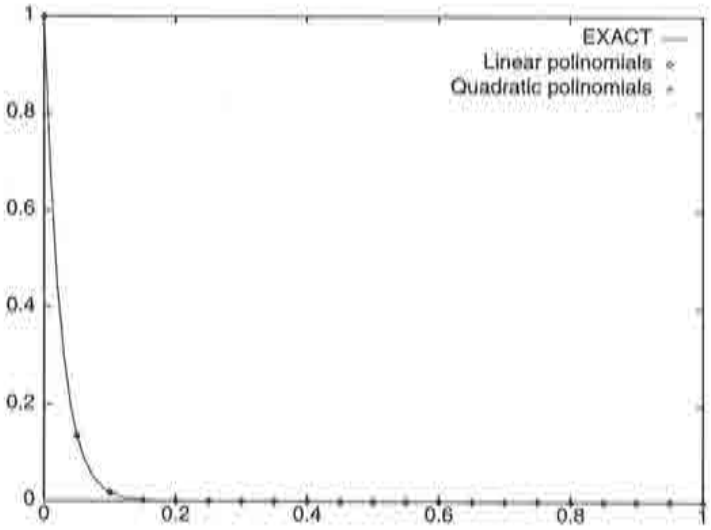
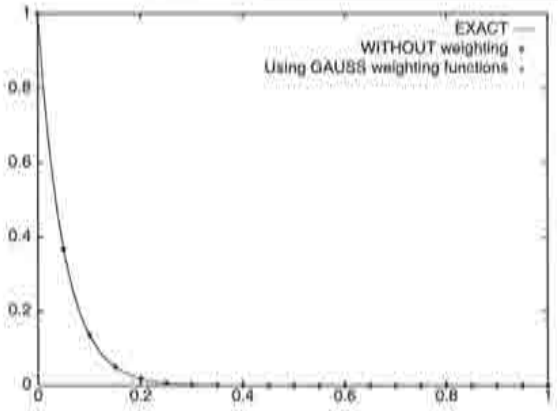
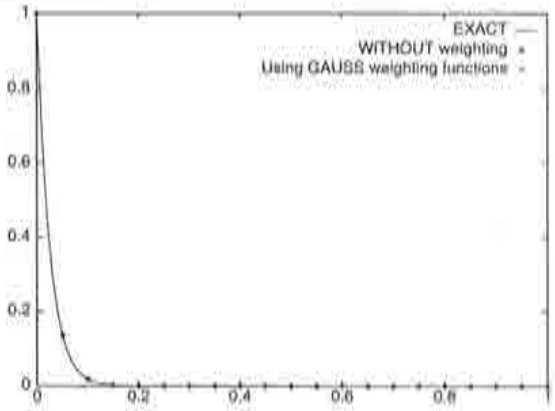


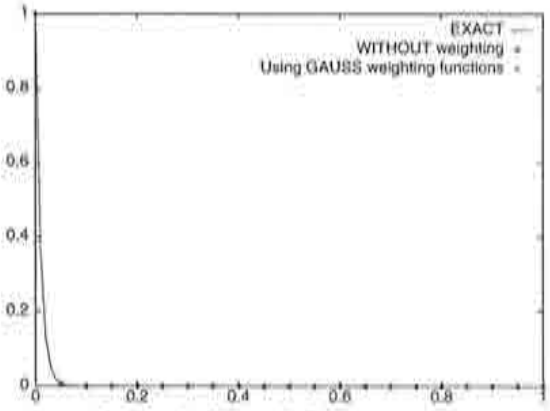
Figure 3.10 : Exact nodal values obtained by the finite point method for $Pe = 1$, $n = 3$. Compared are the use of linear and quadratic basis functions, $m = 2, 3$, with the analytical solution.



3.11 a) $Pe = 0.5$



3.11 b) $Pe = 1$



3.11 c) $Pe = 2,5$

Figure 3.11 : Convection-diffusion equation using linear base functions ($m = 2$) and 3 points per cloud ($n = 3$) for a) $Pe = 0.5$, b) $Pe = 1$ and c) $Pe = 2.5$.

3.2.6 Boundary Conditions and Stability

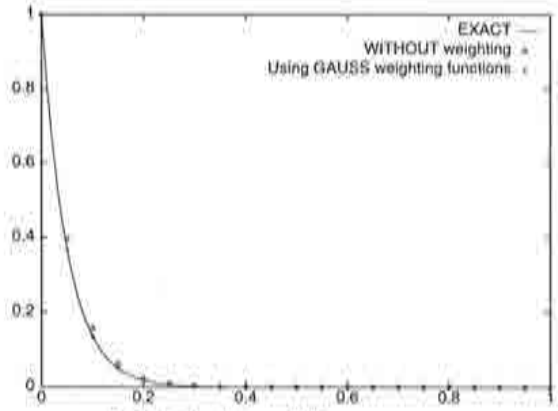
The implications on the formulation of the boundary conditions and stability are the same as described in section 2.7 and section 3.1.2, respectively. The boundary conditions for the numerical examples that follow are of Dirichlet Γ_D type. For the solution of the steady state convection diffusion equation using a time marching scheme, stability must be considered. The stability properties of the scheme 3.69 for 3 noded clouds are the same than the Taylor-Galerkin scheme presented in section 3.1.2 as it is equivalent to linear finite elements. For $n > 3$, this applies if $w_2 \ll w_1$, ie. Gauss weighting functions are used. However, a mathematical proof is not extended here.

3.2.7 Numerical Examples

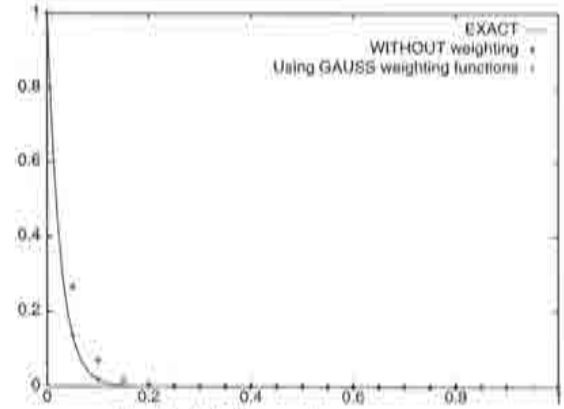
Let us test the solution of the steady convection-diffusion equation using the time marching scheme of equation 3.69 with $\Delta t = \Delta t_{opt}$ of eq. 3.22. Spatial discretization is done by means of clouds of points without weighting and with Gauss weighting functions using linear and quadratic basis functions. Having shown the equivalence of

FPM ($n = 3, m = 2, 3$) with FEM, it should be possible to recover nodally exact values for $n = 3$ and $m = 2, 3$ using the finite point method. Figure 3.10 demonstrates this for $Pe = 1$.

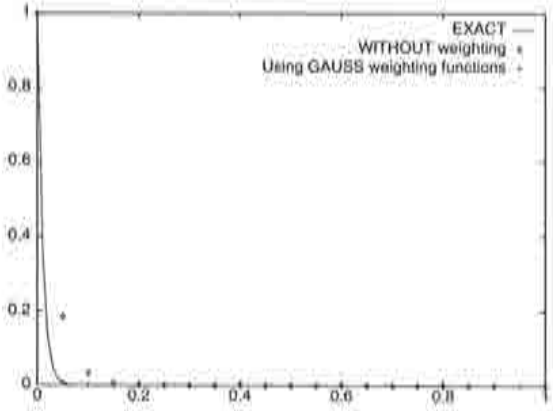
However, if the number of points in the local interpolating domain Ω_i increases ($n > 3$), the algorithm introduces excessive diffusion and the quality of the result deteriorates, especially near stronger gradients, which is the case if the Peclet number is increased.



3.12 a) $Pe = 0.5$



3.12 b) $Pe = 1$

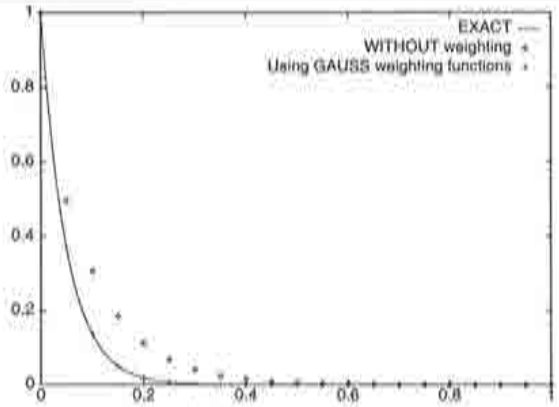


3.12 c) $Pe = 2,5$

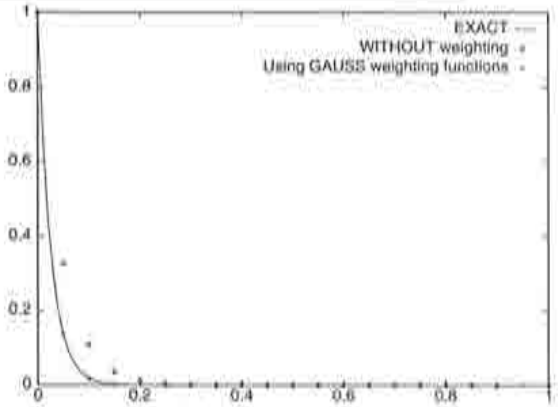
Figure 3.12 : Convection-diffusion equation using linear base functions ($m = 2$) and 4 points per cloud ($n = 4$) for a) $Pe = 0.5$, b) $Pe = 1$ and c) $Pe = 2.5$.

Figure 3.11 shows the solution of the finite point method using linear base functions ($m = 2$) and 3 points per cloud ($n = 3$) for three different Peclet numbers: a) $Pe = 0.5$ b) $Pe = 1.0$ and c) $Pe = 2.5$. Observe that exact nodal values are obtained in all cases.

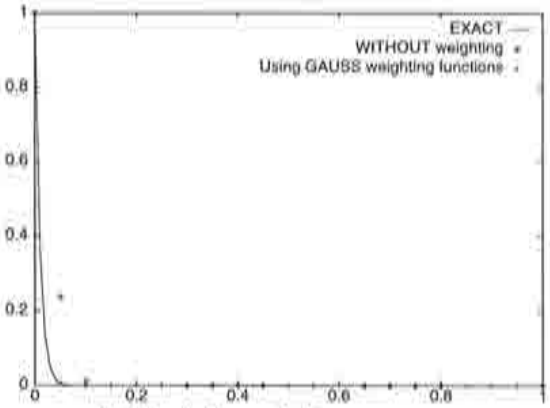
However, as the number of points in the cloud is increased, a deterioration of the solution is visible when no weighting functions are employed. Figures 3.12 and 3.13 demonstrate this behavior for $n = 4$ and $n = 5$. In the same figures, the use of Gaussian weighting functions is compared. The improvement of the solution is impressive. With $\lambda_i = r_{min}$, where r_{min} refers to the minimum distance of r in Ω_i, practically exact nodal values are recovered for this 1D test problem.



3.13 a) $Pe = 0.5$



3.13 b) $Pe = 1$

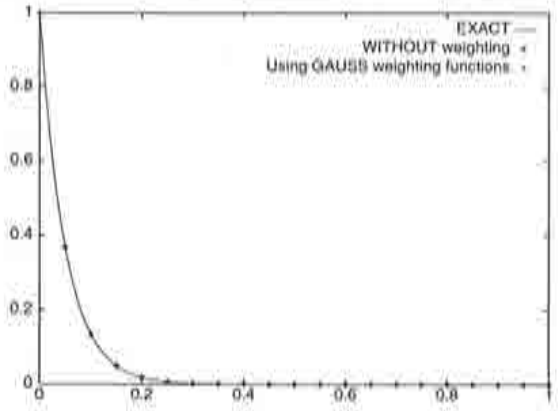


3.13 c) $Pe = 2,5$

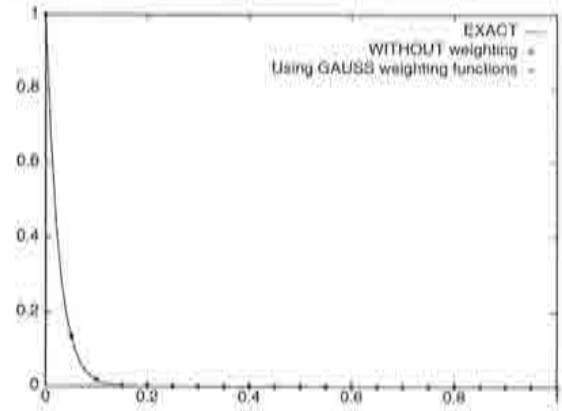
Figure 3.13 : Convection-diffusion equation using linear base functions and 5 points per cloud for a) $Pe = 0.5$, b) $Pe = 1$ and c) $Pe = 2.5$.

The extension to quadratic base functions ($m = 3$) exhibits a more drastic need for using weighting functions. Whereas with 3 noded clouds ($n = 3$) exact nodal values are computed (Figure 3.14), strong oscillations occur as n is increased (Figure 3.15). These oscillations disappear if a weighting interpolation is used.

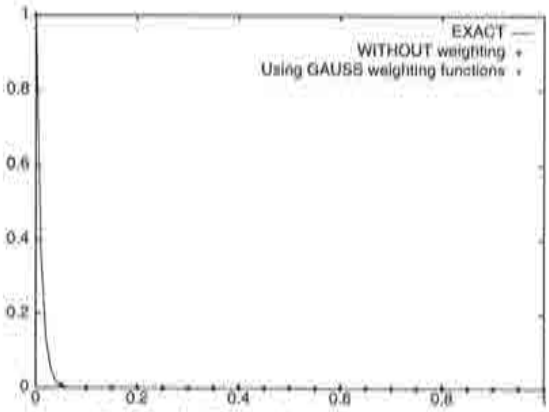
Additionally, it has also been found that the quality of the results worsens as the Peclet number increases if no weighting functions are employed. Note that again nearly exact nodal solutions are recovered by employing Gaussian weighted interpolation.



3.14 a) $Pe = 0.5$



3.14 b) $Pe = 1$



3.14 c) $Pe = 2,5$

Figure 3.14 : Convection-diffusion equation: quadratic base functions and 3 points per cloud for a) $Pe = 0.5$, b) $Pe = 1$ and c) $Pe = 2.5$.

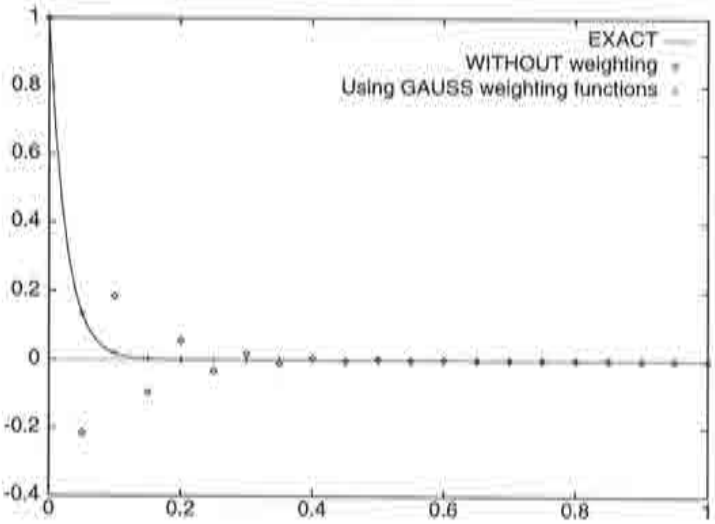


Figure 3.15 : Convection-diffusion equation using quadratic base functions and 5 points per cloud for $Pe = 1$.

3.3 Conclusion

A brief introduction to different discretization techniques has been given using the convection-diffusion equation as an example. Knowing that classical methods such as Petrov-Galerkin and Taylor-Galerkin methods are capable of obtaining exact nodal results of the steady convection-diffusion equation, a new type of discretization scheme using points was introduced and tested with the same examples. It was demonstrated that for particular cases (ie. $n = 3$), the proposed spatial discretization is equivalent to a central difference approximation.

An estimation of the order of accuracy and order of approximation was given. Shown was that the presented scheme is at least second order accurate for equally spaced points and $n = 2, 3$, but reduces to first order for $n > 3$. The addition of appropriately chosen weighting functions, such as a Gauss function, can restore the second order accuracy.

Results were presented for several number of points n per interpolation domain Ω_i. Exact nodal values were computed for $n = 2$ and $n = 3$, because it is equivalent to the Taylor-Galerkin scheme. If the number of points per cloud is increased, weighting functions are needed to be able to obtain sufficiently accurate results for equally spaced elements. Examples for both linear as well as quadratic basis functions were shown for which similar conclusions can be drawn.

References

[1] Hirsch, C. "Numerical Computation of Internal and External Flows", Volume 1, Wiley, Chichester, 1989

[2] Hughes T.J.R., Brooks A. "A Multi-dimensional Upwind Scheme with no Crosswind Diffusion", FEM for Convection Dominated Flows, T.J.R Hughes (ed.), ASME New York, 1979

[3] Heinrich J.C., Huyakorn P.S., Zienkiewicz O.C., Mitchell A.R., "An Upwind Finite Element Scheme for Two-Dimensional Convective Transport Equation", International Journal for Numerical Methods in Engineering, Vol 11, 131-143, 1977

[4] Codina R., "A Finite Element Model for Incompressible Flow Problems", PhD Thesis, Universidad Politècnica de Catalunya, Barcelona, 1992

[5] Kelly D.W., Nakazawa S., Zienkiewicz O.C., Heinrich J.C. "A Note on Upwinding and Anisotropic Balancing Dissipation in Finite Element Approximations to Convective Diffusion Problems", International Journal for Numerical Methods in Engineering, Vol 15, 1705-1711, 1980

[6] Zienkiewicz O.C. and Taylor R.L. "The Finite Element Method" 4th Edition, Volume 2,Mc Graw Hill, 1991

[7] Nayroles, B., Touzot, G. and Villon, P. "Generalizing the Finite Element Method: Diffuse Approximation and Diffuse Elements", Computational Mechanics, 10, 307-318, 1992

[8] Belytschko, T., Lu, Y. and Gu, L. "Element Free Galerkin Methods", Int. Journal for Numerical Methods in Engineering, 37,229-256, 1994

[9] Liu, W.K., Jan, S. and Belytschko "Reproducing Kernel Particle Methods", Int. Journal for Numerical Methods in Engineering (to be published)

[10] Batina J., "A Gridless Euler/Navier Stokes Solution Algorithm for Complex Aircraft Applications", AIAA paper, 93-0333, Reno NV, January 11-14, 1993

[11] Christie I., Griffiths D.F., Mitchell A.R., Zienkiewicz O.C. "Finite Element Methods for Second Order Differential Equations with Significant First Derivatives", International Journal for Numerical Methods in Engineering, Vol 10, 1389-1396, 1976

[12] Donea, J. "A Taylor-Galerkin Method for Convective Transport Problems", International Journal for Numerical Methods in Engineering, Vol 20, 101-119, 1984

[13] Oñate E., Idelsohn S. and Zienkiewicz O.C. "Finite Point Methods in Computational Mechanics", Publication CIMNE No. 67, July 1995

[14] Zienkiewicz O.C. and Taylor R.L. "The Finite Element Method" 4th Edition, Volume 1, Mc Graw Hill, 1989

[15] Oñate E., Idelsohn S., Fischer T., Zienkiewicz O.C. "A Finite Point Method for analysis of fluid flow problems", 9th Int. Conf. on Finite Elements in Fluids, Venezia, Italy, 15-21 October 1995

[16] Peraire, J. "A finite element method for convective dominated flows", PhD Thesis at University College of Swansea, 1986

[17] Jameson, A., Schmidt, W., and Turkel, E. "Numerical simulation of the Euler equations by finite volume methods using Runge-Kutta time stepping schemes" AIAA paper 81-1259. AIAA 5th Computational Fluid Dynamics Conf., 1981

[18] Zienkiewicz, O.C. and Wu, J. "A General Explicit or Semi-Explicit Algorithm for Compressible and Incompressible Flows", Inst. of Num. Meth. in Eng., University College of Swansea, CR/682/91, 1991

[19] Zienkiewicz, O.C., Codina, R., Morgan, K. and Sai, S. "A General Algorithm for Compressible and Incompressible Flow", Inst. of Num. Meth. in Eng., University College of Swansea, CR/842/94, June 1994

[20] Problem 6 of the Workshop on Hypersonic Flows for Reentry Problems, Antibes, France, January 22-25 1990

Chapter 4

Taylor-Galerkin Algorithm for Compressible Flow

4.1 Introduction

This chapter presents the Taylor-Galerkin algorithm and several extensions to solve the equations for compressible flow in more than one dimension. The scheme used is well known and is based on the concepts of the previous chapter. The extension is made to approximate non linear coupled equations in multidimensions capable of simulating different ranges of flows including the resolution of discontinuities.

In the first part, the known general scheme is described for linear triangles in two dimensions. Then, the addition of a Jameson-type artificial diffusion [1] is outlined which incorporates an algorithm originally developed for finite volume schemes. Also, the extension to bilinear quadrilateral elements is presented.

For three dimensions, the Taylor-Galerkin algorithm for unstructured tetrahedral elements is shown [2], and using cylindrical coordinates, the scheme has been extended to cover axisymmetric flows, thus reducing the problem to only two dimensions.

Finally, numerical examples are shown to validate the features and qualities of the basic algorithm and its various extensions. The analytical solution of more complex flows around complex geometries is generally not available, so that a validation must be performed with simpler models or with experiments. An important part is dedicated to prove the accuracy by performing a mesh convergence study in order to underline the order of approximation. Emphasis is also put on error estimation and adaptive unstructured mesh generation.

4.2 Taylor-Galerkin Algorithm

4.2.1 Navier-Stokes Equations

The laminar Navier-Stokes equations for a compressible fluid in conservative form for laminar flux within a two dimensional domain can be written, in the absence of source terms[3]:

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}_i}{\partial x_i} = \frac{\partial \mathbf{g}_i}{\partial x_i} \qquad i = 1, 2 \tag{4.1}$$

where the nodal unknowns $\mathbf{u}$, the advective fluxes $\mathbf{f}_i$ and the viscous fluxes $\mathbf{g}_i$ are:

$$\mathbf{u} = \begin{Bmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho E \end{Bmatrix} \quad \mathbf{f}_i = \begin{Bmatrix} \rho u_i \\ \rho u_i u_1 + p\delta_{1i} \\ \rho u_i u_2 + p\delta_{2i} \\ (\rho E + p) u_i \end{Bmatrix} \quad \mathbf{g}_i = \begin{Bmatrix} 0 \\ \sigma_{1i} \\ \sigma_{2i} \\ k \frac{\partial T}{\partial x_i} + u_j \sigma_{ji} \end{Bmatrix} \tag{4.2}$$

The first row of eqs. 4.1 and 4.2 represent the continuity equation, the second and third row contain the momentum equation and the last row is the energy equation. These equations state the conservation of mass, momentum and energy, respectively. The law of Navier-Poisson for a Newtonian fluid leads to the components of the viscous stress tensor for isotropic media [4]:

$$\sigma_{ij} = \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) + \lambda \frac{\partial u_k}{\partial x_k} \delta_{ij} \qquad k = 1, 2 \tag{4.3}$$

where a bulk viscosity of $(3\lambda + 2\mu) = 0$ is assumed, which is justified by the validity of the Stokes hypothesis for which reversible processes under compression, see [3, 4, 5].

The following list describes the different variables contained in the above equations:

- ρ is the density,
- p the pressure,
- T the temperature,
- u_i the Cartesian components of the velocity,
- E the total energy per unit mass,
- δ_{ij} the Kronecker delta,
- k the material heat conductivity, and
- $\mu = \mu(T)$ is the dynamic coefficient of viscosity which is calculated from Sutherland's equation.

The momentum equation and the energy equation are coupled by the pressure which is achieved by assuming that the fluid is a perfect gas. Thus, the pressure is obtained from the equation of state for a perfect gas:

$$p = (\gamma - 1)\rho(E - 0.5u_j u_j) \tag{4.4}$$

where the ratio of specific heats γ is defined as:

$$\gamma = C_p/C_v \tag{4.5}$$

In the present computations γ is taken to be equal to 1.4 for the simulation of air, which is an adequate choice for subsonic, transonic and supersonic flow where the Mach number is not excessively high and where chemical reactions can be neglected. The variables are transformed into non dimensional values by introducing Pr, the Prandtl number, and Re, the Reynolds number, with details in [3].

4.2.2 One Step Taylor-Galerkin Algorithm

The numerical algorithm for solving the partial differential equation 4.1 is the well known Taylor-Galerkin method, successfully developed and used by different authors [3, 6, 7, 8]. Non structured meshes of linear triangular finite elements are used.

The solution vector $\mathbf{u}$ is written in Taylor series up to second order as:

$$\Delta\mathbf{u}^n = \mathbf{u}^{n+1} - \mathbf{u}^n = \Delta t \frac{\partial \mathbf{u}^n}{\partial t} + \frac{\Delta t^2}{2}\frac{\partial^2 \mathbf{u}^n}{\partial t^2} + O(\Delta t^3) \tag{4.6}$$

The time derivative of eq. 4.6 can be replaced by eq. 4.1 in one step. Hence using equation 4.6 and neglecting the $O(\Delta t^3)$ term the expansion reads:

$$\Delta\mathbf{u}^n = \Delta t \left(\frac{\partial \mathbf{g}_i}{\partial x_i} - \frac{\partial \mathbf{f}_i}{\partial x_i}\right)^n - \frac{\Delta t^2}{2}\left[\frac{\partial}{\partial x_i}\left(\mathbf{G}_i \frac{\partial \mathbf{f}_k}{\partial x_k}\right) - \frac{\partial}{\partial x_i}\left(\mathbf{A}_i \frac{\partial \mathbf{f}_k}{\partial x_k}\right)\right]^n \tag{4.7}$$

where $i, k = 1, 2$, $\mathbf{A}_i = \partial \mathbf{f}_i/\partial \mathbf{u}$ and $\mathbf{G}_i = \partial \mathbf{g}_i/\partial \mathbf{u}$ are matrices of vector gradients, and where derivatives or products of order higher than two have been neglected. Superscript n means values at time instant n. Equation 4.7 can be now discretized in finite element manner using standard Galerkin weighting functions and the weak form becomes:

$$\begin{aligned}\int_\Omega \mathbf{N}^T \Delta\mathbf{u} d\Omega &= \Delta t \int_\Omega \mathbf{N}^T \left[\frac{\partial \mathbf{g}_i}{\partial x_i} - \frac{\partial \mathbf{f}_i}{\partial x_i}\right]^n d\Omega \\ &- \frac{\Delta t^2}{2}\int_\Omega \mathbf{N}^T \left[\frac{\partial}{\partial x_i}\left(\mathbf{G}_i \frac{\partial \mathbf{f}_k}{\partial x_k}\right) - \frac{\partial}{\partial x_i}\left(\mathbf{A}_i \frac{\partial \mathbf{f}_k}{\partial x_k}\right)\right]^n d\Omega\end{aligned} \tag{4.8}$$

Integration by parts using Green's theorem in equation 4.8 gives:

$$\int_\Omega \mathbf{N}^T \Delta \mathbf{u} d\Omega =$$

$$\Delta t \left\{ \int_\Omega \mathbf{N}^T \left(-\frac{\partial \mathbf{f}_i}{\partial x_i} \right) d\Omega - \int_\Omega \frac{\partial \mathbf{N}^T}{\partial x_i} \mathbf{g}_i d\Omega + \int_\Gamma \left(\mathbf{N}^T \mathbf{g} \right)_n d\Gamma \right\}^n - \tag{4.9}$$

$$\frac{\Delta t^2}{2} \left\{ \int_\Omega \frac{\partial \mathbf{N}^T}{\partial x_i} \left(\mathbf{A}_i \frac{\partial \mathbf{f}_k}{\partial x_k} - \mathbf{G}_i \frac{\partial \mathbf{f}_k}{\partial x_k} \right) d\Omega - \int_\Gamma \mathbf{N}^T \left(\mathbf{A}_i \frac{\partial \mathbf{f}_k}{\partial x_k} - \mathbf{G}_i \frac{\partial \mathbf{f}_k}{\partial x_k} \right)_n d\Gamma \right\}^n$$

Subscript n indicates normal projections. Eq. 4.9 can now be discretized using standard finite element shape functions (eq. 3.8) with subscript i denoting nodal values:

$$\mathbf{u}^n = \sum_i N_i \bar{u}_i^n = \mathbf{N}\bar{\mathbf{u}}^n \tag{4.10}$$

where the overbar denotes the nodal values of the unknowns, but is omitted in what follows. This leads to a system of algebraic equations which can be compactly written in the following form:

$$\mathbf{M}\, \Delta \mathbf{u}^n = \mathbf{RHS}^n \tag{4.11}$$

where **RHS** is the right hand side of eq. 4.9 after discretization but is not detailed here for clarity. The components of the mass matrix $\mathbf{M}$ are defined by

$$\mathbf{M} = \int_\Omega \mathbf{N}^T \mathbf{N} d\Omega \tag{4.12}$$

In order to reduce the computational effort, the explicit form of the algorithm can be preserved for steady state solutions by solving equation 4.11 using a lumped mass matrix. The diagonalization is performed using the row sum method. For time accurate calculations, a Jacobi-type iteration procedure may be performed to recover the properties of the full mass matrix [3, 6].

Eq. 4.11 is called the Taylor-Galerkin one-step scheme. However, it would be necessary to compute matrices of vector gradients, $\partial \mathbf{f}_i / \partial \mathbf{u}$ and $\partial \mathbf{g}_i / \partial \mathbf{u}$ which is time and memory consuming. Instead, a two-step scheme is implemented which is equivalent to the one-step scheme, but avoids the computation of these Jacobians.

4.2.3 Two-step Algorithm

In the first step, a Taylor expansion at time $n + 1/2$ is performed and using eq. 4.1 (neglecting the viscous terms):

$$\mathbf{u}^{n+1/2} \simeq \mathbf{u}^n + \frac{\Delta t}{2} \frac{\partial \mathbf{u}^n}{\partial t} = \mathbf{u}^n - \frac{\Delta t}{2} \frac{\partial \mathbf{f}_i^n}{\partial x_i} \tag{4.13}$$

From this step $\mathbf{u}^{n+1/2}$ is obtained. For the second step, again the use of Taylor expansion for $\mathbf{f}_i^{n+1/2}$ and equation 4.1 without third order derivatives, provides the tools to

replace the Jacobians by values of fluxes at time $n+1/2$ [3]. The complete system of equations now becomes:

$$\mathbf{M}\frac{\Delta \mathbf{u}^n}{\Delta t} = \int_\Omega \frac{\partial \mathbf{N}^T}{\partial x_i}\left(\mathbf{f}_i^{n+1/2} - \bar{\mathbf{g}}_i^n\right) d\Omega - \int_\Gamma \mathbf{N}^T \left(\mathbf{f}_i^{n+1/2} - \bar{\mathbf{g}}_i^n\right)_n d\Gamma \tag{4.14}$$

where $\mathbf{f}_i^{n+1/2}$ can be directly obtained from the $\mathbf{u}^{n+1/2}$ values computed in the first step. The intermediate steps for reaching eq. 4.14 and its practical equivalence to eq. 4.11 can be read in [3].

Subscript n refers to boundary normal values and $\mathbf{M}$ is again evaluated using a lumped form to preserve the explicitness of the scheme. The overbar refers to mean elemental values. The definition of the shape functions can be obtained from [9] and are detailed in section 4.7.

4.3 Artificial Dissipation

To account for discontinuities in the solution of fluid flow using a high order scheme, a certain kind of artificial viscosity or diffusion must be introduced to avoid unphysical oscillations especially near strong gradients. First an algorithm of the Lapidus kind [10] and then an approach similar to Jameson [11] are described.

4.3.1 Lapidus Algorithm

The form by which a Lapidus type diffusion may be added to the Euler and Navier-Stokes equation is described next. The original algorithm has been developed by Lapidus [10] and has been successfully extended to multidimensional compressible flow by Löhner [12].

The idea is based on the correction of the solution obtained at every time step by incorporating a smoothing diffusion of the form

$$\Delta \mathbf{u}_s = \mathbf{u}_s^{n+1} - \mathbf{u}^{n+1} = \Delta t \frac{\partial}{\partial \mathbf{l}}\left(k^\star \frac{\partial (\mathbf{v}^{n+1})}{\partial \mathbf{l}}\right) \tag{4.15}$$

$k^\star$ is variable to determine the amount of viscosity necessary, $\mathbf{l}$ is the normalized vector of velocity gradients, $\Delta \mathbf{u}_s$ are incremental smoothed values, $\mathbf{u}_s^{n+1}$ is the smoothed solution at time $n+1$, and $\mathbf{v}$ is the vector of velocity.

$k^\star$ is obtained as follows:

$$k^\star = C_k h^2 \left|\frac{\partial (\mathbf{v}\mathbf{l})}{\partial \mathbf{l}}\right| \tag{4.16}$$

where C_k is the so called Lapidus constant chosen by the user, h is the element length chosen as the minimum height of the triangle and the vector $\mathbf{l}$, indicating the direction of the strongest gradient of velocity, is

$$\mathbf{l} = \left\{ \begin{array}{c} l_1 \\ l_2 \end{array} \right\} = \frac{1}{\left| \nabla \, |\mathbf{v}|^2 \right|} \left\{ \begin{array}{c} \frac{\partial |\mathbf{v}|^2}{\partial x} \\ \frac{\partial |\mathbf{v}|^2}{\partial y} \end{array} \right\} \tag{4.17}$$

The result is a diffusion proportional to the velocity gradient in the direction of maximum change in velocity. The advantage of this type of scaling is that the diffusion is large near strong gradients such as shocks while reducing a smearing of the solution in the boundary layer by reducing the crosswind diffusion. In this work, the application of a Lapidus type diffusion is not shown explicitly by examples, as it has been tested by others [3, 5, 7, 12].

4.3.2 2nd and 4th Order Artificial Dissipation

Instead of the artificial viscosity of Lapidus, an artificial dissipation of a blend of second and fourth order can be chosen [1, 11]. However, the original formulation must be adequately modified for use with finite elements.

Consider the Navier-Stokes equations 4.1 written in compact form

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \mathbf{f}^{\star} = 0 \tag{4.18}$$

where $\nabla \cdot \mathbf{f}^{\star}$ is the divergence of $\mathbf{f}^{\star}$ which contains the advective fluxes $\mathbf{f}$, the viscous fluxes $\mathbf{g}$ and the added dissipative fluxes $\mathbf{f}_d$:

$$\mathbf{f}^{\star} = \left\{ \begin{array}{c} \mathbf{f}_x - \mathbf{g}_x - \mathbf{f}_d \\ \mathbf{f}_y - \mathbf{g}_y - \mathbf{f}_d \end{array} \right\} \tag{4.19}$$

where the subscripts x and y refer to the direction of the flux.

One Dimension

For simplicity, the way to calculate the diffusive fluxes $\mathbf{f}_d$ is shown in one dimension and extended to two dimensions in a heuristic way. The basic expression for the diffusive fluxes $\mathbf{f}_d$ is taken from [24] as a blend of first and third order derivatives in 1D:

$$\mathbf{f}_d = \alpha^{(2)} \Delta x \lambda \frac{\Delta x^2}{\hat{p}} \left| \frac{\partial^2 p}{\partial x^2} \right| \frac{\partial \mathbf{u}}{\partial x} - \alpha^{(4)} \Delta x^3 \lambda \frac{\partial^3 \mathbf{u}}{\partial x^3} \tag{4.20}$$

In eq. 4.20, $\hat{p}$ is an average pressure of the elements surrounding a node and the coefficients α are chosen by the user according to the problem and detailed later on. Δx is the element length and λ is defined by the maximum eigenvalue of the advective flux Jacobian $\frac{\partial \mathbf{f}}{\partial \mathbf{u}}$ which is [24]:

$$\lambda = \lambda_{max} = |\mathbf{v}| + c \tag{4.21}$$

where $|\mathbf{v}|$ is the modulus of the velocity and $c = \sqrt{\gamma \frac{p}{\rho}}$ is the speed of sound. Derivation of the fluxes in eq. 4.18 leads to second and fourth order dissipation terms after regrouping:

$$\frac{\partial \mathbf{f}_d}{\partial x} = \alpha^{(2)}\lambda\frac{\Delta x^2}{\hat{p}}\left|\frac{\partial^2 p}{\partial x^2}\right|\Delta x^2\frac{\partial^2 \mathbf{u}}{\partial x^2} - \alpha^{(4)}\Delta x^2\lambda\Delta x^2\frac{\partial^4 \mathbf{u}}{\partial x^4} \tag{4.22}$$

The evaluation of the diffusion in eq. 4.22 requires the calculation of second and fourth derivatives, which can be computationally expensive. However, through the use of a more compact stencil, second derivatives can accurately be approximated given a regular mesh. It can be shown that for the unidimensional solution of node i, a second derivative of the unknown can be obtained using mass matrices [7, 25] (the equality holds for a mesh of equal spacing):

$$C\left[\frac{\partial}{\partial x}(\Delta x^2\frac{\partial \phi}{\partial x})\right]_i \simeq \left[\mathbf{M}_l^{-1}(\mathbf{M}_c - \mathbf{M}_l)\bar{\phi}\right]_i \tag{4.23}$$

- $\mathbf{M}_l$ is the diagonalized or lumped mass matrix,
- $\mathbf{M}_c$ is the consistent mass matrix,
- C is a constant equal to $\frac{1}{6}$ for linear 1D elements, and
- ϕ is any unknowns variable and $\bar{\phi}$ is the vector containing the nodal values of the unknown function within the element.

Replacing the expression for the evaluation of second derivatives in eq. 4.22, one gets omitting the overbar

$$\frac{\partial \mathbf{f}_d}{\partial x} \simeq \alpha^{(2)}\lambda\frac{\Delta x^2}{\hat{p}}\left|\frac{\partial^2 p}{\partial x^2}\right|\left(\mathbf{M}_l^{-1}(\mathbf{M}_c - \mathbf{M}_l)\mathbf{u}\right) - \alpha^{(4)}\lambda\left(\mathbf{M}_l^{-1}(\mathbf{M}_c - \mathbf{M}_l)\mathbf{u}\right)\Delta x^2\frac{\partial^2 \mathbf{u}}{\partial x^2} \tag{4.24}$$

where the constant C is absorbed in the coefficients $\alpha^{(2)}$ and $\alpha^{(4)}$. Replacing again $\Delta x^2\frac{\partial^2 \mathbf{u}}{\partial x^2}$ by $\mathbf{M}_l^{-1}(\mathbf{M}_c - \mathbf{M}_l)\mathbf{u}$ and omitting the constant C, we obtain after simplification:

$$\frac{\partial \mathbf{f}_d}{\partial x} \simeq \left[\alpha^{(2)}\lambda S - \alpha^{(4)}\lambda\left(\mathbf{M}_l^{-1}(\mathbf{M}_c - \mathbf{M}_l)\mathbf{u}\right)\right]\left(\mathbf{M}_l^{-1}(\mathbf{M}_c - \mathbf{M}_l)\mathbf{u}\right) \tag{4.25}$$

where the pressure switched term $S = \frac{\Delta x^2}{\hat{p}}\left|\frac{\partial^2 p}{\partial x^2}\right|$ was introduced which is a sensor for discontinuities. The assembly of eq. 4.25 is performed element wise and the flux $\mathbf{f}_d$ is accumulated at each node of the mesh.

Extension to Multidimensions

In a heuristic approach this idea can be extended to two and three dimensions. Then, the general expression for the dissipation flux is as follows in finite element fashion:

$$\frac{\partial_i(\mathbf{f}_d)}{\partial x_i} \simeq \mathbf{M}_l^{-1}\left[\epsilon^{(2)} - \epsilon^{(4)}\left(\mathbf{M}_l^{-1}(\mathbf{M}_c - \mathbf{M}_l)\mathbf{u}\right)\right]((\mathbf{M}_c - \mathbf{M}_l)\mathbf{u}) \tag{4.26}$$

where the coefficients ϵ have been introduced which will be detailed below. The following mass matrices for linear triangles for a smooth mesh are used:

$$\mathbf{M}_c = \frac{A}{12}\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} \text{ and } \mathbf{M}_l = \frac{A}{3}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.27}$$

and for linear tetrahedral elements:

$$\mathbf{M}_c = \frac{V}{20}\begin{bmatrix} 2 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix} \text{ and } \mathbf{M}_l = \frac{V}{4}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.28}$$

where A and V are the element area and element volume, respectively.

Remarks

- The summation over the elements using elemental mass matrices and spreading the values to the nodes is exactly equivalent to performing a summation over the sides in the mesh, ie. [13].
- Eq. 4.26 is the additional term to be added to all the equations 4.1 with the modification of the energy equation where the enthalpy is constant at steady state. Hence, the last component of $\mathbf{u}$, ρE, is replaced by the expression ρH where $H = E + \frac{p}{\rho}$ is the enthalpy.

The two coefficients, $\epsilon^{(2)}$ and $\epsilon^{(4)}$, are obtained from the expressions

$$\epsilon^{(2)} = \alpha^{(2)} \lambda S \tag{4.29}$$

$$\epsilon^{(4)} = \max\left[0, \alpha^{(4)} - \alpha^{(2)} S\right] \tag{4.30}$$

The terms $\alpha^{(2)}$ and $\alpha^{(4)}$ are user defined constants that are adjusted according to the desired amount of diffusion to be added. $\alpha^{(2)}$ is of the order of unity, generally chosen between 0.2 and 1 for flows with shocks, whereas the coefficient $\alpha^{(4)}$ is chosen between 0.05 and 0.6 for inviscid flows and 0.005 and 0.1 for viscous flows.

A pressure switched diffusion coefficient S_i for each node is calculated according to the following term:

$$S_i = \left[\sum^{Nel} \frac{|(\mathbf{M}_c - \mathbf{M}_l)\ \mathbf{p}|}{\mathbf{M}_c\ \mathbf{p}}\right]_i \tag{4.31}$$

where $\mathbf{p}$ is the pressure of each node within an element and N_{el} is the number of elements in the domain Ω. This formulation is well suited for both supersonic and subsonic flow. The second order dissipation accounts for strong discontinuities such as

shocks. The fourth order terms introduce a background dissipation to avoid oscillations where $|\mathbf{v}| \to 0$, ie. stagnation zone. If no shocks occur, accurate results can be obtained using only the fourth order background dissipation only [13]. Note however, that now the formulation has changed slightly in comparison to the original formulation, as the pressure switch is taken as the maximum within each element instead of the side.

4.4 Boundary Conditions

4.4.1 Euler Equations

The discretized equation system, eq. 4.14, assumes a computational domain Ω surrounded by a boundary Γ. So far, the algorithm only describes the contributions of each element across the integral Ω but does not yet state how to incorporate the boundary conditions for integral Γ.

In this thesis, two types of boundaries exist: the wall boundary through which mass flux is not possible and the inflow/outflow boundary through which mass flux is possible. As mentioned in chapter 2, the boundary conditions must be applied in a compatible form with the equations to be solved. Performing a one-dimensional linearized characteristic analysis of the Euler equations normal to the boundary Γ, the conditions for inflow outflow and solid wall can be deduced.

A transformation into characteristic variables similar to section 2.5 for eq. 2.18 can be done by chosing $\Lambda = \mathbf{X}^{-1}\mathbf{A}\mathbf{X}$, where Λ contains the eigenvalues and $\mathbf{X}$ contains the eigenvectors of the matrix $\mathbf{A} = \frac{\partial \mathbf{f}}{\partial \mathbf{u}}$. This leads to

$$\frac{\partial \mathbf{V}}{\partial t} + \Lambda \frac{\partial \mathbf{V}}{\partial x} = 0 \tag{4.32}$$

where $\mathbf{V} = \mathbf{X}^{-1}\mathbf{u}$ are called the characteristic variables. The system is now decoupled, so that the boundary conditions can be applied according to the signal of the characteristics of each equation. Details can be found in [24]. For the two dimensional case the characteristic variables and its eigenvector become [25]:

$$\mathbf{V} = \begin{bmatrix} \rho - \frac{p}{c^{*2}} \\ u_t \\ u_n + \frac{p}{\rho^* c^*} \\ -u_n + \frac{p}{\rho^* c^*} \end{bmatrix} \quad \text{and} \quad \Lambda = \begin{bmatrix} u_n \\ u_n \\ u_n + c \\ u_n - c \end{bmatrix} \tag{4.33}$$

The asterisc denotes the linearized state at the time step t^n, subscript n and t denote inward normal and tangential direction, respectively. The sign of the components of Λ in eq. 4.33 determines whether variables are prescribed or not.

- Supersonic inflow

  All components of Λ are positive, hence all variables must be prescribed, ie. by infinity values $\mathbf{V}_C = \mathbf{V}_\infty$. (subscript C denotes the corrected state).

- Supersonic outflow

  All components of Λ are negative, hence all variables are left free. They are taken as predicted values at time step $t^{n+1/2}$: $\mathbf{V}_C = \mathbf{V}^{n+1/2}$ within the program.

- Subsonic inflow

  The last component of Λ is negative ($0 < u_n < c$) and, hence, the last component of $\mathbf{V}$ is left free. The other components of $\mathbf{V}$ are consequently prescribed by infinity values which are the velocity and the pressure.

- Subsonic outflow

  The third component of Λ is positive and has to be prescribed while the other components are left free. For external flows, the outer boundary is placed at a far distance, ie. $>$ 10 chord lengths for airfoils, and the free stream pressure is prescribed there.

- Wall boundary

  As the third component of Λ is positive, it must be set. The wall boundary condition requires that no mass flux crosses the wall or $u_n = 0$.

A detailed description and the resulting corrective states after mathematical expansions for the conservative variables $\mathbf{u}$ can be found in [15]. The application of these conditions is done in a weak form in eq. 4.14 by correcting the fluxes at time $n + 1/2$ in the integral Γ.

4.4.2 Navier-Stokes Equations

The determination of the boundary conditions of the Navier-Stokes equations is similar than for the Euler equations. However, the steady momentum and energy equations are elliptic and its modeling is more complex. Details are dealt with in [24] and different possibilities are exposed.

One possibility is a direct extrapolation from the interior for supersonic outflow boundaries, so that the corrected states can be found. The physical wall boundary conditions which are imposed which is the no slip condition $u_i = 0$. An additional condition is the type of wall which is simulated. If an adiabatic wall is modeled, then the heat flux across the wall is zero, whereas if an isothermal wall is given, the temperature is set. The other boundary conditions are the same as for the Euler equations for the type of external flows considered here.

4.5 Stability

The extension of the stability properties of the numerical scheme for non linear systems are extended from the findings in section 3.1.2. In the case of stability limits in the convective case the limit Courant number was defined by $C = |A|\frac{\Delta t}{h} \leq 1$. For non linear convective equation systems such as the Euler equations, a decoupling is performed like

eq. 4.32 using characteristic variables and each equation can be analyzed separately. So, $|A|$ of eq. 3.23 for the Courant number is replaced by the maximum eigenvalue of the Jacobian $\mathbf{A} = \frac{\partial \mathbf{f}}{\partial \mathbf{u}}$ which is for eq. 2.18 $\lambda = |\mathbf{v}| + c$ obtained from Λ of eq. 4.33. Hence, according to section 3.1.2 the stability of the algorithm is preserved if the time step meets the following criteria for the inviscid algorithm

$$\Delta t_I = \frac{c_s h}{(|\mathbf{v}| + c)} \quad \text{and } c_s \leq 1 \tag{4.34}$$

where the subscript I refers to the inviscid time increment, c_s is a safety factor, usually close to unity and h is the element length. Details can be found elsewhere [3, 7, 24] Including the viscous terms, the time step criteria is

$$\Delta t = \frac{c_s h}{(|\mathbf{v}| + c) + \frac{4\mu\gamma^{3/2} M_\infty}{\rho_{min} \mathrm{PrRe} h}} \quad \text{and } c_s \leq 1 \tag{4.35}$$

where ρ_{min} is the minimum density within the element surrounding node i, M_∞ is the free stream Mach number, Pr is the non dimensional Prandtl number and Re is the Reynolds number. The other variables have the same meaning as defined earlier. Details can be found in [3, 24, 13]:

Local Time Stepping

In order to enhance the performance of the program and to accelerate the solution towards steady state, local time stepping can be used. Then, time steps of different size are used within each element individually according to the local stability limit. As the mesh size increases, ie. with the distance from the body, the time increment will also increase. However, this option is only permissible if the transient solution is not of interest.

4.6 Adaptivity and Error Estimation

The solution program has the possibility to increase the efficiency of the unstructured solution by adapting the element sizes in the mesh. This reduces the amount of work without sacrificing accuracy. For instance, strong variations may appear in very localized regions of the domain, whereas the rest of the solution remains smooth. A way to perform remeshing is by estimating *a posteriori* the error [16]. Although this process has been developed for elliptic problems, practical applications have shown that it works well and efficient for many compressible flow applications as well.

The element mean quadratic error for unidimensional elliptic problems is:

$$E^{el} = Ch^2 \left| \frac{\partial^2 \phi}{\partial x^2} \right|_{el} \tag{4.36}$$

h is the 1D element length, ϕ is the variable chosen, and C is a constant. Eq. 4.36 is used as the basis for estimating the error in advective-diffusive and compressible flow

problems. Although there is not a sound mathematical proof of this extension, it has been verified in practice that it provides accurate estimates of the regions where the changes in the numerical solution are higher, ie. where mesh refinement is necessary [3, 7, 26, 27].

If the criterion of equidistribution of the error among the elements is adopted, then:

$$h^2 \left| \frac{\partial^2 \mathbf{v}}{\partial x^2} \right|_{el} = \text{constant} \tag{4.37}$$

For 2D problems eq. 4.37 may be extended as follows:

$$(\delta_i)^2 |\lambda_i| = \text{constant} \qquad i = 1, 2 \tag{4.38}$$

where λ_i are the eigenvalues of the Hessian matrix $C_{ij} = \frac{\partial^2 \phi}{\partial x_i \partial x_j}$ and δ_i is the corresponding new element size.

This has the advantage of providing directionality to the error estimator which can produce stretched elements (ie. $\delta_1 \neq \delta_2$) [3]. It is a desirable property, since some discontinuities present a strong variation in one direction and a smooth behavior in the direction normal to the first.

A scalar variable should be chosen for the error estimation like the Mach number, density, etc. The new element sizes δ_1 and δ_2 in the principal directions are obtained according to:

$$(\delta_1)^2 \, |\lambda_1| = (\delta_2)^2 \, |\lambda_2| = \delta_r \, \lambda \text{max} \tag{4.39}$$

λ_1 and λ_2 being the eigenvalues of the Hessian matrix $\mathbf{C}$, δ_r is a user specified constant (minimum element size) and λ_{max} is the maximum eigenvalue over the whole mesh. The same process is easily extended to 3D problems.

4.6.1 Mesh Generation

A finite element mesh of linear triangles of 3 nodes is generated using the advancing front technique [3, 7, 26]. The element sizes are decided in accordance with the sizes specified by a background grid. The initial background grid may be a very simple mesh, and the subsequent grids are those used for the latest or seemingly best computation, where the sizes are decided as specified in the previous section. For details of the preprocessor, the reader is referred to the above references.

4.7 Quadrilateral Elements

The finite element formulation of the compressible Navier-Stokes equations based on a two-step explicit Taylor-Galerkin scheme is being extended to treat unstructured quadrilateral bilinear elements. Adaptive mesh refinement as well as shock capturing techniques can used to enhance the numerical solution in the vicinity of steep gradients, such as shocks.

As shown in the previous section, the advantage of the use of a two-step algorithm with a lumped mass matrix formulation is a reduction of stored memory [3] resulting in a lean, explicit algorithm. The aim of this report is to extend these features mentioned above to the solution of quadrilateral elements. Three main advantages are apparent which will be discussed in the examples

- More flexibility using more types of elements
- Cost reduction by reducing the total number of elements
- Improved accuracy through more structure near boundaries

However, a disadvantage of unstructured quadrilaterals is the difficulty to control the quality of the elements, necessary for accurate results, and the higher generation cost.

4.7.1 Time Integration

Consider again the Taylor-Galerkin method [3] already described. Recall the time integration using a Taylor expansion up to second order just like eq. 4.6. The two step scheme divides the time integration into two parts. The resulting equation system is given by a similar one as obtained from eq. 4.13 and eq. 4.14. The first step of the computation gives an estimation of the unknown vector at time step $n + 1/2$ and the second part completes the time integration by using the evaluated values at time $n + 1/2$.

4.7.2 Space Discretization

Performing the spatial discretization using Galerkin shape functions, we rewrite eq. 4.14 and we get for the second step:

$$\mathbf{M}\frac{\Delta \mathbf{u}^n}{\Delta t} = \int_\Omega \frac{\partial \mathbf{N}^T}{\partial x_i}\left(\mathbf{f}_i^{n+1/2} - \bar{\mathbf{g}}_i^n\right) d\Omega - \int_\Gamma \mathbf{N}^T \left(\mathbf{f}_i^{n+1/2} - \bar{\mathbf{g}}_i^n\right)_n d\Gamma \tag{4.40}$$

The terms appearing in eq. 4.40 have the same meaning than in eq. 4.14, except that $\mathbf{M}$ is solved by the lumped mass matrix of quadrilaterals which will be detailed in section 4.7.3. The appearing derivatives of the shape functions are expressed for quadrilaterals [9]. By recalling the use of triangles, the differences in the calculation of the shape functions for quadrilateral elements become clear:

Shape Functions for Linear Triangles

For linear triangles with area A and nodes $i = 1, 2, 3$, the shape functions and its derivatives can be calculated in a straightforward manner:

$$N_i = \frac{1}{2A}(a_i + b_i x + c_i y) \tag{4.41}$$

Deriving the shape functions with respect to x and y we obtain the following constants for a linear triangle $1, 2, 3$:

$$\frac{\partial N_i}{\partial x} = \frac{b_i}{2A} = \frac{y_2 - y_3}{2A} \tag{4.42}$$

and

$$\frac{\partial N_i}{\partial y} = \frac{c_i}{2A} = \frac{x_3 - x_2}{2A} \tag{4.43}$$

Quadrilaterals

The evaluation of equation 4.13 and equation 4.14 may be done by using numerical integration at one Gauss point, which preserves the simplicity of the algorithm. The integral along Γ is evaluated similar to triangles but the first integral of equation 4.14 over the elements is rewritten using numerical integration as:

$$\int_\Omega \frac{\partial \mathbf{N}^T}{\partial x_j} \left(\mathbf{f}_j^{n+1/2} - \mathbf{g}_j^n \right) d\Omega \;=\; \sum^{Nel} \left[\sum_{i=1}^{4} \frac{\partial N_i}{\partial x_j} \left(f_j^{n+1/2} - g_j^n \right)_i \; w_i \right] |\mathbf{J}| \tag{4.44}$$

In this case the weight function w_i has been introduced for numerical integration and is equal to two at the center of the element for one Gauss point. $|\mathbf{J}|$ is the determinant of $\mathbf{J}$, defined further down.

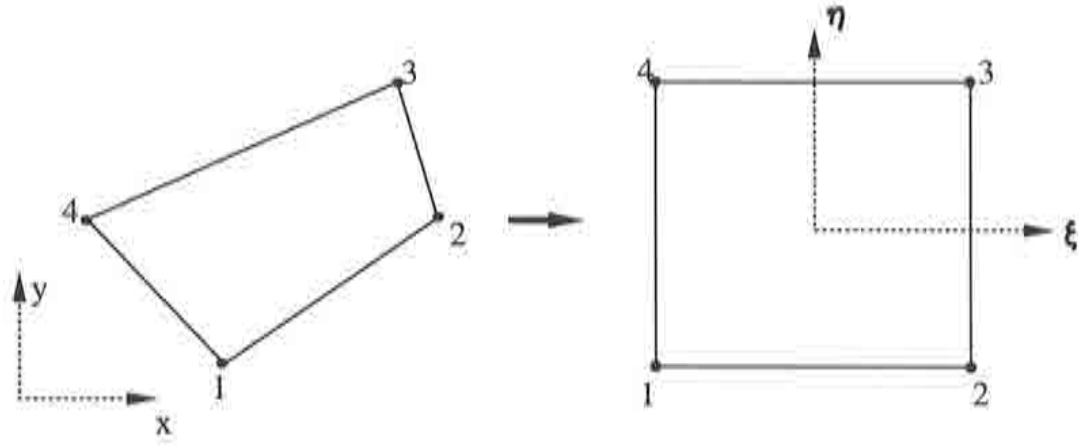


Figure 4.1 : Coordinate transformation of Cartesian coordinates to local coordinates

Shape Functions for Bilinear Quadrilaterals

Consider the transformation of Cartesian coordinates (x, y) into local coordinates (ξ, η) as shown in Figure 4.1. By means of the Jacobian transformation the following relation between $\boldsymbol{\xi}$ and $\mathbf{x}$ can be stated:

$$\mathbf{x} = \begin{Bmatrix} x \\ y \end{Bmatrix} = \mathbf{J}\boldsymbol{\xi} \tag{4.45}$$

or

$$\boldsymbol{\xi} = \begin{Bmatrix} \xi \\ \eta \end{Bmatrix} = \mathbf{J}^{-1} \begin{Bmatrix} x \\ y \end{Bmatrix} \tag{4.46}$$

where $\mathbf{J}$ is the Jacobian matrix, defined as

$$\mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial y}{\partial \eta} & \frac{\partial x}{\partial \eta} \end{bmatrix} \tag{4.47}$$

Hence, from the local coordinate system, the shape functions can be expressed as

$$N_i(\xi, \eta) = 0.25(1 + \xi\xi_i)(1 + \eta\eta_i) \ , \quad i = 1, 2, 3, 4 \tag{4.48}$$

For the calculation of derivatives in Cartesian coordinates, obviously the process becomes more complicated than for triangles:

$$\frac{\partial N_i}{\partial x} = \frac{1}{|\mathbf{J}|} \left(\frac{\partial N_i}{\partial \xi} \frac{\partial y}{\partial \eta} - \frac{\partial N_i}{\partial \eta} \frac{\partial y}{\partial \xi} \right) \tag{4.49}$$

and

$$\frac{\partial N_i}{\partial y} = \frac{1}{|\mathbf{J}|} \left(-\frac{\partial N_i}{\partial \xi} \frac{\partial x}{\partial \eta} + \frac{\partial N_i}{\partial \eta} \frac{\partial x}{\partial \eta} \right) \tag{4.50}$$

where

$$\frac{\partial N_i}{\partial \xi} = 0.25(1 + \eta\eta_i)\xi_i \ , \quad i = 1, 2, 3, 4 \tag{4.51}$$

and

$$\frac{\partial N_i}{\partial \eta} = 0.25(1 + \xi\xi_i)\eta_i \ , \quad i = 1, 2, 3, 4 \tag{4.52}$$

and the determinant of the Jacobian matrix is

$$|\mathbf{J}| = \frac{\partial x}{\partial \xi} \frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta} \frac{\partial y}{\partial \xi} \tag{4.53}$$

Using 1 Gauss point the evaluation of the determinant of the Jacobian matrix is:

$$|\mathbf{J}| = \frac{A}{4} \tag{4.54}$$

where A is the area of the quadrilateral element. The shape function derivatives reduce for node 1 as (with similar expressions for the other nodes):

$$\frac{\partial N_1}{\partial x} = \frac{1}{8|\mathbf{J}|}(y2 - y4) \ , \quad \frac{\partial N_1}{\partial y} = \frac{1}{8|\mathbf{J}|}(x4 - x2) \tag{4.55}$$

Incorporating $|\mathbf{J}| = \frac{A}{4}$ the following shape function derivatives for a quadrilateral element with the nodes i,j,k and l is obtained for one Gauss point:

$$\frac{\partial N_i}{\partial x} = \frac{1}{2A}(y_j - y_l) \ , \quad \frac{\partial N_i}{\partial y} = \frac{1}{2A}(x_l - x_j) \tag{4.56}$$

Equivalence to Finite Volume Scheme

The finite volume method for the Euler equations 2.18 can be written as [17]

$$\frac{\partial}{\partial t}\int_{\Omega_V} \mathbf{u} d\Omega_V \ + \ \int_\Gamma \sum_{i=1}^{N_D} \mathbf{f}_i n_i d\Gamma \ = \ 0 \tag{4.57}$$

Different forms for calculating and storing the fluxes of this scheme exist. For instance, if a central scheme is used, the cell vertex finite volume method is equivalent to the Galerkin finite element method using a lumped mass matrix for both triangles and quadrilaterals [17]. This is important to know, because the cell vertex finite volume method is very popular among other authors to date. Hence, extensive developments and testing exist which can be used or compared to the finite element scheme used in this thesis. In chapter 6 more detailed comparisons are made.

4.7.3 Artificial Dissipation

Again, the addition of artificial viscosity is needed to stabilize the solution in regions where the flow field is not smooth. The two previous possibilities exist for adding the necessary dissipation: Lapidus [10] and Jameson [11]. The routine for the calculation of the Lapidus diffusion is straightforward and does not need further explanation. However, in case of the Jameson diffusion there appears the difference of the consistent and lumped mass matrix, which must not be identical to the mass matrix of bilinear quadrilaterals:

$$(\mathbf{M}_c - \mathbf{M}_l) = \frac{A}{32}\begin{bmatrix} 3 & 2 & 1 & 2 \\ 2 & 3 & 2 & 1 \\ 1 & 2 & 3 & 2 \\ 2 & 1 & 2 & 3 \end{bmatrix} - \frac{A}{32}\begin{bmatrix} 8 & 0 & 0 & 0 \\ 0 & 8 & 0 & 0 \\ 0 & 0 & 8 & 0 \\ 0 & 0 & 0 & 8 \end{bmatrix} = \frac{A}{32}\begin{bmatrix} -5 & 2 & 1 & 2 \\ 2 & -5 & 2 & 1 \\ 1 & 2 & -5 & 2 \\ 2 & 1 & 2 & -5 \end{bmatrix} \tag{4.58}$$

It can also be approximated by

$$(\mathbf{M}_c - \mathbf{M}_l) \simeq \frac{A}{24}\begin{bmatrix} -3 & 1 & 1 & 1 \\ 1 & -3 & 1 & 1 \\ 1 & 1 & -3 & 1 \\ 1 & 1 & 1 & -3 \end{bmatrix} \tag{4.59}$$

4.7.4 Adaptive Solution and Mesh Generation

The adaptive remeshing procedure is similar to triangles described in section 4.6 and does not need further description.

A finite element mesh of linear unstructured quadrilaterals of 4 nodes is generated using the advancing front technique [3, 7] or a method proposed by Rank's conversion technique [30, 31]. The element sizes are decided by the sizes specified in a background grid, which may come from the error indicator.

4.8 Three Dimensional Fluid Flow

The Euler/Navier-Stokes equations in three dimensions can be written in conservative form (in case of the Euler equations $\mathbf{g} = 0$):

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}_i}{\partial \mathbf{x}_i} = \frac{\partial \mathbf{g}_i}{\partial \mathbf{x}_i} \qquad i = 1,3 \tag{4.60}$$

where the $\mathbf{u}$, $\mathbf{f}_i$ and $\mathbf{g}_i$ are the nodal unknowns, advective fluxes and viscous fluxes, respectively.

$$\mathbf{u} = \begin{Bmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho u_3 \\ \rho E \end{Bmatrix} \quad \mathbf{f}_i = \begin{Bmatrix} \rho u_i \\ \rho u_i u_1 + p\delta_{1_i} \\ \rho u_i u_2 + p\delta_{2_i} \\ \rho u_i u_3 + p\delta_{3_i} \\ (\rho E + p)u_i \end{Bmatrix} \quad \mathbf{g}_i = \begin{Bmatrix} 0 \\ \sigma_{1i} \\ \sigma_{2i} \\ \sigma_{3i} \\ k\,\frac{\partial T}{\partial x_i} + u_j\,\sigma_{ji} \end{Bmatrix} \tag{4.61}$$

The variables in the above above equations have the same meaning as in eq. 4.2. The numerical scheme to resolve the three dimensional Euler/Navier-Stokes equations is similar to that presented in section 4.2 and [18]. The spatial discretization is performed by means of linear tetrahedral elements.

The mass matrices appearing in the several equation of the procedure are now defined by:

$$\mathbf{M}_c = \frac{V}{20}\begin{bmatrix} 2 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix} \quad \text{and} \quad \mathbf{M}_l = \frac{V}{4}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.62}$$

where V is the volume of the tetrahedral element. The error estimation is performed in analogy to the scheme of section 4.2 using three dimensions ($i = 1,3$).

4.9 Axisymmetric Fluid Flow

Some axisymmetric flow configurations can be simplified by reducing the order of the dimension if both the geometry and the flow are axisymmetric. This can simplify greatly the cost and increase the accuracy for a test problem.

The three dimensional Navier-Stokes equations from eq. 4.60 are modified by the addition of the axisymmetric approximations similar to Brückner [28]. In order to be able to simplify the three dimensions (x, y, z) of eq. 4.60 into two axisymmetric dimensions (x, r), the equation system is rewritten by introducing in the cylindrical coordinate system (x, r, Θ), using polar coordinates:

$$r = \sqrt{(y^2 + z^2)} \quad \text{and} \quad \Theta = arctan(y/z). \tag{4.63}$$

The terms in the circumferential direction are eliminated according to the symmetry condition, ie. $\partial/\partial\Theta$=0 as well as the velocity component in Θ-direction, which is zero. Hence, the Navier-Stokes eqs. 4.60 reduce to:

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}_x}{\partial x} + \frac{\partial \mathbf{f}_r}{\partial r} + \frac{\mathbf{f}_r}{r} = \frac{\partial \mathbf{g}_x}{\partial x} + \frac{\partial \mathbf{g}_r}{\partial r} + \frac{\mathbf{g}_r}{r} \tag{4.64}$$

The nodal unknowns $\mathbf{u}$ and the fluxes $\mathbf{f}$, $\mathbf{g}$ are again given by eq. 4.2, with the difference that the axisymmetric stress tensor changes:

$$\sigma_{xx} = \frac{4}{3}\mu\frac{\partial u_x}{\partial x} - \frac{2}{3}\mu\left(\frac{\partial u_r}{\partial r} + \frac{u_r}{r}\right) \tag{4.65}$$

$$\sigma_{rr} = \frac{4}{3}\mu\frac{\partial u_r}{\partial r} - \frac{2}{3}\mu\left(\frac{\partial u_x}{\partial x} + \frac{u_r}{r}\right) \tag{4.66}$$

$$\sigma_{xr} = \sigma_{rx} = \mu\left(\frac{\partial u_x}{\partial r} + \frac{\partial u_r}{\partial x}\right) \tag{4.67}$$

where μ is defined earlier.

Description of the numerical algorithm

The technique used to discretize eq. (4.64) is now similar to the two dimensional Taylor-Galerkin method, described in section 4.2.

In the first step, a Taylor expansion at time $n + 1/2$ is performed (neglecting the viscous terms), including the axisymmetric contribution:

$$\mathbf{u}^{n+\frac{1}{2}} = \mathbf{u}^n + \frac{\Delta t}{2}\frac{\partial \mathbf{u}^n}{\partial t} = \mathbf{u}^n - \frac{\Delta t}{2}\left(\frac{\partial \mathbf{f}_i^n}{\partial x_i} + \frac{\mathbf{f}_r^n}{r}\right) \tag{4.68}$$

For the second step, the equation system becomes

$$\mathbf{M}\frac{\Delta \mathbf{u}^n}{\Delta t} = \int_\Omega \frac{\partial \mathbf{N}^T}{\partial \mathbf{x}_i}\left(\mathbf{f}_i^{n+\frac{1}{2}} - \mathbf{g}_i^n\right) d\Omega - \int_\Gamma \mathbf{N}^T\left(\mathbf{f}_i^{n+\frac{1}{2}} - \mathbf{g}_i^n\right)_n d\Gamma - \int_\Omega \mathbf{N}^T\left(\frac{\mathbf{f}_r^n}{r} - \frac{\mathbf{g}_r^n}{r}\right) d\Omega \tag{4.69}$$

Note the axisymmetric contributions as the added term on the right. The coordinates (x_1, x_2) refer now to (x, r). The rest of the notation is the same as in section 4.2. In

the limit $r \to 0$, the use of l'Hôpital replaces the right term by the derivative and it becomes $\int_\Omega \mathbf{N}^T \left(\frac{\partial \mathbf{f}_r^n}{\partial r} - \frac{\partial \mathbf{g}_r^n}{\partial r} \right) d\Omega$.

The use of artificial dissipation, identical to section 4.3.2, stabilizes the solutions near discontinuities and avoids unphysical oscillations.

4.10 Numerical Examples

4.10.1 Two Dimensional Validation

The Taylor-Galerkin scheme has had difficulties with the resolution in the stagnation area for which $|\mathbf{v}| \to 0$ [19]. Even the addition of an artificial viscosity has not been able to resolve these problems. So, the following test cases have been performed to validate the incorporation of the Jameson type artificial diffusion model described in previous sections and to demonstrate the excellent behavior when used in conjunction with the Taylor-Galerkin scheme.

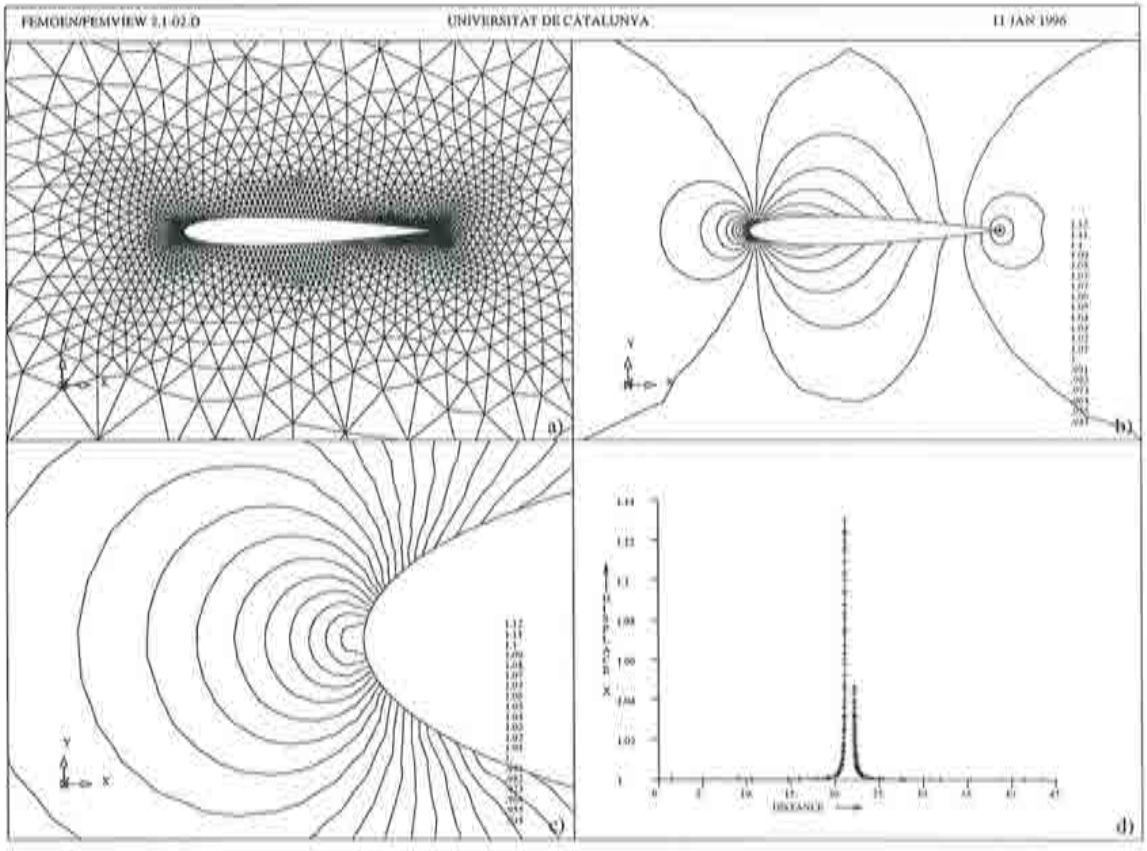


Figure 4.2 : NACA0012 profile for inviscid test case $M_\infty = 0.5$, $\alpha = 0°$: a) Finite element mesh of 2556 nodes and 4902 elements, b) density contours, c) close up density lines in the stagnation area and d) density values along the stagnation stream line. Note that no oscillations are present in the solution.

Subsonic Compressible Flow

- NACA0012: $M_\infty = 0.5$, inviscid flow at an angle of attack of $\alpha = 0°$.

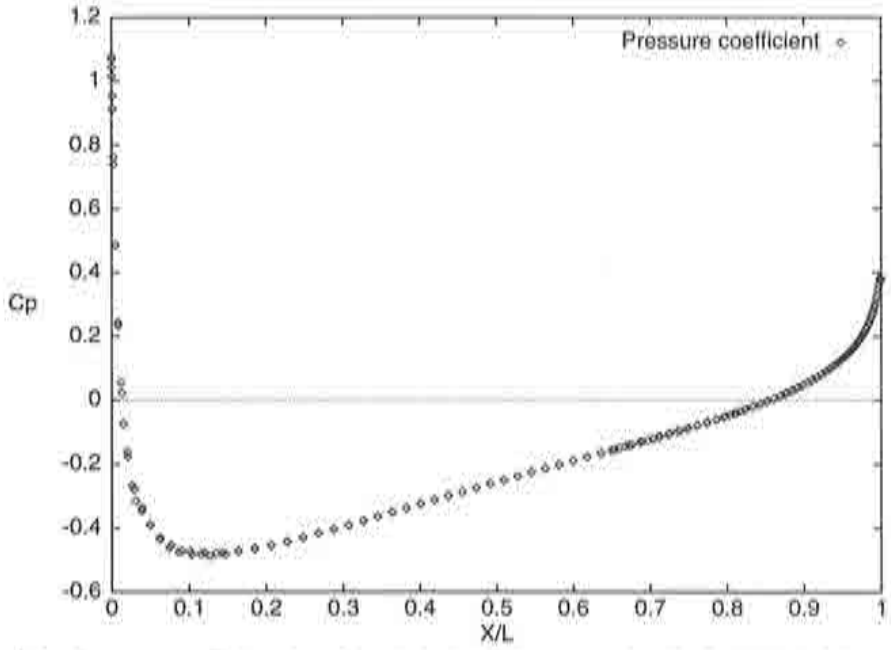


Figure 4.3 : Pressure coefficient for subsonic inviscid flow around a NACA0012 airfoil at $0°$ angle of attack and $M_\infty = 0.5$.

The first test case illustrates the quality of the flow solution for two dimensional inviscid subsonic compressible flow. Using a rather coarse unstructured mesh of 2556 nodes and 4902 elements shown in Figure 4.2 a), already reasonable results are obtained. In Figure 4.2 b) and c) density contours and a zoom in the stagnation area are shown, respectively. No oscillations are observed, neither in the global solution neither along the stagnation stream line of Figure 4.2 d).

A test for accuracy is the value of the density at the stagnation point for which the analytical result is known:

$$\rho_0 = \rho_\infty \left(1 + \frac{\gamma - 1}{2} M_\infty^2\right)^{\frac{1}{\gamma - 1}} \tag{4.70}$$

where $\rho_\infty = 1$ and $\gamma = 1.4$ for these calculations. Inserting $M_\infty = 0.5$, we obtain 1.129726 as the analytical value. The numerical result for this test case was obtained as 1.1302 which differs about 0.04% from the analytical value. The fourth order diffusion constant was chosen as $\alpha^{(4)} = 0.01$, whereas the second order diffusion was switched off ($\alpha^{(2)} = 0$). The pressure coefficient can be observed in Figure 4.3

Another test for accuracy are the drag and lift coefficients which must be zero for subsonic shock free symmetrical flow around a symmetrical airfoil. The values for c_D and c_L are 0.0015 and 0.0011 yield a very precise result for this test case

using a non symmetrical mesh without adaptivity. The reduction of the residuals is shown in Figure 4.29 and compared to a similar solution for a quadrilateral mesh.

A more detailed analysis of this test case, also performing a mesh convergence study, using different meshes and different values $\alpha^{(4)}$ is treated in section 6.4.2.

- NACA0012: $M_\infty = 0.5$, laminar viscous flow with a Reynolds number of $Re = 5000$ at an angle of attack of $\alpha = 0°$.

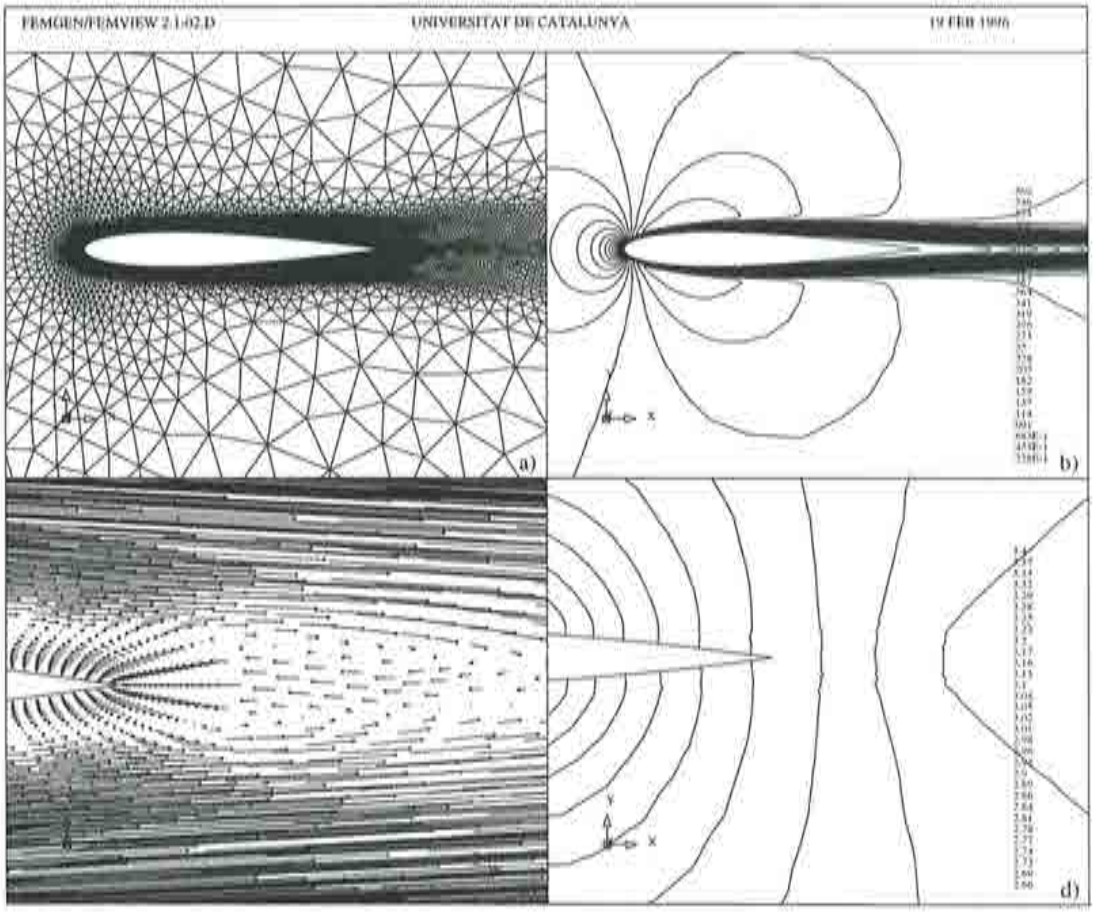


Figure 4.4 : Viscous flow solution around a NACA0012 airfoil (Re=5000, M_∞=0.5, α=0°) on an unstructured mesh of 15092 nodes.

In order to resolve this test case two meshes were generated. The first mesh (which is not shown) contains 7970 nodes and 15666 elements including 20 O-type structured layers near the boundary from which an initial result has been obtained. Performing adaptive refinement using the Mach number as the error variable, a new and finer mesh has been obtained with 15092 elements and 29701 elements that are mainly clustered in the wake region, see Figure 4.4 a). Note the 20 structured boundary layer elements along the surface of the airfoil.

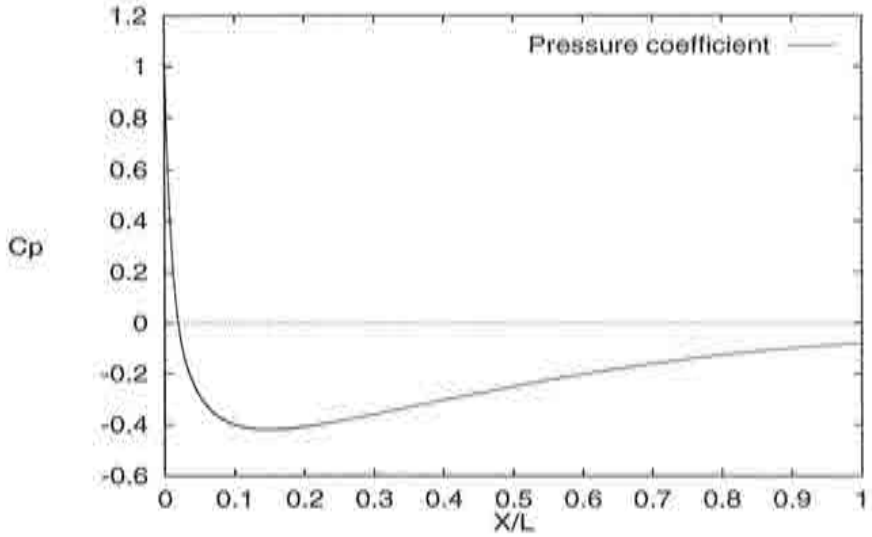


Figure 4.5 : Pressure coefficient along the airfoil for the solution of viscous flow around a NACA0012 airfoil (Re=5000, M_∞=0.5, α=0°).

The solution obtained for the adapted mesh using $\kappa^{(4)} = 0.02$ is presented in the following displays. Mach number contours of Figure 4.4 b) shows overall excellent agreement with the reference result [13]. Figure 4.4 c) illustrates the recirculation bubble at the trailing edge by velocity vectors. Each vector represents the modulus and direction of velocity at each node of the unstructured mesh. Pressure contours are shown in Figure 4.4 d). The fact that the lines are parallel to each other with almost no oscillations indicates good quality of the results.

A more severe test of accuracy is the plot of the pressure coefficient c_p and the skin friction coefficient c_f along the airfoil, shown in Figure 4.5 and Figure 4.6, respectively. The c_p agrees with the reference result whereas the c_f slightly underpredicts the peak value of the reference result. The pressure drag coefficient is $c_D = 0.0235$, which is slightly above the reference results lying between $c_D = 0.022$ and $c_D = 0.023$. A close-up view (Figure 4.7) of c_f indicates the location of separation which is also used to test for accuracy. The value of 81.8% chord is in excellent agreement with the comparisons made in the reference paper, ranging from 81%-83% chord. However, the reference meshes were much finer (> 20000 nodes), so that a similar order of accuracy is achieved using unstructured stretched meshes with less nodes.

The convergence history for this test case is plotted in Figure 4.8. After 8000 iterations a reduction of 7 order in magnitude of the L2-norm of the density residuals has been reached.

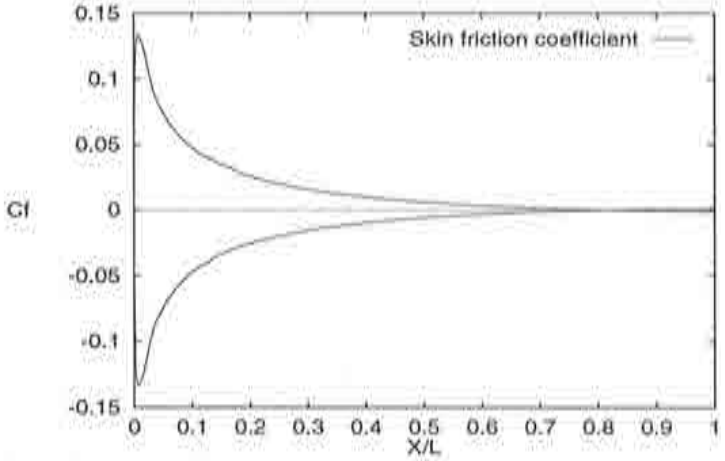


Figure 4.6 : Skin friction coefficient along the airfoil for the solution of viscous flow around a NACA0012 airfoil (Re=5000, M_∞=0.5, α=0°).

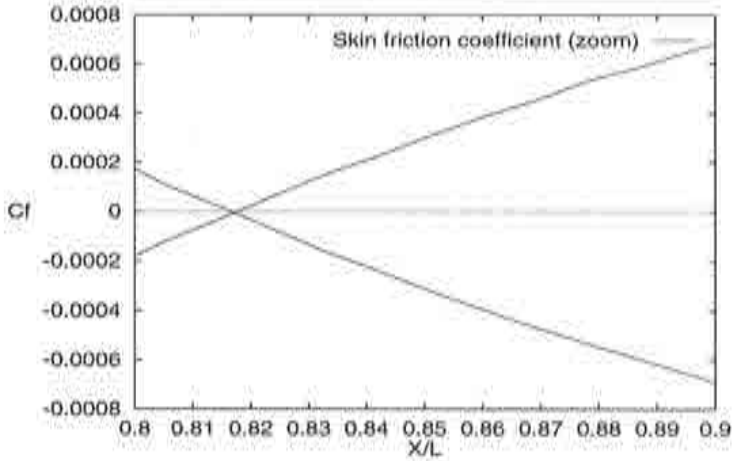


Figure 4.7 : Close-up of the skin friction coefficient along the airfoil to show the location of the separation point, yielding about 81.8% for the solution of viscous flow around a NACA0012 airfoil (Re=5000, M_∞=0.5, α=0°).

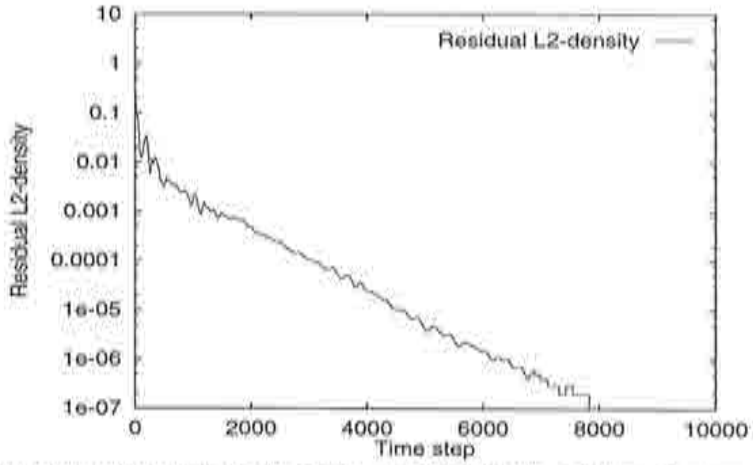


Figure 4.8 : Convergence history of the density residual for the solution of viscous flow around a NACA0012 airfoil (Re=5000, M_∞=0.5, α=0°).

Transonic Flow

When shocks occur, the incorporation of a pressure switched second order diffusion, makes it possible to capture a good resolution of the discontinuity without distorting the solution in smooth parts of the flow. The following examples illustrate this:

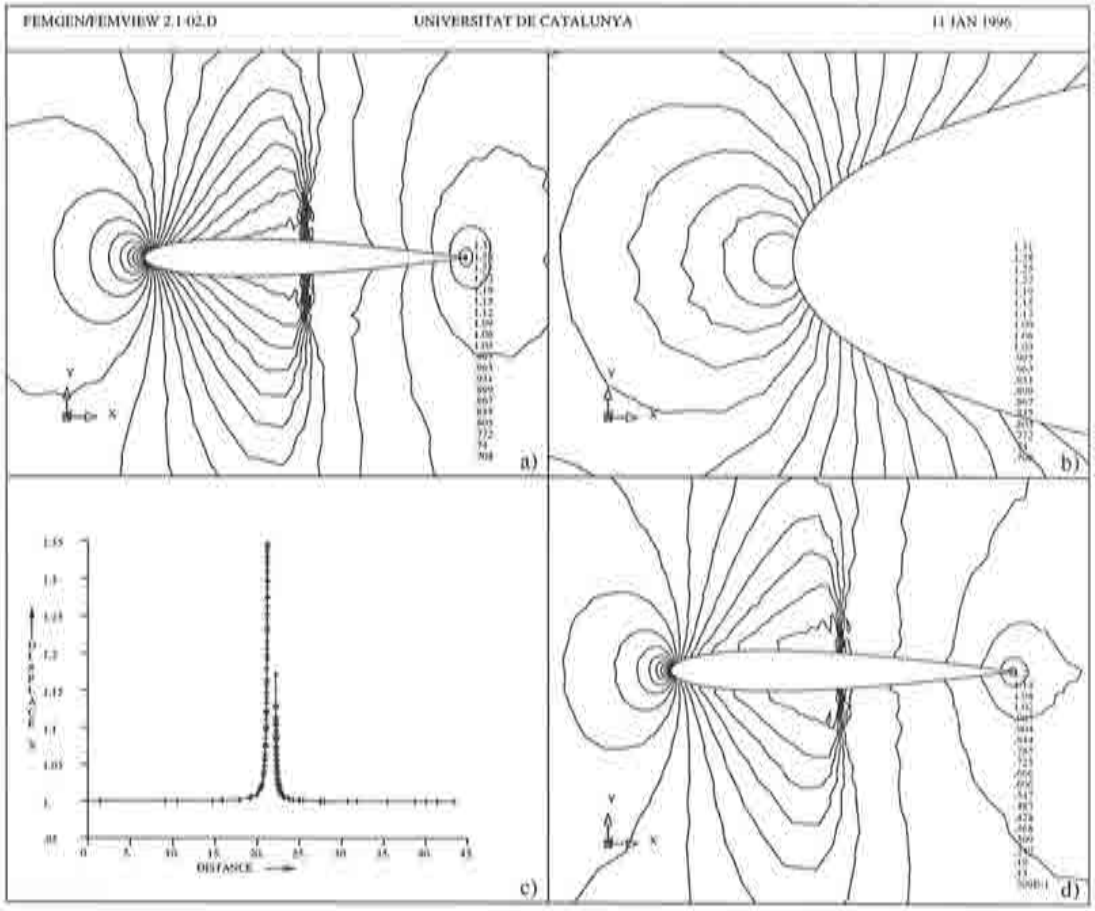


Figure 4.9 : NACA0012 profile for transonic inviscid test case $M_\infty = 0.8$, $\alpha = 0°$ using a mesh of 2556 nodes and 4902 elements: a) density contours, b) close up density lines in the stagnation area and c) density values along the stagnation stream line. d) Mach number contours.

- NACA0012: $M_\infty = 0.8$, at an angle of attack of $\alpha = 0°$.

  This test case demonstrates the quality of the flow solution for two dimensional inviscid transonic compressible flow. Using the same coarse unstructured mesh of 2556 nodes and 4902 elements of Figure 4.2 a), the shock is captured within three elements and two nodes without oscillations. In Figure 4.9, the a) density contours, b) density lines in the stagnation area, c) The stagnation stream line and d) Mach number plots are shown. Figure 4.9 c) presents the density values along the stagnation stream line showing no oscillations showing a peak of about 1.35. The pressure coefficient in Figure 4.10 clearly indicates the definition of the shock and its location of about 0.5 chord.

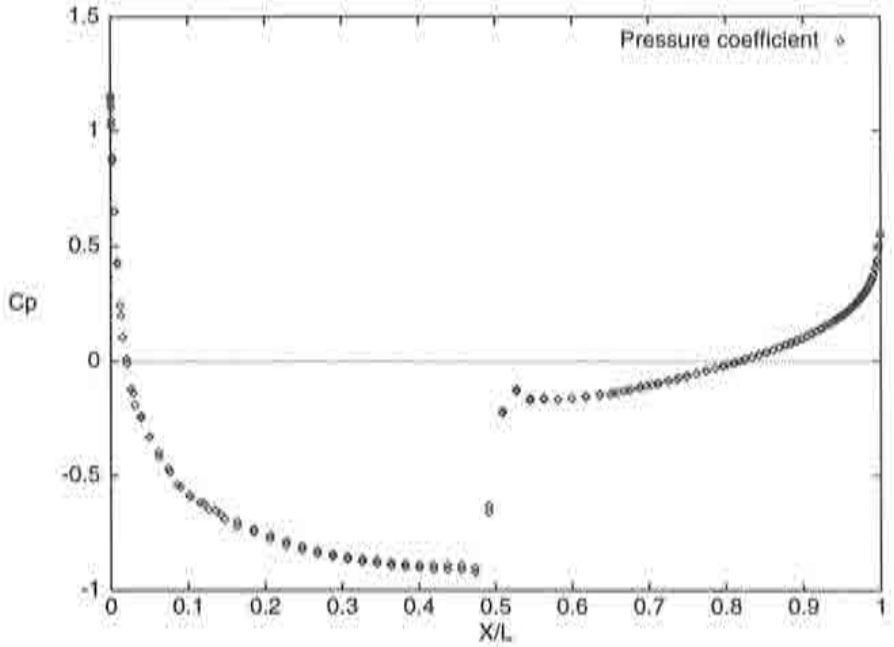


Figure 4.10 : Pressure coefficient for transonic inviscid flow around a NACA0012 airfoil at 0° angle of attack and $M_\infty = 0.8$.

The analytical value of the density in the stagnation point from eq. 4.70 is 1.351 which agrees well with the calculated result of 1.346 which underpredicts the exact value of only 0.41%. The fourth order diffusion constant was chosen as $\alpha^{(4)} = 0.02$, whereas the second order diffusion coefficient was ($\alpha^{(2)} = 0.35$). The lift coefficient which should be zero is predicted at $c_L = 0.0054$ which is within reasonable limits for a non symmetric mesh without adaptation.

- NACA0012: $M_\infty = 0.8$, at an angle of attack of $\alpha = 1.25^\circ$.

  This test case is taken from the reports of the AGARD working group 07 [32] which have contributed to a data base of transonic and supersonic airfoil solutions using numerical methods.

  Again, the same mesh of 2556 nodes and 4902 elements as seen in Figure 4.2 a) has been used. The artificial constant $\alpha^{(2)}$ and $\alpha^{(4)}$ has to be increased to 1.0 and 0.05, respectively, in order to compensate for the stronger shock on the upper surface of the airfoil. Figure 4.11 shows the density contours from which the location of the shock can be observed. The shock itself is captured within 2 elements with minor oscillation before and after the stronger shock, see the pressure coefficient in Figure 4.12. The result is graphically compared with the results of [32]. Considering that the reference results of [32] were obtained on a adapted mesh using 36400 points, the results are in good agreement for the coarse mesh of 2556 points.

  The drag and lift coefficients are $c_D = 0.0209$ and $c_L = 0.3176$, respectively. The underprediction of the lift coefficient is as expected. This is due to the reduced distance of the outer boundary which is only 20 chords compared to 48 chords of

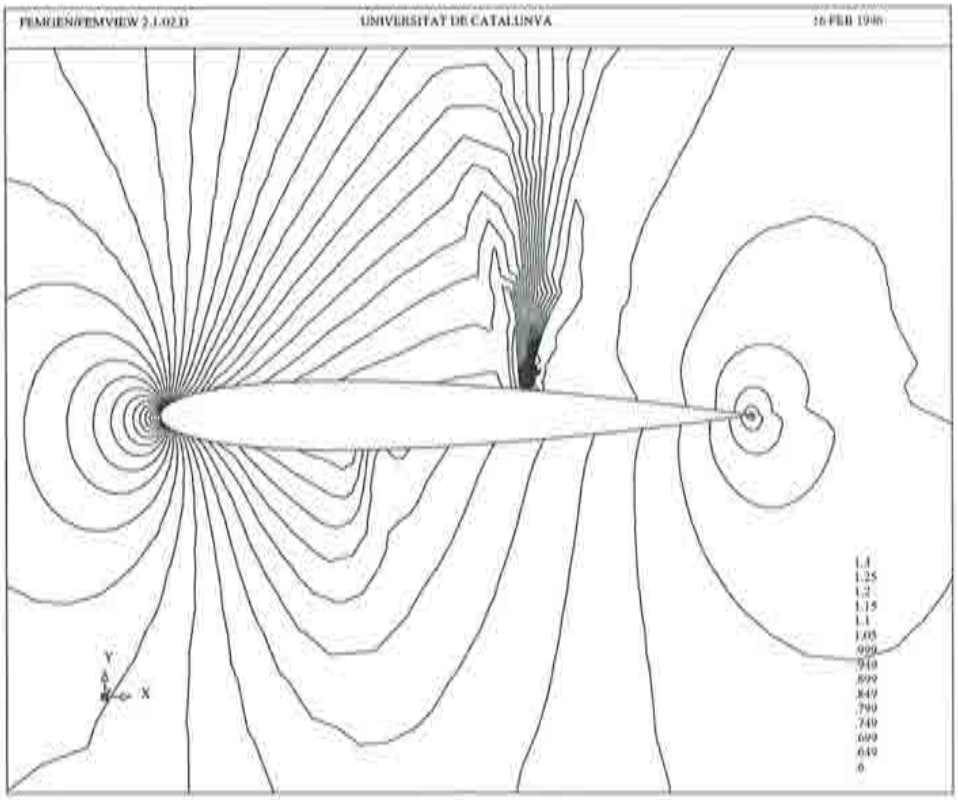


Figure 4.11 : Density contours for a NACA0012 profile for transonic inviscid flow ($M_\infty = 0.8$, $\alpha = 1.25°$) using a mesh of 2556 nodes and 4902 elements.

the reference test case. In [32] the influence of the outer distance is studied and shows an underprediction of about 6% in the lift coefficient. Thus, correction of the lift coefficient gives a value of $c_L = 0.3367$ which is slightly lower than the reference results using finer and adapted meshes. The drag coefficient (without correction) is also underpredicted where values near 0.0225 are computed. The convergence of the residuals (L2 norm of the density) has dropped by more than eight orders of magnitude as shown in Figure 4.13.

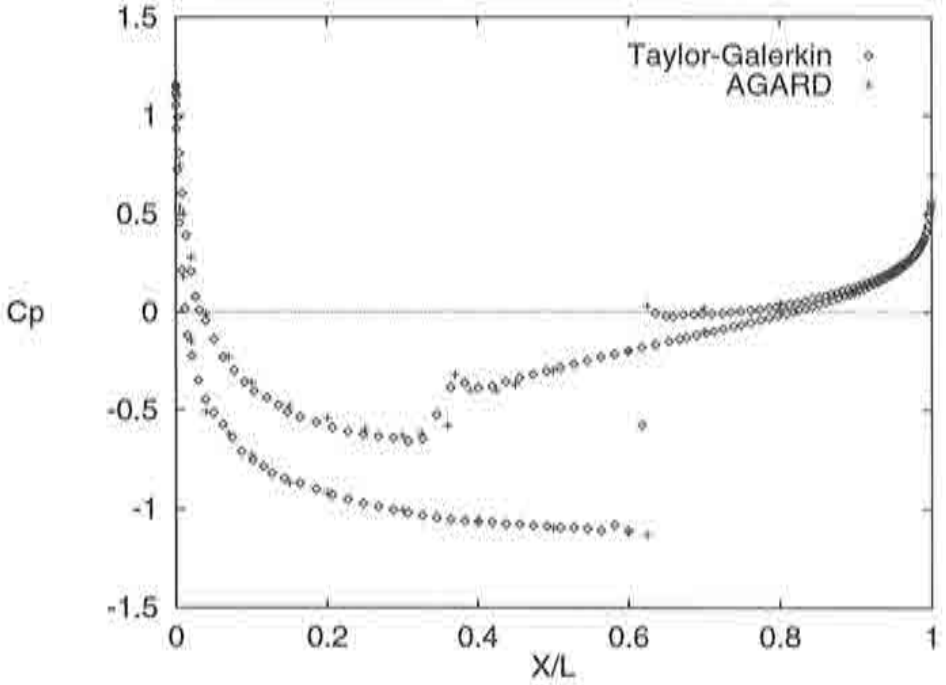


Figure 4.12 : Pressure coefficient for transonic inviscid flow around a NACA0012 airfoil at 1.25° angle of attack and $M_\infty = 0.8$ (AGARD test case #1 [32]).

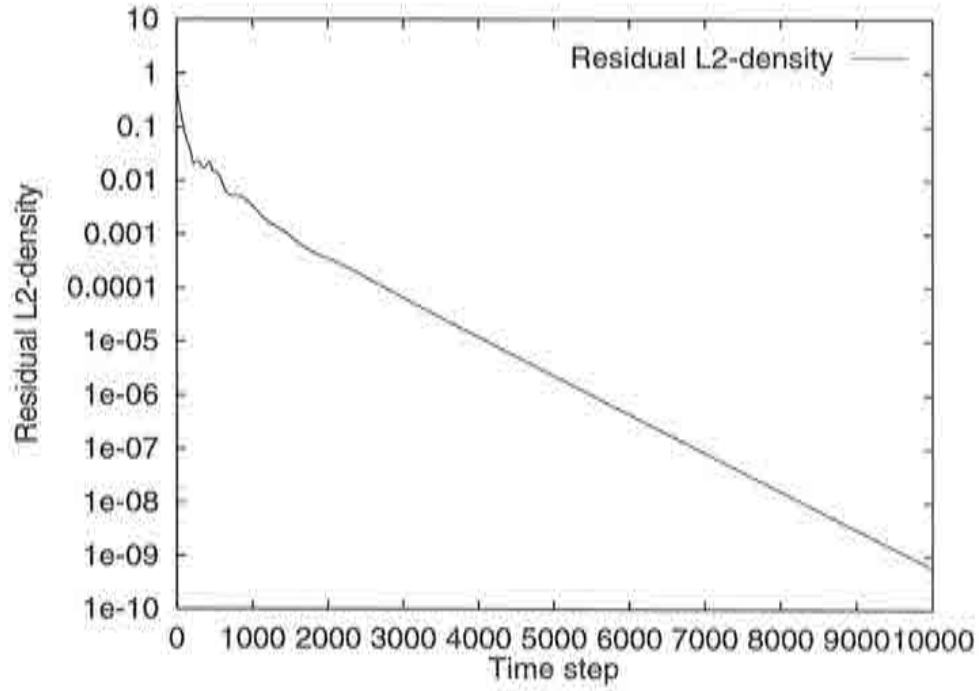


Figure 4.13 : Convergence history for transonic inviscid flow around a NACA0012 airfoil at 1.25° angle of attack and $M_\infty = 0.8$.

Supersonic Flow

Two geometries are selected for tests using supersonic fluid flow calculations. The first geometry is a ramp with a given angle and the other geometry is a double ellipse taken from a known test case of a workshop.

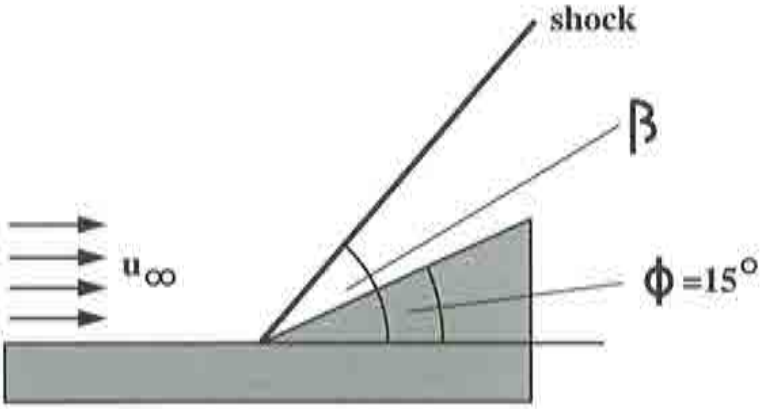


Figure 4.14 : Inviscid flow past a ramp of $\Phi = 15°$ inclination at $M_\infty = 2$.

- Corner flow: $M_\infty = 2$, at an angle of attack of $\alpha = 0°$, the ramp angle is $\Phi = 15°$. Schematically this can be seen in Figure 4.14. From fluid dynamics theory we can derive certain analytical results for this test case, such as the shock location or the Mach number downstream. In order to obtain the shock location angle it is necessary to solve the following third order equation equation for the angle β:

$$tan^3(\beta) - \frac{L^2-1}{tan(\Phi)} tan^2(\beta) + \left(1 + \frac{2}{\gamma+1} L^2\right) tan(\beta) + \frac{1 - \frac{\gamma-1}{\gamma+1} L^2}{tan(\Phi)} = 0 \qquad (4.71)$$

where L is the Laval number, Φ is the ramp angle. The Laval number is a function of the Mach number:

$$L = \sqrt{\frac{\gamma+1}{\gamma - 1 + \frac{2}{M^2}}} \qquad (4.72)$$

Hence, for a ramp angle of 15 degrees at $M_\infty = 2$, a Laval number of 1,633 results. Solving equation 4.71, a shock angle of $\beta = 45,344°$ results analytically. The calculations on a mesh of 1535 points and 2908 elements of similar size (Figure 4.15 yields an angle of approximately 45 degrees, which means that the shock location on this mesh is accurately captured. Figure 4.16 displays Mach number contours which indicate the shock inclination. The shock is captured within three elements or two nodes, if analyzed closely. At the wall boundary this can also be observed from pressure coefficient which is compared to the exact value in Figure 4.17. The convergence history for this triangular mesh is compared later on to quadrilaterals in Figure 4.18.

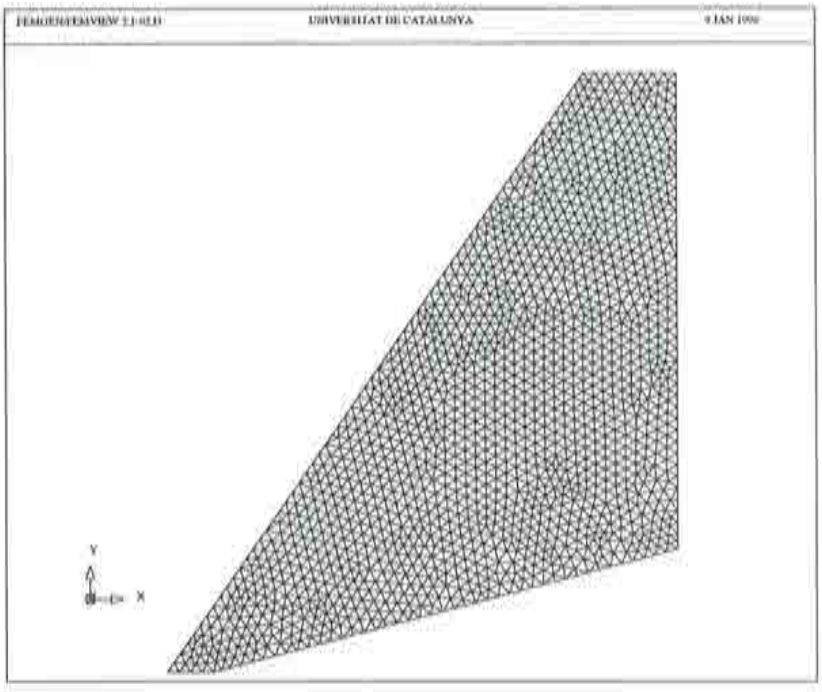


Figure 4.15 : Mesh of 1535 points and 2908 elements of similar size and shape, for $M_\infty = 2$.

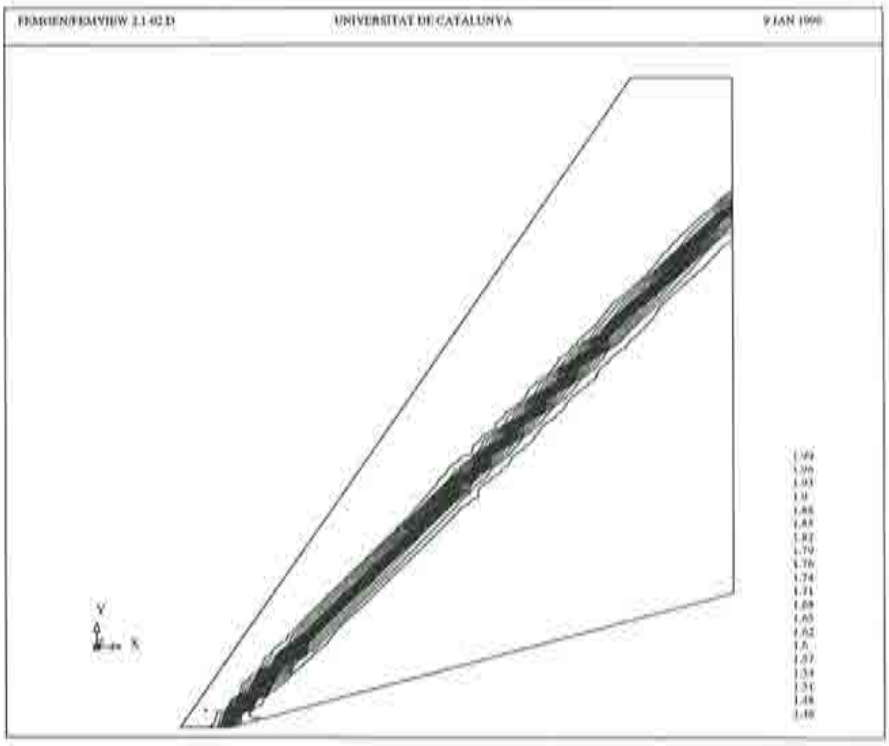


Figure 4.16 : Mach number contours for the corner flow using a mesh of 1535 points and 2908 elements at $M_\infty = 2$. Note the shock angle is approximately $45°$.

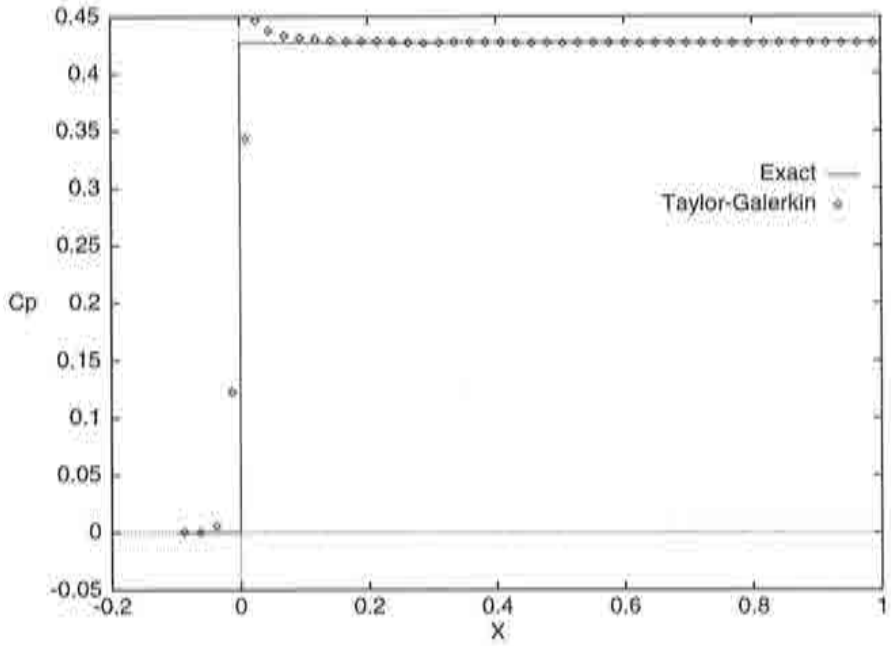


Figure 4.17 : Pressure coefficient compared to the analytical result at $M_\infty = 2$. Note the shock is captured within three elements and two nodes.

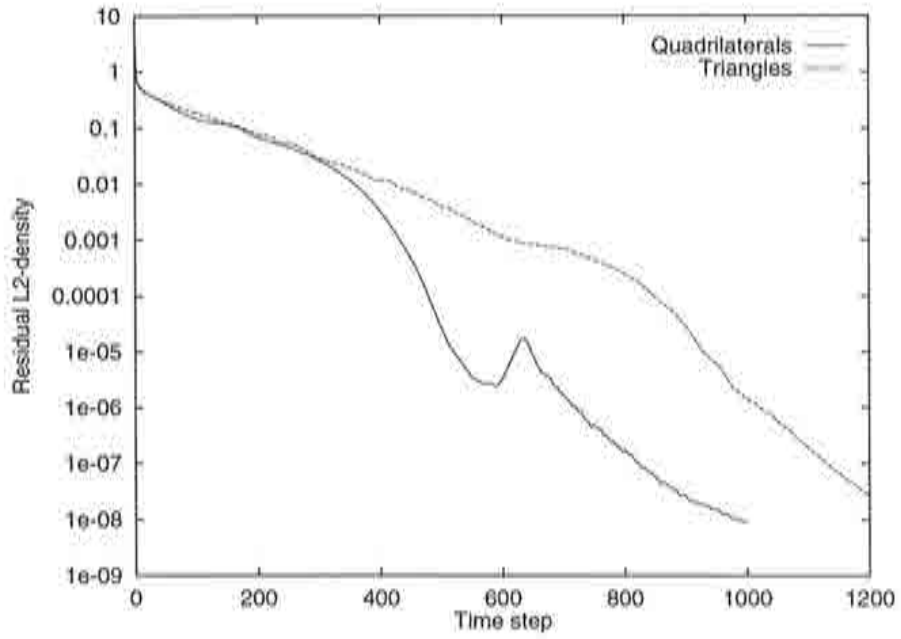


Figure 4.18 : Convergence history for supersonic inviscid flow past a ramp at 0° angle of attack and $M_\infty = 2$. Shown is the L2 norm of the density residuals for triangular and quadrilateral meshes of 1535 and 2163 nodes, respectively.

- Viscous hypersonic flow past a double ellipse geometry at $M_\infty = 8.15$, $Re/m_\infty = 1.67 \times 10^7$, $T_\infty = 56$K and $\alpha = 30°$ (L=0.06m).

  This example corresponds to a workshop test problem of the Workshop for Reentry Problems held in Antibes 1990 and 1991 (test case: 6.1.2) [20] and [21], hence it provides a good basis for comparison with other contributors, for instance references [22, 23].

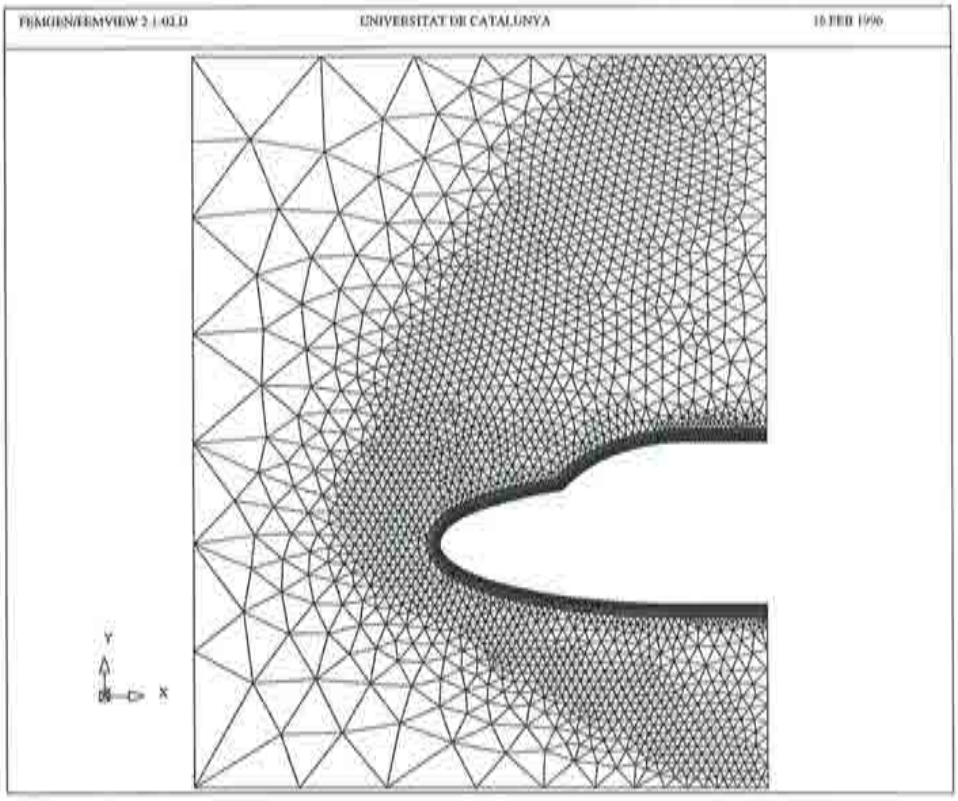


Figure 4.19 : Initial mesh of 2987 points and 5757 elements for the adaptive solution of hypersonic viscous flow.

The sequence of meshes can be seen in Figures 4.19, 4.20 and 4.21. A detail of the boundary element layer can be observed in Figure 4.22. The adaptive refinement process can be observed for which the Mach number was used as the scalar error variable.

Figure 4.23 shows Mach number contours where a clear definition of the shock can be appreciated. No wiggles are present in the contour lines and only a slight overshoot of 1.7% in the Mach number is observed. The next graph (Figure 4.24) presents density contours. No oscillations are present in the stagnation area. A better test for accuracy is the pressure coefficient along the surface of the object which is presented in Figure 4.25. The plot agrees well woth the results of the workshop where a peak of nearly $c_P = 1.8$ is reached. The convergence history of the maximum residual is presented in Figure 4.26.

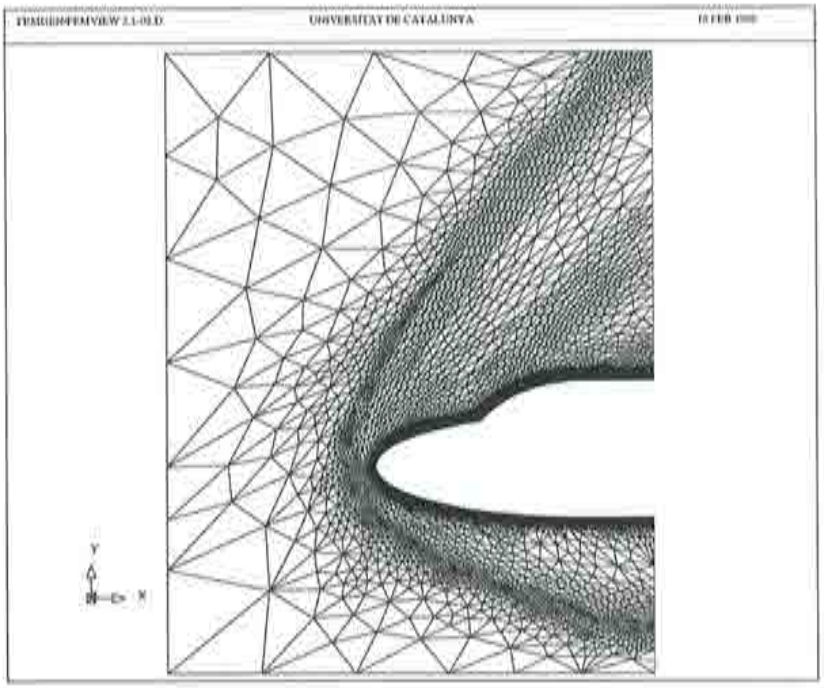


Figure 4.20 : First adapted mesh of 4300 points and 8252 elements for the adaptive solution of hypersonic viscous flow.

The results obtained for this test case using the adaptive Taylor-Galerkin scheme to solve laminar viscous flow are very good. This provides a good basis for the extension of the algorithm to include real gas effects, chemical reactions and or even turbulence in order to better simulate and to obtain a more precise prediction of real reentry problems.

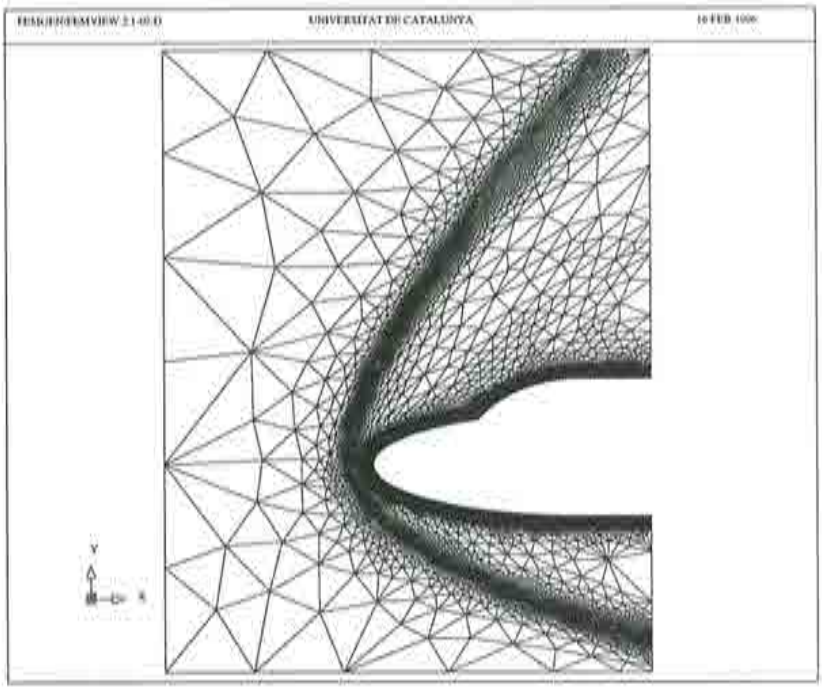


Figure 4.21 : Second adapted mesh of 8983 points and 17410 elements for the adaptive solution of hypersonic viscous flow.

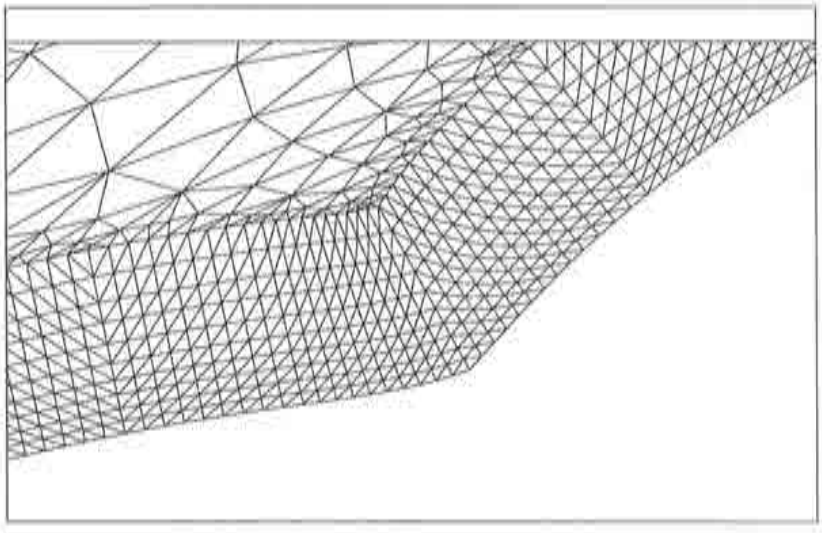


Figure 4.22 : Detail of the structured layer of elements along the boundary near the canopy. 10 layers are displayed.

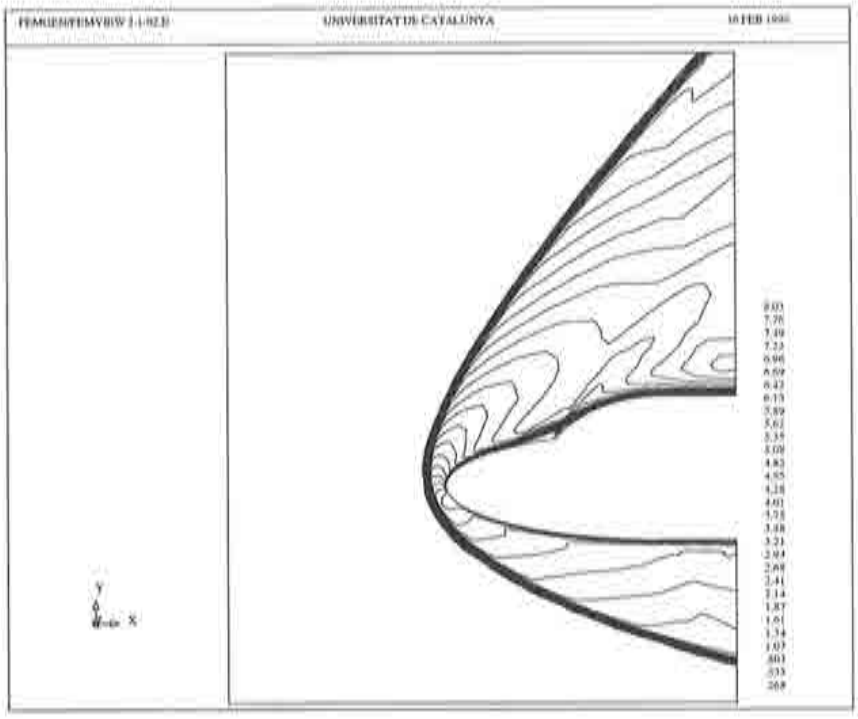


Figure 4.23 : Mach number contours for the hypersonic viscous flow around a double ellipse at an incidence of 30° for the final mesh.

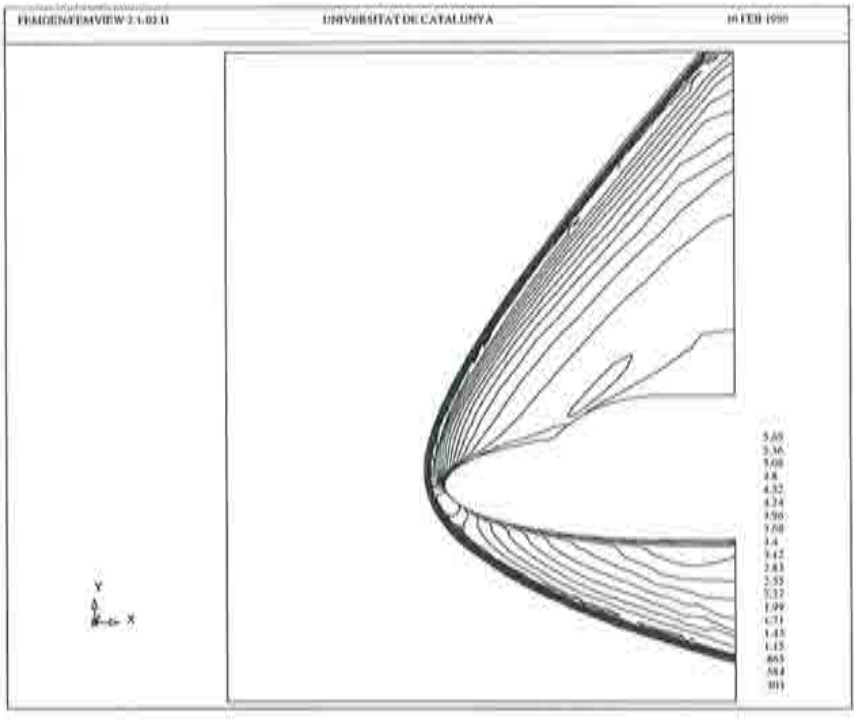


Figure 4.24 : Density contour lines for the hypersonic viscous flow around a double ellipse at an incidence of 30° for the final mesh.

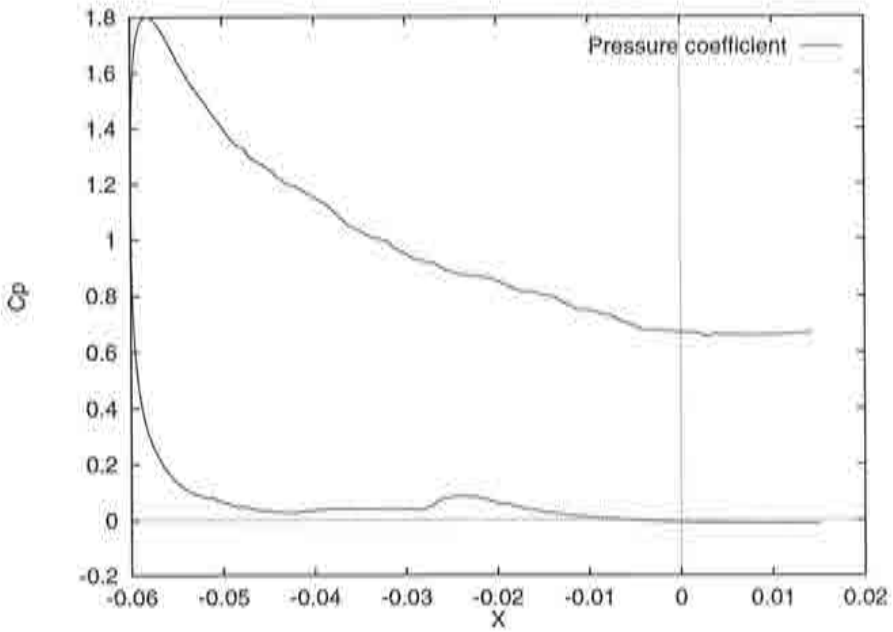


Figure 4.25 : Pressure coefficient for hypersonic viscous flow around a double ellipse for the adapted mesh (30° angle of attack and $M_\infty = 8.15$).

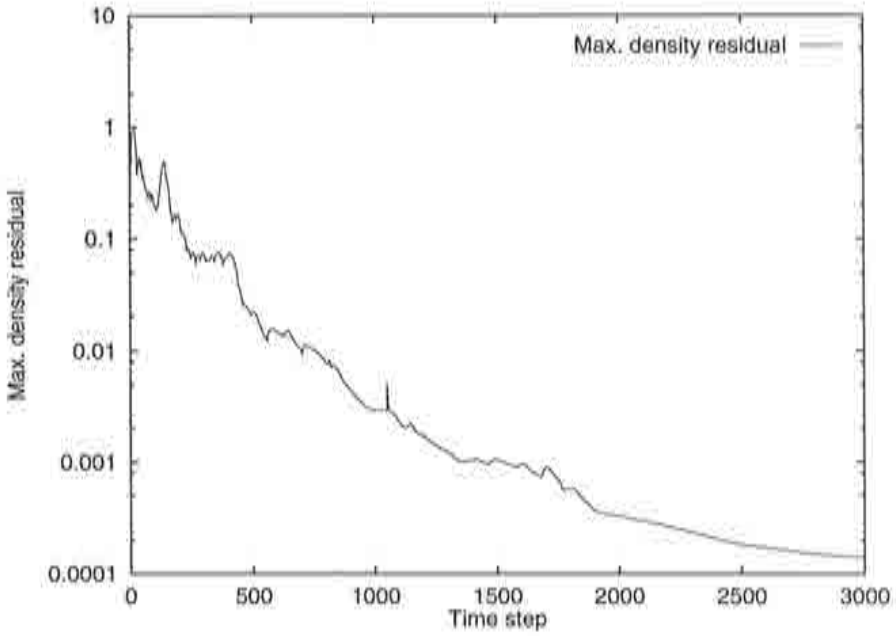


Figure 4.26 : Convergence history of the maximum density residual for hypersonic viscous flow around a double ellipse for the adapted mesh (30° angle of attack and $M_\infty = 8.15$).

4.10.2 Quadrilaterals

Subsonic Test Case

NACA0012: $M_\infty = 0.5$, inviscid flow at an angle of attack of $\alpha = 0°$.

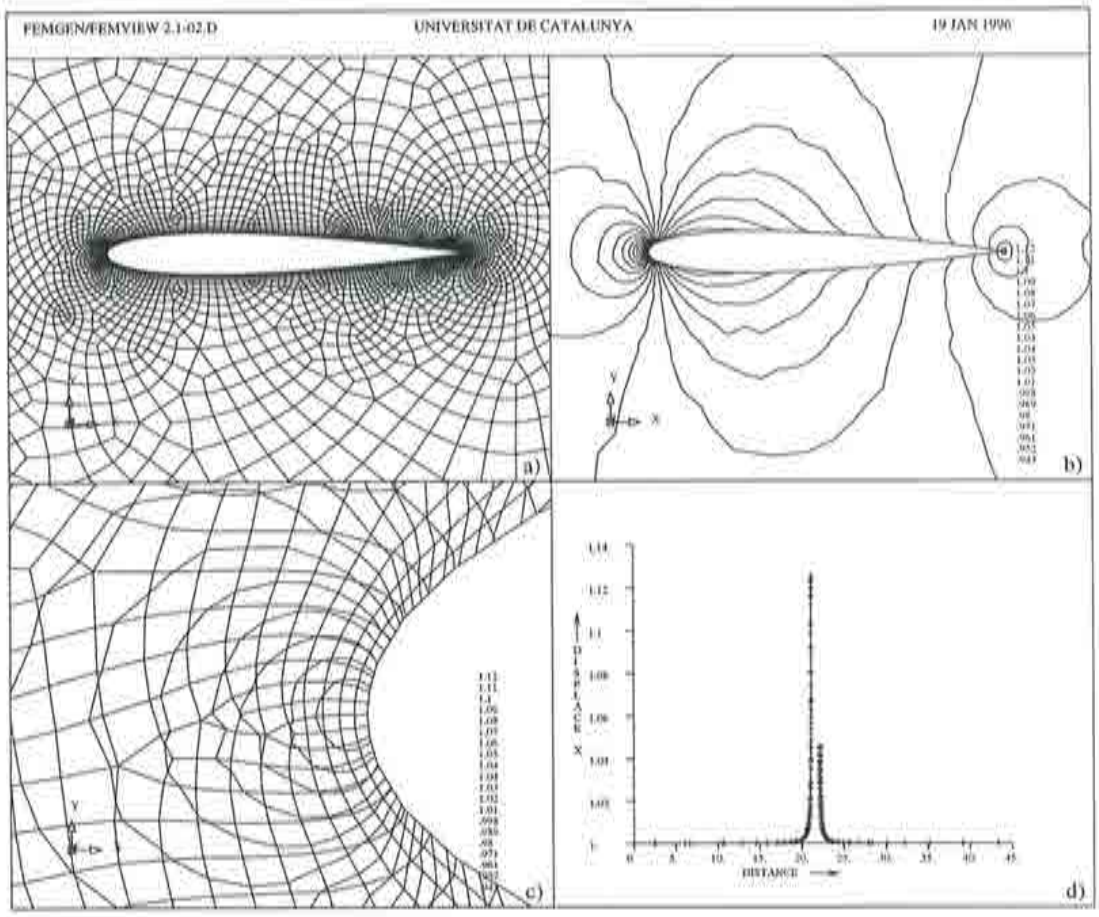


Figure 4.27 : NACA0012 profile for inviscid test case $M_\infty = 0.5$, $\alpha = 0°$: a) Finite element quadrilateral mesh of 4412 nodes and 4288 elements, b) density contours, c) close up density lines in the stagnation area and d) density values along the stagnation stream line.

The same test case as for linear triangles shows the quality of the solution for two dimensional inviscid subsonic compressible flow. For the unstructured mesh of 4412 nodes and 4288 elements shown in Figure 4.27 a), good results are obtained. In Figure 4.27 b) and c) density contours and their close up values in the stagnation area are shown, respectively. No oscillations are observed, neither in the global solution neither along the stagnation stream line of Figure 4.27 d).

Again the test for accuracy is the value of the density at the stagnation point for which the analytical result is known $\rho_0 = 1.129726$. The numerical result for this test case was obtained as 1.1294 which is about 0.03% less than the analytical value. The fourth order diffusion constant was $\alpha^{(4)} = 0.6$ and the second order diffusion was switched off ($\alpha^{(2)} = 0$). The pressure coefficient can be observed in Figure 4.28

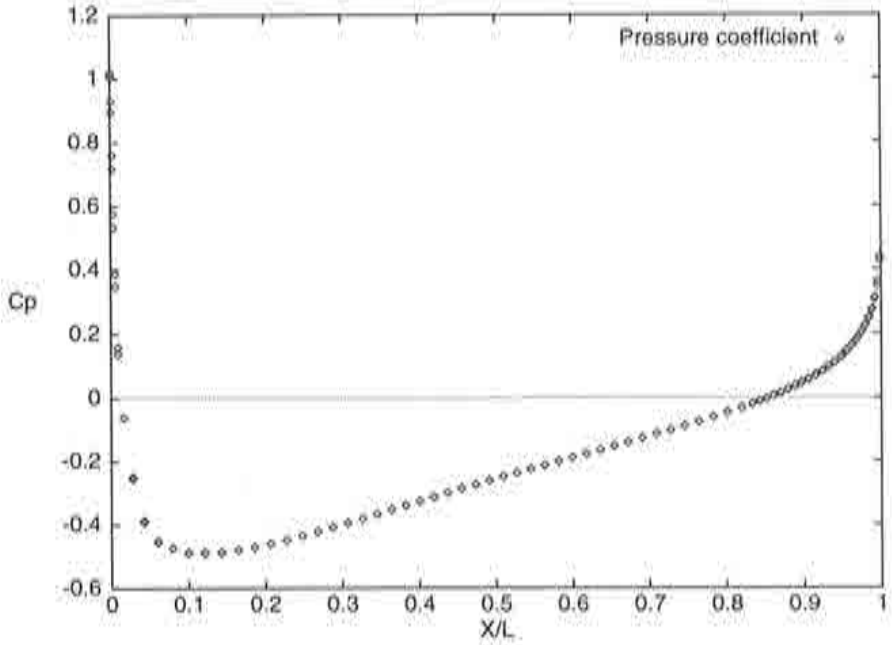


Figure 4.28 : Pressure coefficient for subsonic inviscid flow around a NACA0012 airfoil using quadrilaterals at 0° angle of attack and $M_\infty = 0.5$.

Another test for accuracy are the drag and lift coefficients which must be zero for subsonic shock free symmetrical flow around a symmetrical airfoil. The values for the aerodynamic coefficients of drag and lift are $c_D = 0.0018$ and $c_L = 0.0006$. This is a very precise result for this test case using a non symmetrical mesh without adaptivity. Convergence of the residuals is shown in Figure 4.29 and compared to a triangular mesh of 2556 nodes. The convergence versus the cpu time is shown in Figure 4.30. Obviously, as more nodes are used to obtain similar accuracy, the performance is slightly lower for quadrilaterals. A complete analysis and mesh convergence study for this test case using different meshes and different coefficients $\alpha^{(4)}$ is presented in section 6.4.2.

Comparing the results with those of the triangular mesh, there is a dependency of the quality of the results upon the mesh which can be detected from Figure 4.27 c). This is not so visible for the linear triangles because the mesh quality of the unstructured triangular mesh is much better. The reason for this is that the accuracy of the solution depends on the smoothness and quality of the mesh which determines the order of precision of the numerical scheme [29]. Generally speaking, unstructured quadrilateral meshes are of lower quality than similar triangular meshes. A study and discussion of the quality of quadrilateral meshes can be found in [30].

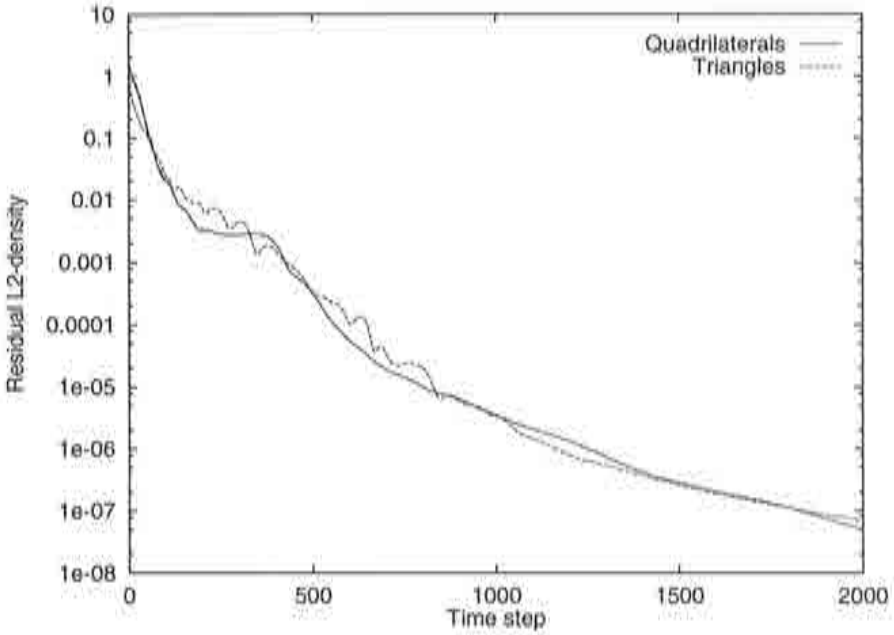


Figure 4.29 : Convergence history for subsonic inviscid flow around a NACA0012 airfoil at 0° angle of attack and $M_\infty = 0.5$. Shown is the L2 norm of the density residuals versus the time step for triangular and quadrilateral meshes of 2556 and 4412 nodes, respectively.

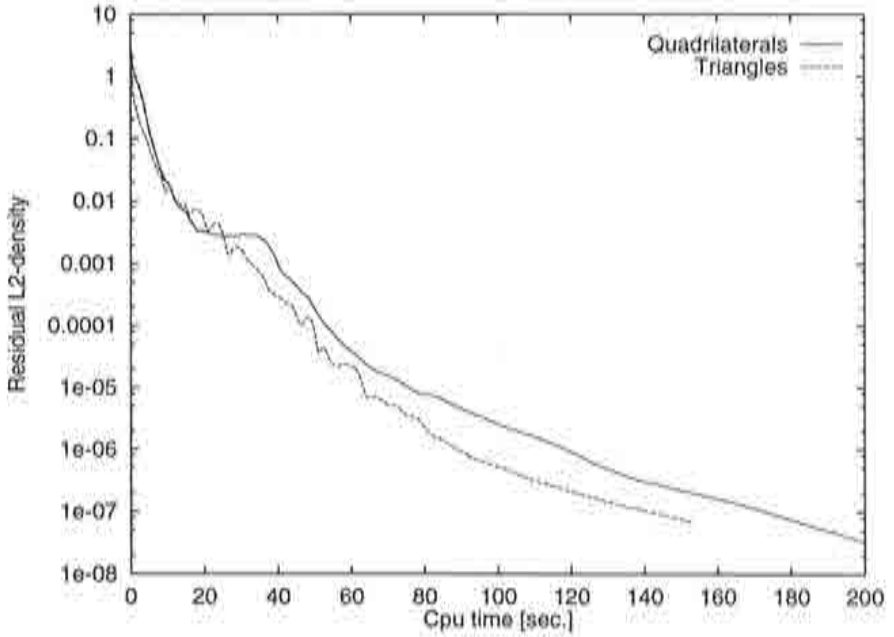


Figure 4.30 : Convergence history for subsonic inviscid flow around a NACA0012 airfoil at 0° angle of attack and $M_\infty = 0.5$. Shown is the L2 norm of the density residuals versus cpu time for triangular and quadrilateral meshes of 2556 and 4412 nodes, respectively.

Supersonic Test Case

Corner flow: Supersonic inviscid flow of $M_\infty = 2$, at an angle of attack of $\alpha = 0°$, the ramp angle is $\Phi = 15°$.

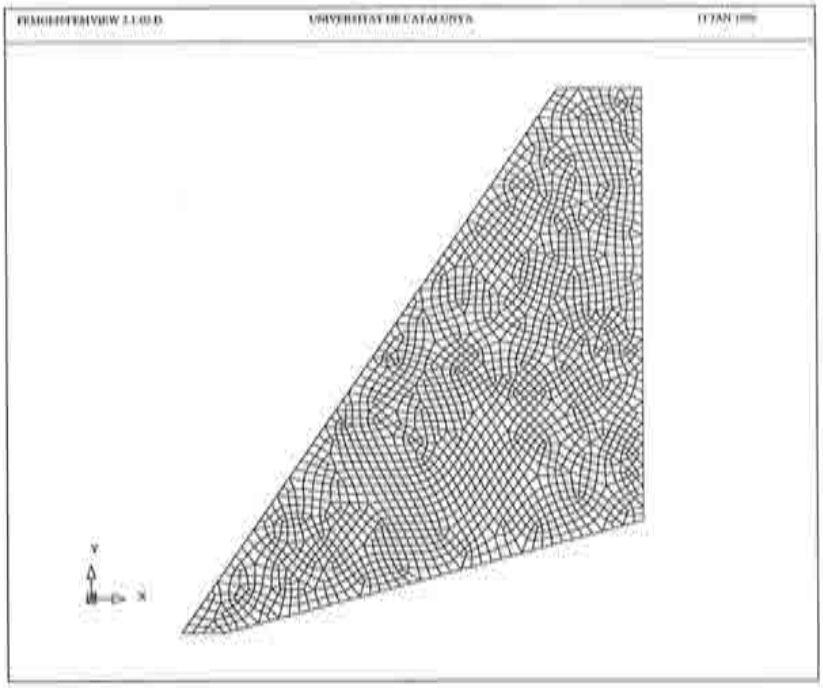


Figure 4.31 : Quadrilateral mesh of 2163 points and 2082 elements of similar size and shape, for $M_\infty = 2$.

This is the same test case for the ramp flow as described in section 4.10.1. Schematically the test case is shown in Figure 4.14. Recall that the analytical shock location can be obtained from eq. 4.71. For $\beta = 15°$ a shock angle of $\Phi = 45,344°$ results analytically. The calculations on a mesh of 2163 points and 2082 elements of similar size (Figure 4.31 yield an angle of approximately 45 degrees, which means that the shock location on this mesh is accurately captured. Figure 4.32 displays Mach number contours which indicate the shock inclination. The shock is captured within three elements or two nodes, if analyzed closely. At the wall boundary this can be also observed from pressure coefficient which is compared to the exact value in Figure 4.33. Convergence of the solution for this quadrilateral mesh is compared to the results obtained on a similar triangular mesh shown earlier in Figure 4.18.

Note that the quality of the quadrilaterals is very good and that no distorted elements appear. Thus, comparing for instance the pressure coefficient with that of the triangular mesh we observe that the result of the quadrilateral mesh is equal or better, using approximately the same number of nodes but half the number of elements. This shows that the solver of quadrilaterals is more efficient than the use of triangles for the same mesh quality and the same claim of precision. This will be analyzed more closely in chapter 6.

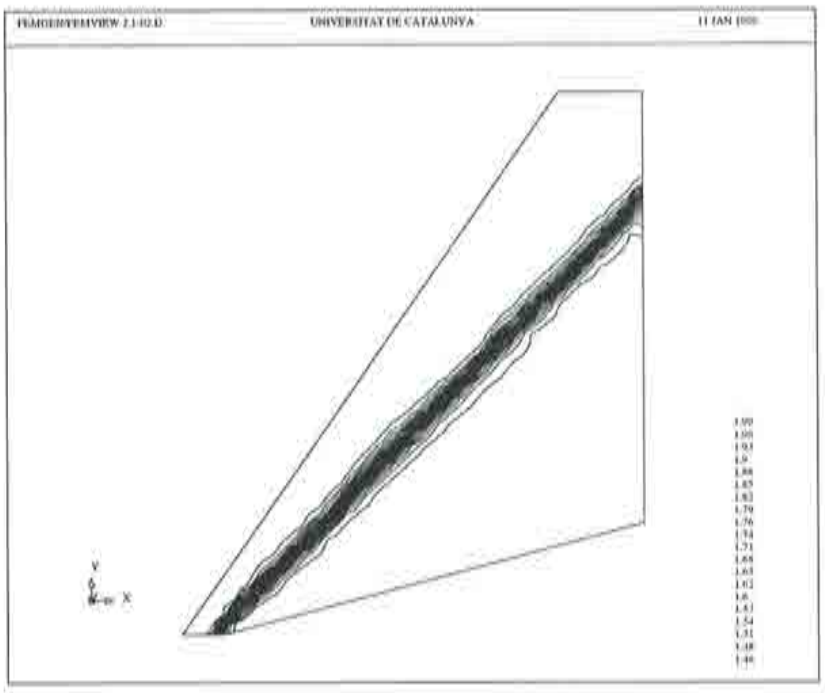


Figure 4.32 : Mach number contours for the corner flow using a quadrilateral mesh of 2163 points and 2082 elements at $M_\infty = 2$. Note the shock incidence is approximately 45°.

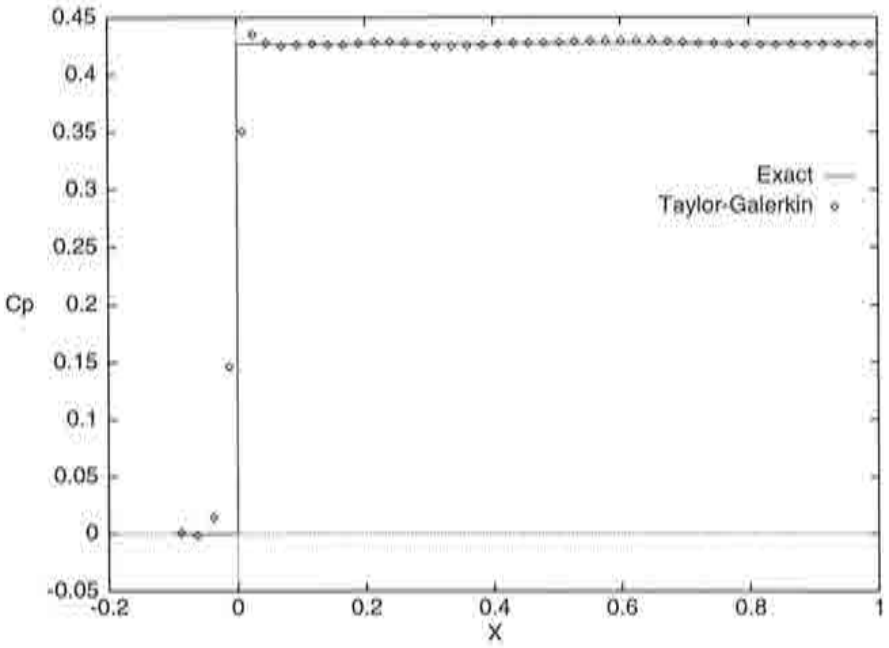


Figure 4.33 : Pressure coefficient compared to the analytical result at $M_\infty = 2$. Note the shock is captured within three elements and two nodes.

4.10.3 Three Dimensional Test Problem

Three dimensional results are only presented for supersonic flows. The particular objective of this test problem was to analyze the capabilities to solve the Euler equations on three dimensional unstructured meshes using adaptive remeshing techniques and compare them to other known solutions used by other authors employing different techniques. A well known test case was chosen for comparison, test case 6.1.7 of the I. Workshop on Hypersonic Flows for Reentry Problems. The two seemingly good computations from ONERA [33] and DLR [34] presented at the workshop were chosen for comparison of the pressure coefficient. ONERA uses a MUSCL type finite volume method on structured grids, whereas DLR uses a Jameson type Runge-Kutta finite volume method on structured grids, both of which are well known methods in the CFD community.

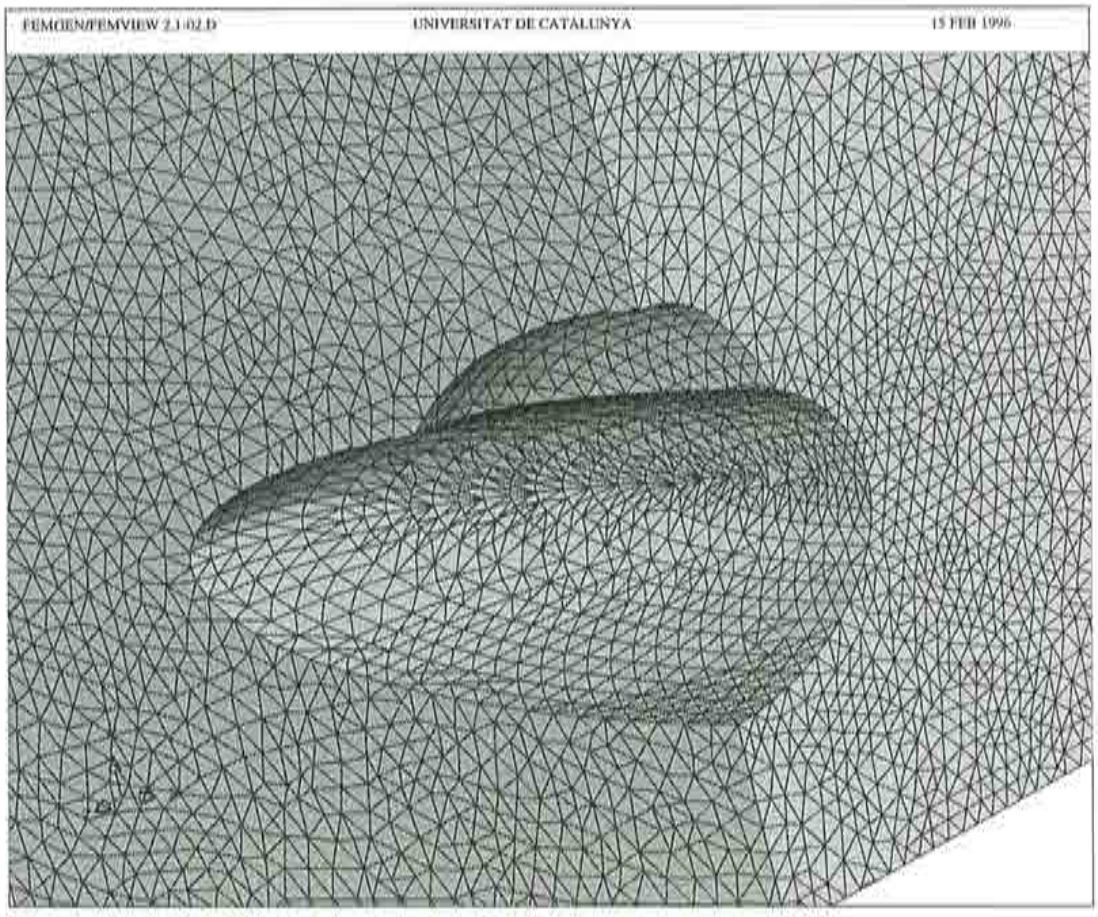


Figure 4.34 : Initial mesh for the calculation of hypersonic flow around a double ellipsoid (M_∞=8.15, α=30°) on an unstructured mesh of 38734 nodes.

The test example corresponds to the hypersonic inviscid flow past a double ellipsoid in three dimensions under the following conditions: $M_\infty = 8.15$, $\alpha = 30°$. All the calculations were obtained on a Silicon Graphics R4000 workstation, with the exception of the largest mesh which was run on a CRAY-YMP.

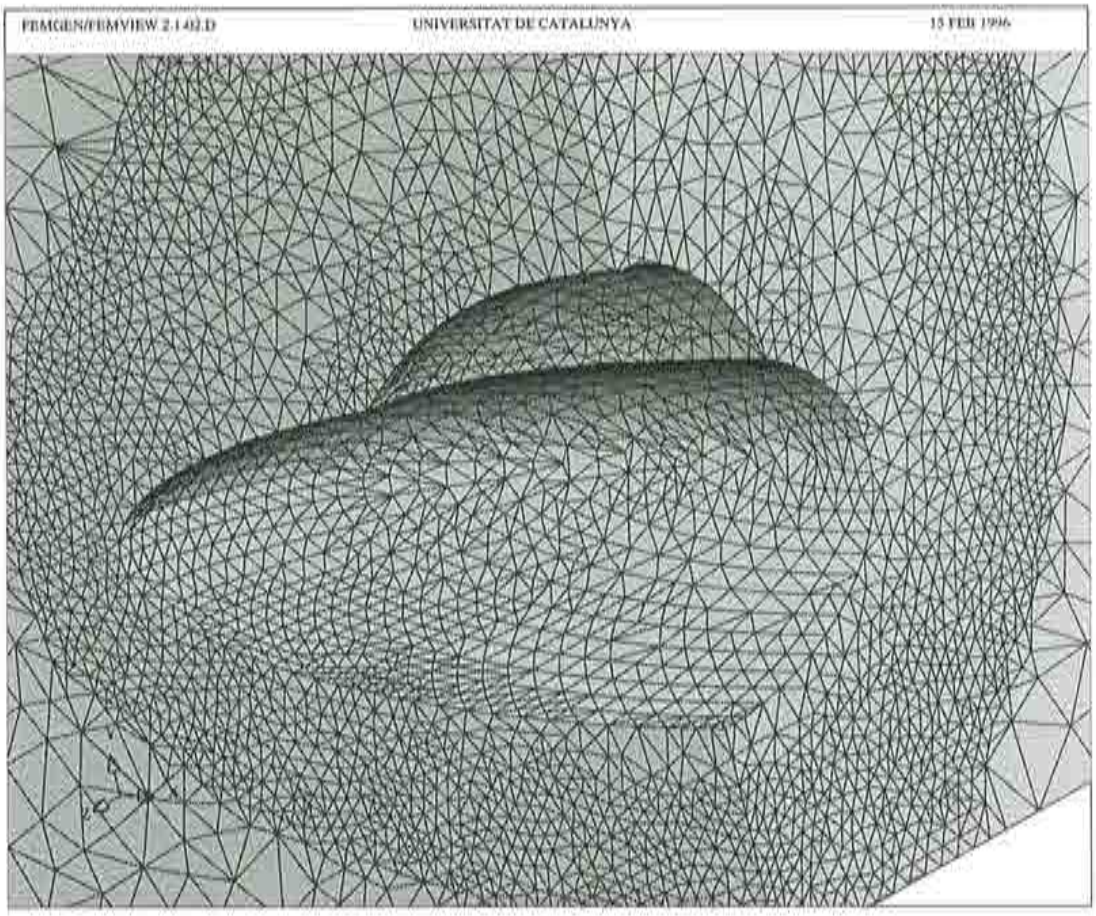


Figure 4.35 : Refined mesh for the calculation of hypersonic flow around a double ellipsoid (M_∞=8.15, α=30°) on an unstructured mesh of 35808 nodes.

The results for only three different meshes are presented for clarity even though several meshes have generated. The sequence of the initial fine mesh and the final refined mesh can be seen in Figures 4.34 and 4.35. The initial mesh with equal sized element contains 38734 nodes. After the refinement process two meshes were obtained, a coarse refined mesh of 35808 nodes and a fine refined mesh of 75006 nodes which is not shown. Note at this stage, the remeshing process has produced many small elements in the vicinity of the two shocks, according to the error estimator.

Figures 4.36 and 4.37 shows Mach number contours for the initial and coarse refined mesh respectively, where a clear improvement of the resolution of the shock can be seen. Note also that the canopy shock has been captured much more precisely using the finer grid, also the principal shock shows a crisper pattern than the coarse mesh. Figure 4.38 compares the pressure coefficient (c_p) of the three meshes for the plane of symmetry along the body to the results of different authors. The pressure peak is much better resolved using the more refined mesh and compares very well to other codes using structured meshes.

The final figures document the evolution of residuals and forces of the finest grid.

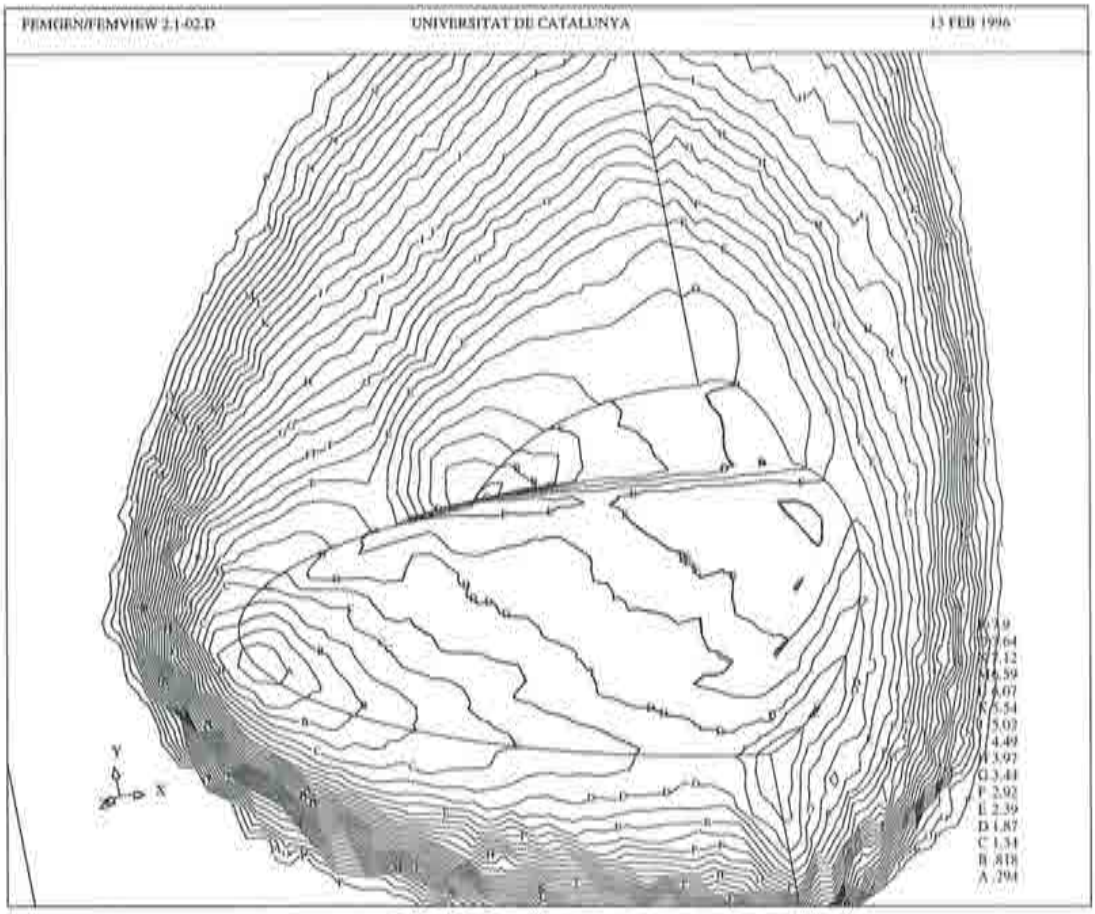


Figure 4.36 : Mach number lines for the initial mesh for the hypersonic flow around a double ellipsoid (M_∞=8.15, α=30°).

The convergence was of 5 orders of magnitude (Fig. 4.39), although some oscillations are present at some stage of the calculation. The convergence of the non dimensional forces Fx and Fy is represented in Figures 4.40 and 4.41. The cpu-time of the final calculation was about 7,6 hours on a Cray YMP.

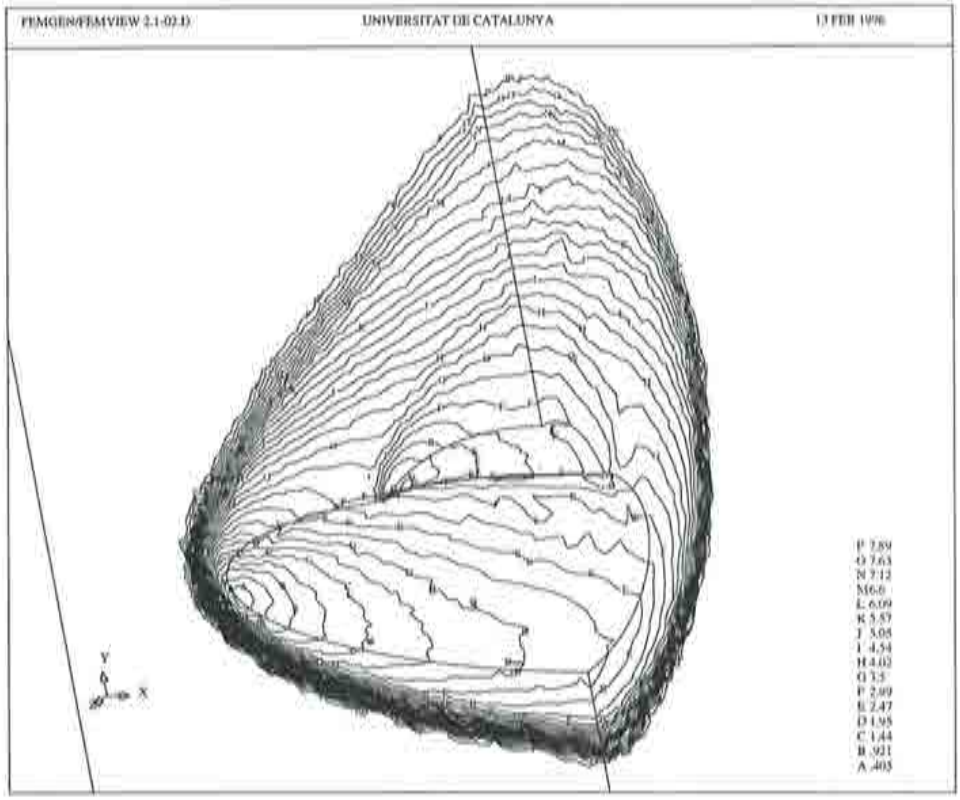


Figure 4.37 : Mach number lines for the refined mesh for the hypersonic flow around a double ellipsoid (M_∞=8.15, α=30°).

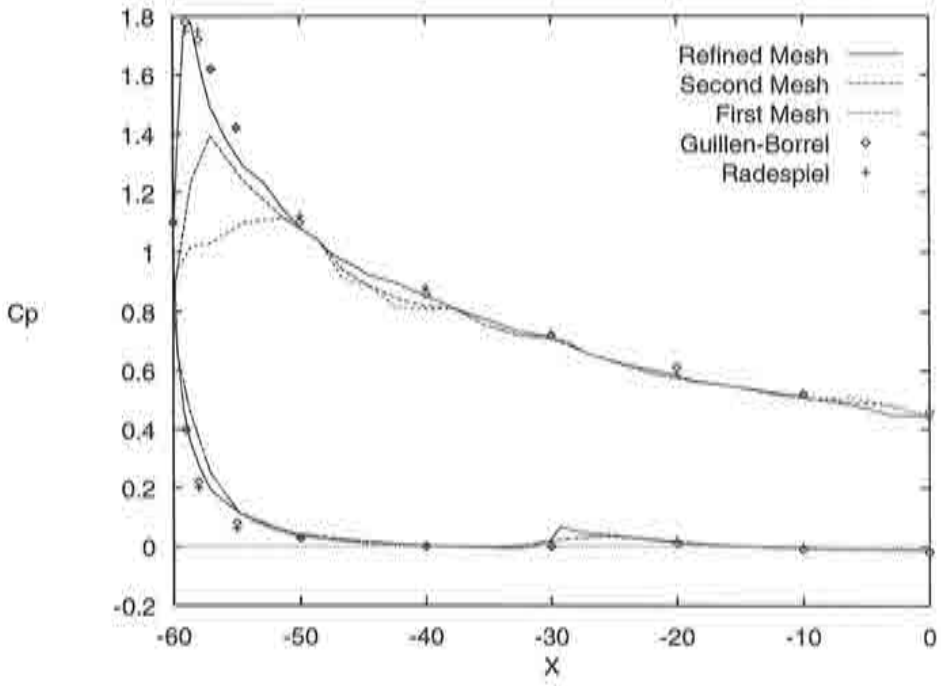


Figure 4.38 : Pressure coefficient for different meshes compared to other authors.

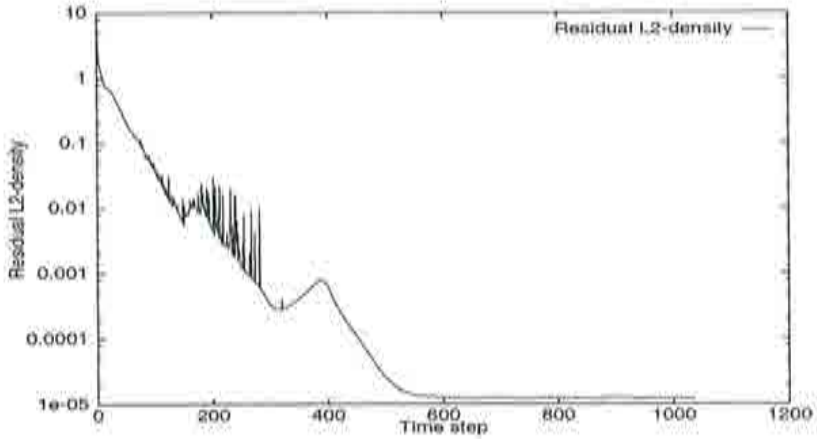


Figure 4.39 : Convergence history of the residual decrease of the L2 norm of the density versus the time step.

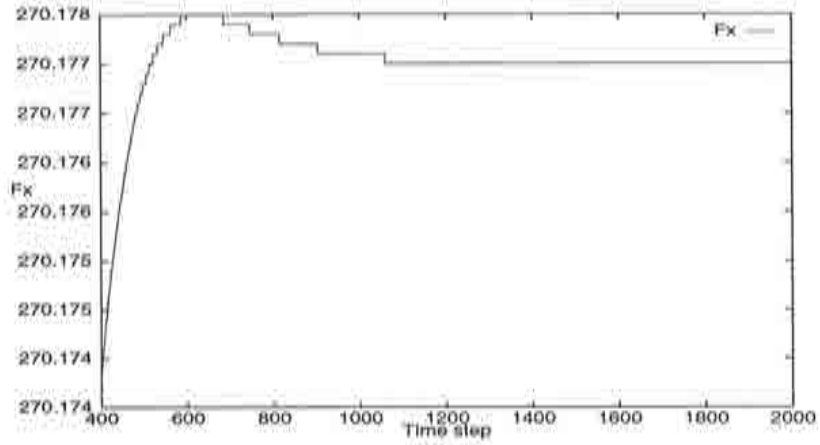


Figure 4.40 : Convergence of the non dimensional pressure force F_x versus the time step. Note that the solution does not change anymore after 1400 iterations.

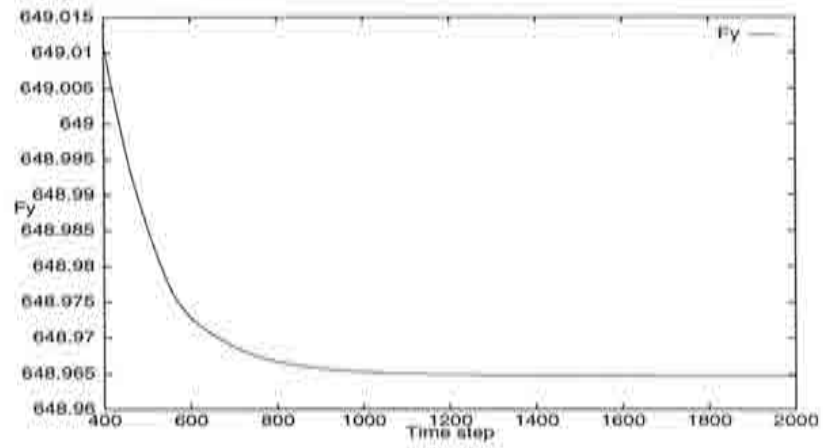


Figure 4.41 : Convergence of the non dimensional pressure force F_y versus the time step.

4.10.4 Axisymmetric Examples

Numerical results are presented for two test cases of hypersonic axisymmetric flows around a hyperboloid flare. The numerical results were stored in the EHVDB database at INRIA. In addition, experimental results are available which are also stored in the database. We have used the adaptive remeshing technique, local time stepping and artificial dissipation of the Jameson type to resolve these test cases. Both cases studied were run on a Connection Machine CM-200 which is a massive parallel computer. Its implementation will be discussed in the final chapter.

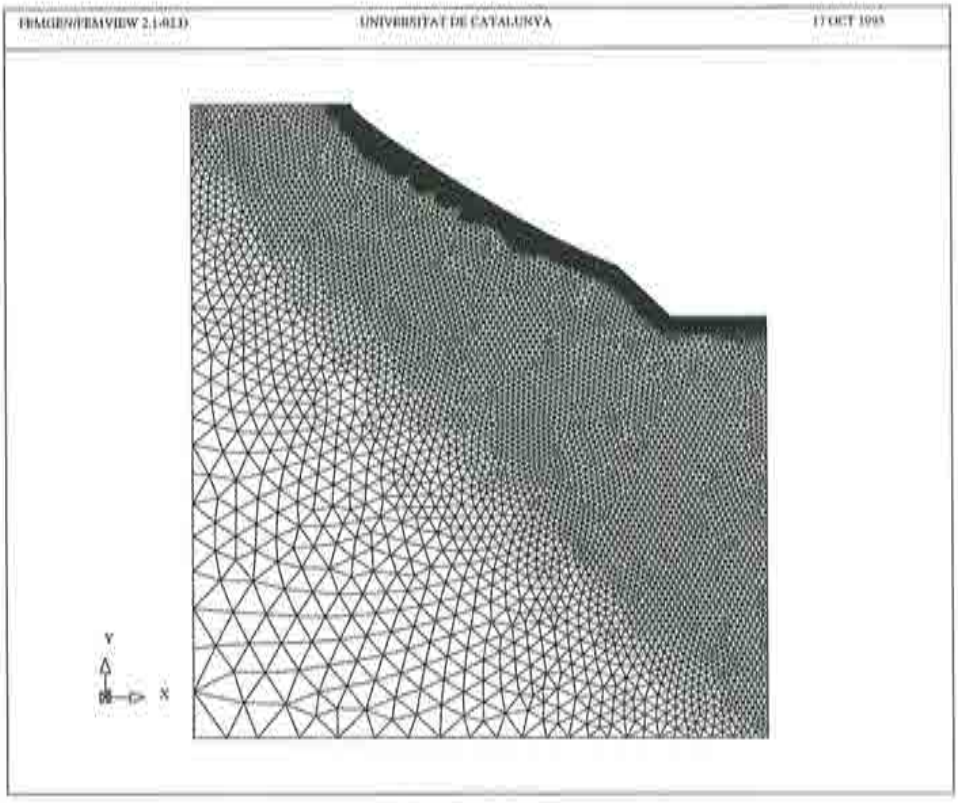


Figure 4.42 : Initial mesh of the refinement process for both test cases: H2K and LTB

H2K test case

The free-stream conditions for test case H2K were the following: Mach number M_∞= 8.7, Reynolds number Re_∞= 37025, isothermal wall T_w= 310 K and the temperature T_∞= 72.69 K

A total of three unstructured triangular grids are used to discretize the flow domain in the adaptive remeshing process; the initial and final mesh of a total of three are shown in Figures 4.42 and 4.43, respectively. The final mesh had 23155 elements, employing 14 structured layers of stretched elements along the surface of the object to capture boudary layer effects more accurately and efficiently. The results show a shock closer

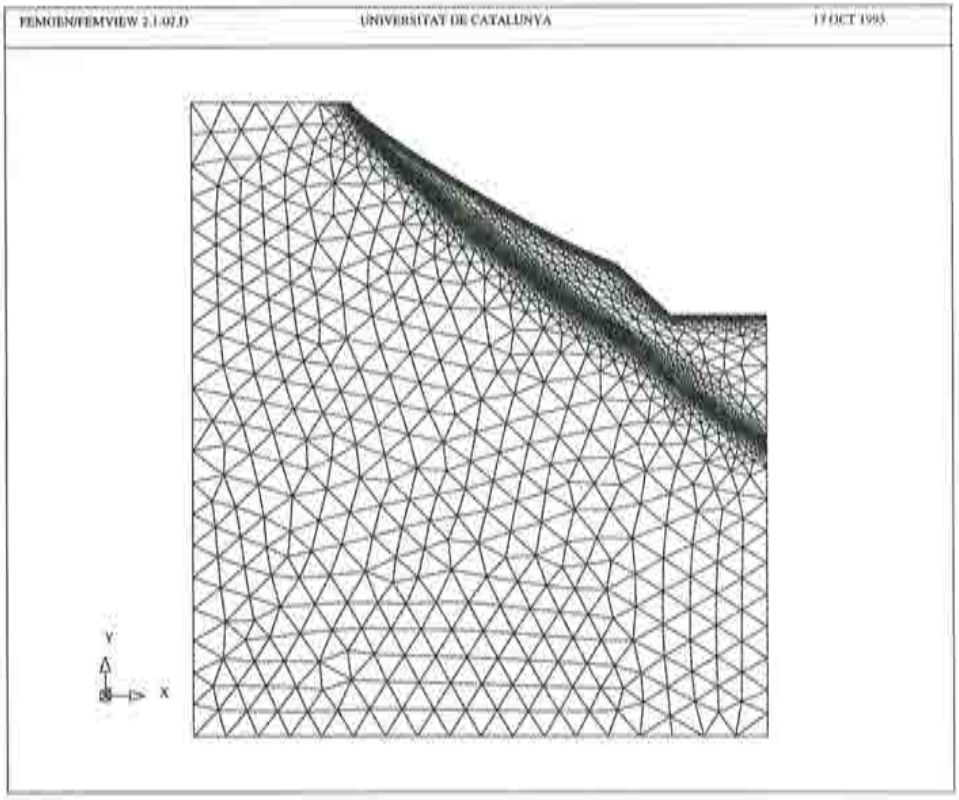


Figure 4.43 : Final adapted mesh for test case H2K

to the body for this axisymmetric geometry than one would expect in a regular two dimensional solution.

The solution has converged more than five orders of magnitude for the density residual (L2-norm) which is compared to the following test case LTB in Figure 4.50. Both, the aerodynamic coefficients do not show any significant change and the length of the separation bubble has reached a steady position. Figs. 4.44, 4.45 and 4.46 present the pressure coefficient, skin friction and Stanton number along the surface of the object, respectively. These values coincide closely to the experimental and numerical results which were presented at the workshop in November. In the plots of Figs. 4.44 and 4.46, the available experimental data obtained from the database is represented. It was not possible to obtain other experimental data yet.

The pressure coefficient c_P coincides well with the experiments, except for the area of the recirculation zone. In fact, one would expect a pressure increase as predicted by the numerical solutions, but this is not reflected in the experiments. All numerical solutions from other contributers for that test case predicted this pressure rise as well. On the other hand, the solutions for the Stanton number St coincides much better with the experiments, where only the peak values are not quite reached. The proceedings of this workshop which have not been published yet may give a more detailed explanation [35].

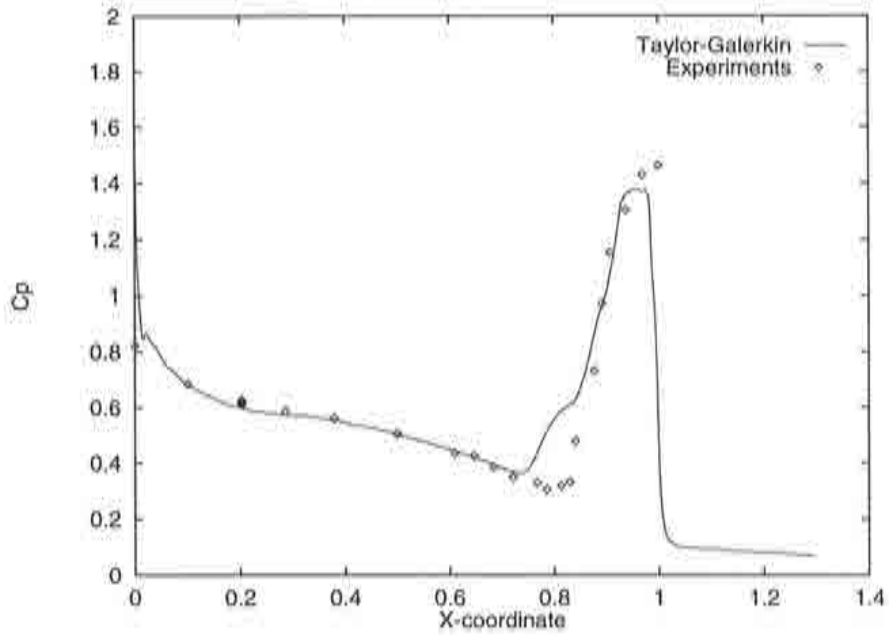


Figure 4.44 : Pressure coefficient for test case H2K. The computations using the Taylor-Galerkin scheme is compared to experimental results obtained from the database.

LTB test case

The free-stream conditions for the LTB test cse were: Mach number M_∞= 6.83, Reynolds number Re_∞= 414680, isothermal wall T_w= 310 K and the temperature T_∞= 67.765 K

Again, three meshes were used to resolve this case, similar to the ones shown in Figures 4.42 and 4.43. The size of the final mesh was 24026 elements with 14 structured layers of elements along the surface, which was similar to the grid shown in Figure 4.43. In this case the solution has converged to only three orders of magnitude for the density residual (L2-norm). A steady state has not been found indicated by the oscillations after 2000 iterations. Also the mesh does not seem to have sufficient points in that region which is reflected by small oscillations in the skin friction coefficient and the Stanton number. The cpu time for this case was almost four hours in the CM-200 at a CFL number of one. The pressure coefficient, skin friction coefficient and Stanton number for test case LTB can be observed in Figures 4.47, 4.48 and 4.49. Note the oscillations that are present, indicating that the solution has not reached a steady state. This was also observed by other contributors of numerical solutions.

A comparison of convergence histories for both test cases is presented in Figure 4.50. Clearly, test case H2K approaches a steady solution after a residual drop of five orders of magnitude at 10000 iterations, although the gradient after 4000 iterations has decreased due to the slow build up of the viscous recirculation bubble. Contrarily, test case LTB does not converge to a final steady solution, where the recirculation bubble in the corner of the flap moves back and forth.

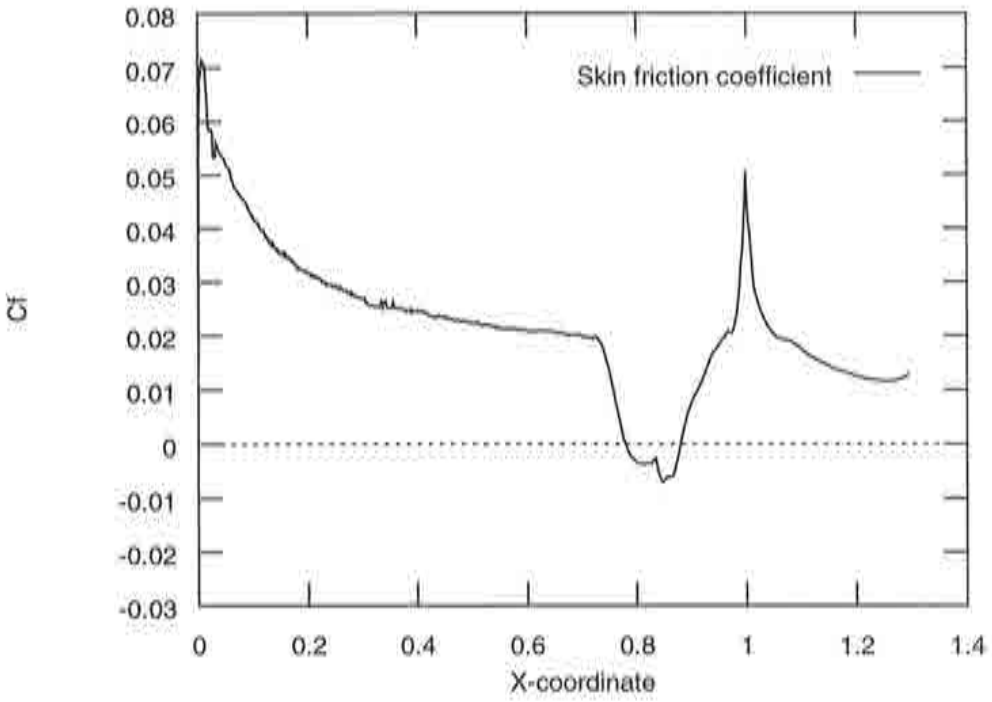


Figure 4.45 : Skin friction coefficient for test case H2K

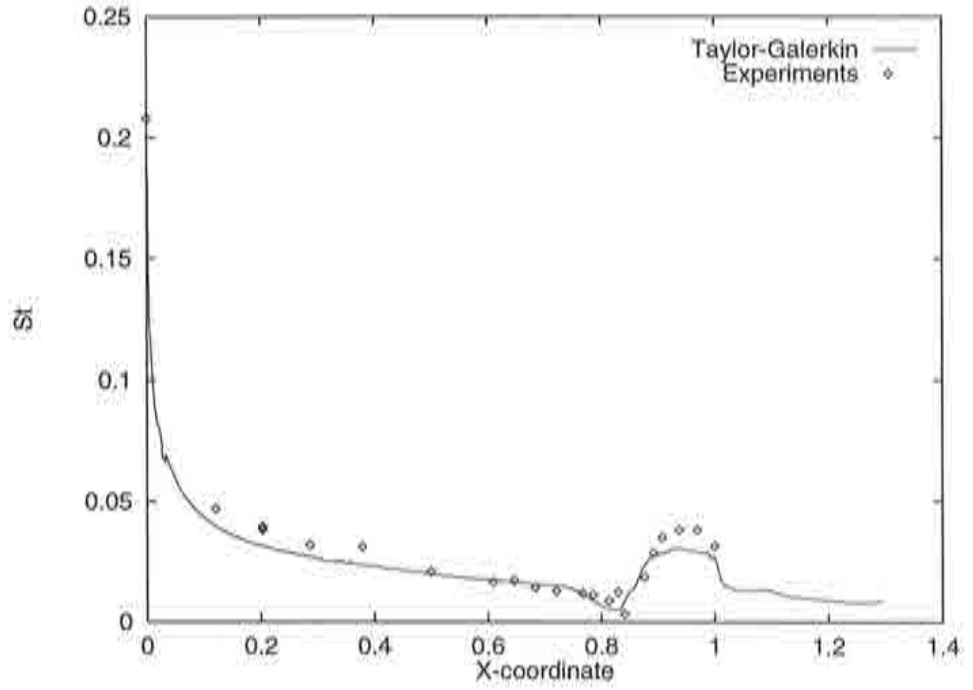


Figure 4.46 : Stanton number for test case H2K. The computations using the Taylor-Galerkin scheme is compared to experimental results obtained from the database.

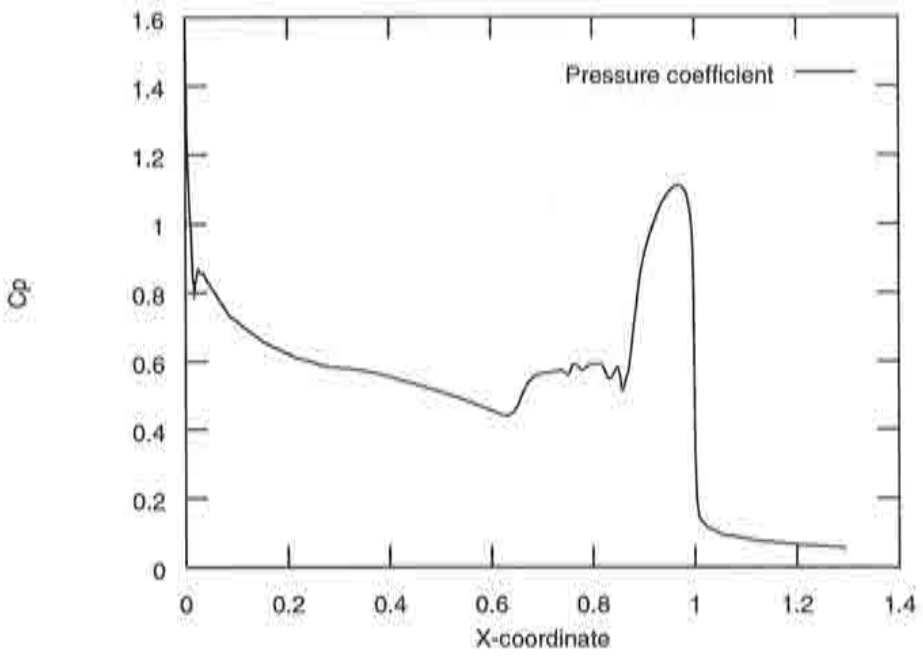


Figure 4.47 : Pressure coefficient for test case LTB

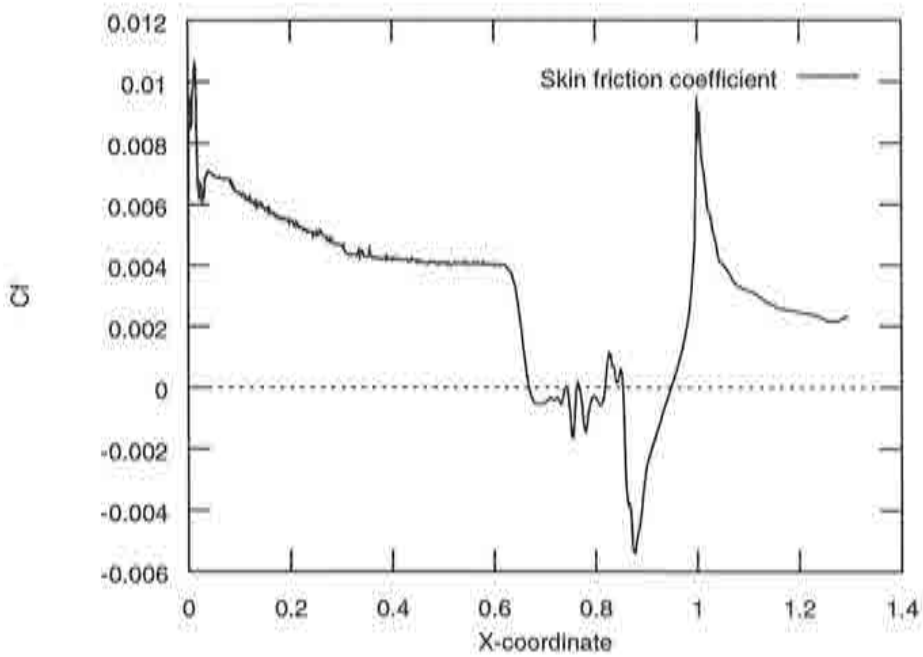


Figure 4.48 : Skin friction coefficient for test case LTB

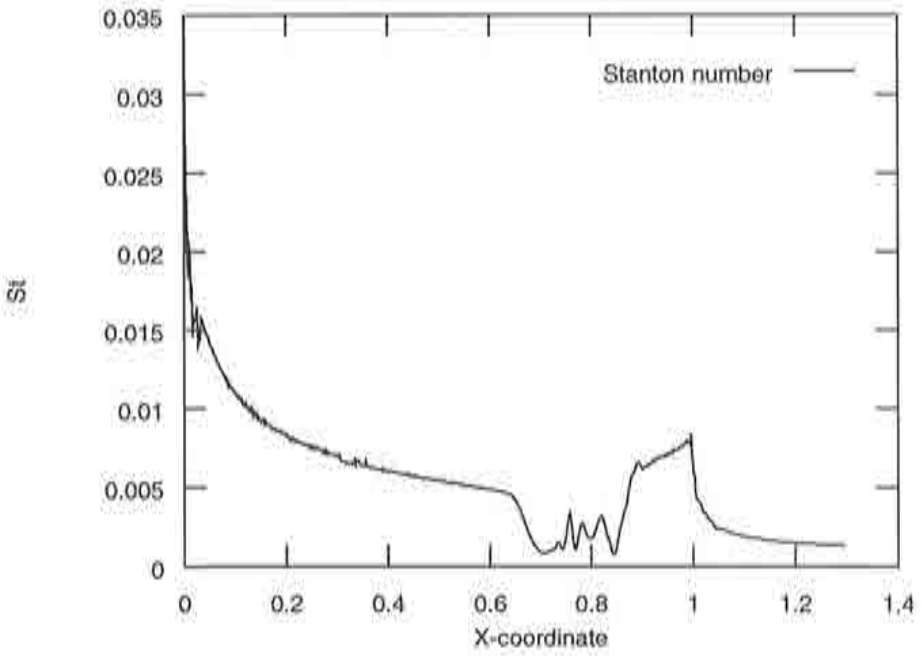


Figure 4.49 : Stanton number for test case LTB

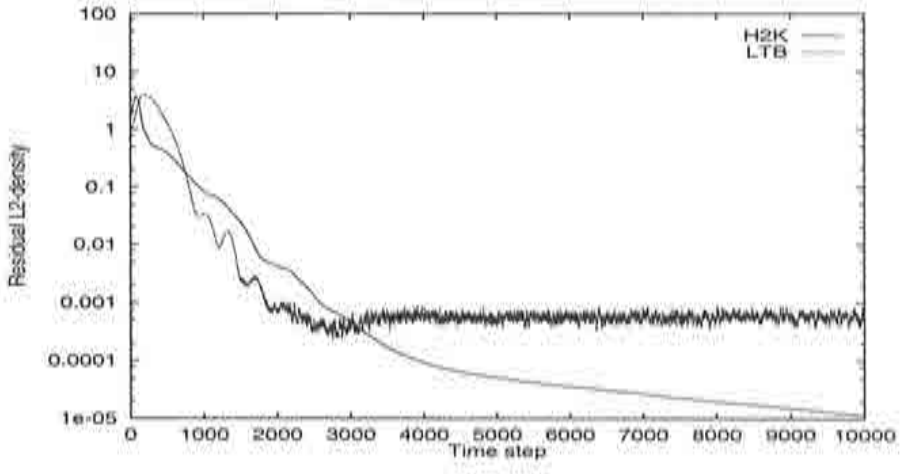


Figure 4.50 : Comparison of convergence histories of the L2-norm density residual for both test cases, H2K and LTB.

4.10.5 Accuracy

This section demonstrates the accuracy of the Taylor-Galerkin scheme for a subsonic test case by performing a mesh convergence study using four triangular meshes and four different artificial diffusion constants $\alpha^{(4)}$. The test case considered is again the subsonic flow around a NACA0012 with no incidence and $M_\infty = 0.5$.

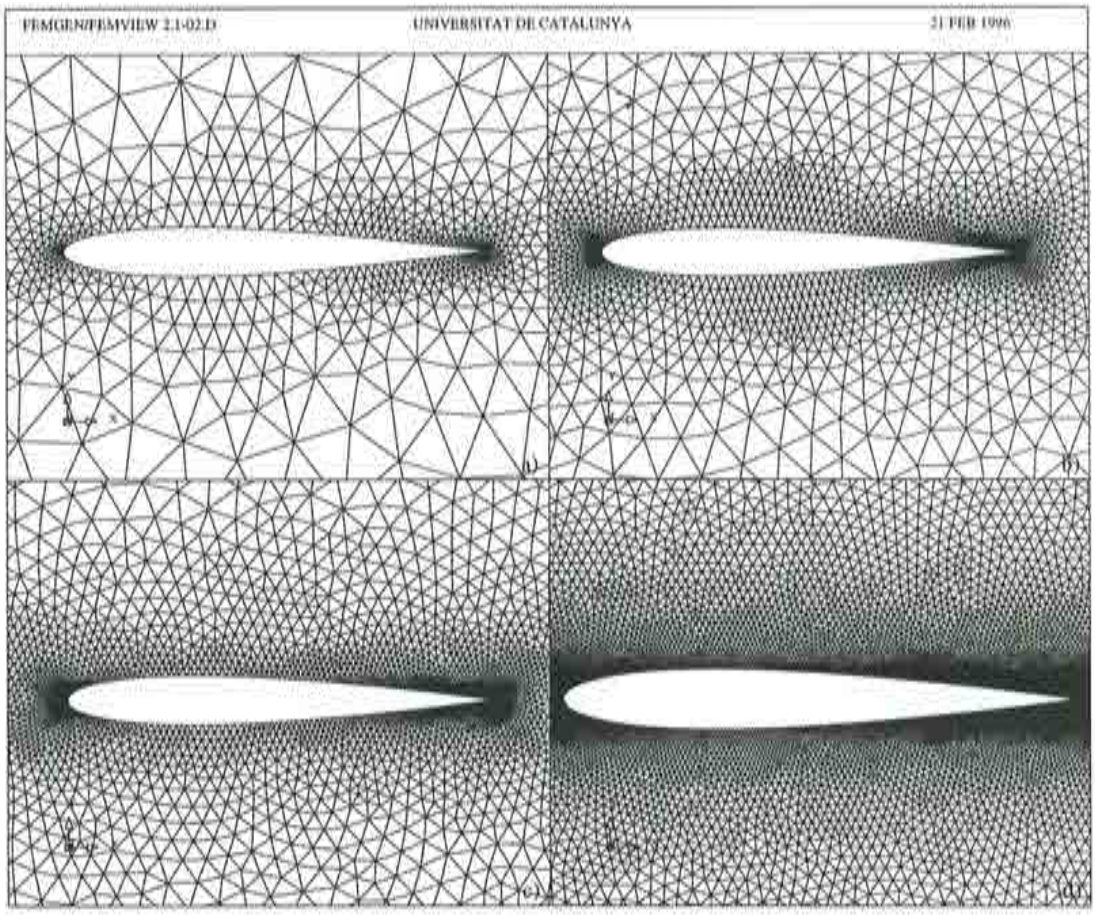


Figure 4.51 : Four triangular meshes for the mesh convergence study of a NACA0012 profile for inviscid test case $M_\infty = 0.5$, $\alpha = 0°$: a) 932 nodes and 1740 elements, b) 2556 nodes and 4902 elements, c) 4022 nodes and 7756 elements and d) 14836 nodes and 29108 elements.

The four meshes of unstructured triangles used in this study are shown in Figure 4.51. The density contours in the stagnation area for the four meshes are shown in Figure 4.52.

Numerical comparisons of the drag coefficient c_D and the stagnation density ρ_0 are shown in Table 4.1. Graphically, the stagnation density ρ_0 is presented in Figure 4.53 which is compared to the analytical value of 1.129726. The comparison of the convergence of the drag coefficient c_D is shown in Figure 4.53 and its analytic value is $c_D = 0.0$. The abscissa shows the inverse of the number of points per mesh. So, the result for an infinitely fine mesh can be extrapolated by extending each of the lines until they cross the ordinate. The vanishing influence of the artificial dissipation can

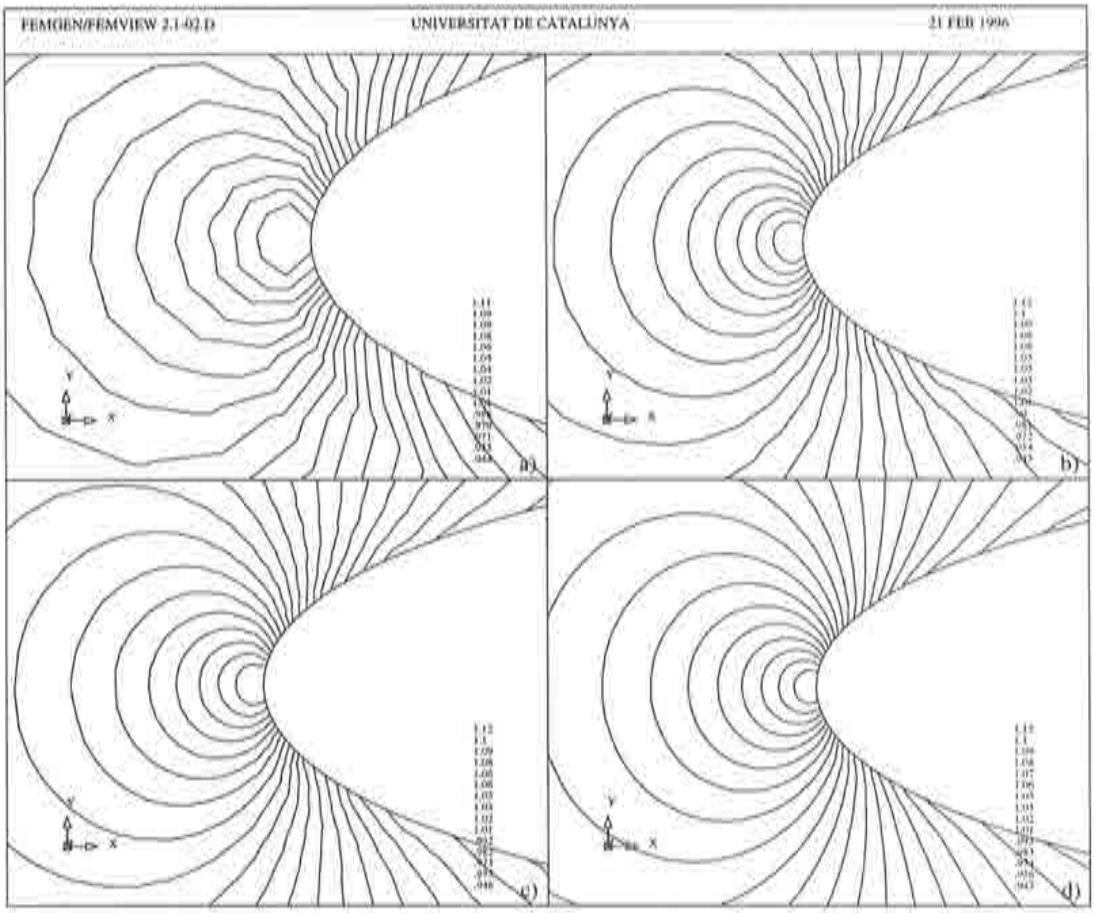


Figure 4.52 : Density contours in the stagnation area obtained from the respective triangular meshes in Figure 4.51 for the NACA0012 profile for inviscid test case $M_\infty = 0.5$, $\alpha = 0^\circ$.

also be observed as the meshsize goes to zero. This is demonstrated by the fact that the distance which separates the curves reduces.

The error in the stagnation density is shown in Figure 4.55 indicating quadratic convergence for as h decreases. The error in the stagnation density ρ_0 is defined by:

$$\text{Error}(\rho_0) = |\rho_0 - 1.129726| \tag{4.73}$$

$h = 1$ in Figure 4.55 is the spacing in the stagnation area of the coarsest mesh of the four. Quadratic convergence is seen from the fact that if h decreases by one half, the error diminishes to one quarter. A comparison of the drop in the L2-norm of the density residuals for $\alpha^{(4)} = 0.2$ using the different meshes is shown in Figure 4.56.

Another comparison of the same test case using triangles and quadrilaterals for the Runge-Kutta Galerkin scheme presented in chapter 6 is shown in section 6.4.2.

	$\kappa^{(4)} = 0.05$		$\kappa^{(4)} = 0.1$		$\kappa^{(4)} = 0.2$		$\kappa^{(4)} = 0.4$	
Npoin	c_D	ρ_0	c_D	ρ_0	c_D	ρ_0	c_D	ρ_0
932	0.00059	1.1111	0.00065	1.1131	0.00078	1.1137	0.00091	1.1140
2556	0.00028	1.1269	0.00029	1.1272	0.00034	1.1271	0.00045	1.1270
4022	0.00017	1.1273	0.00019	1.1274	0.00022	1.1277	0.00028	1.1279
14836	0.00007	1.1295	0.00008	1.1296	0.00010	1.1296	0.00013	1.1297

Table 4.1: Comparison of the drag coefficient c_D and the stagnation density ρ_0 for 4 different triangular meshes and 4 different diffusion constants $\alpha^{(4)}$. The analytic results are $c_D = 0.0$ and $\rho_0 = 1.1297$.

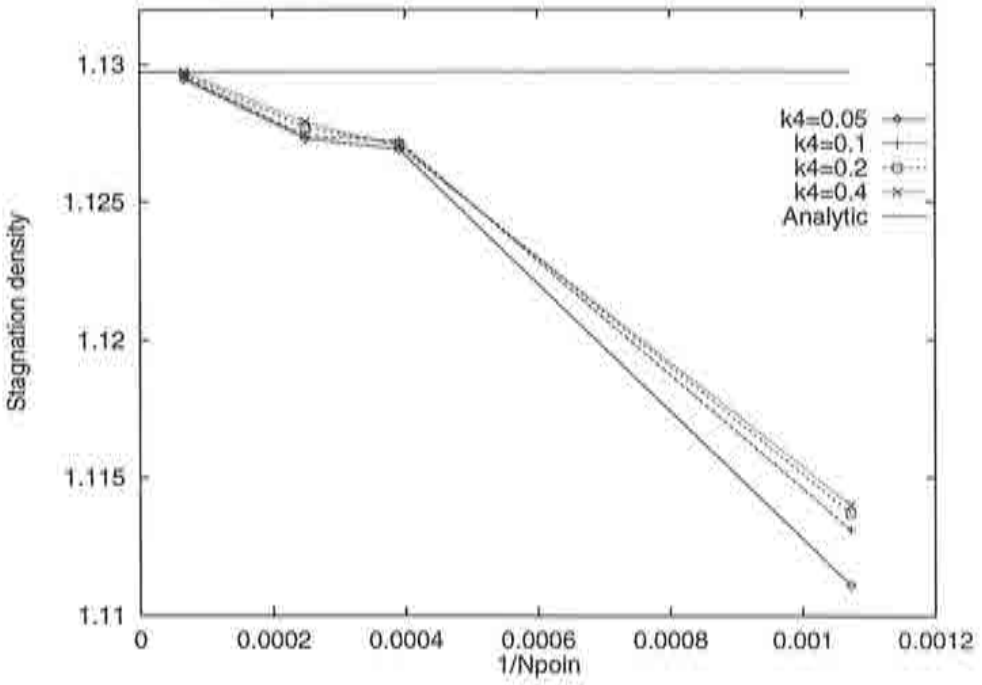


Figure 4.53 : Comparison of the stagnation density ρ_0 for four different meshes and three different artificial diffusion constants to analytical value of 1.129726. Plotted are the density values versus the the inverse of the number of points.

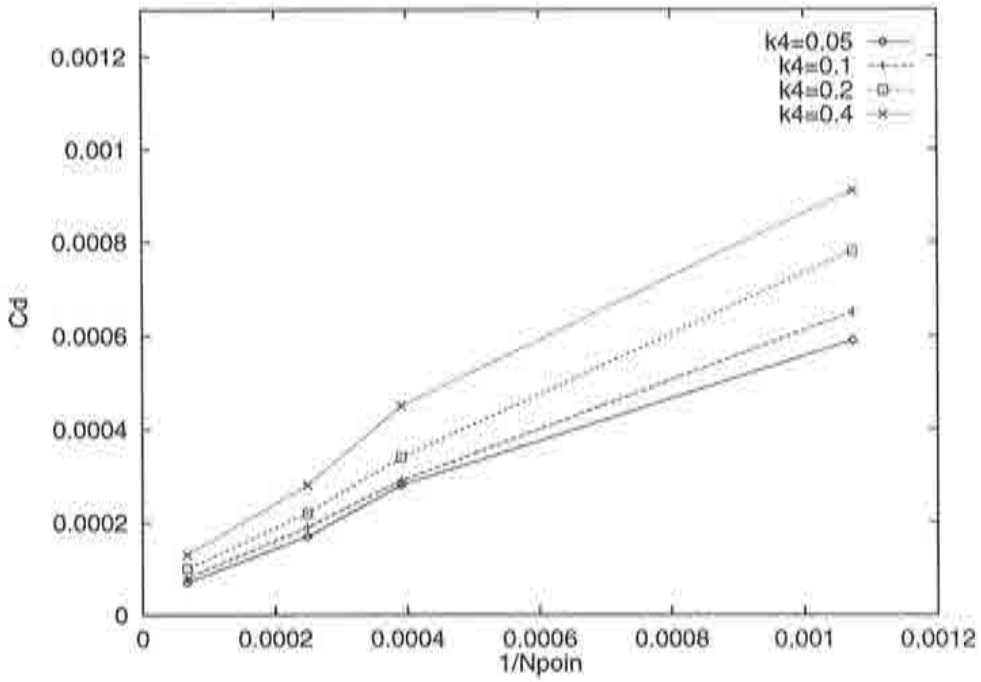


Figure 4.54 : Comparison of the drag coefficient for four different meshes and three different artificial diffusion constants to the analytical value of zero for inviscid flows. The plots show the c_D versus the inverse of the number of points.

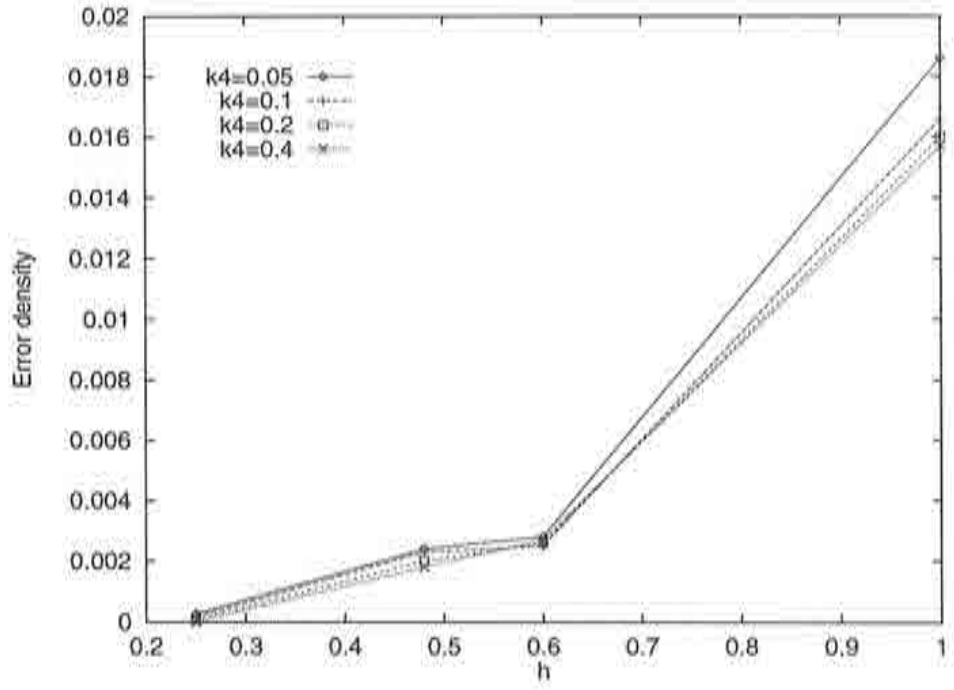


Figure 4.55 : Plot of the error of the stagnation density with respect to the mesh size h. The non dimensional value $h = 1$ is taken from the coarsest mesh.

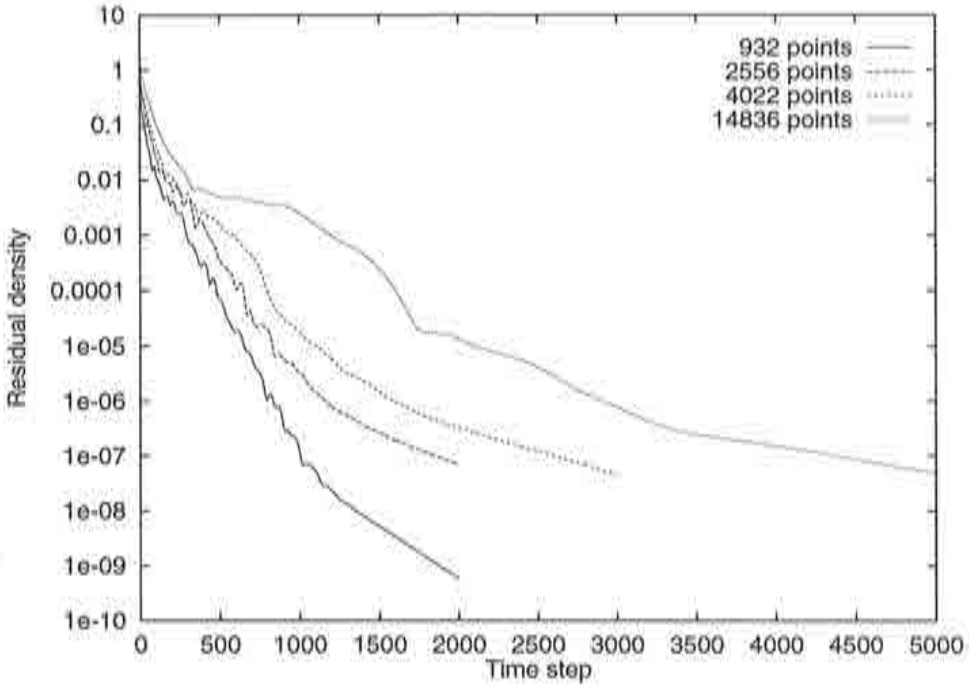


Figure 4.56 : Comparison of the convergence histories for the four different meshes using different diffusion constants $\alpha^{(4)} = 0.2$.

4.11 Conclusions

The classical Taylor-Galerkin algorithm for linear unstructured triangles and tetrahedra has been extended for bilinear quadrilaterals and a Jameson-type artificial diffusion [1]. The three dimensional scheme has been simplified using cylindrical coordinates to also cover axisymmetric flows. Therefore, a wide range of applications for fluid flow simulations is possible with this procedure.

Different types of numerical examples have been shown to validate the features of the new algorithms for subsonic, transonic and supersonic flow. Some results were compared to the analytical values whereas others were compared to experiments or other numerical solutions. Also the analysis of precision and the estimation of the order of accuracy for a certain test cases have been done by performing a mesh convergence study.

The Jameson type artificial dissipation routine used for finite elements has led to stable and more accurate solutions in regions where $|\mathbf{v}| \to 0$, such as stagnation areas. Also, the influence on accuracy of the artificial diffusion constants has been analyzed with respect to the test problem, mesh size and element type. Shocks were resolved within three elements and yields therefore superior results as the Lapidus type diffusion, especially at higher Mach numbers.

The use of quadrilateral elements has shown to add more precision with respect to the number of nodes used. However, the demands on the quadrilateral mesh generator are higher with respect to the quality of the generated elements because distorted quadrilaterals greatly degrade the quality of the solution whereas regular quadrilaterals lead to more accurate results than similar triangles.

Error estimation and adaptive unstructured mesh generation have improved the relation cost and accuracy, especially for flows with shocks.

References

[1] Jameson, A., Schmidt, W., and Turkel, E. "Numerical simulation of the Euler equations by finite volume methods using Runge-Kutta time stepping schemes" AIAA paper 81-1259. AIAA 5th Computational Fluid Dynamics Conf., 1981.

[2] Peraire, J., Peiro, J., Morgan K. "A 3D Finite Element Multigrid Solver for the Euler Equations", AIAA Paper 92-0449, 1992.

[3] Peraire, J. "A Finite Element Method for Convective Dominated Flows". PhD Thesis at University College of Swansea, 1986.

[4] Schade H., Kunz E. "Strömungslehre", Walter de Gruyter, Berlin, 1980.

[5] Quintana, J.F.A. "Análisis de Problemas de Flujo Compresible de Advección Dominante por el Método de Elementos Finitos", PhD Thesis at Universitat Politècnica de Catalunya, Spain, March 1993.

[6] Donea, J. "A Taylor-Galerkin Method for Convective Transport Problems", International Journal for Numerical Methods in Engineering, Vol 20, 101-119, 1984.

[7] Zienkiewicz O.C., Taylor "The Finite Element Method" 4th Edition, Volume 2,Mc Graw Hill, 1991.

[8] Fischer, T.,Codina R., Miquel, J. and Oñate E. "Adaptive Finite Element Computations for Viscous High Speed Flows", Proceeding of "Finite Elements in Fluids", Eds. K. Morgan, E. Oñate, J. Périaux, J. Peraire and O.C. Zienkiewicz, 1993.

[9] Oñate, E. "Cálculo de Estructuras por el Método de Elementos Finitos", CIMNE, Barcelona, January 1992.

[10] Lapidus A. 'A Detached Shock Calculation by Second Order Finite Differences'. J. of Computational Physics 2, 154-177, 1967.

[11] Jameson A., Baker T.J., Weatherill N.P. "Calculation of Inviscid Transonic Flow over a Complete Aircraft", AIAA Paper 86-0103, AIAA 24th Aerospace Sciences Meeting, Reno, 1986

[12] Löhner R., Morgan K. and Peraire J. "A Simple Extension to Multidimensional Problems of the Artificial Viscosity due to Lapidus", Communications in Applied Numerical Methods, Vol 1, pp 141-147, 1985.

[13] Mavripilis P., Jameson A., Martinelli L., "Multigrid Solution of the Navier-Stokes Equations on Triangular Meshes", ICASE Rep. No 89-11, February 1989.

[14] Lacor, C. Eliasson, P., Hirsch, C., Lindblad, I "Study of the Efficiency of a Parallelized Multigrid/Multiblock Navier-Stokes Solver on Different MIMD Platforms", 2nd European CFD Conference, Stuttgart, 5-8 September, 1994.

[15] Usab W. J., Murman E., "Embedded Mesh Solutions of the Euler Equations Using a Multiple Grid Method", Advances in Computational Transonics (W.G. Habashi, Ed.), Pineridge Press (UK) 447-472, 1985.

[16] Oñate, E. "Error Estimations and Adaptive Refinement Techniques for Structural and Fluid Flow Problems" in Mecánica Computational Vol. 11, S. Idelsohn, Asociación Argentina de Mecánica Computational, 1991.

[17] Hirsch, C. "Numerical Computation of Internal and External flows", Volume 1, Wiley, Chichester, 1989.

[18] Morgan K., Peraire J., Peiro, J., Hassan O., "The Computation of Three-Dimensional Flows Using Unstructured Grids", Computer Methods in Applied Mechanics and Engineering 87, 335-352, 1991.

[19] Zienkiewicz O.C., Morgan K.,Sai S., Codina R., Vázquez M. "A General Algorithm for Compressible and Incompressible Flow Part II, Tests on the Explicit Form", IJNME Vol 20, 887-919, 1995.

[20] "Workshop on Hypersonic Flows for Reentry Problems", Proceedings, Antibes, 22-25 January 1990.

[21] Abgrall R., Désidéri J.-A., Glowinski R., Mallet M., Périaux J. "Hypersonic Flows for Reentry Problems", Volume 3, Springer-Verlag, Berlin Heidelberg, 1992.

[22] Chalot F., Hughes T.J.R, Johan Z., Shakib F. "Application of the Galerkin/Least-Squares Formulation to the Analysis of Hypersonic Flows", Workshop on Hypersonic Flows for Reentry Problems, Antibes, 22-25 January 1990.

[23] Wang Z, and Richards B.E. *High Resolution Schemes for Steady Hypersonic Flow*, Proceeding of Workshop on Hypersonic Flows for Reentry Problems, Antibes, 22-25 January 1990.

[24] Hirsch, C. "Numerical Computation of Internal and External flows", Volume 2, Wiley, Chichester, 1989.

[25] Peiro, J. "A Finite Element Procedure for the Solution of the Euler Equations on Unstructured Meshes". PhD Thesis at University College of Swansea, 1989.

[26] Bugeda, G. "Utilización de técnicas de estimación de error y generación automática de mallas en procesos de optimización estructural". PhD Thesis at Universitat Politècnica de Catalunya, Spain, 1990.

[27] Löhner R. "An Adaptive Finite Element Scheme for Transient Problems in CFD", Computer Methods in Applied Mechanics and Engineering 61, 323-338, 1987.

[28] Brueckner F.P., "Finite Element Analysis of High Speed Flows with Application to the Ram Accelerator Concept", Phd Thesis, University of Arizona, 1991.

[29] Swanson R.C., Radespiel R., "Cell Centered and Cell Vertex Multigrid Schemes for the Navier-Stokes equations", AIAA Journal Vol. 29, No.5 , 1991.

[30] Kreiner R., "Generation of Unstructured Finite Element Meshes", PhD Thesis, Universität Stuttgart, 1995.

[31] Rank E., Schweingruber M., Sommer M., "Adaptive Mesh Generation and Transformation of Triangular to Quadrilateral Meshes", Comm. Num. Meth. Eng., 9, 121-129, 1993.

[32] Pulliam T.H., Barton J.T., "Euler Computations of AGARD Working Group 07 Airfoil Test Cases", AIAA 23rd Aerospace Sciences Meeting, Reno, Nevada, AIAA Paper 85-0018, 1985.

[33] Guillen Ph., Borrel M. "Contribution to the workshop on hypersonic flows for reentry problems", Workshop on Hypersonic Flows for Reentry Problems, Antibes, 22-25 January 1990

[34] Radespiel R., Herrmann U., Longo J.M.A. "Problem 6: Flow over a double ellipsoid", Workshop on Hypersonic Flows for Reentry Problems, Antibes, 22-25 January 1990

[35] Désidéri J.A. "Contributors to the EHVDB Workshop, ESTEC, Noordwijk, Nov. 24-25th, 1994", Letter requesting participation in the proceedings of the workshop, July 21st, 1995

Chapter 5

Finite Point Method in 2D

In computational mechanics, there currently is a trend towards using clouds of points in a multidimensional context, in which the advantages of not using a traditional mesh are more apparent than in one dimension. Many possibilities exist [1, 2, 3, 4] for performing this new type of spatial discretization and to come up with a new procedure. As mentioned earlier, this work applies these ideas in the context of 2D compressible flow.

The scheme derived for the one dimensional problem of section 3.2 is extended to the solution of the non linear two dimensional Euler and Navier-Stokes equations by using the concepts of the Taylor-Galerkin algorithm of section 4.2:

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}_k}{\partial x_k} = \frac{\partial \mathbf{g}_k}{\partial x_k} \quad k = 1, 2 \tag{5.1}$$

where the nodal unknowns and the advective fluxes are given by eq. 4.2. Subscript k instead of i is used to avoid confusion with the central point i of each local interpolation domain Ω_i which is defined further down.

5.1 Time Discretization

Consider the second order two-step scheme of section 4.2 in order to advance the solution in time, which avoids the calculation of the flux Jacobian. The first step is by replacing eq. 5.1 (neglecting viscous terms)

$$\mathbf{u}^{n+\frac{1}{2}} = \mathbf{u}^n - \frac{\Delta t}{2} \frac{\partial \mathbf{f}_k}{\partial x_k} \quad k = 1, 2 \tag{5.2}$$

and the second step becomes using eq. 5.1 and in analogy to section 4.2 neglecting third order derivatives or higher:

$$\mathbf{u}^{n+1} = \mathbf{u}^n - \Delta t \left(\frac{\partial \mathbf{f}_k^{n+\frac{1}{2}}}{\partial x_k} - \frac{\partial \mathbf{g}_k^n}{\partial x_k} \right) \simeq \mathbf{u}^n - \Delta t \frac{\partial}{\partial x_k} \left(\mathbf{f}_k(\mathbf{u}^{n+\frac{1}{2}}) - \mathbf{g}_k^n \right) \quad k = 1, 2 \tag{5.3}$$

5.2 Space Discretization

The concepts of section 3.2 are extended to two dimensions and the differences are outlined in what follows.

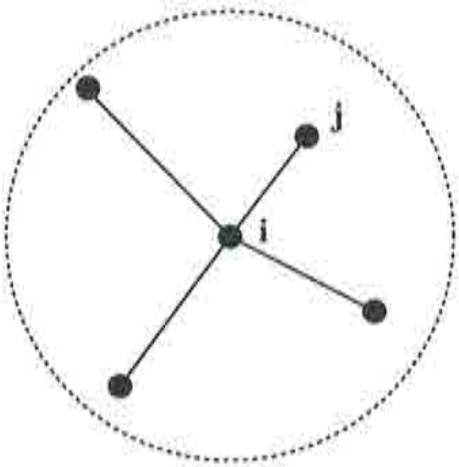


Figure 5.1 : Local interpolation domain Ω_i, indicated by the dashed curve, for 4 nodes j plus the central node i.

Let us describe a local interpolating domain Ω_i, also called "cloud of points", by a set of $j = 1, n$ points including the central node $i = 1$ as seen in Figure 5.1. With the help of these points a sought function, now of x and y, $u(x, y)$ can be approximated by the function $\hat{u}(x, y)$

$$u(x, y) \simeq \hat{u}(x, y) \tag{5.4}$$

Performing a polynomial expansion of order 2 like eq. 3.37, the function $\hat{u}(x, y)$ can be described within Ω_i, ie. for quadratic polynomials ($m = 6$)as

$$\hat{u}(x, y) = a_1 + a_2 x + a_3 y + a_4 x^2 + a_5 y^2 + a_6 xy \tag{5.5}$$

or in vectorial form

$$\hat{u}(x) = \mathbf{p}^T(x)\mathbf{a} \tag{5.6}$$

where the polynomial coefficients a_m have been cast into the vector $\mathbf{a}$ and the base functions $\mathbf{p}$ for the linear and quadratic case are, respectively:

$$\mathbf{p}^T = [1, x, y] \quad \text{for } m = 3 \tag{5.7}$$

$$\mathbf{p}^T = [1, x, y, x^2, y^2, xy] \quad \text{for } m = 6 \tag{5.8}$$

in two dimensions [5].

The process for obtaining the polynomials follows that of section 3.2, so that we can write directly equation 3.52:

$$\mathbf{a} = \mathbf{A}^{-1}\mathbf{B}\mathbf{u} \tag{5.9}$$

where $\mathbf{A}$ and $\mathbf{B}$ are now composed of the two dimensional base functions $\mathbf{p}$ and are given by eqs. 3.53 and 3.54. Eq. 5.9 is resolved in the same fashion as in section 3.2, optionally with the incorporation of weighting functions. Then, $\mathbf{A}$ and $\mathbf{B}$ are given by eqs. 3.61 and 3.62.

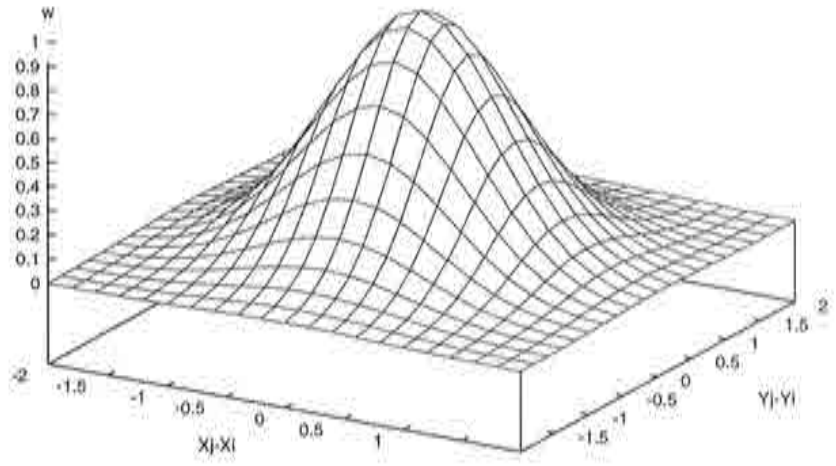


Figure 5.2 : This figure shows the two dimensional cone for Gauss weighting functions for $\lambda_i c = 1$ and $p = 2$ and the central node $(x_i, y_i) = (0, 0)$. Note, as the distance from the central node increases, the weight decreases rapidly.

5.2.1 Weighting functions

The weighting functions are again a function of the distance, and a possible choice is the Gauss function of eq. 3.64:

$$w(r_j) = e^{-\left(\frac{r_j}{\lambda_i c}\right)^p} \tag{5.10}$$

where λ_i is a characteristic length of the local interpolating domain Ω_i defined below, p is a constant chosen equal to 2 and r_j is defined in two dimensions as the absolute distance between the nodes j and the central node i:

$$r_j = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} \tag{5.11}$$

Graphically, the Gauss distribution in two dimensions can be seen in Figure 5.2.

5.2.2 Stability

The extension of the stability properties of the Taylor-Galerkin scheme presented in previous sections, to this explicit two step scheme using clouds of points is done in a heuristic way. The result is therefore a conditionally stable explicit second order algorithm with the following limit in inviscid flow for Δt_I which is obtained from eq. 4.34:

$$\Delta t_I = \frac{c_s h}{|\mathbf{v}| + c} \quad \text{and} \quad c_s \leq 1 \tag{5.12}$$

The condition including the viscous fluxes for Δt is given by eq. 4.35.

A difficulty in a multidimensional context arises from the determination of h in a given cloud of points Ω_i. In finite elements, using linear triangular elements, h is defined according to the minimum length within each element [6]. In meshless methods, a clear definition has not been presented yet. In this work, a prudent choice for h has been taken equal to the minimum distance of the points j to the center point i within each interpolation domain Ω_i:

$$h = \lambda_i = \min(r_j) \quad j = 2, n \tag{5.13}$$

5.3 Balancing Dissipation

Since the hyperbolic Euler equations do not contain any diffusion terms, some balancing damping must be added to prevent unphysical oscillations. Following Jameson [7], 2nd and 4th order diffusion terms are added to the fluxes as described in chapter 4. The formulation is adopted directly for isotropic diffusion from [12] by multiplication of weighting functions w_j with the dissipation flux. The dissipation flux $\mathbf{f}_d$ for each node i which is added to eq. 4.18 of section 4.3.2 is then constructed as

$$\mathbf{f}_d = -\frac{(|\mathbf{v}| + c)}{r_j} \sum_{j=2}^{n} (\mathbf{e}_j - \mathbf{e}_i) \frac{w_j}{\sum_{k=2}^{n} w_k} \tag{5.14}$$

where

$$\mathbf{e}_i = \epsilon_i^{(2)} \mathbf{u}_i - \epsilon_i^{(4)} \sum_{j=2}^{n} (\mathbf{u}_j - \mathbf{u}_i) \tag{5.15}$$

where $\mathbf{u}$ are the nodal unknowns, $(|\mathbf{v}| + c)$ is the maximum eigenvalue of $\frac{\partial \mathbf{f}}{\partial \mathbf{u}}$, w_j are the same weighting functions used in the interpolations of eq. 5.10 and the coefficients ϵ_i of eq. 5.15 are obtained as:

$$\epsilon_i^{(2)} = \alpha^{(2)} \frac{\sum_{j=2}^{n} |p_j - p_i|}{\sum_{j=2}^{n} p_j + p_i} \tag{5.16}$$

where p_j denotes the nodal pressures and

$$\epsilon_i^{(4)} = \max(0, \alpha^{(4)} - \epsilon_i^{(2)}) \tag{5.17}$$

$\alpha^{(2)}$ and $\alpha^{(4)}$ are user defined constants, chosen similar to section 4.3.2. The summation j extends across the number of points in each cloud and is accumulated at both the central point i and the point j. In subsonic flows $\epsilon_i^{(2)}$ is generally switched off.

5.4 Selection of Points

In a multidimensional domain, the difficulty arises on how to define each local interpolating domain. In section 3.2, a necessary condition for the selection of the minimum number of nodes has been stated in eq. 3.43: $n \geq m$. However, this is not sufficient in a multidimensional context as can be seen from a simple example. Let's assume that we are using linear base functions $m = 3$ in two dimensions and we choose three nodes $n = 3$ which lie in a straight line to comply with condition of eq. 3.43. However, if we assemble matrix $\mathbf{A}$, we find that it is singular and it can not be inverted.

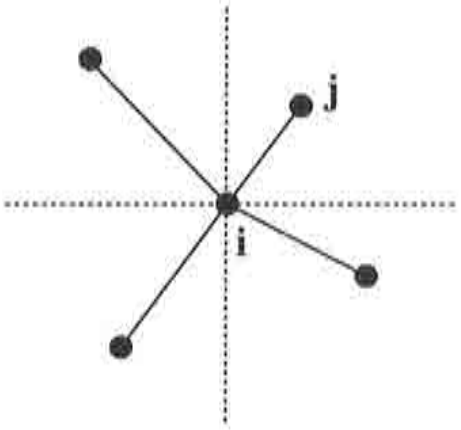


Figure 5.3 : Selection of points using a criteria of quadrants. At least the closest point in each quadrant is chosen.

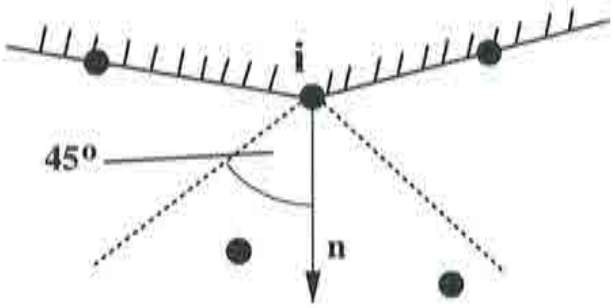


Figure 5.4 : Selection of points along the boundary using a cone of around the point normal.

In order to avoid $\mathbf{A}$ being singular, several conditions upon the selection of points

have been imposed and even though weighting functions are employed, it is still necessary to choose the most significant points for each Ω_i. Therefore, the following conditions are imposed for the proper selection of points for the results of this thesis:

- The central point plus the closest points are chosen.
- A condition of quadrants is imposed such that there must be at least one point in every quadrant of orthogonal axes (Figure 5.3). Hence, the closest point in each quadrant is selected. This leads to a minimum of 5 points per cloud.
- At the boundary, the two points adjacent to the central point on the boundary plus at least the 2 closest closest points within a cone along the boundary point normal are chosen (Figure 5.4).
- Another condition is that no boundary section is crossed so that points from the opposite side are not chosen. For instance, at the trailing edge of an airfoil, the closest points to a point on one side of the airfoil may lie across the wall on the other side (Figure 5.5). Since there is a physical separation of these points, they are not included in the same interpolation domain.

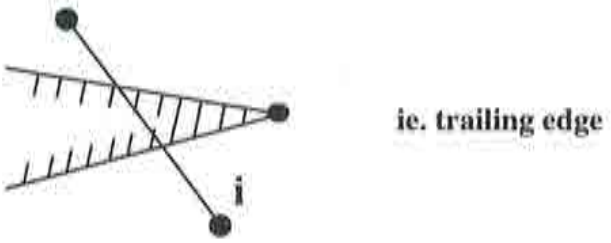


Figure 5.5 : Point selection across physical separations such as a wall boundary is inhibited.

Remark

1. The proposed criteria for the selection of points is by no means optimal, and can be optimized for improved accuracy and performance. But these conditions work well in practice; they are easy to implement and they are inexpensive. For the scope of this thesis 11 noded clouds of points (the central point, plus the 4 of the quadrants, plus the rest of the 6 closest points) are stored in a data file. This relieves the need to recalculate the definitions of the local interpolation domains every time a solution run is performed.
2. In addition the above way for selecting points avoids that $\mathbf{A}$ is singular in the case of linear basis functions. For the use of quadratic basis functions, it is not rigorous and more than the minimum set of points ($n = m = 6$) should be chosen. In practice, it was found that 9 or 10 points for use with quadratic basis functions was sufficient. A more rigorous way to select nodes for quadratic basis functions would be the use of octants, however this was not tested in this work.

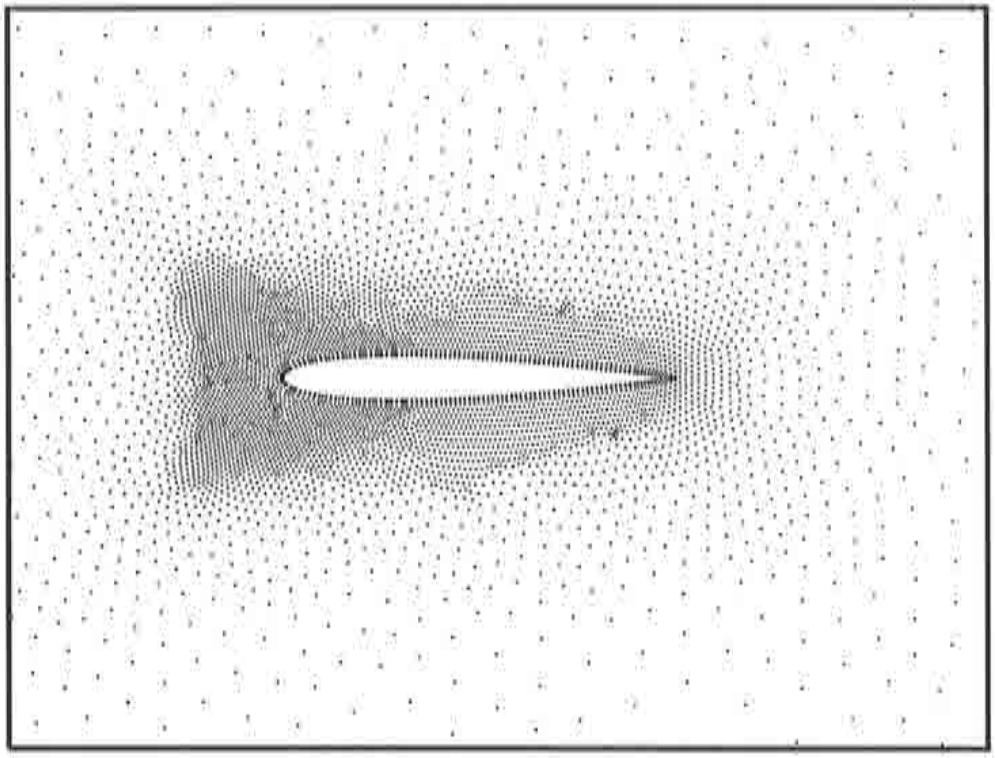


Figure 5.6 : Initial point distribution around a NACA0012 profile with 6694 points.

5.5 Numerical Examples

The flow around the NACA0012 airfoil is again used as a test case to validate two-dimensional flow solutions, because extensive analytical and numerical results are known. The following test cases compare the solution obtained by the FPM to some reference numerical solutions with the focus on accuracy.

5.5.1 Subsonic Inviscid Flow

The first test case which is presented here is an example of subsonic nearly incompressible flow of $M_\infty = 0.3$ and at an angle of 10 degrees. The fact of the flow being incompressible and inviscid is beneficial for comparison with a solution approximated by a potential solver.

In Figure 5.6, the grid of 6694 points around the NACA0012 airfoil is illustrated. The points have been taken from a previous generated mesh of finite elements for convenience of comparison. The aim of this test case was to show the influence of weighting functions upon the solution of an unstructured grid of points. In this test case, the Gauss weighting functions have only been applied to the interpolation functions for the evaluation of the polynomial coefficients $\mathbf{a}$ for clarity. The diffusion terms have not been submitted to the use of weighting functions. The number of nodes per cloud was taken $n = 5$ and linear basis functions were employed ($m = 3$).

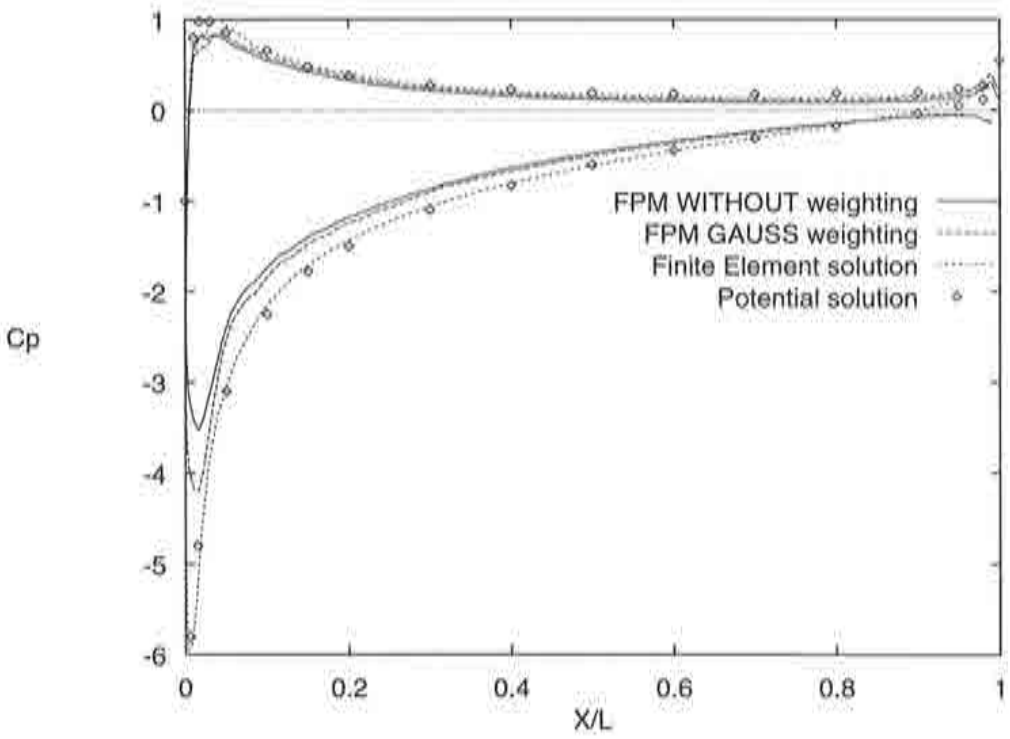


Figure 5.7 : NACA0012: Comparison of the pressure coefficient around the airfoil of the FPM ($n = 5, m = 3$) with a reference potential solution and a finite element solution. Note the improvement for weighting functions.

Figure 5.7 shows a comparison of the solution of constant (or no) weighting and the influence of using Gauss weighting functions with a reference solution obtained from a potential flow solver and a finite element solution using Runge Kutta time integration. The finite element mesh consisted of the same mesh points as the finite point mesh, so a direct comparison between the solutions is possible. Clearly the improvement of weighting functions is appreciated, however the solution is not yet of excellent quality as can be seen from the wiggles of the c_p in the stagnation area and near the trailing edge. A comparison of the convergence properties of the FPM versus the Taylor-Galerkin two-step scheme is shown in Figure 5.8. Clearly, both solutions of the FPM converge much slower than the corresponding finite element solution. The residuals drop by approximately 5 orders of magnitude after about 15000 iterations.

Part of the low quality of the result is due to the low number of points on the boundary and their uneven distribution. Performing adaptive refinement using the previous solution, a new grid is obtained with even less nodes (5463) as shown in Figure 5.9.

Figure 5.10 compares the density contours obtained for both meshes and for employing weighting functions or not. Note the global improvement for each mesh, respectively, after the use of Gauss weighting functions. Oscillations have reduced strongly. The computed pressure coefficient along the surface of the airfoil can now be seen from

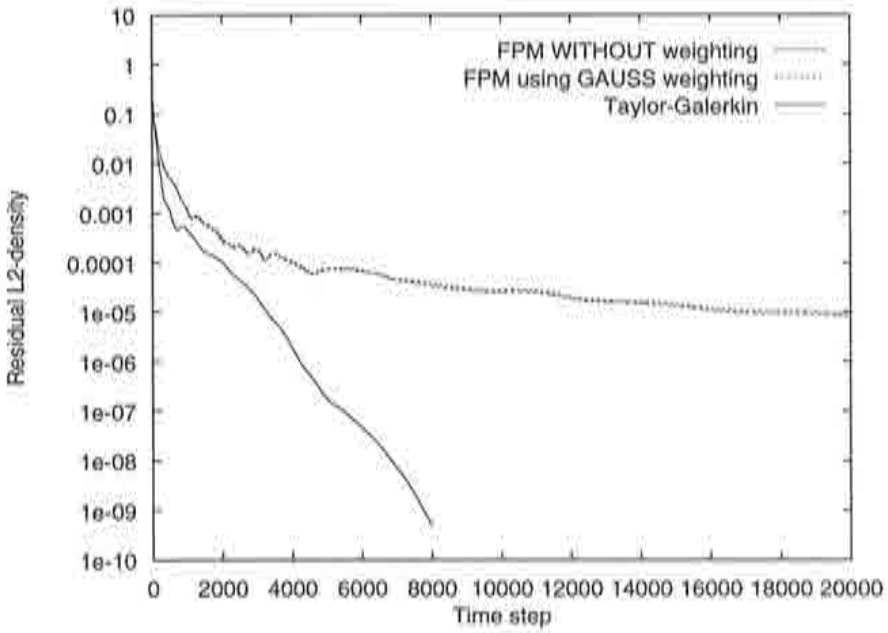


Figure 5.8 : NACA0012: Comparison of the convergence history of the FPM ($n = 5, m = 3$) without weighting and using Gauss weighting functions with the reference Taylor-Galerkin solution.

Figure 5.11. The improvement is evident, but again it seems that the use Gauss weighting functions has contributed toward an improved resolution of the test case. Both the stagnation area as well as the flow around the trailing edge have improved.

Quadratic basis functions

The incorporation of quadratic basis functions $m = 6$ has also a favorable effect upon the accuracy. Again the use of weighting functions was compared. For the density values, similar graphs to Figure 5.10 are obtained. Much better conclusions are drawn from the comparison of the pressure coefficient presented in Figure 5.12. A more drastic improvement is achieved for the use of Gauss weighting functions, as $n = 11$ points per clouds are introduced.

Convergence

A plot of the convergence history for the use of weighting functions and without weighting functions can be observed from Figure 5.13. Note that the use of weighting functions slightly increases the rate of convergence. For clarity, the convergence for quadratic basis functions $m = 6$ is omitted, however it is similar to the linear case $m = 3$.

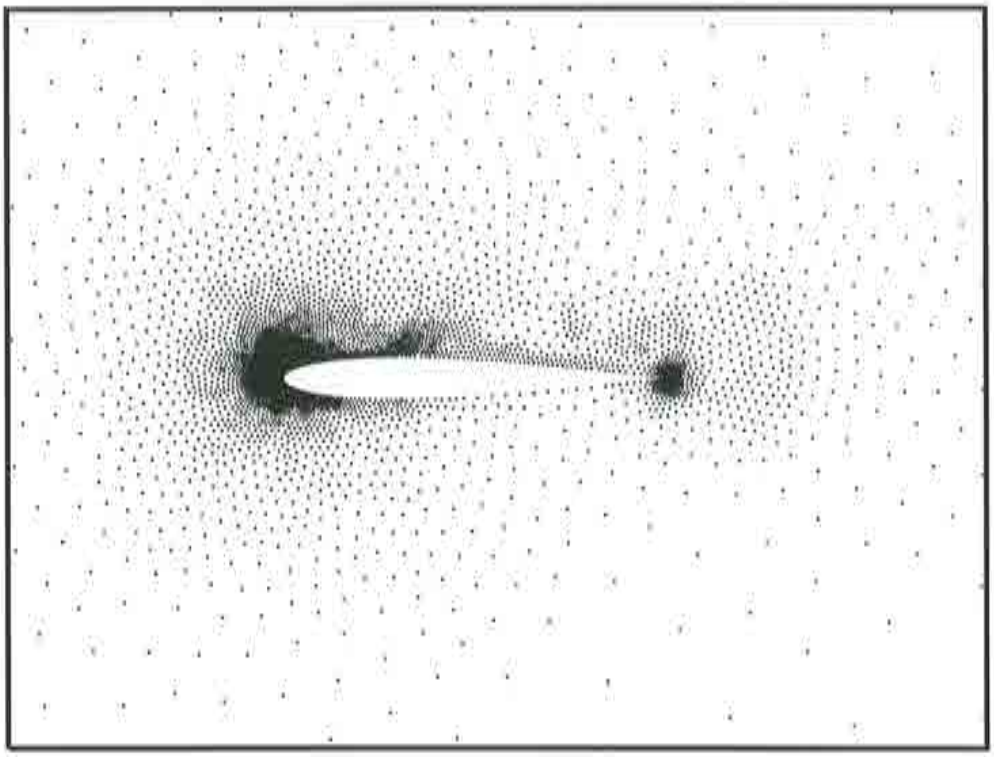


Figure 5.9 : Point distribution around a NACA0012 profile with 5436 points after adaptive remeshing. Note the clustering of points at the leading and trailing edge.

Gauss Weighted Diffusion

So far, the incorporation of Gauss weighting functions has been limited to the interpolation values **a**. Let us test the use of Gauss weighting functions also for the diffusion terms just like in eq. 5.14. The first unrefined mesh is again subjected to same initial conditions of the test case. Figure 5.14 illustrates the improvement of the results especially in the stagnation are when compared to the potential solution and the finite element solution.

Sensitivity to the number of points n

The behavior of the algorithm with respect to the number of point per cloud has a similar effect than for the one dimensional case, so it is not expressly repeated here. The results for this test case of a flow of $\alpha = 10°$ has been resolved with 5 noded clouds and has been increased up to $n = 11$, resulting in a strong deterioration of the results if no weighting functions are employed at all. However, if weighting functions are used, the results are quite insensitive to the number of the nodes. As more nodes are added less weight are given to them because they are generally farther away, so the global quality of the result is retained. In order to show the use of more nodes per cloud, the next test case is resolved with $n = 7$.

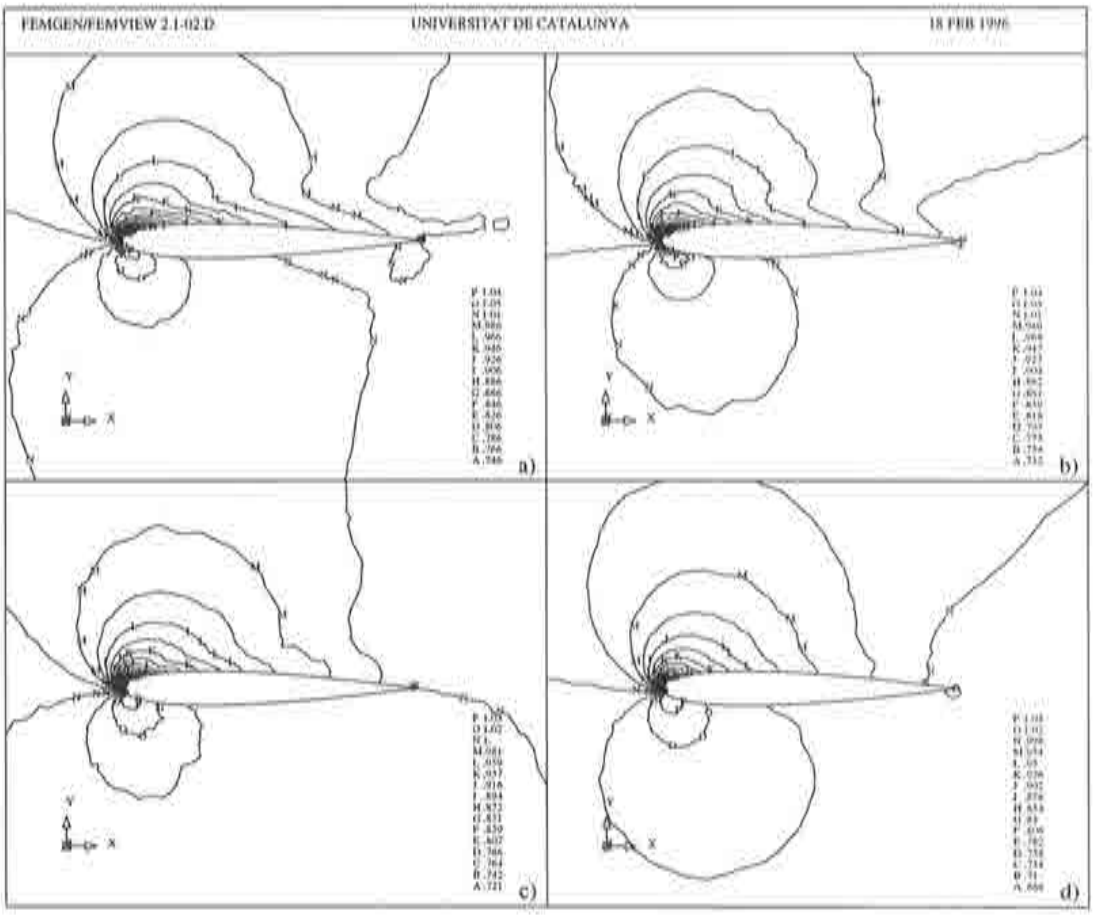


Figure 5.10 : NACA0012: Comparison of density contours ($n = 5, m = 3$): a) FPM without weighting functions for the first mesh, b) FPM using Gauss weighting functions for the first mesh, c) FPM without weighting functions for the adapted mesh, d) FPM using Gauss weighting functions for the adapted mesh.

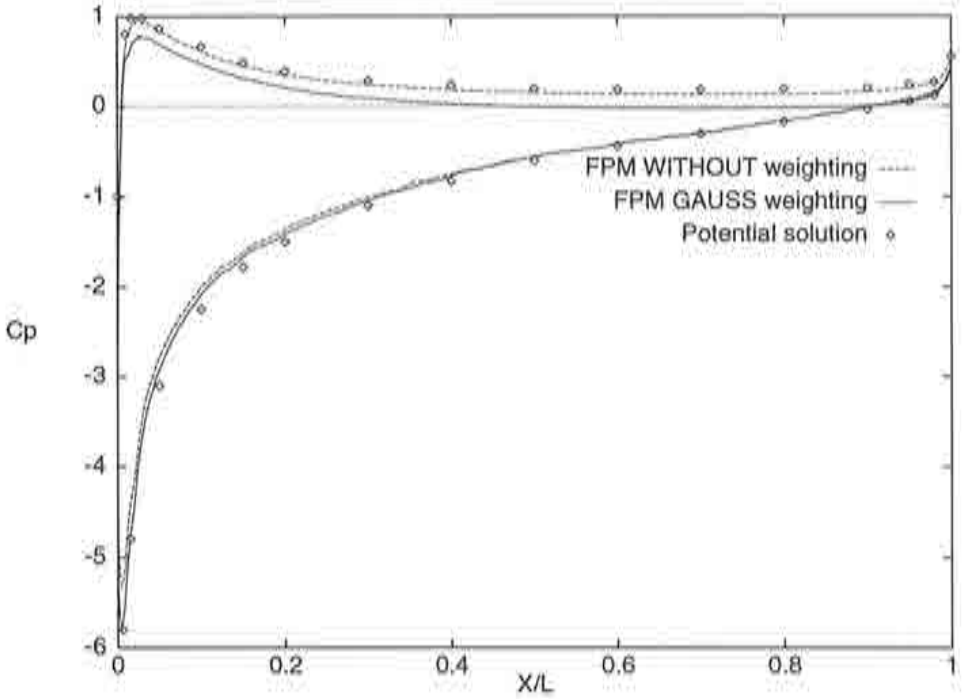


Figure 5.11 : NACA0012: Comparison of the pressure coefficient around the airfoil of the FPM ($n = 5$) using linear basis functions ($m = 3$) with a reference potential solution. Note again the superior performance when using weighting functions.

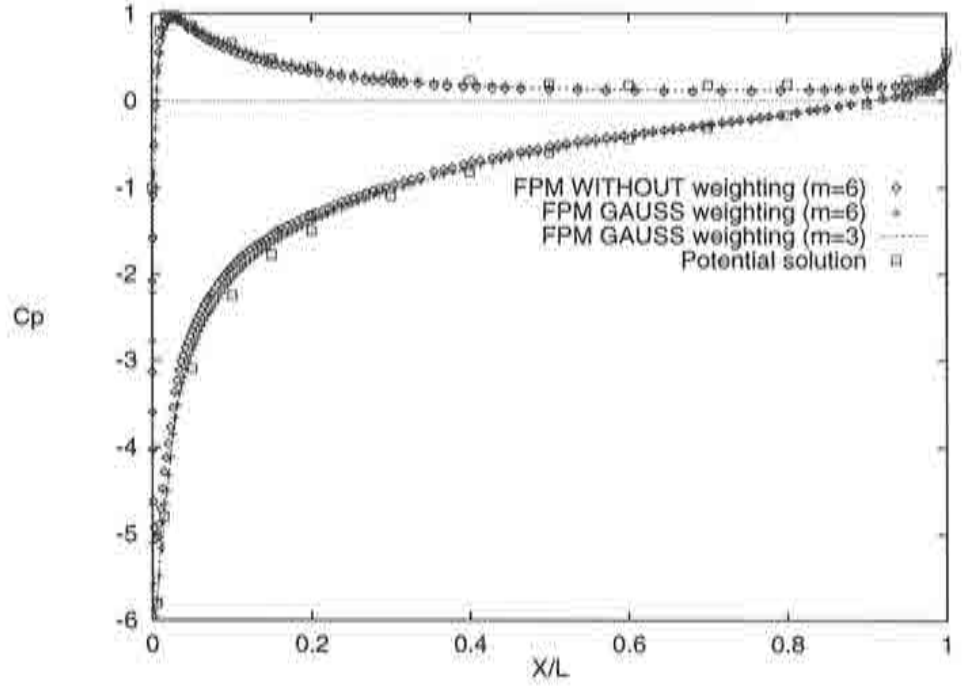


Figure 5.12 : NACA0012: Comparison of the pressure coefficient around the airfoil of the FPM using quadratic basis functions ($n = 5, m = 6$) with the linear solution ($n = 5, m = 3$) and a reference potential solution.

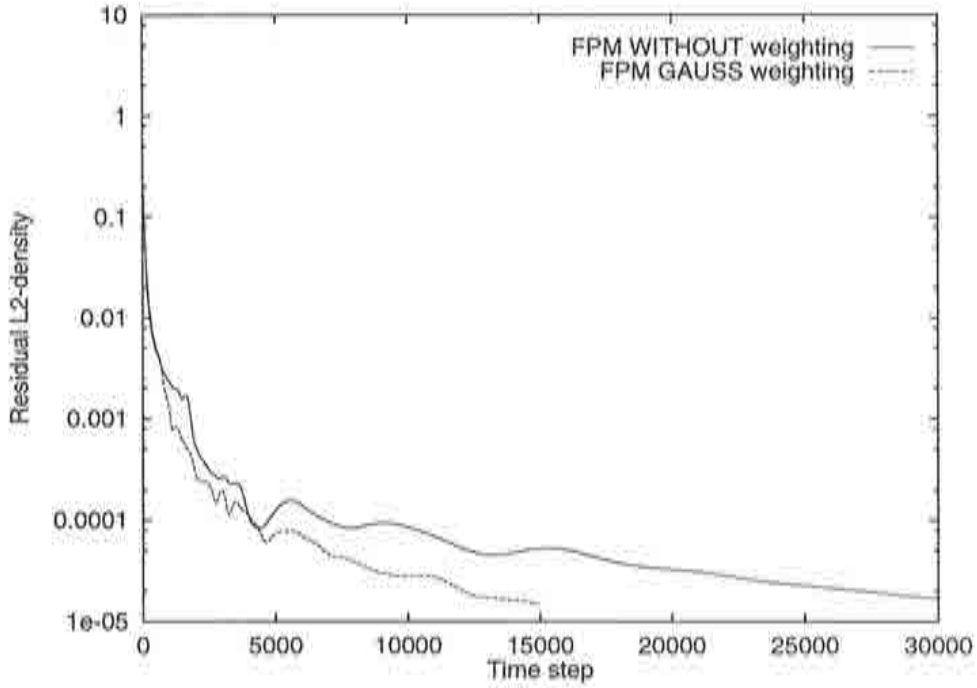


Figure 5.13 : NACA0012: Comparison of convergence histories of the FPM ($n = 5, m = 3$) without weighting functions and using Gauss weighting functions for the adapted mesh.

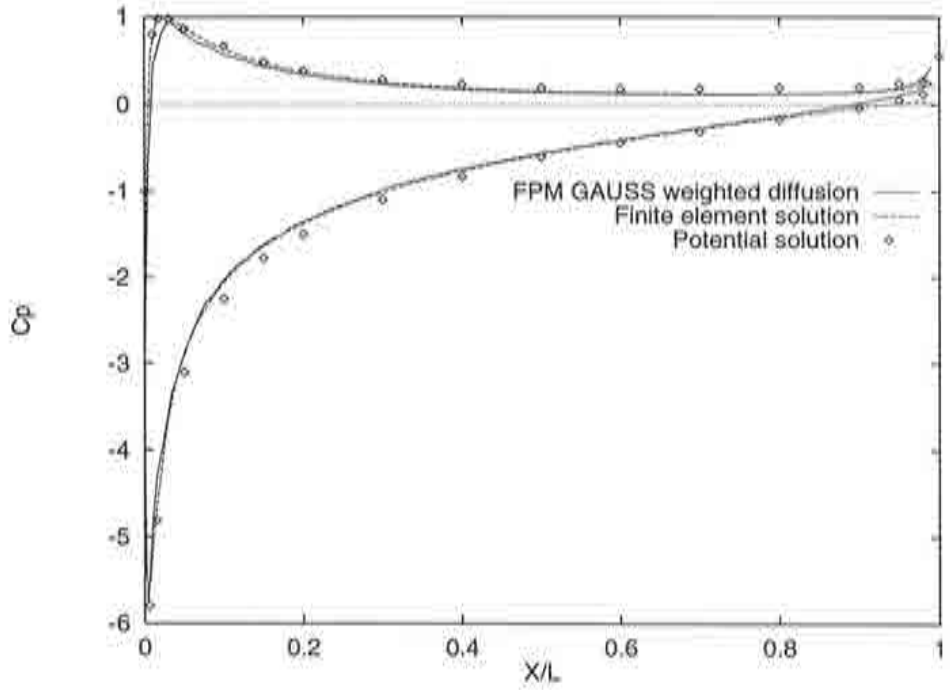


Figure 5.14 : NACA0012: This plot illustrates the influence of weighting functions also for the diffusion terms using $n = 5, m = 3$. The result is again compared with a reference potential solution and a finite element solution. Note the improvement in the stagnation area compared to the results in the previous figures

5.5.2 Subsonic Inviscid Compressible Flow

From the above test case we have seen that it is important to use weighting functions for both the fluxes and the balancing diffusion terms. Let us analyze this behavior more closely on another classic test case. It is known that a numerical scheme can not be justified accurate if it is not able to reproduce solutions on coarse grids with an appropriate precision. The objective is to show that sufficient accuracy can be obtained also on a coarse mesh using weighted balancing diffusion (WBD) terms.

The following test case uses a mesh with the same points for both finite point and finite element solution and compares the use of Gauss weighting functions also for the diffusion terms. The test case consists of a subsonic compressible flow around a NACA0012 profile with a free stream Mach number of 0.5 and 0 degrees angle of attack, analyzed by [11]. Again, in order to compare solutions, a finite element mesh and its solution has been taken for comparison. The meshless grid of 2556 points is shown in Figure 5.15. The same points have been used for the FE solution on the equivalent unstructured triangular mesh obtained using a standard advancing front technique [8, 9].

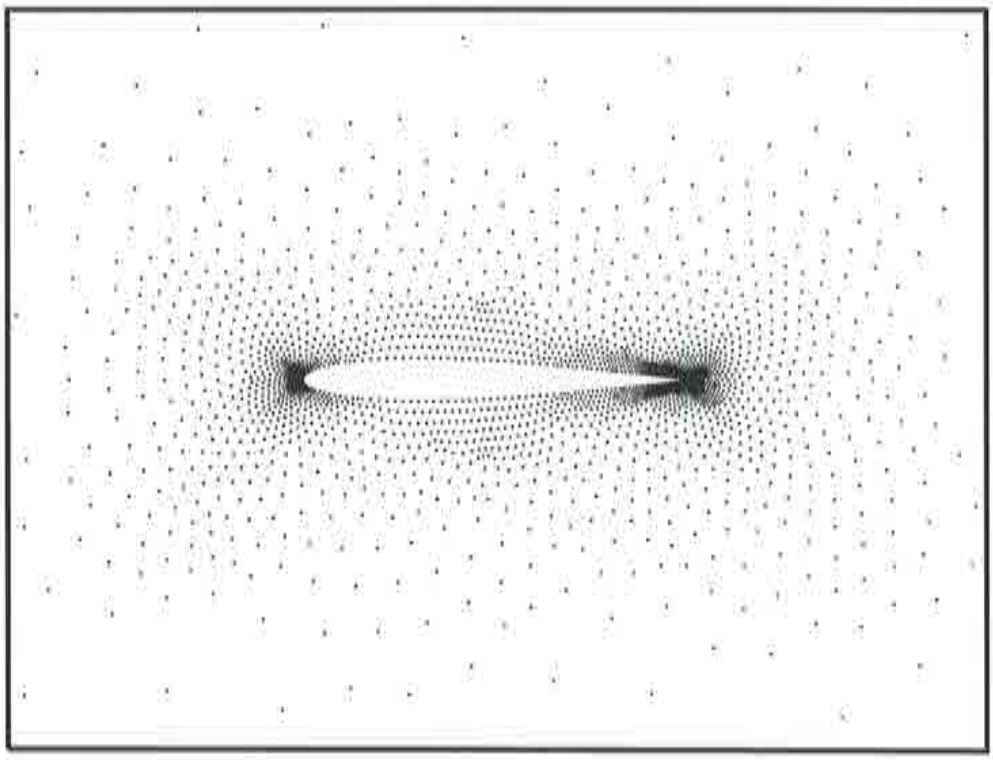


Figure 5.15 : Point distribution around a NACA0012 profile using only 2556 points

The objective is to compare the influence of the weighting functions in the finite point approximation. In the previous section, results have been shown proving the superiority of weighting functions in a 2D context, but without using weighting functions

for the balancing diffusion terms (except for the final figure). Here, the benefit of the weighted diffusion terms shall be presented, see eq. 5.14.

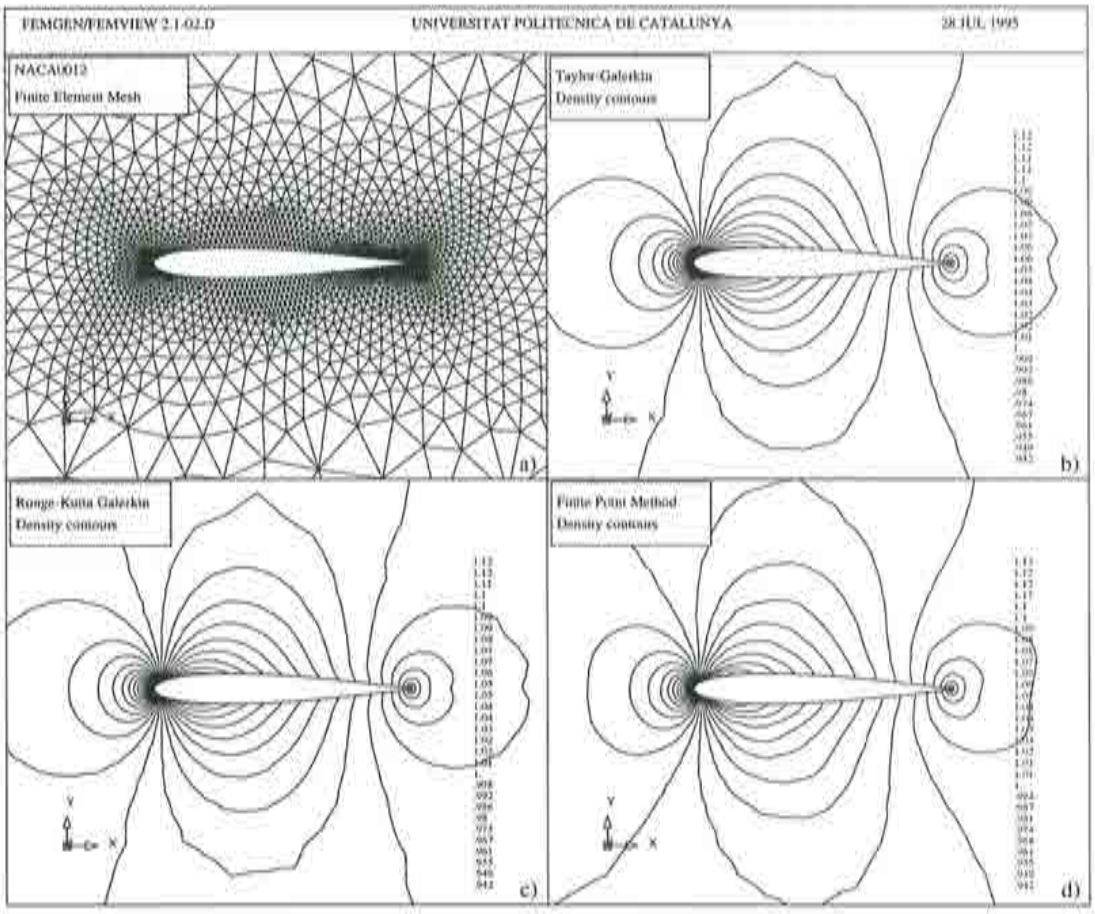


Figure 5.16 : NACA0012 profile: a) Mesh, b) TG solution, c) RK solution and d) FPM solution are shown.

The results of the FPM were obtained by employing 7 nodes in Ω_i, $\lambda = \lambda_{min}$, $c = 1$ and linear base functions ($m = 3$). A global comparison of the meshless solution is shown in Figure 5.16. In Figure 5.16 a), the finite element mesh is shown, in b), c) and d), density contours are presented for the Taylor-Galerkin solution, a four-stage Runge-Kutta Galerkin result (detailed in chapter 6) and the FPM solution, respectively. Qualitatively, all results are very similar. In Figure 5.17, close-up views in the stagnation area enhance the comparison of density contours and therefore the quality of the results. The following list describes each part of Figure 5.17:

a) Finite point method using Gauss weighting functions for the space interpolation but without WBD,

b) Finite point method with full Gauss weighting including the diffusion terms,

c) RK-Galerkin finite element and

d) Taylor Galerkin finite element scheme.

Note the improvement of solution b) using Gauss weighting function also for the diffusion terms when compared to solution a). The density contours of b) do not exhibit any oscillations in the stagnation area through the use of weighted diffusion terms.

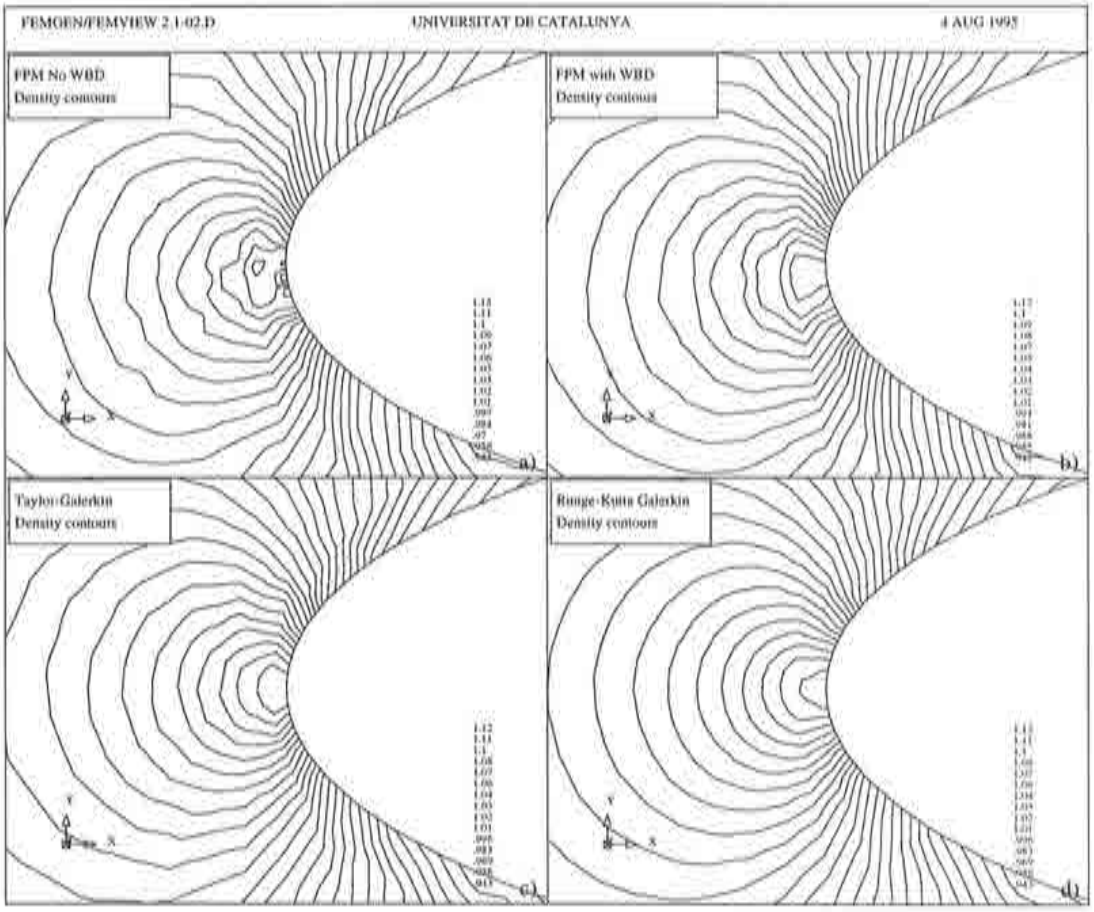


Figure 5.17 : NACA0012 profile: Close up density contours in the stagnation area for a) FPM without WBD, b) FPM with WBD, c) Runge-Kutta solution and d) Taylor-Galerkin solution (WBD: weighted balancing diffusion).

In Figure 5.18 and Figure 5.19, velocity vectors and velocity contours in the stagnation zone are displayed for the FPM using weighted balancing diffusion, respectively. The quality of the results are again very good, always keeping in mind that a very coarse grid is used.

A test of accuracy is comparing the pressure coefficient c_p of the FPM with the finite element solution. Figure 5.20 shows the comparison of FPM without WBD and using Gauss WBD with the Taylor-Galerkin result. Note that without WBD, oscillations and a deviation from the finite element result is visible, whereas the use of Gauss weighting for the balancing dissipation the result improves.

The other test of accuracy is error in the stagnation density which is for the FPM

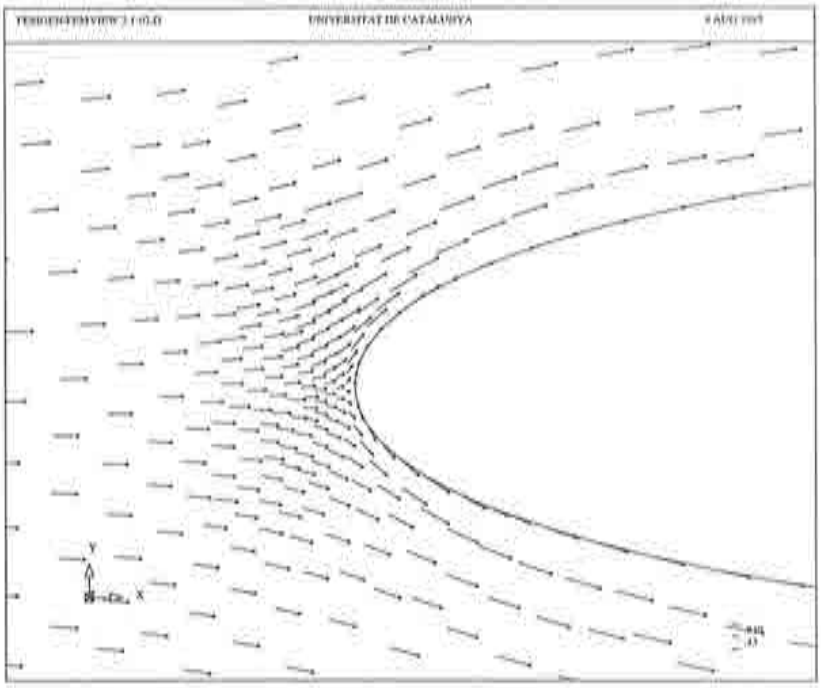


Figure 5.18 : NACA0012 profile: velocity vectors in the stagnation zone are shown for the FPM with Gaussian weighted balancing diffusion terms for $M_\infty = 0.5$ and $\alpha = 0°$.

without WBD: 0.47%, using Gauss weighted diffusion: 0.06%, for the Taylor-Galerkin scheme 0.04% and for the Runge-Kutta Galerkin scheme 0.02%. The drag coefficient is without WBD: $c_D = -0.0040$ and using Gauss WBD: $c_D = -0.0017$, compared to $c_D = 0.0028$ for the Taylor-Galerkin result and $c_D = -0.0015$ for the Runge-Kutta-Galerkin scheme.

The overall accuracy is similar to the finite element solutions for this test case if weighted diffusion terms are incorporated. The objective was to show the importance of the WBD for precise results and no oscillations.

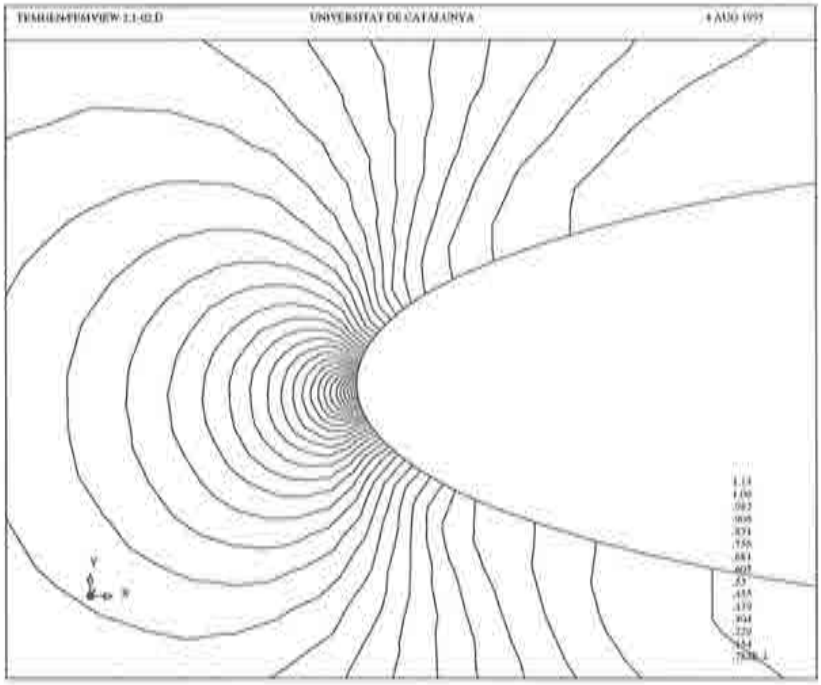


Figure 5.19 : NACA0012 profile: velocity contours in the stagnation zone are shown for the FPM with Gaussian weighted balancing diffusion terms for $M_\infty = 0.5$ and $\alpha = 0°$.

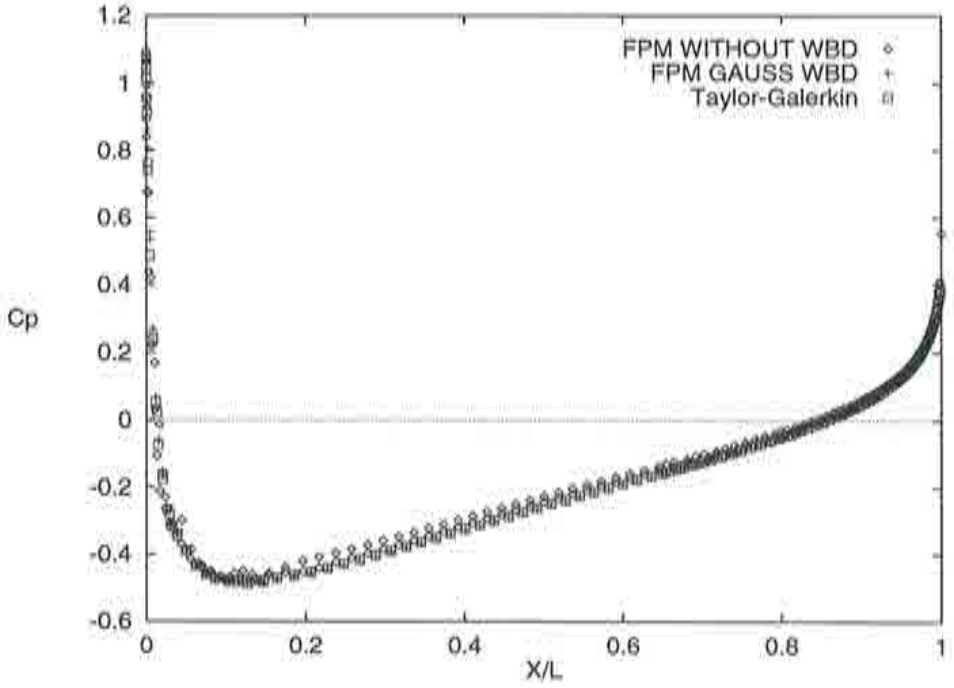


Figure 5.20 : NACA0012: Comparison of the pressure coefficient around the airfoil for the FPM with a reference finite element solution. Note the improvement for weighting functions for the diffusion terms. (WBD: weighted balancing diffusion.)

5.5.3 Subsonic Viscous Compressible Flow

Numerical experiments have been performed for the following test case already analyzed by finite elements in section 4.10.1: laminar Navier-Stokes solution in two dimensions (NACA0012; $M_\infty = 0.5$; $Re = 5000$; α=0°) with the reference in [12].

The grid of points (Figure 5.21) that was used to resolve the test problem consisted of 14106 points which were expanded in structured form away from the boundary. The points were obtained from the same mesh as were used for the results of section 6.4. The clouds of points exhibit a strong stretching close to the boundary and in the wake.

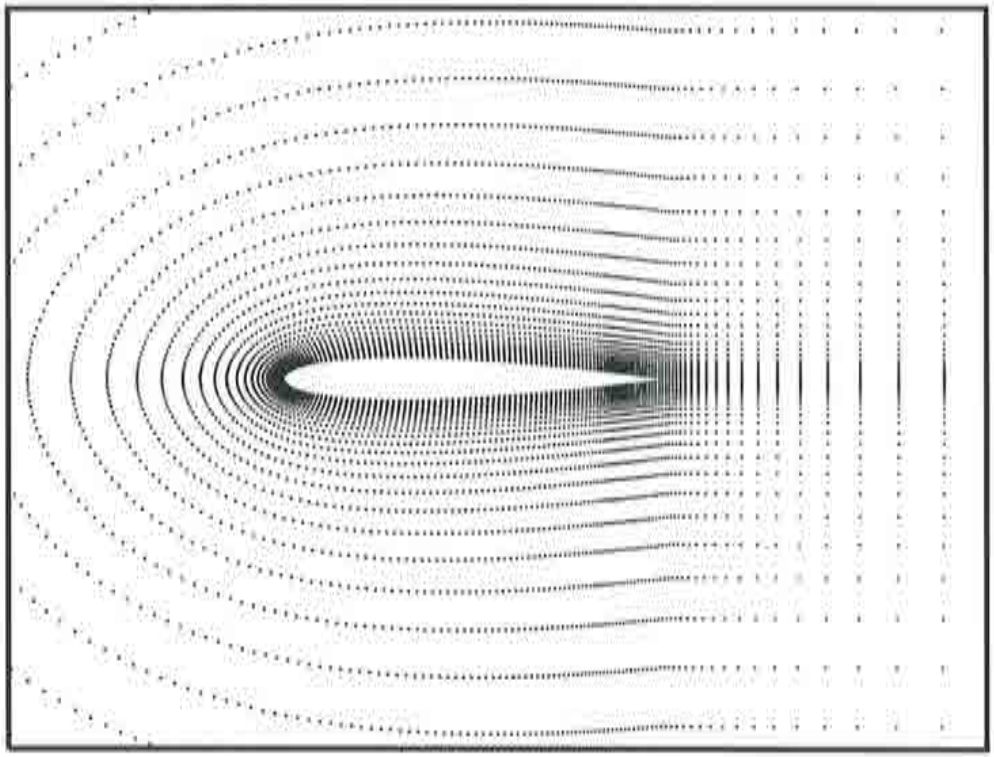


Figure 5.21 : Grid of 14106 points for the solution of viscous flow around a NACA0012 airfoil (Re=5000, M_∞=0.5, α=0°).

The computation using 5 points per cloud with weighting functions for the spatial discretization as well as for the diffusion terms show that the overall features of the flow are very similar to the results obtained from previous calculations in section 4.10.1. The Mach number contours (Fig. 5.22) around the airfoil or velocity vectors along the trailing edge (Fig. 5.23) show good quality of the solution as no differences can be visualized. The pressure coefficient (Fig. 5.24) and the separation point of 84.0% chord are within reasonable limits for this difficult test case. The pressure drag coefficient shows a very nice agreement with 0.0225 compared to 0.0228 of [12].

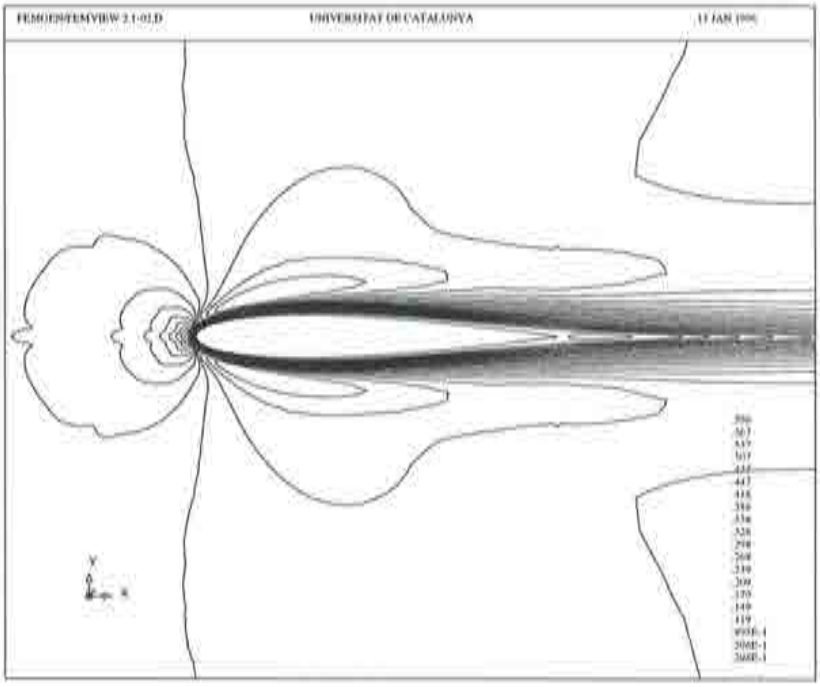


Figure 5.22 : Mach number contours for the solution of viscous flow around a NACA0012 airfoil (Re=5000, M_∞=0.5, α=0°) using 14106 points.

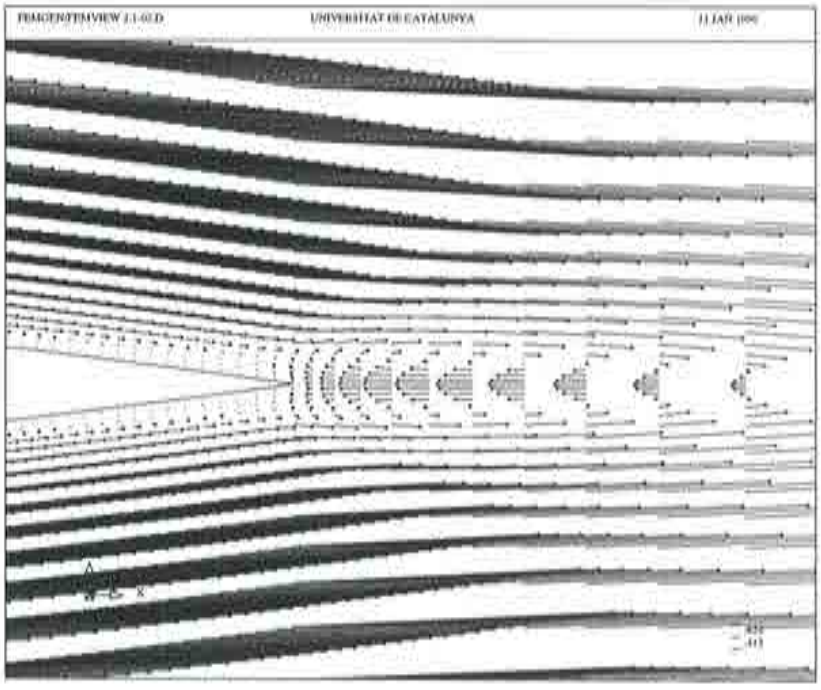


Figure 5.23 : Velocity vectors at the trailing edge for the solution of viscous flow around a NACA0012 airfoil (Re=5000, M_∞=0.5, α=0°). Note the recirculation at tip of the trailing edge.

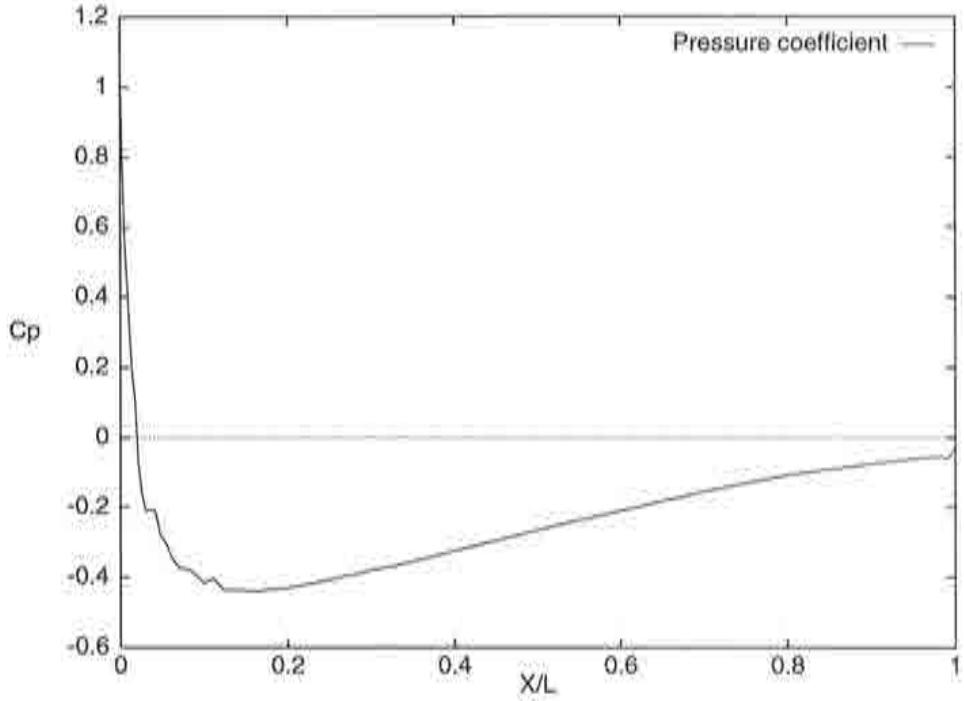


Figure 5.24 : Pressure coefficient along the airfoil for the solution of viscous flow around a NACA0012 airfoil (Re=5000, M_∞=0.5, α=0°).

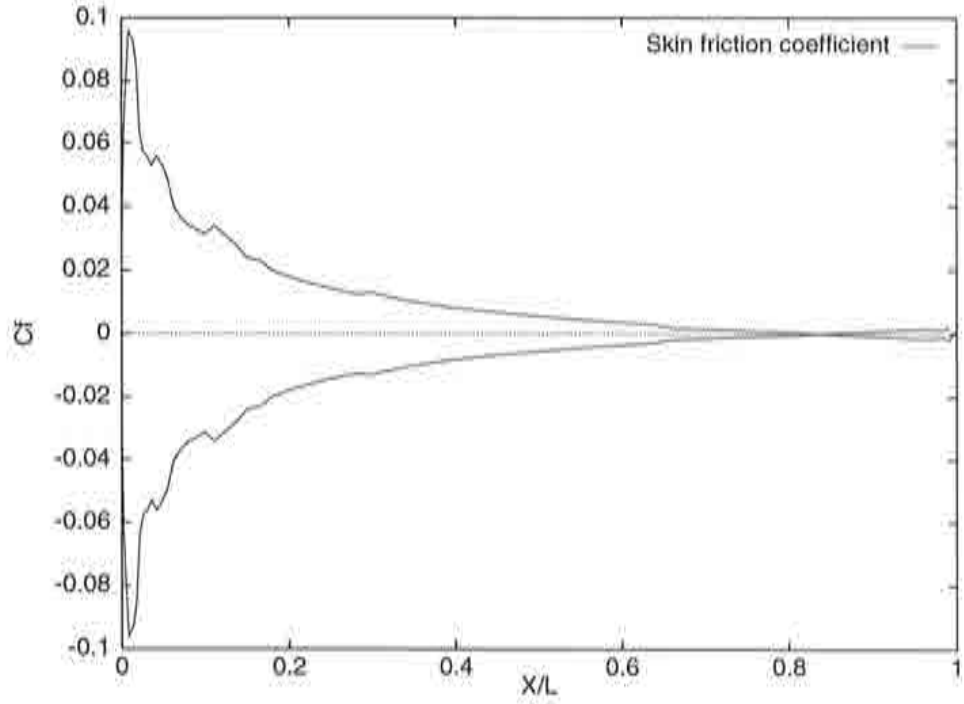


Figure 5.25 : Skin friction coefficient along the airfoil for the solution of viscous flow around a NACA0012 airfoil (Re=5000, M_∞=0.5, α=0°).

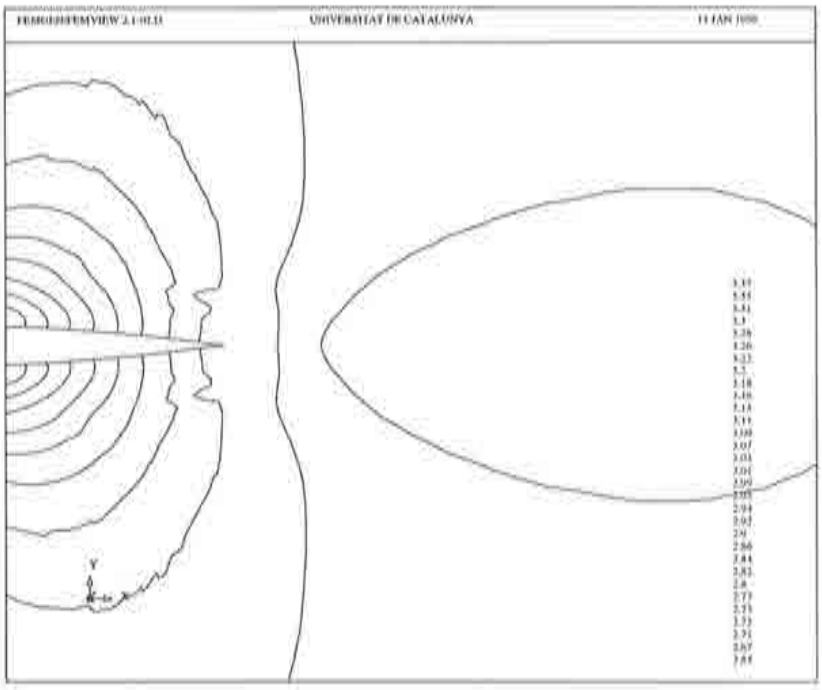


Figure 5.26 : Pressure lines in the wake for the solution of viscous flow around a NACA0012 airfoil (Re=5000, M_∞=0.5, α=0°). Note their parallel position to each other in the wake.

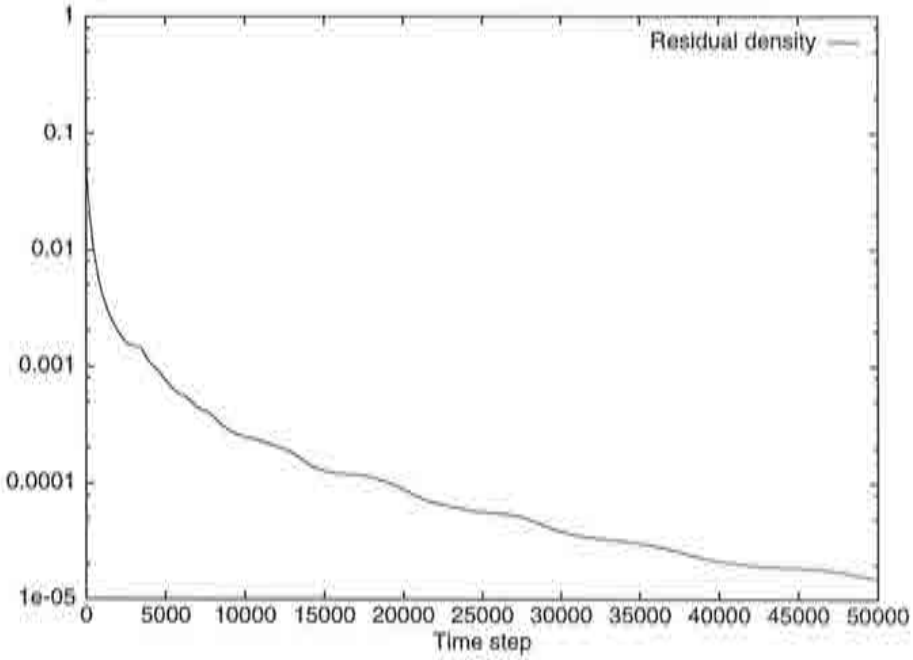


Figure 5.27 : Convergence history of the density residual for the solution of viscous flow around a NACA0012 airfoil (Re=5000, M_∞=0.5, α=0°).

5.5.4 Transonic Inviscid Flow

In the following, the test case of transonic flow of section 4.10.1 ($M_\infty = 0.8$, $\alpha = 0°$) is repeated. The points of the same coarse mesh as for the finite element solution are used and shown in Figure 5.15. The solution of the pressure coefficient c_p can be seen in Figure 5.28.

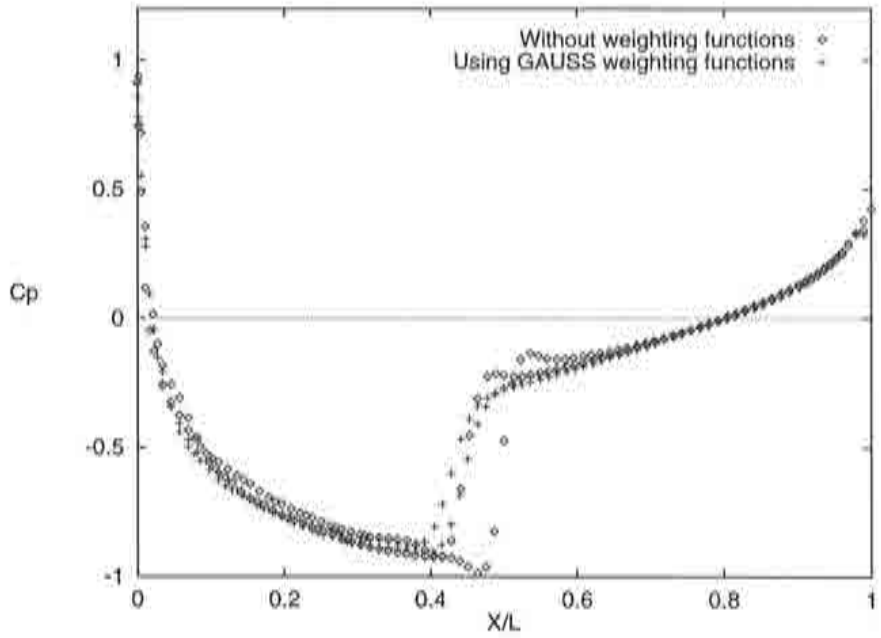


Figure 5.28 : Comparison of the pressure coefficient c_p for the solution of a transonic flow ($M_\infty = 0.8$, $\alpha = 0°$) without weighting and using Gauss weighting functions, including WBD.

The resolution of the shocks of the upper and lower surface should be in the same location. The shock location for the upper and lower surface are not the same and, hence, the results are wrong. Obviously, the result for flows with shocks depend on the grid of points, because they are not generated in the same form on the upper and lower surface. The use of weighting functions has improved the solution but has not been able to capture the shock in the right position. The reason for this is that the method presented in this work is non conservative and therefore is not able to resolve shocks correctly, because vital physical shock relations such as the Rankine-Hugeniot conditions are violated, see chapter 2 and [13]. For subsonic flows, conservation of mass, moment and energy do not play such a vital role as when shocks appear. This is observed from the good results for subsonic flow.

5.5.5 Supersonic Inviscid Flow

It has been shown that the finite point method so far described is non conservative, which is prohibitive for flows in which shocks occur because the Rankine-Hugeniot condition can not accurately fulfilled. However, the following test case shows that a

solution of high speed flows may be obtained even though no conservative correction has been applied. This has been achieved by generating a regular grid with an equal spacing size all over the grid.

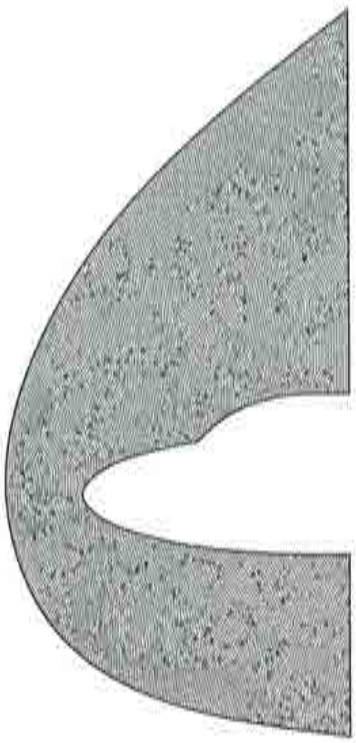


Figure 5.29 : Finite point grid of 11000 points for the resolution of hypersonic flow around a double ellipse.

So, the next 2D test case is again a hypersonic inviscid flow of Mach 8.15 around a double ellipse at $\alpha = 30°$, as described earlier and which is documented by the proceedings of the workshop in Antibes, 1990 [14]. The solution is characterized by a strong primary bow shock and a weaker canopy shock.

To solve this problem, a grid of 14943 points was generated as shown in Figure 5.29. Linear base functions ($m = 3$) and $n = 7$ point clouds with Gaussian weighting were used. Figure 5.30 presents the Mach number contours and density lines, respectively. Note that the solution is very smooth and the location of the canopy shock is well captured. The numerical overshoot of about 3% in Mach number is within reasonable limits.

Figure 5.31 illustrates density contours in the vicinity of the stagnation area showing no oscillations. Figure 5.32 displays the pressure coefficient c_p on the boundary of the double ellipse which compares well to other contributors [15].

The residuals of the solution have been reduced to more than five orders of magnitude, see Figure 5.33. The initial bump is due to the increase in the safety factor after 1000 iterations. The low safety factor for the first 1000 time iterations avoids negative pressures.

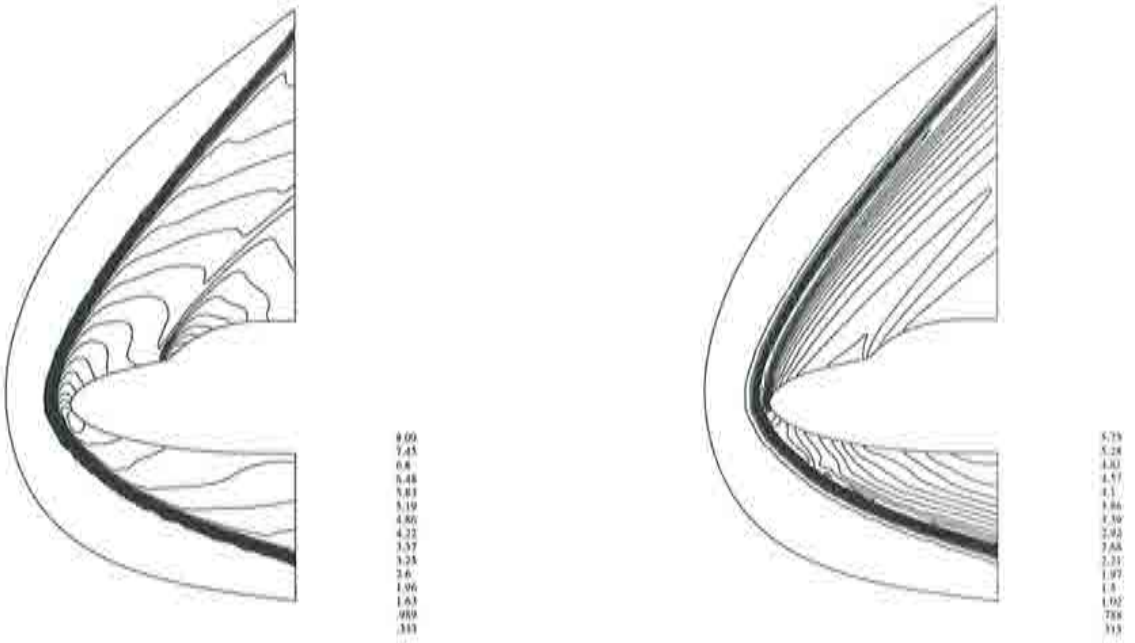


Figure 5.30 : Mach number and density contours and for the hypersonic flow around a double ellipse at $M_\infty = 8.15$ and $\alpha = 30°$.

Discussion

By use of an equal spacing of the elements, the error committed compared to a conservative scheme can be kept small. A conservative scheme would integrate across the volume of the domain. Since each contribution of integration is of similar size for this problem, it finally cancels out. This explains the good solution obtained in this test case. Obviously, it is not the idea to use equally spaced points for this method, so that suitable modifications should be made to assure conservation. That, however, is not dealt with in this thesis.

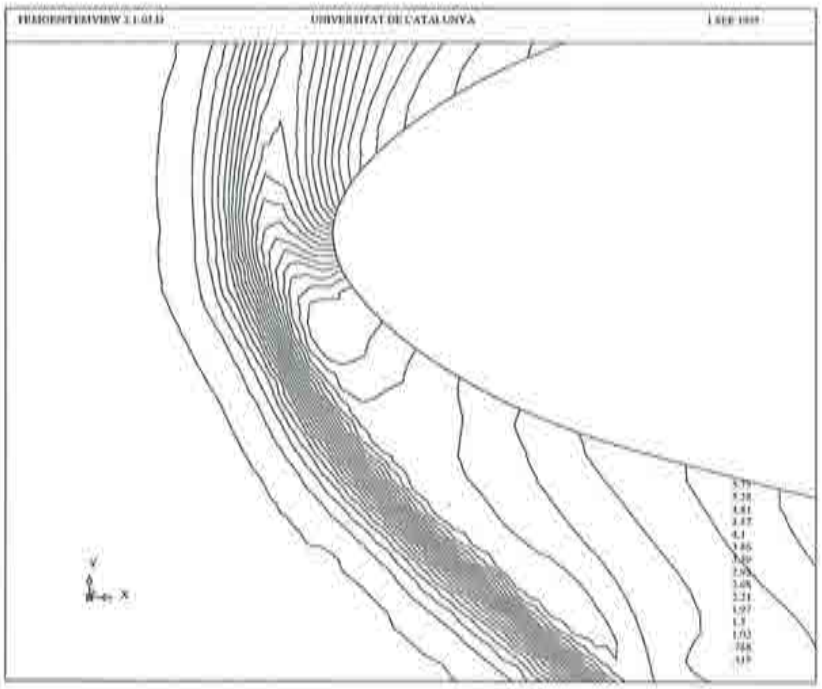


Figure 5.31 : Density contours in the stagnation zone for the hypersonic flow around a double ellipse at $M_\infty = 8.15$ and $\alpha = 30^\circ$.

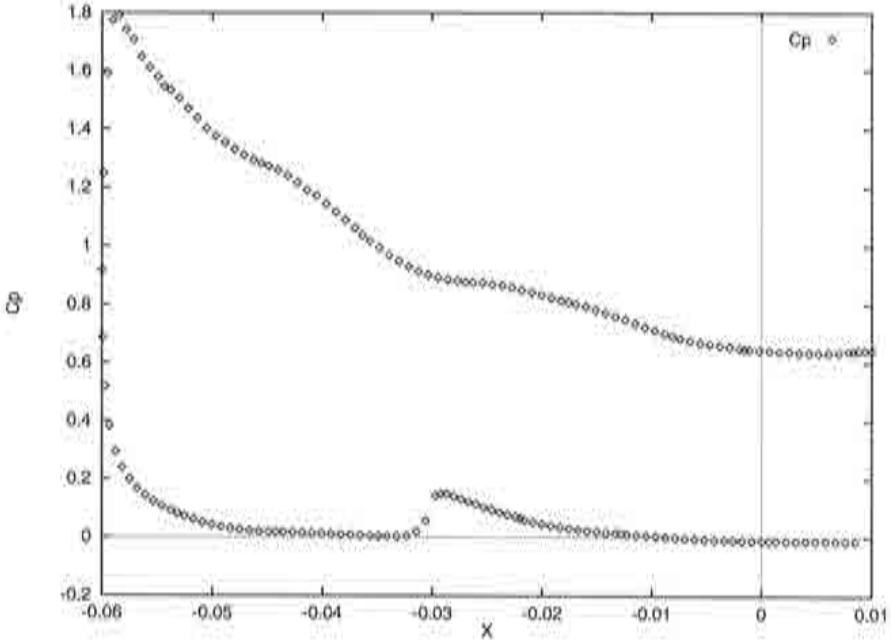


Figure 5.32 : Pressure coefficient along the surface of the body for the hypersonic flow around a double ellipse at $M_\infty = 8.15$ and $\alpha = 30^\circ$.

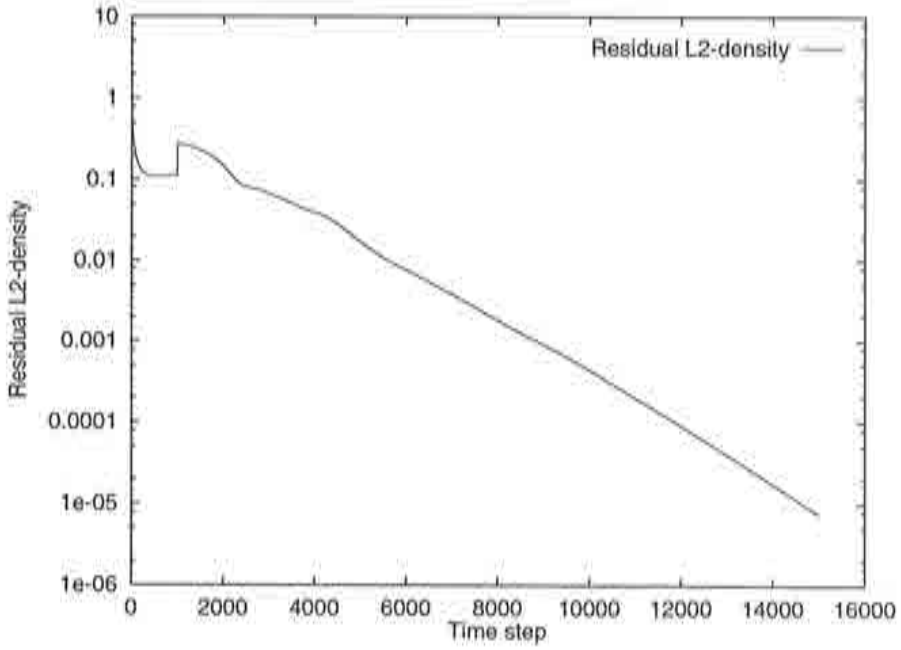


Figure 5.33 : Convergence history of the density residuals (L2-norm) for the hypersonic flow around a double ellipse at $M_\infty = 8.15$ and $\alpha = 30°$.

5.6 Conclusions

The use of clouds of points is extended to solve compressible flow problems in two dimensions. A global concept for linear and quadratic interpolations is presented which avoids the use of defined connectivities among points. The two step algorithm of the Taylor-Galerkin scheme advances the solution in time. The necessary balancing diffusion which avoids oscillations is of Jameson type and is adequately modified for this type of formulation.

Several conditions for the adequate selection of points is used to locally formulate the spatial discretizations in order to avoid singularities for the setup of matrices. Weighting functions are added for both the basic discretization and the diffusion terms.

The test cases of subsonic flows without shocks have been accurately resolved with respect to the number of points employed. It has been demonstrated that the incorporation of weighting functions, also for the diffusion terms, is necessary to maintain the high level of accuracy. Even though only isotropic weighting terms have been used with meshes containing clouds with high aspect ratios for the solution of laminar viscous flow, good results are obtained. The incorporation of anisotropic weighting functions may improve these results, especially near the boundary and in the wake.

However, if shocks appear such as transonic flows, the formulation so far presented is not able to reproduce them in the correct location. This great drawback is due to the non conservative character of the method. In addition, the convergence of the residuals for the solution using clouds of points is slower when compared to the finite element results using a Taylor-Galerkin procedure. The cost efficiency is therefore lower, also because of the additional evaluations of the weighting terms.

Further research in this area is therefore needed to enhance the solution in these cases. However, the advantages of a more flexible discretization technique, especially in the field of mesh generation and adaptivity, may compensate the disadvantages. As this young concept of spatial discretization is in its infancy, many possibilities exist to improve the method and to avoid some of the drawbacks encountered in the present scheme.

References

[1] Nayroles, B., Touzot, G. and Villon, P. "Generalizing the Finite Element Method: Diffuse Approximation and Diffuse Elements", Computational Mechanics, 10, 307-318, 1992

[2] Belytschko, T., Lu, Y. and Gu, L. "Element Free Galerkin Methods", Int. Journal for Numerical Methods in Engineering, 37,229-256, 1994

[3] Liu, W.K., Jan, S. and Belytschko "Reproducing Kernel Particle Methods", Int. Journal for Numerical Methods in Engineering (to be published)

[4] Batina J., "A Gridless Euler/Navier Stokes Solution Algorithm for Complex Aircraft Applications", AIAA paper, 93-0333, Reno NV, January 11-14, 1993

[5] Oñate E., Idelsohn S. and Zienkiewicz O.C. "Finite Point Methods in Computational Mechanics", Publication CIMNE No. 67, July 1995

[6] Oñate, E. "Cálculo de Estructuras por el Método de Elementos Finitos", CIMNE, Barcelona, January 1992

[7] Jameson A. 'Transonic Aerofoil Calculations Using the Euler Equations', in Numerical Methods in Aeronautical Fluid Dynamics, 1982

[8] Peraire, J. "A Finite Element Method for Convective Dominated Flows". PhD Thesis at University College of Swansea, 1986

[9] Peiro, J. "A Finite Element Procedure for the Solution of the Euler Equations on Unstructured Meshes". PhD Thesis at University College of Swansea, 1989

[10] Oñate E., Idelsohn S., Fischer T., Zienkiewicz O.C. "A Finite Point Method for analysis of fluid flow problems", 9th Int. Conf. on Finite Elements in Fluids, Venezia, Italy, 15-21 October 1995

[11] Zienkiewicz O.C., Morgan K.,Sai S., Codina R., Vázquez M. "A General Algorithm for Compressible and Incompressible Flow Part II, Tests on the Explicit Form", IJNME Vol 20, 887-919, 1995

[12] Mavripilis P., Jameson A., Martinelli L., "Multigrid Solution of the Navier-Stokes Equations on Triangular Meshes", ICASE Rep. No 89-11, February 1989.

[13] Courant R., Friedrichs K.O. "Supersonic Flows and Shock Waves", Interscience Publishers, New York, 1948

[14] "Workshop on Hypersonic Flows for Reentry Problems", Proceedings, Antibes, 22-25 January 1990

[15] Wang Z, and Richards B.E. *High Resolution Schemes for Steady Hypersonic Flow*, Proceedings of Workshop on Hypersonic Flows for Reentry Problems, Antibes, 22-25 January 1990

Chapter 6

Performance and Accuracy

6.1 Introduction

This chapter contains different practical methods which are used to exploit the available resources as far as possible. First of all, a description is made of how the convergence properties of the numerical scheme can be enhanced by introducing the Runge-Kutta Galerkin algorithm together with residual smoothing. In addition, an enthalpy damping routine is used to improve convergence to steady state for inviscid flows.

In the final section, the work in the field of parallel computing is presented together with some numerical examples. The computing architectures employed in this thesis have been serial, vector and SIMD parallel. Most of the serial calculations have been performed on a SGI R4000 workstation. Some of the results for three dimensional flows of section 4.10.1 were done on the Cray-YMP of CESCA (Centre de Supercomputació de Catalunya). All results using parallel computing were obtained on a CM-200 of CEPBA (Centre de Paral·lelització de Barcelona). The implementation of the programs developed in this thesis wore done on Silicon Graphics workstations and Cray-YMP are not explicitly detailed here, because their form of programming using Fortran is fairly standard. However, the details of the installation of the Taylor-Galerkin scheme on the CM-200 are explained, because new data structures and communication algorithms have to be defined.

6.2 Runge-Kutta Galerkin

The two step Taylor-Galerkin procedure described in sections 3.1.2 and 4.2 has a limitation of the Courant number C:

$$C \leq 1 \tag{6.1}$$

This limit can be increased by applying higher order time integration procedures, such as the explicit Runge-Kutta scheme [2]. In addition, a residual averaging procedure can increase the limit of stability, which again increases the CFL limit. Finally, a

simple enthalpy damping procedure for inviscid flows greatly improves convergence to steady state. A detailed description of this methodology for finite volume schemes can be found from other authors [2, 14, 3]. The very effective multigrid strategy for convergence acceleration [4, 5] or the use of a side based data structure [6] has not been considered in this work.

6.2.1 Runge-Kutta Time Integration

A k-stage Runge-Kutta time integration may be written as

$$\mathbf{u}^{(i)} = \mathbf{u}^n - \kappa_i \mathbf{M}_l^{-1} \Delta t \left[\mathbf{f}^* \left(\mathbf{u}^{(i-1)} \right) \right] \quad i = 1, k \tag{6.2}$$

$$\mathbf{u}^{n+1} = \mathbf{u}^{(k)} \tag{6.3}$$

where the superscript (i) represents each integration stage, κ_i are the coefficients which are chosen according to the number of stages employed. $\mathbf{M}_l$ is the lumped mass matrix and the function of the fluxes $\mathbf{f}^*$ is given by eq. 4.19.

The advantages of using higher order integration are to increase the order of time accuracy of the scheme as well as increasing the stability limit. For instance, fourth order time accuracy can be obtained by a 4-stage Runge-Kutta scheme with a condition for C as [2, 3]:

$$C \leq 2\sqrt{2} \tag{6.4}$$

This is almost three times greater that conventional two step explicit schemes. The coefficients of this 4-stage algorithm are:

$$\kappa_1 = \frac{1}{4}, \quad \kappa_2 = \frac{1}{3}, \quad \kappa_3 = \frac{1}{2}, \quad \kappa_4 = 1 \tag{6.5}$$

The advective fluxes are evaluated at every stage whereas the evaluation of the viscous fluxes is performed only once per time iteration. The evaluation of the dissipative fluxes $\mathbf{f}_d$ of eq. 4.19 are evaluated either once or twice at each time iteration. This combination together with the increase in Courant number provides an increase in convergence of generally more than two.

6.2.2 Residual Smoothing

Another form of increasing the limit for the Courant number is by means of implicitly averaging the residuals [7], by introducing $\mathbf{r} = \frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t}$:

$$\bar{\mathbf{r}} = \mathbf{r} - \epsilon \nabla^2 \bar{\mathbf{r}} \tag{6.6}$$

where ϵ is chosen about 0.2 and the overbar refers to the final state of the residuals. This equation can be solved approximately by performing two explicit Jacobi iterations. The explicit iterations are then formulated in an isotropic form by extending eq. 4.23 to two dimensions for any function ϕ:

$$C\left[\nabla^2\phi\right]_i \simeq \left[\mathbf{M}_l^{-1}(\mathbf{M}_c - \mathbf{M}_l)\bar{\phi}\right]_i \tag{6.7}$$

where the terms are the same as for eq. 4.23 except for h and the mass matrices which are now defined by eq. 4.27.

6.2.3 Enthalpy Damping

For inviscid calculations, the stagnation enthalpy in the energy equation remains constant all over the flow field at steady state.

$$H_0 = \text{constant} = H_\infty \tag{6.8}$$

where $H = E + \frac{p}{\rho}$ is defined as the enthalpy. This property is important for the incorporation of a damping function. Hence, the addition of a damping term does not alter the result at steady state ($\frac{\partial \mathbf{u}}{\partial t} = 0$), because $H = H_\infty$ [2]. In fact, this damping does have considerable convergence acceleration properties. The non dimensional damping term which is implemented for the continuity and momentum equation is, slightly modified from [2] by division of H_∞:

$$\mathbf{u} = \mathbf{u}\frac{1}{1 + \alpha\frac{H - H_\infty}{H_\infty}} \tag{6.9}$$

where α is a constant, usually chosen 0.1. The energy equation contains a the term similar to the above but is modified [2, 3] to avoid a quadratic term in H, resulting in:

$$\rho E = \frac{\rho(H - H_\infty)}{1 + \frac{\alpha}{H_\infty}} - p + \rho H_\infty \tag{6.10}$$

Details can be found in the referenced literature.

6.3 Parallel computation

6.3.1 Introduction

In recent years, the development of finite element and finite volume methods for the simulation of fluid flow has reached a stage at which robust solvers are available to deal with even complex flow situations. Even some of the more complex problems in CFD can be accurately solved. However, future demands in CFD, especially for aerodynamic design, require a huge amount of computational power and memory, see Figure 6.1, that can only be attended by parallel systems in the next years [1]. For this, parallel solvers need to be developed which provide the same degree of accuracy and robustness as their serial versions to be able to meet some of the objectives in CFD in the next years.

Parallel CFD is still a research topic and the practical performance of different parallel platforms running different applications are still not yet tested completely. Many possibilities exist to cluster a number of processors in order to form a parallel

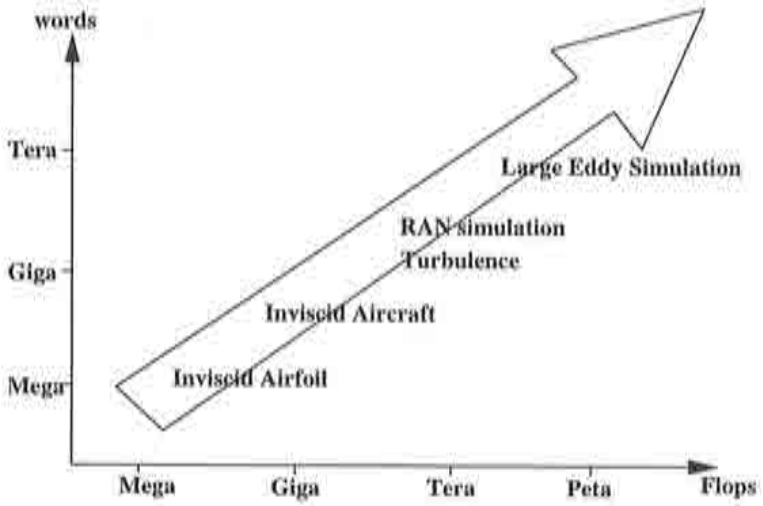


Figure 6.1 : Estimate of future demands on computational power in CFD [9]. This plot shows computational versus memory needs for a given problem approximately. RAN refers to the solution of the Reynolds averaged Navier-Stokes equations which includes turbulence modeling.

environment where different variables influence its structure. These are, for example, memory, cache, disk storage and communication bus. The choice to assemble a parallel computing network lies in the fact of sharing these items or locally owning these devices and how fast communication can be done among them.

In the following several definitions of parallel vocabulary are used, but are not explained here in detail. A thorough definition of the employed expressions can be found in [1]. Especially, the classification of parallel computing architectures according to Flynn's taxonomy for different processor clusters and Bell's taxonomy describing memory divisions are explained there.

6.3.2 Classification of Parallel Computing Architectures

One platform is the current state of the art RISC-processor environment, which is quite inexpensive because of mass production and very popular for modern workstations. Most of the two-dimensional examples were calculated such a processor system built into a SGI Indigo workstation and can be classified as serial processing or SISD. There is a current trend to cluster these workstations or the processors to form a MIMD parallel computing system, such as the IBM SP2 or the Silicon Graphics Power Challenge, respectively. Another platform is the powerful vector processor environment capable of executing several similar tasks at a time, provided that the program has been properly vectorized, ie. Convex C3480 or the Cray-YMP, which was used to calculate some three dimensional examples presented in chapter 4. Finally, there exists the SIMD massive parallel computing environment, such as the Connection Machine CM200 with 2048 processors, for which numerical experiments and performance estimations are presented.

SIMD

Two of the general groups of parallel architectures have been distinguished: The single instruction multiple data (SIMD) environment is capable of performing the same instruction on different data at the same time ie. multiplication of a vector with a scalar.

MIMD

The multiple instruction multiple data (MIMD) environment can execute different instructions on different data at a time, thus providing more flexibility. Many times for numerical methods running on MIMD type architectures, domain decomposition procedures are applied in order to divide the number of elements into subdomains of elements in accordance with the number of processors. Then, the equations are first solved in each subdomain and communicated with the rest. The best performance is obtained by balancing the workload of each processor in order to avoid as far as possible that processors run idle.

6.3.3 Computational Platform

Because of the multiples of choices and subsets of possible architectures, the optimal choice for a computational algorithm is not easy. But obviously, the above classification has its implication on performance and programming style for each scheme.

This is a contribution to demonstrate the strengths and weaknesses of an explicit algorithm using unstructured triangular meshes, the Taylor-Galerkin scheme in a SIMD environment. An advantage of the two-step algorithm with a lumped mass matrix formulation is the explicit form of the algorithm without matrix solver and low memory demands. This avoids a global restructuring of the program and reduces complexity. However, the unstructured mesh connectivities lead to more communication overhead.

For the scope of this thesis, I will limit myself to describe the work in the field of parallelism using the Connection Machine CM-200 which exploits a SIMD-type architecture. The main parallel computing environment and tools consisted of:

- Computer: Connection-Machine CM-200 with 2048 processors and 1 Mbit memory.
- Communication: Router software libraries (method FASTGRAPH).
- Peak performance: 640 MFlops
- Programming language: Fortran 90 and CMF Fortran

6.3.4 Parallel Implementation

The implementation involves the use of local time stepping to accelerate convergence to the steady state solution. However, other convergence acceleration techniques such

as enthalpy damping or implicit residual averaging have not been incorporated. The following list summarizes the main features of the Taylor-Galerkin scheme (eqs. 4.13 and 4.14) used as a basis for parallelization which was described earlier in section 4.2

- 2D Taylor-Galerkin finite element formulation
- Two-step explicit scheme
- Addition of balancing dissipation (Lapidus and Jameson)
- Local time steps
- Adaptive remeshing and error estimation

Once the algorithm is determined, some thought must be given to the data structure and interprocessor communication to be able to exploit the speed of the massive parallel environment of the CM-200.

In the present form of the program, no attempt has been made to decompose the domain into partitions, according to the number of processors. This practice is common for MIMD platforms, but seems to be impractical for SIMD machines with more than 2000 processors, especially when using completely unstructured meshes. However, it must be mentioned that Farhat *et al* [15] have shown that it is possible to partition the mesh into structured submeshes of 16 elements each which coincide with the structure of the processor connectivity. This is to reduce the communication bandwidth, and the promising results presented by Farhat *et al* [16] are based on calculations of irregular grids. However, with the advent of a new communication library, called FASTGRAPH, a decomposition is preprocessed once by the system and stored as the communication pattern for performance enhancements [17].

Data Structure

It is important to find the adequate assignments of the vectors involved in order maximize performance. Three different parameters are important that affect the decision of the data structure: vectors with a size of multiples of number of points, elements and boundary sides. The optimal layout for assigning these parameters in parallel to the processors of the CM-200 is sought for.

The optimal choice is dividing the main vectors into multiples of number of points, elements and boundary sides and keeping any other dimension in serial. This is justified by the fact that most operations occur at the level of these vector combinations having the same dimension in one direction but different dimensions in the other direction. An example would be the multiplication of the vector $\mathbf{u}$ with the scalar Δt.

Consider, for instance, the conservative variables $\mathbf{u}$ with the following dimensionalization:

```
dimension u(4,npoin)
```

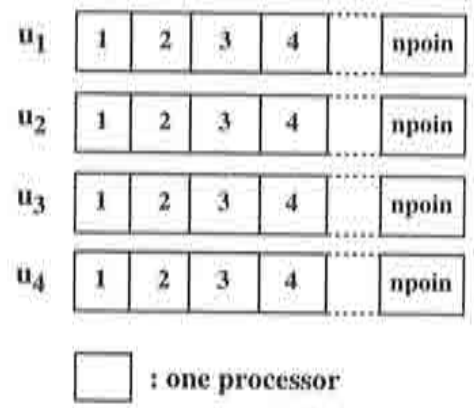


Figure 6.2 : Layout of the conservative variables u where each box assigns one processor to each variable and each node.

By means of the following compiler directive, we are able to control the layout and performance of that vector:

```
cmf$layout u(:serial,:news)
```

We obtain 4 virtually parallel vectors assigned to each processor as shown in figure 6.2. It is interesting to note that the serial version of the code has exactly the opposite structure in order to optimize the cache alignment:

```
dimension u(npoin,4)
```

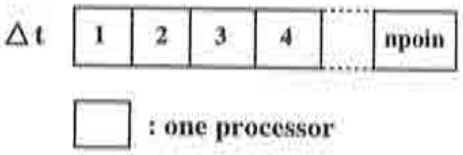


Figure 6.3 : Layout of the time step increment Δt where each box assigns one processor to each Δt of each node.

The advantage of inverting the structure is that this vector can be combined with a vector of the same size, but with a different dimension. For example the time step increment Δt with a dimension and layout as shown in figure 6.3. The assignment in the program code is controlled by the following statements:

```
dimension Δt(npoin)
```

and

```
cmf$layout   Δt(:news)
```

Any other distribution would provide additional difficulties in combining the two vectors. A similar form of data structure has also been advocated in [18], also using unstructured meshes to explicitly solve laminar viscous flows.

Interprocessor Communication

Once the layout of the vectors has been defined, some data communication between vectors of different layouts remains which requires that different processors exchange their contents with others. Therefore, interprocessor communication is an essential part in parallel computations, both on MIMD architectures as well as SIMD systems such as the Connection Machine CM-200. In the current implementation, between 35 and 45 percent of the cpu time (depending on the mesh) is devoted to shift data from elemental values to nodal values and vice versa. In particular these are gather and scatter routines [19].

Although the refined structure of the program has avoided the exchange of elemental and nodal values as far as possible, a minimum data transfer still remains which is the bottleneck of the parallel codification of eqs. 4.13 and 4.14.

Basically, two possibilities exist for establishing interprocessor links on the CM-200. One is the North East South West (NEWS) system and the other is the Router system.

NEWS System

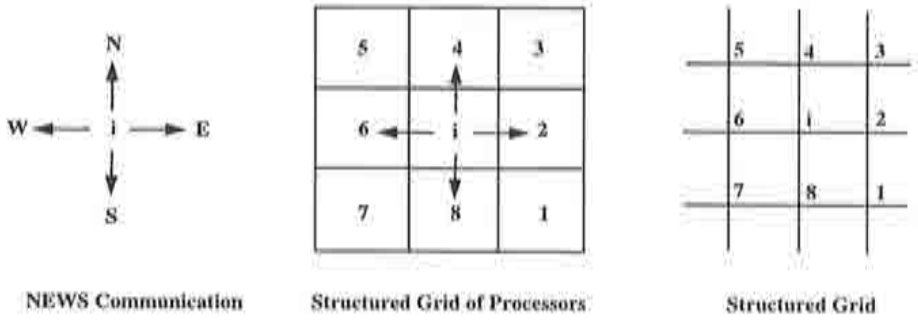


Figure 6.4 : Schematic drawing of the interprocessor communication system for data exchange among vectors of different layouts.

The NEWS system allows a structured data exchange between the neighboring processors making it attractive for flow calculations using structured meshes. The way data exchange is handled by the NEWS system can be appreciated from figure 6.4. Every processor can perform a data exchange only between the next four neighbors separated by the edge of each element. Thus, only a structured grid can be mapped exactly on to the structured grid of processors.

However, communication based on the NEWS system for unstructured meshes becomes less evident and was shown to be slower than the Router system in an attempt to optimize interprocessor communication [19].

Router System

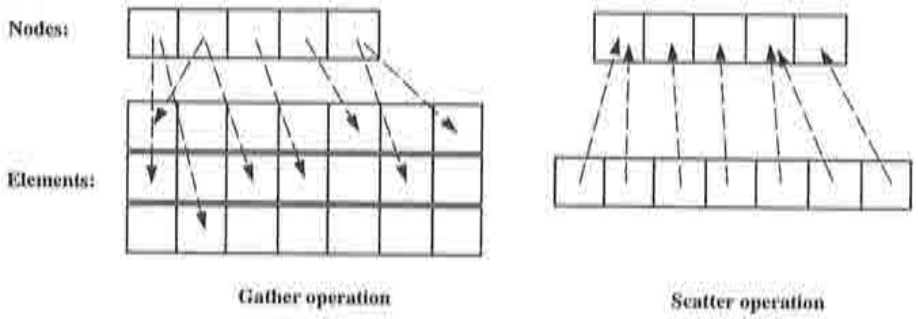


Figure 6.5 : Schematic drawing of the interprocessor communication system for data exchange among vectors of different layouts.

For the current version of the program the Router system has been preferred for simplicity even though parts of the mesh could be generated in a structured way using the NEWS system. In order to invoke the Router communication system it is necessary to use the automatic communication routines provided by the CM system library routines [17]. It allows for arbitrary data exchange between processors as shown in figure 6.5 where source and target arrays do not have to have the same rank. Figure 6.5 also indicates both gather and scatter routines which are conventionally used in finite element calculations. A gather routine collects nodal values and stores them at the elements whereas scatter routines spread elemental values to the nodes.

We have considered three available gather and scatter routines in this context. The first one is provided by the Connection Machine Scientific Software Library (CMSSL), *sparse_util_gather* or *sparse_util_scatter* modules. The other two possibilities would be the *cmf_send_add* module with option FASTGRAPH or NOP. FASTGRAPH has a speed advantage of about factor 3 for a reasonable size mesh, but with the limitation of long setup times and large memory requirements. The setup for option NOP is fast and less memory intensive but performance is slower. A comparison of these routines can be found in [20].

6.4 Numerical Examples

The first examples correspond to the use of the Runge-Kutta Galerkin scheme for the solution of the inviscid compressible flow around an airfoil using different triangular and quadrilateral meshes. The objective is to demonstrate performance and accuracy and compare the results to the Taylor-Galerkin scheme. The final examples demonstrates

accuracy and performance for the Taylor-Galerkin scheme running on the parallel computer Connection machine CM-200. The test case is the compressible laminar viscous flow around a NACA0012 profile already exposed in section 4.10.1.

6.4.1 Performance

The improvement in performance of the proposed Runge-Kutta Galerkin algorithm using residual smoothing and enthalpy damping is demonstrated for the same test case of section 4.10.1: compressible inviscid flow of $M_\infty = 0.5$ at $\alpha = 0°$ around a NACA0012 airfoil using a triangular mesh of 2556 points and 4902 elements. The accuracy is dealt with in the following section 6.4.2, so the plots are only focusing on cpu time and convergence. All the constants that were used are: $\alpha^{(2)} = 0$, $\alpha^{(4)} = 0.1$, $c_s = 0.95$ for the Taylor-Galerkin scheme and $c_s = 2.8$ for the Runge-Kutta Galerkin scheme. First, Figure 6.6 displays the convergence histories for the Taylor-Galerkin scheme (TG) using local time stepping. The plot compares the unmodified TG scheme, the TG with residual smoothing, the TG with enthalpy damping and TG using both residual smoothing and enthalpy damping. Clearly, the acceleration properties of the enthalpy damping routine become visible. Also residual smoothing improves convergence, but only marginally. A combination of both routines does not increase the performance compared to the convergence of the enthalpy damping procedure alone.

Then, Figure 6.7 presents the convergence histories for the Runge-Kutta Galerkin scheme (RK) using local time stepping. The graph again compares the RK scheme, the RK with residual smoothing, the RK with enthalpy damping and RK using both residual smoothing and enthalpy damping. The positive acceleration properties of the enthalpy damping routine is again apparent. The residual smoothing procedure increases convergence more than for the Taylor-Galerkin scheme. A combination of both routines yields slightly slower performance for convergence up to 7 orders of magnitude.

In summary, the RK scheme is about twice as fast for the convergence of the residuals. Together with residual smoothing or enthalpy damping, convergence may be accelerated. For inviscid flows, the enthalpy damping routine dramatically increases performance. However, enthalpy damping can not be used for flows for which diffusive effects can not be neglected. For the acceleration of viscous flows, the residual smoothing routine may be used. The effect of residual smoothing may be limited, though, if strongly stretched elements are employed, because the formulation for residual smoothing is isotropic. Both routines have neglectable effects on accuracy at steady state. However, the use of RK slightly alters the results when compared to TG, because the time integration scheme is different. The results for RK is shown in the next example.

6.4.2 Mesh Convergence and Accuracy

From CFD, the design engineer needs an accurate result as fast as possible. However, often the most accurate result would lead him to generate a mesh of millions of points which is not feasible or too expensive. Therefore, the computational solution should

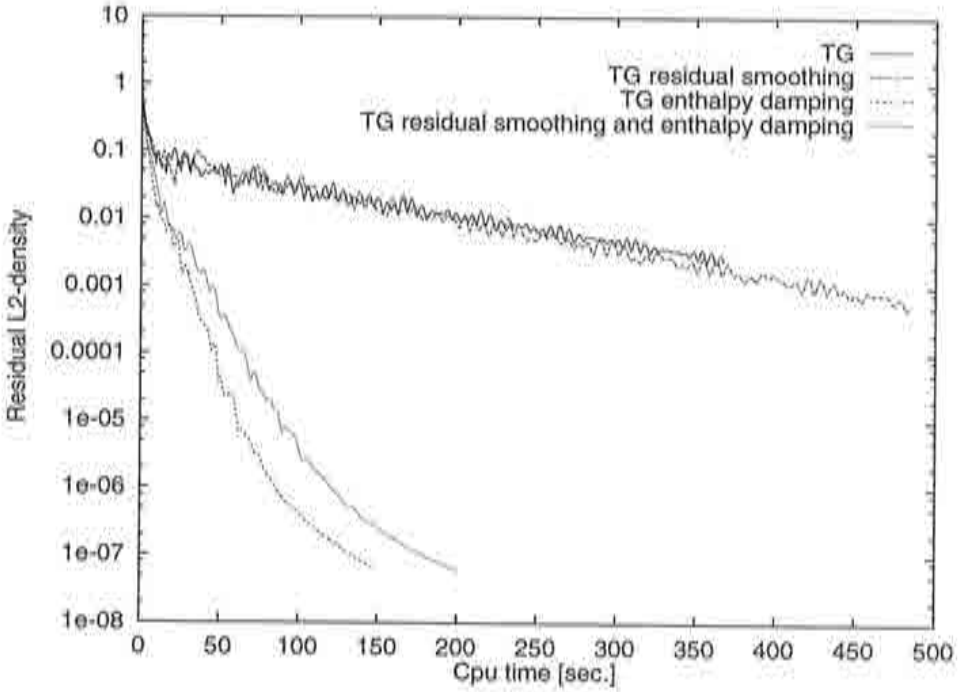


Figure 6.6 : Comparison of the convergence histories versus cpu time for different Taylor-Galerkin schemes (TG) for $\alpha^{(4)} = 0.1$: unmodified TG scheme, the TG with residual smoothing, the TG with enthalpy damping and TG using both residual smoothing and enthalpy damping.

indicate how precise the solution is for a given mesh size. This information can be difficult to obtain, especially when the exact solution is not known. But using the knowledge of the order of approximation of the numerical algorithm, an estimation of accuracy can be performed by a mesh convergence study [29].

	$\kappa^{(4)} = 0.05$		$\kappa^{(4)} = 0.1$		$\kappa^{(4)} = 0.2$		$\kappa^{(4)} = 0.4$	
Npoin	c_D	ρ_0	c_D	ρ_0	c_D	ρ_0	c_D	ρ_0
932	-0.00064	1.1244	-0.00073	1.1222	-0.00082	1.1202	-0.00091	1.1118
2556	-0.00015	1.1269	-0.00021	1.1274	-0.00029	1.1274	-0.00041	1.1266
4022	-0.00008	1.1288	-0.00012	1.1287	-0.00017	1.1283	-0.00024	1.1275
14836	-0.00004	1.1294	-0.00005	1.1293	-0.00008	1.1292	-0.00011	1.1289

Table 6.1: Comparison of the drag coefficient c_D and the stagnation density ρ_0 for 4 different triangular meshes and 4 different diffusion constants $\alpha^{(4)}$. The analytic results are $c_D = 0.0$ and $\rho_0 = 1.1297$.

As a means for estimating accuracy, the convergence of the key variables of the numerical solution is not a sufficient indicator in order to have confidence in the precision of the solution. It only shows that the solution on a given mesh does not change anymore but to assure accuracy, a mesh convergence study should be performed. This shows that the solution is mesh independent within given error bounds and that the solution for an infinitely fine mesh can be estimated. Then, we also speak of a mesh

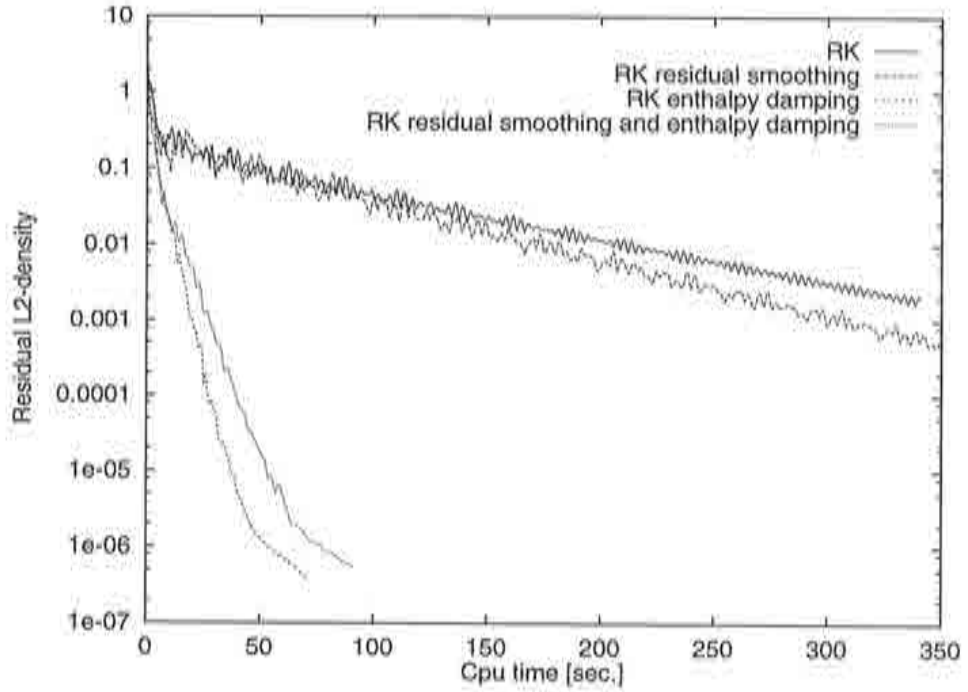


Figure 6.7 : Comparison of the convergence histories versus cpu time for different Runge-Kutta Galerkin schemes (RK) for $alpha^{(4)} = 0.1$: unmodified RK scheme, the RK with residual smoothing, the RK with enthalpy damping and RK using both residual smoothing and enthalpy damping.

converged solution which means that the result depends on the numerical scheme and not mesh. Moreover, by means of a mesh convergence study, the order of the numerical scheme can be demonstrated.

The test case considered is again the subsonic flow around a NACA0012 with no incidence and $M_\infty = 0.5$. In total four triangular and quadrilateral meshes were generated containing between 900 and 19000 nodes. In addition to accuracy, the Runge-Kutta Galerkin scheme is used to enhance the performance of the scheme along with enthalpy damping.

Mesh Convergence using Triangles

The same four meshes of unstructured triangles of section 4.10.5 (Figure 4.51) have been subjected to the Runge-Kutta Galerkin scheme. Different values for the fourth order diffusion constant $\alpha^{(4)}$ were analyzed as well. The density contours in the stagnation area for the four meshes is shown in Figure 6.8.

Numerical comparisons of the drag coefficient c_D and the stagnation density ρ_0 are shown in Table 6.1. Graphically, the stagnation density ρ_0 is presented in Figure 6.9 which is compared to the analytical value of 1.129726. The comparison of the convergence of the drag coefficient c_D is shown in Figure 6.9 and its analytic value is $c_D = 0.0$. The abscissa shows the inverse of the number of points per mesh. So, the result for an infinitely fine mesh can be extrapolated by extending each of the lines until they cross the ordinate. The vanishing influence of the artificial dissipation can also be observed

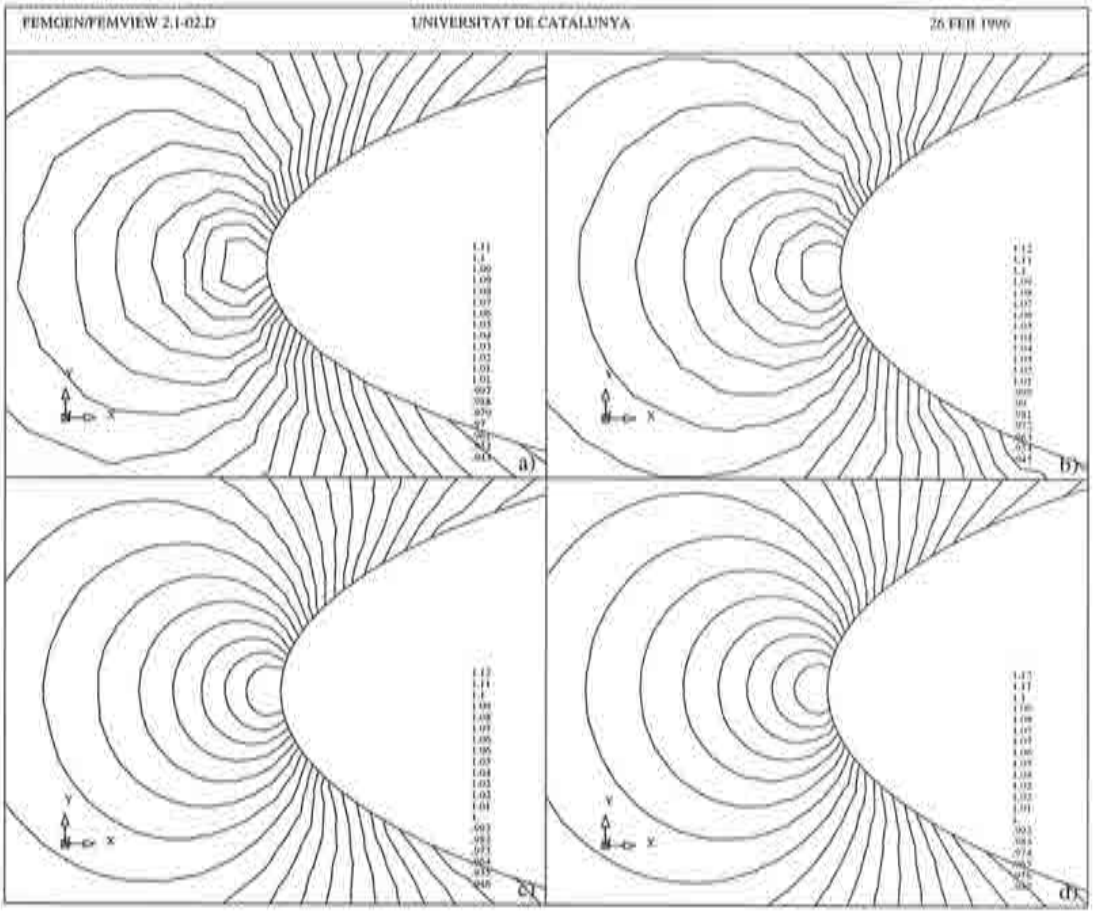


Figure 6.8 : Density contours in the stagnation area obtained from the different triangular meshes in Figure 4.51 using the Runge-Kutta Galerkin scheme for the inviscid test case: $M_\infty = 0.5$, $\alpha = 0°$.

as the meshsize goes to zero. This is demonstrated by the fact that the distance which separates the curves reduces. $h = 1$ in Figure 6.11 is the spacing in the stagnation area of the coarsest mesh of the four. Quadratic convergence is seen from the fact that if h decreases by one half, the error diminishes to one quarter. The history of convergence versus the time step for four different diffusion constants $\alpha^{(4)}$ using the third mesh of 4022 points is plotted in Figure 6.12. The convergence history for different meshes using the same $\alpha^{(4)} = 0.2$ is shown in Figure 6.13.

In summary for the Runge-Kutta Galerkin scheme, a stronger influence of the artificial diffusion constant $\alpha^{(4)}$ is observed, if compared to the Taylor-Galerkin scheme. Convergence is higher and accuracy is improved slightly for the drag coefficient c_D. The error in the stagnation density is generally lower except for $\alpha^{(4)} = 0.4$, where the error is approximately the same. The second order accuracy of the scheme is again manifested by the fact that as the mesh size h decreases by one half, the error in the stagnation density reduces to a quarter or more and the error in the drag coefficient reduces to a quarter for all meshes except for the fine mesh for which the error is nearly a third depending on $\alpha^{(4)}$.

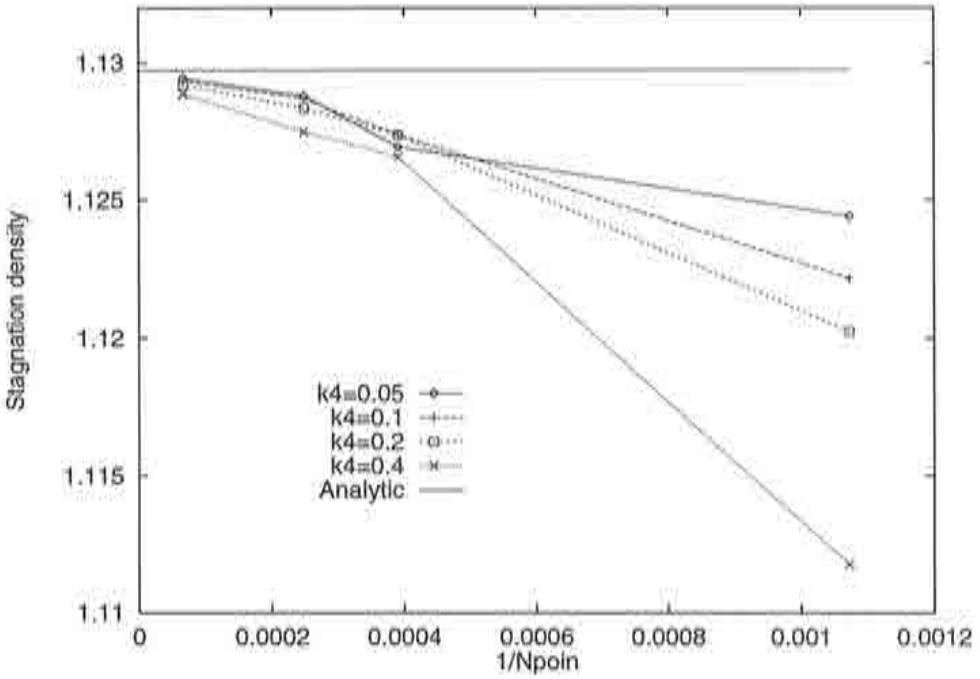


Figure 6.9 : Comparison of the stagnation density ρ_0 for four different triangular meshes and four different artificial diffusion constants k4 = $\alpha^{(4)}$ to analytical value of 1.129726 using the Runge-Kutta scheme. Plotted are the density values versus the the inverse of the number of points.

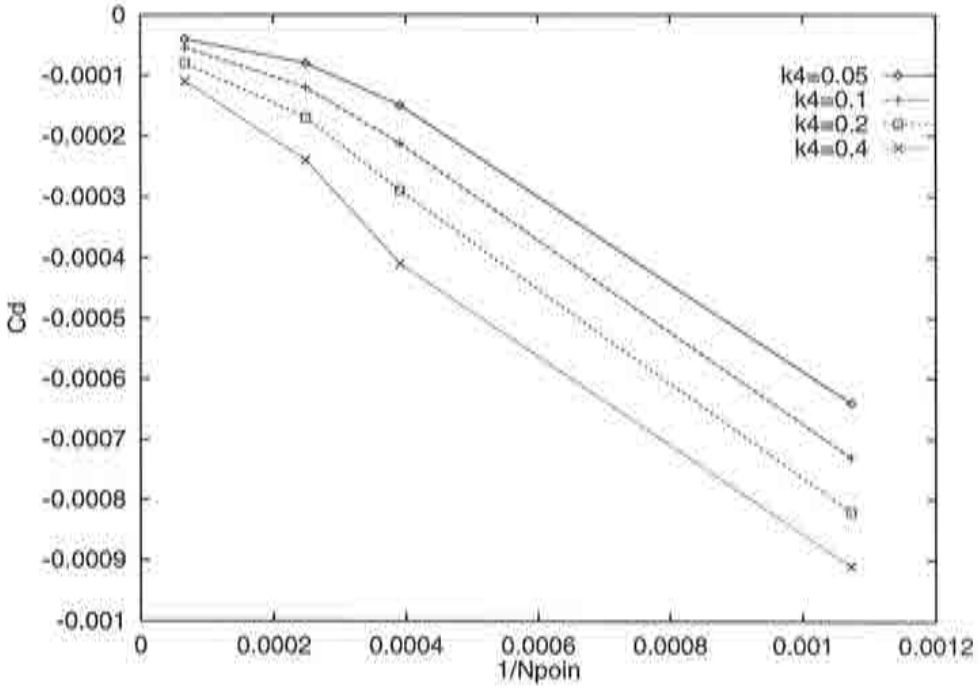


Figure 6.10 : Comparison of the drag coefficient for four different triangular meshes and four different artificial diffusion constants k4 = $\alpha^{(4)}$ using a Runge-Kutta Galerkin scheme. The analytical value is $c_D = 0$ for inviscid flows. The plots show the c_D versus the inverse of the number of points.

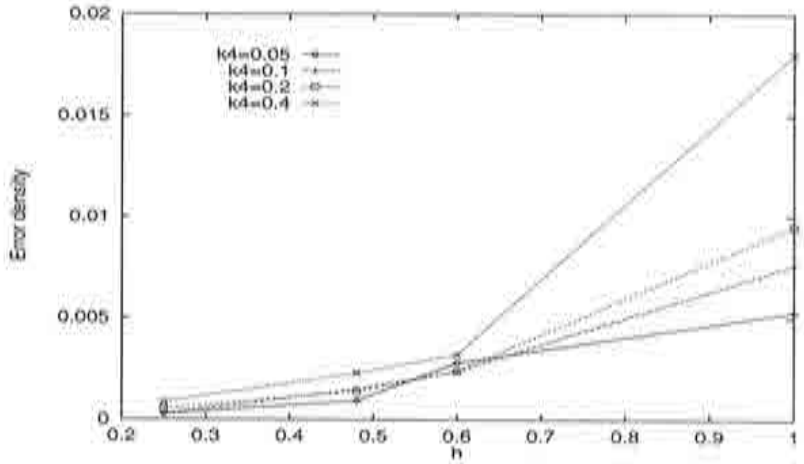


Figure 6.11 : Plot of the error of the stagnation density with respect to the mesh size h for the Runge-Kutta Galerkin scheme using different k4 = $\alpha^{(4)}$. The non dimensional value $h = 1$ is taken from the coarsest mesh.

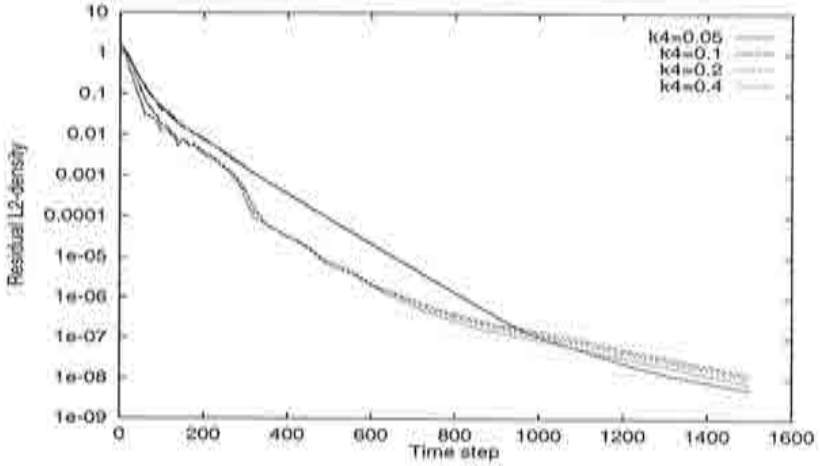


Figure 6.12 : Comparison of the convergence histories for four different diffusion constants k4 = $\alpha^{(4)}$ using the Runge-Kutta Galerkin scheme.

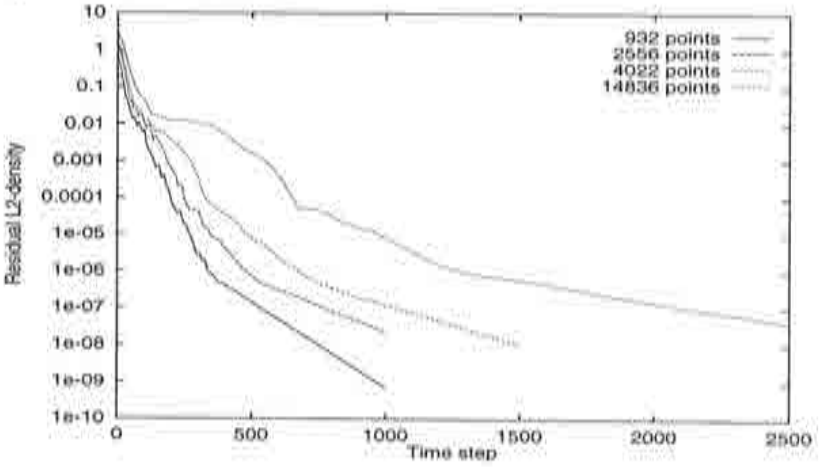


Figure 6.13 : Comparison of the convergence histories for the four different triangular meshes using $\alpha^{(4)} = 0.2$ using the Runge-Kutta Galerkin scheme.

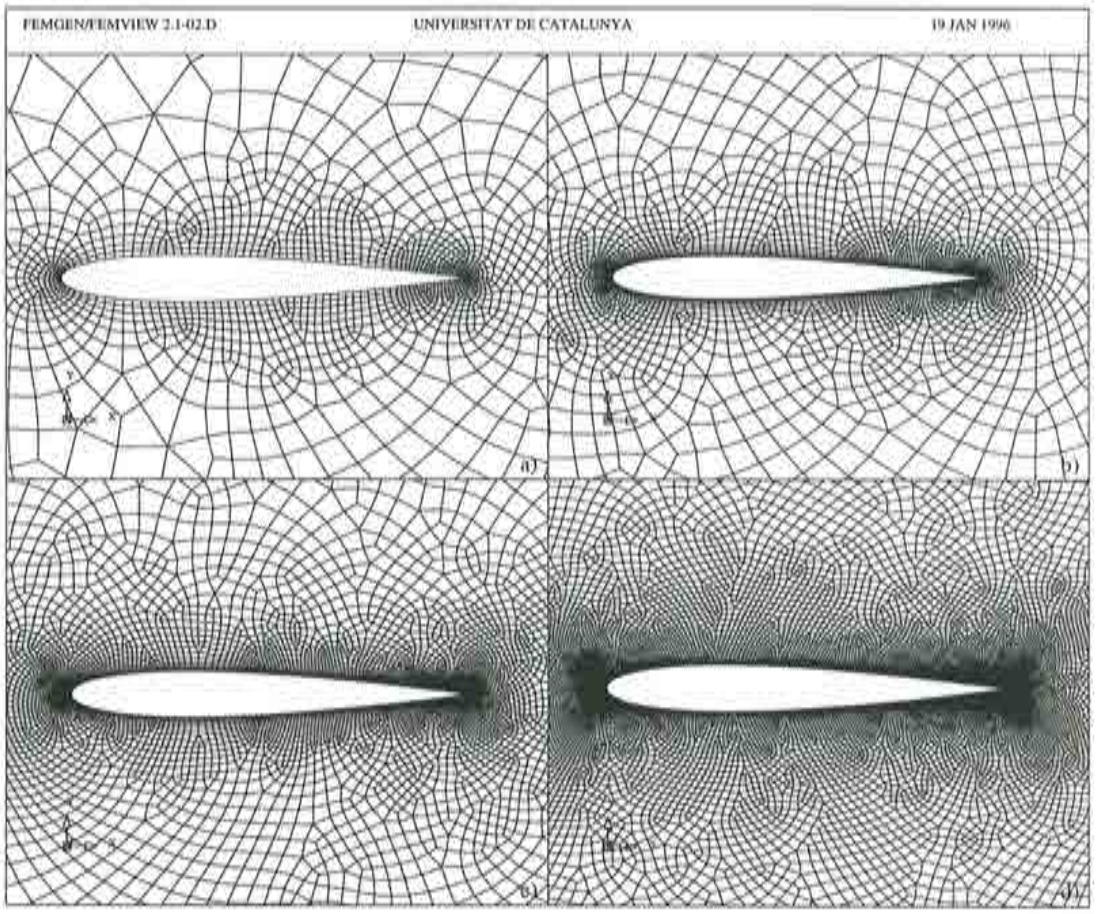


Figure 6.14 : Four quadrilateral meshes for the mesh convergence study of an inviscid flow around a NACA0012 profile ($M_\infty = 0.5$, $\alpha = 0°$): a) 2150 nodes and 2058 elements, b) 4412 nodes and 4288 elements, c) 8070 nodes and 7900 elements and d) 18974 nodes and 18701 elements.

Mesh Convergence of Quadrilaterals

The four quadrilateral meshes are shown in Figure 6.14. Solutions were obtained for three different values of $\alpha^{(4)} = 0.2$, 0.4 and 0.6 to also show the influence of the artificial diffusion. The density contours in the stagnation area for the four meshes can be appreciated in Figure 6.15.

Numerical comparisons of the stagnation density and the drag coefficient are shown in Table 6.2. Graphically, the stagnation density ρ_0 is presented in Figure 6.16 and compared to the analytical value of 1.129726. The comparison of the convergence of the drag coefficient c_D is shown in Figure 6.17. The vanishing influence of the artificial dissipation can again be observed as $h \to 0$.

The error in the stagnation density is shown in Figure 6.18 indicating quadratic convergence for as h decreases. The error in the stagnation density ρ_0 is again defined by eq. 4.73. $h = 1$ in Figure 6.18 is the spacing in the stagnation area of the coarsest mesh of the four, so quadratic convergence is also observed in this case.

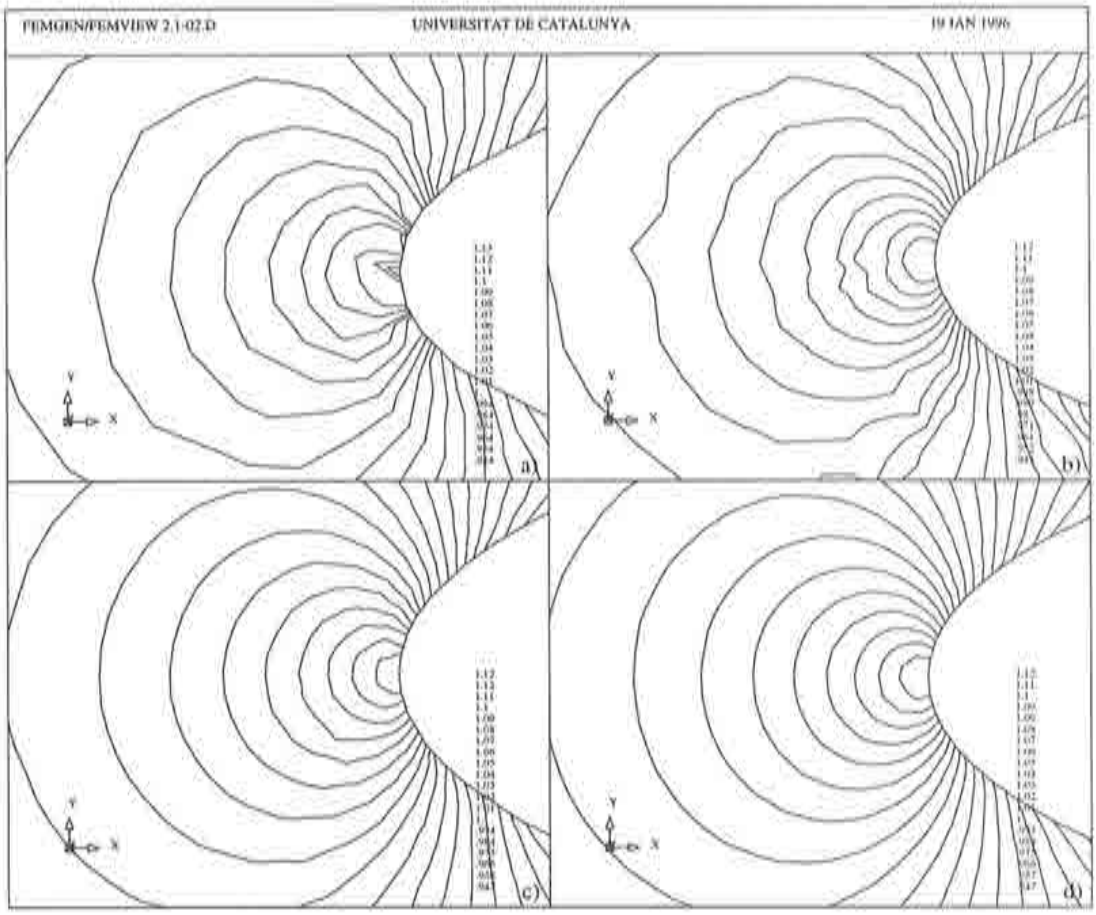


Figure 6.15 : Density contours in the stagnation area obtained from the respective quadrilateral meshes in Figure 6.14 for the NACA0012 profile for inviscid test case $M_\infty = 0.5$, $\alpha = 0°$ using the Runge-Kutta Galerkin scheme.

Again, a stronger influence on the accuracy of the artificial diffusion constant $\alpha^{(4)}$ is observed. Compared to triangles, the accuracy for quadrilaterals is slightly lower for this test case which is due to the lower quality of the unstructured quadrilateral meshes containing elements with high angles and strong deformation, whereas the quality of the triangular meshes is globally higher and much more regular. In almost all cases quadratic convergence is reached for both the error in c_D and ρ_0, demonstrating again second order accuracy of the algorithm.

The comparison of the convergence histories for quadrilaterals for different $\alpha^{(4)}$ is not plotted here, because similar conclusions can be drawn as for triangles. However, an interesting comparison is the direct comparison of the convergence of the residuals to that of triangles which is shown in Figure 6.19. Three different curves for similar accuracy are drawn which are: Taylor-Galerkin (TG) using triangles, Runge-Kutta Galerkin scheme (RK) with triangles and Runge-Kutta scheme with quadrilaterals. From that plot, it is observed that the convergence for RK time intergration scheme for triangles is superior that of TG triangles and RK quadrilaterals if compared to the time step. So, for similar accuracy quadrilaterals converge slower, because more number of

	$\kappa^{(4)} = 0.2$		$\kappa^{(4)} = 0.4$		$\kappa^{(4)} = 0.6$	
Npoin	c_D	ρ_0	c_D	ρ_0	c_D	ρ_0
2150	-0.00045	1.1432	-0.00068	1.1347	-0.00090	1.1294
4412	-0.00019	1.1398	-0.00032	1.1346	-0.00044	1.1320
8070	-0.00012	1.1339	-0.00018	1.1321	-0.00022	1.1309
18974	-0.00008	1.1310	-0.00012	1.1305	-0.00015	1.1301

Table 6.2: Comparison of the drag coefficient c_D and the stagnation density ρ_0 for 4 different quadrilateral meshes and 3 different diffusion constants $\alpha^{(4)}$ using the Runge-Kutta Galerkin scheme. The analytic results are $c_D = 0.0$ and $\rho_0 = 1.1297$.

nodes are used.

In terms of performance, the consumed cpu time is compared for each scheme for the convergence curves $\alpha^{(4)} = 0.2$. The residual drop in density versus the cpu time is shown in Figure 6.20. Again for the fact that more nodes are used, the convergence rate for quadrilaterals is slower than for triangles if the same rate of accuracy is reached. The best form to analyze this behavior is comparing the solution on a triangular mesh with that of a quadrilateral mesh using the same points which is presented in the following example.

Triangular versus Quadrilateral Mesh Using the Same Points

In order to create a good basis for comparison, a triangular and quadrilateral mesh using the same points has been used. The triangular mesh has been obtained from the structured quadrilaterals by subdivision of each element. In total 14106 points were used giving 27636 triangles or 13818 quadrilaterals. The points that were used for both meshes are shown in Figure 5.21 which have already been applied to resolve another test case.

The results now indicate that, in fact, the quadrilateral mesh is more cost efficient than triangles. In terms of accuracy, the $c_D = 0.000016$ for quadrilaterals and $c_D = 0.000039$ for triangles, the error in the stagnation density was 0.00040 for quadrilaterals and 0.00031 using the triangular mesh. Hence, accuracy is much better for the drag prediction but slightly lower for ρ_0. However, performance is now almost twice as high for the same amount of points and similar accuracy using quadrilaterals. Figure 6.21 demonstrates this for the residual decrease in the L2 norm of the density versus cpu time.

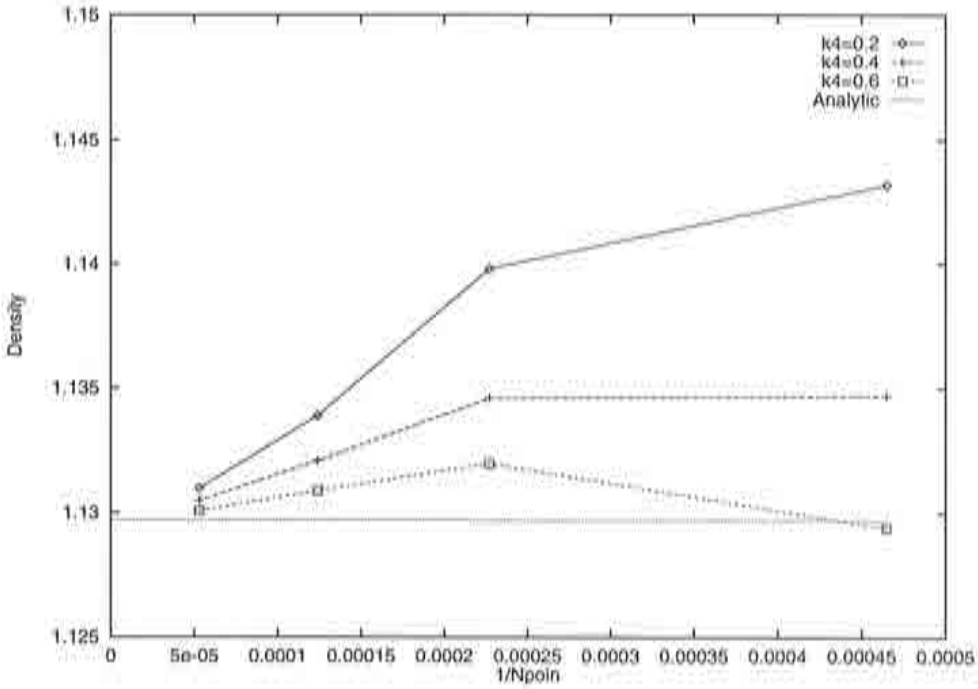


Figure 6.16 : Comparison of the stagnation density for four different quadrilateral meshes and three different artificial diffusion constants k4 = $\alpha^{(4)}$ to analytical value of 1.129726 using the Runge-Kutta Galerkin scheme. Plotted are the density values versus the the inverse of the number of points.

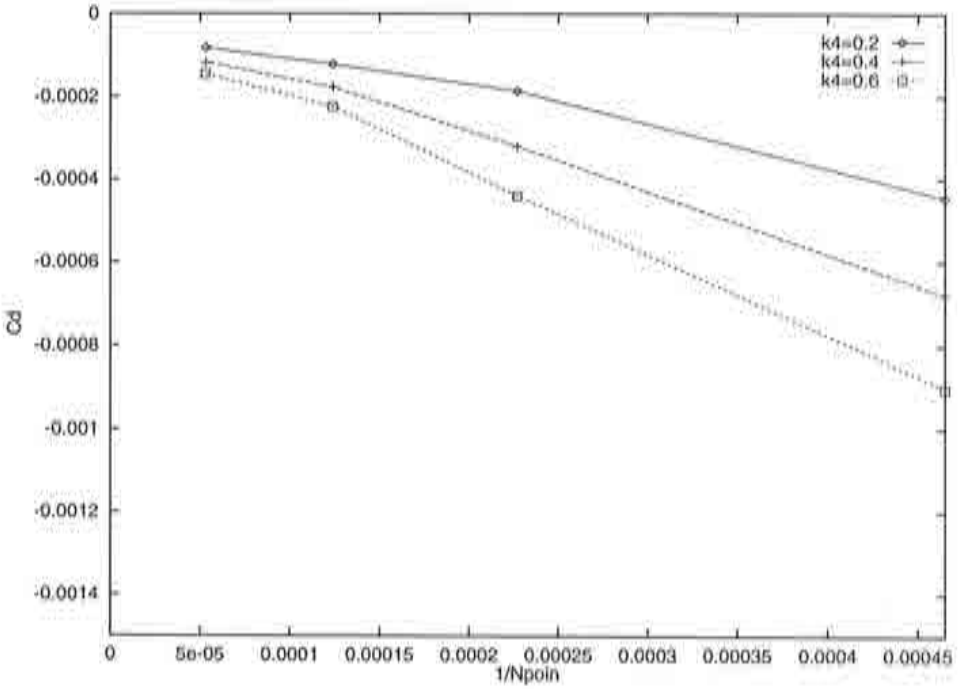


Figure 6.17 : Comparison of the drag coefficient for four different quadrilateral meshes and three different artificial diffusion constants to the analytical value of zero for inviscid flows using the Runge-Kutta Galerkin scheme. The plots show the c_D versus the inverse of the number of points.

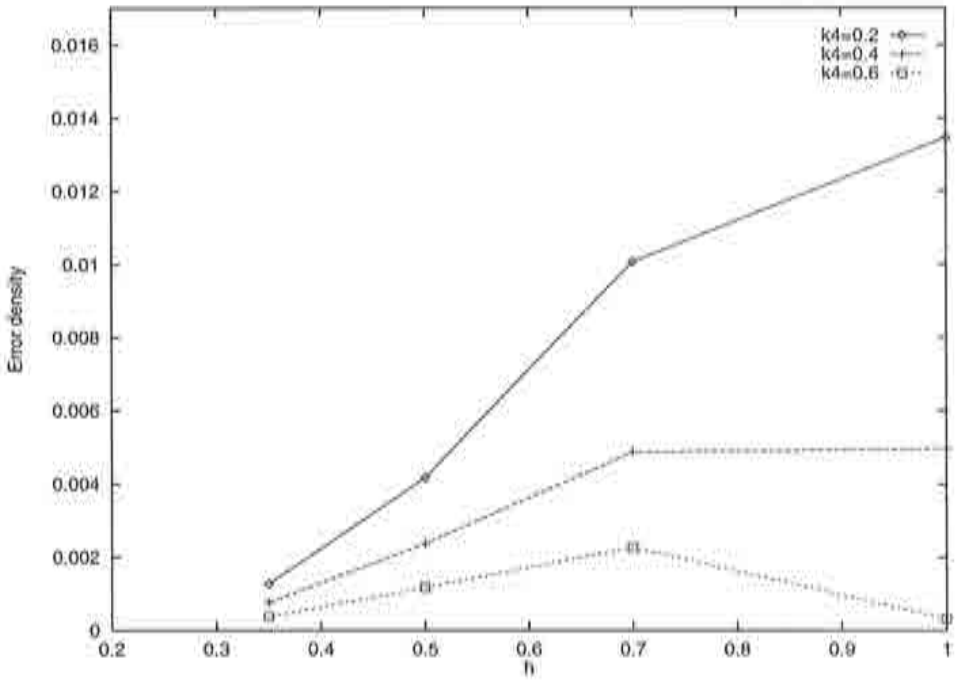


Figure 6.18 : Plot of the error of the stagnation density with respect to the mesh size h for four different quadrilateral meshes and three different k4 = $\alpha^{(4)}$ using the Runge-Kutta Galerkin scheme. The non dimensional value $h = 1$ is taken from the coarsest mesh.

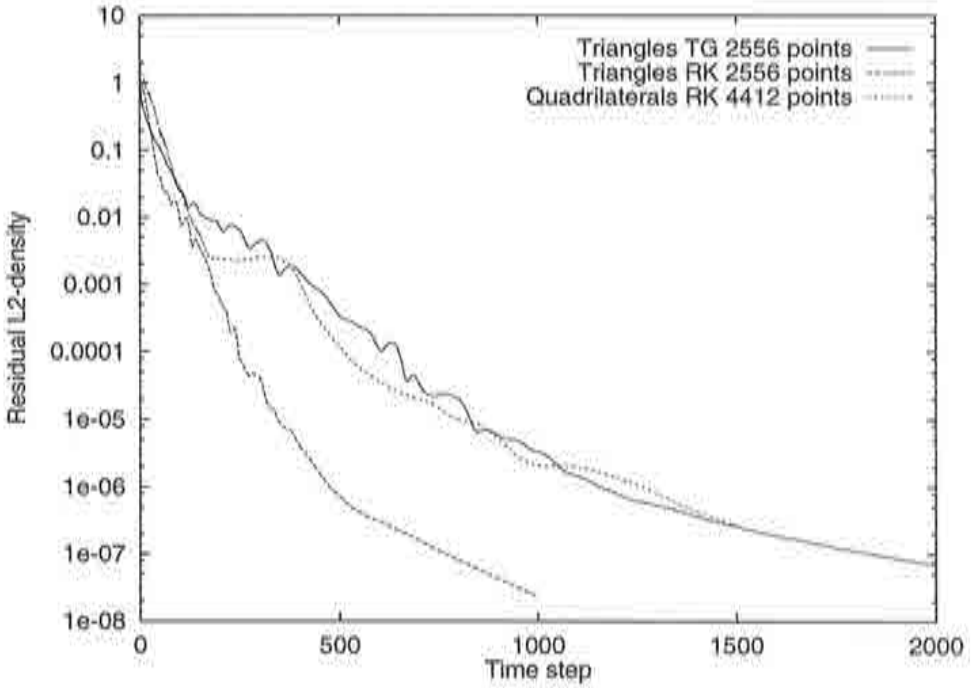


Figure 6.19 : Comparison of the convergence histories for three different schemes for similar accuracy: TG triangles 2556 nodes, RK triangles 2556 points and RK quadrilaterals 4412 points (TG: Taylor-Galerkin, RK: Runge-Kutta Galerkin) where $\alpha^{(4)} = 0.2$.

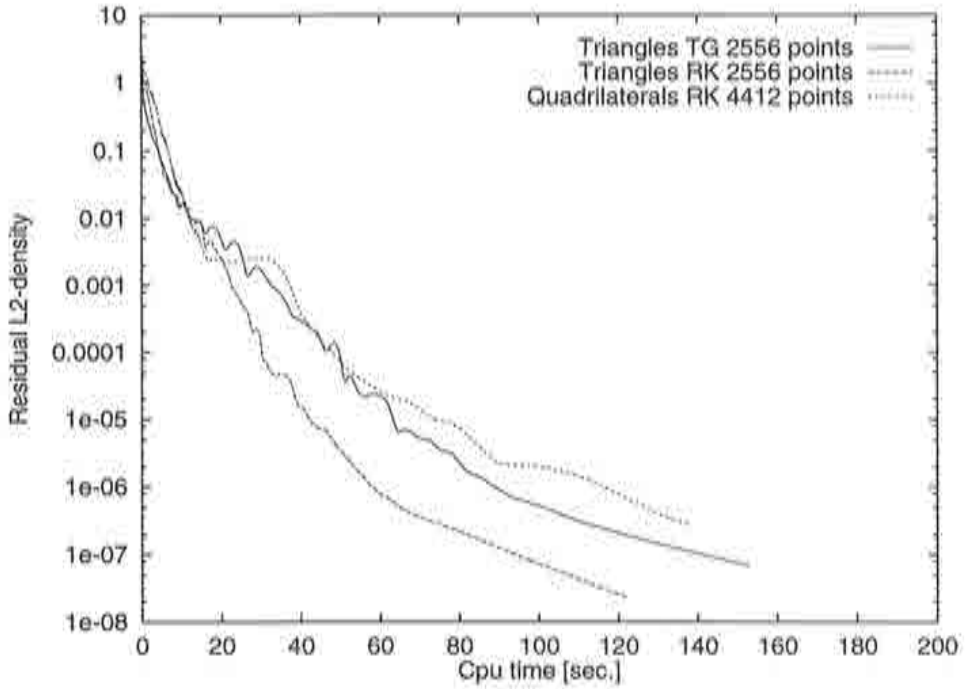


Figure 6.20 : Comparison of the convergence histories versus cpu time for three different schemes of similar accuracy: TG triangles 2556 nodes, RK triangles 2556 points and RK quadrilaterals 4412 points (TG: Taylor-Galerkin, RK: Runge-Kutta Galerkin) where $\alpha^{(4)} = 0.2$.

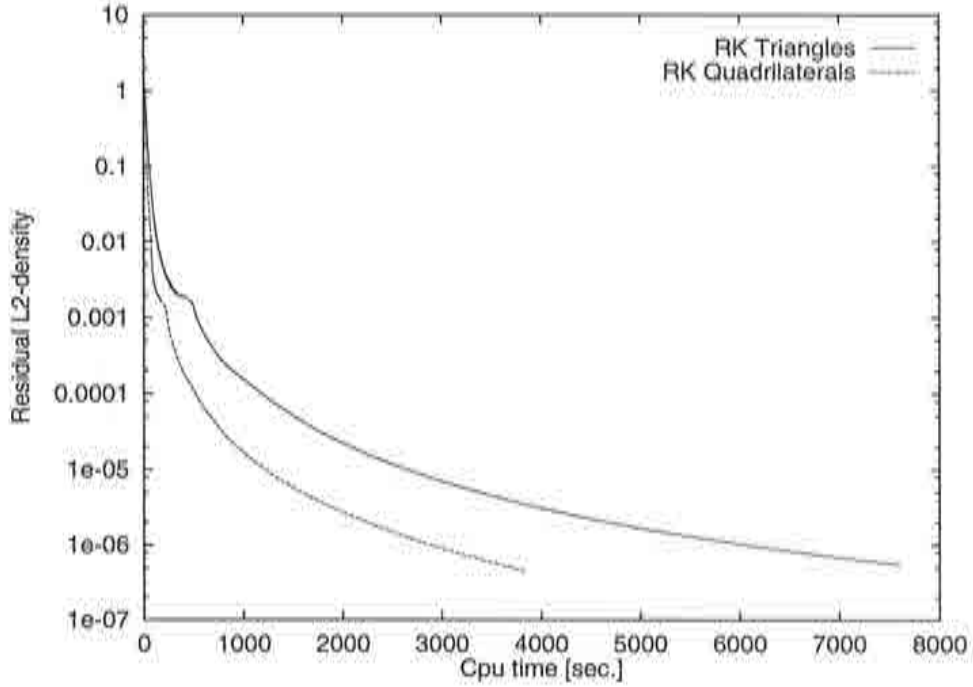


Figure 6.21 : Comparison of the convergence histories versus cpu time for triangles and quadrilaterals of similar accuracy using the same 14106 points for the Runge-Kutta Galerkin scheme.

6.4.3 Example Using Parallel Computing

Numerical experiments using the parallel computing environment of the Connection Machine CM-200 (CMSSL, FASTGRAPH (FG) and NOP) have been performed for the hypersonic viscous flow around a double ellipse and resolving the subsonic laminar viscous flow past a NACA0012 airfoil. Both test cases have been solved earlier in section 4.10.1.

Hypersonic Flow Around a Double Ellipse

Some results of the hypersonic test case around the double ellipse have been run using the same meshes as earlier, yielding nearly identical results and therefore are not repeated here. However, performance measurements are presented for the calculation of several 100 time steps using four different meshes of 5450, 17410, 31380 and 54866 elements. The number of elements was controlled by the variation of elements in the structured layer section.

		2048 Processors			4096 Processors		
Nelem	SGI	CMSSL	FG	NOP	CMSSL	FG	NOP
5450	0.75	0.41	0.26	0.53	0.29	0.19	0.33
17410	2.49	1.37	0.52	1.55	0.89	0.33	1.01
31380	4.6	2.21	0.69	2.57	1.55	0.45	1.81
54866	7.9	3.31	0.97	3.79	2.25	0.65	2.78

Table 6.3: Performance comparison of different communication routines for different numbers of processors with a serial Silicon Graphics Indigo workstation (SGI). The values are in seconds per time step.

The main indicator that was chosen for comparison is cpu-time per time step because the number of time steps to converge to steady state is different for each mesh and depends particularly on the elements sizes. Also the setup effects for the different communication routines are ignored.

The performance study involves three available communication possibilities of the Router data transfer system in the CM-200 in comparison to the speed of Silicon Graphics workstations. Moreover, the use of 2048 and 4096 processors of the CM-200 are also compared. We had the possibility to use 4096 processors for a short period of time, based on a rental agreement. So, only the following results were run using 4096 processors. A summary of the results can be obtained from Table 6.3.

A comparison in terms of speed up ratios versus one single processor are not possible for this type of computer architecture. A different method comparing with a known serial processor is made instead. The speed increase for different Router constellations and number of processors can be observed from that table as well as from Figures 6.22 and 6.23. Note the extreme difference of the FG utility. The CM-200 shows a declining slope as the number of elements increases, whereas the serial system presents a linear

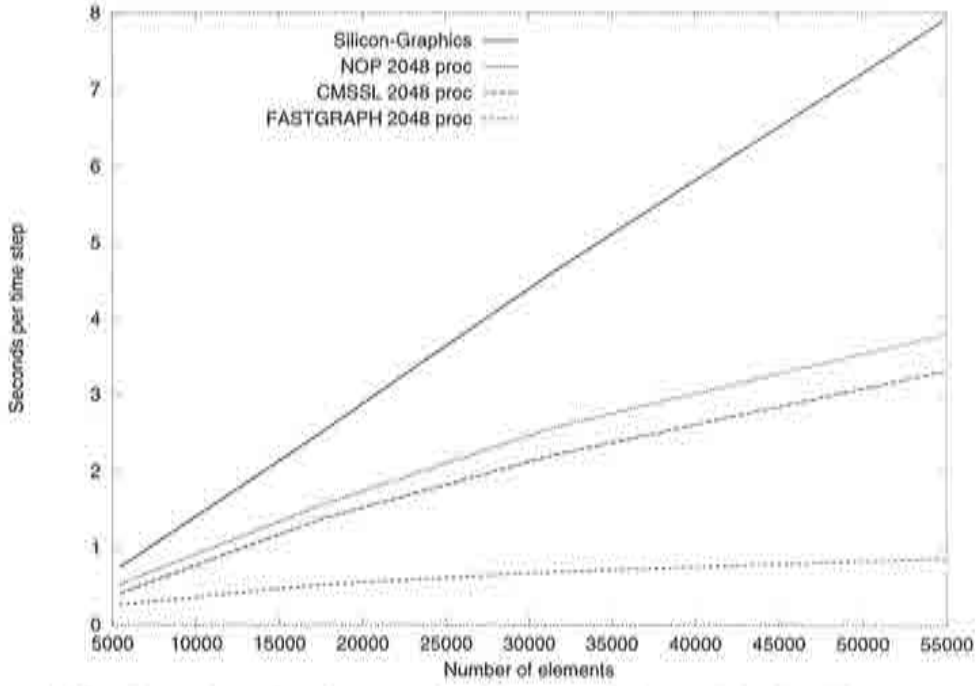


Figure 6.22 : Comparison of performance for different mesh sizes and for four different computing configurations (serial, CMSSL, FG, NOP). The results for the parallel system were performed using 2048 processors.

relationship of time versus number of elements. This is devoted to the increase of communication efficiency as the mesh size increases. This non-linear relationship can best be analyzed if we compute the ratio, speed factor, which is the cpu-time of the serial computer divided by the cpu-time of the FG utility, tabulated in Table 6.4. Figure 6.24, shows this effect graphically for 2048 and 4096 processors.

However, doubling the number of processors does not provide double speed in the parallel environment. This is again due to the loss in efficiency, especially in communication. This limits the potential power for parallel systems when scaled because of scalar operations performed within the parallel system, also known as Amdahl's law [1]. (For the CM-200, 2048 processors are taken as one processing unit for the calculation using the formula of Amdahl's law.)

Nelem	2048 Processors	4096 Processors
5450	2.88	3.95
17410	4.79	7.55
31380	6.67	10.22
54866	8.14	12.15

Table 6.4: Ratio of the speed factor of the cpu time per time step for serial workstation over parallel system using FG.

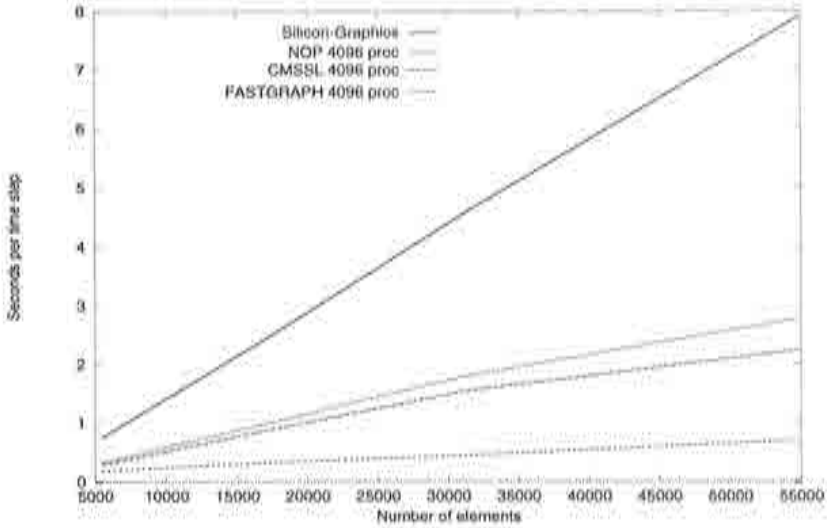


Figure 6.23 : Comparison of performance for different mesh sizes and for four different computing configurations (serial, CMSSL, FG, NOP). The results for the parallel system were performed using 4096 processors.

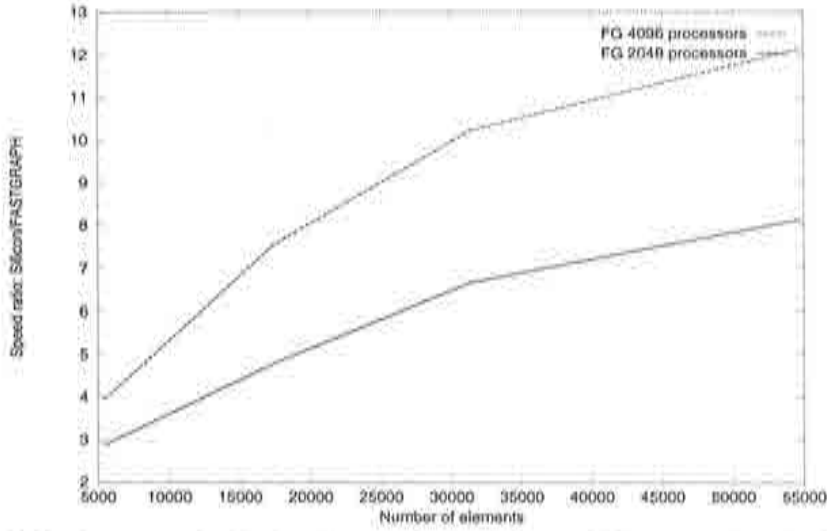


Figure 6.24 : Performance plot showing the speed ratio of the parallel system using the FASTGRAPH communication routine (FG) versus the serial cpu time The speed ratio is the cpu time per iteration used by the SGI workstation divided by the cpu time of the FG utility.

Subsonic Viscous Flow around NACA0012

This test case again resolves laminar Navier-Stokes solution in two dimensions around a NACA0012 airfoil ($M\infty = 0.5$; $Re = 5000$; α=0°) as referenced in [7]. A reference mesh was used for resolving this test case obtained from a database containing 14106 nodes and 27636 elements. This mesh, although unstructured, has been obtained by subdivision of a structured quadrilateral mesh with a high aspect ratio in the wake region. In addition, the results were prepared in digital form to comply with the proposed database format by the Institut Nationale de Recherche en Informatique et en Automatique (INRIA) and presented for comparison with other contributors at a workshop held on 11.07.1995 at INRIA Sophia-Antipolis, France.

The contribution to the database includes two dimensional results (Mach number, pressure, density etc.) as well as some 1D curves (pressure coefficient c_p, skin friction coefficient c_f, convergence history, etc).

Accuracy

The aim of this test case was to compare the solution for the Taylor-Galerkin scheme with the very well documented reference test case [7]. Since the blended 2nd and 4th order artificial diffusion model is very similar to the one of the reference solution, it was very easy to establish a basis for comparison of the results.

All graphs are presented for the seemingly best computation which used $\alpha^{(4)} = \frac{1}{120}$ if not otherwise stated. The calculations show that the overall features of the flow are very similar, if not identical, to the results obtained from the reference solution [7]. In the two dimensional plots like Mach number contours (Fig. 6.25) around the airfoil or velocity vectors along the trailing edge (Fig. 6.26) no differences can be visualized. Also, the pressure coefficient (Fig. 6.27) and the separation point of 82.7% chord agrees extremely well with the reference solution on unstructured meshes (82.4%) given roughly the same values for the fourth order diffusion coefficient. Fig. 6.28 shows the location of the separation point as a function of the iteration time step for two different coefficients $\alpha^{(4)}$: $\frac{1}{60}$ and $\frac{1}{120}$. Thus, as $\alpha^{(4)}$ is increased the location of the separation moves toward the rear of the airfoil. Also the pressure drag coefficient shows a very nice agreement with 0.02286 compared to 0.0228 of [7].

From the 2D analysis during the workshop, parallel pressure isolines and a clear definition of the recirculation area were observed which demonstrate the quality of the results. Note also the parallel pressure lines without oscillations in the wake (Fig. 6.29) which underline the high quality of the results. However, the main difference that was observed at the workshop was comparing the skin friction coefficient c_f with the reference results and other partners (Fig. 6.30). Especially near the leading edge the peak value of almost $c_f = 0.15$ has not been obtained using this scheme. Most other contributors had similar results which underpredict the peak. A reason for this is could be the different meshes that have been used. The normal mesh spacing in Jameson's mesh is about 0.0002 chords, whereas in the mesh used for this calculation it is roughly 8 times higher (0.0017 chords).

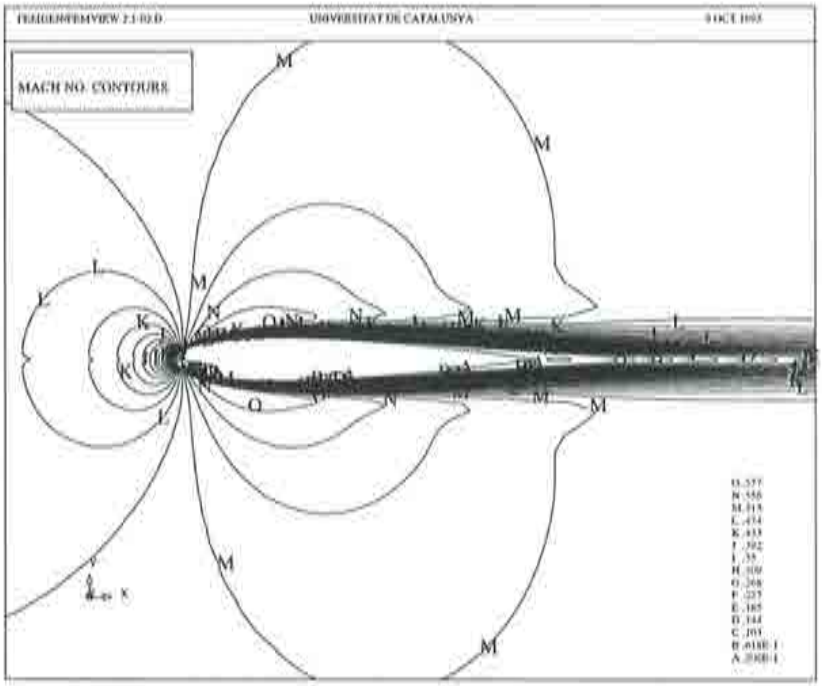


Figure 6.25 : Mach number contours for the solution of viscous flow around a NACA0012 airfoil (Re=5000, M_∞=0.5, α=0°) on the common unstructured mesh with 14106 nodes using the TG scheme on the CM-200.

In summary, the results using a parallel computing environment show a successful implementation of the Taylor-Galerkin scheme using triangular meshes. It has been shown that accuracy is not sacrificed.

Parallel performance

The other objective of the workshop was not only to assess the accuracy of different flow solvers, but also to give a measure of efficiency. In order to give an estimate of the efficiency of each solution scheme, a good indicator is the convergence curve for which a given accurate solution has been obtained. In this context it is necessary to show, for instance, the residual drop in orders of magnitude versus work units in cpu-time. Thus, Fig. 6.31 shows the convergence history of the energy-residual versus cpu time where convergence to about 5 orders of magnitude can be observed. Total time to reach a steady state where no more significant change is appreciable can be reached for this mesh in about 30 minutes for $\alpha^{(4)} = \frac{1}{60}$ and 90 minutes $\alpha^{(4)} = \frac{1}{120}$.

Due to the nature of using unstructured finite element meshes, a large amount of gather, scatter and scatter-add operations can be expected. Thus, we can expect a strong amount of interprocessor communication for this type of problems.

A comparison of the performance, ie. speed-up rates, is very difficult for this type of parallel architecture, because the comparison with one processor would be meaningless.

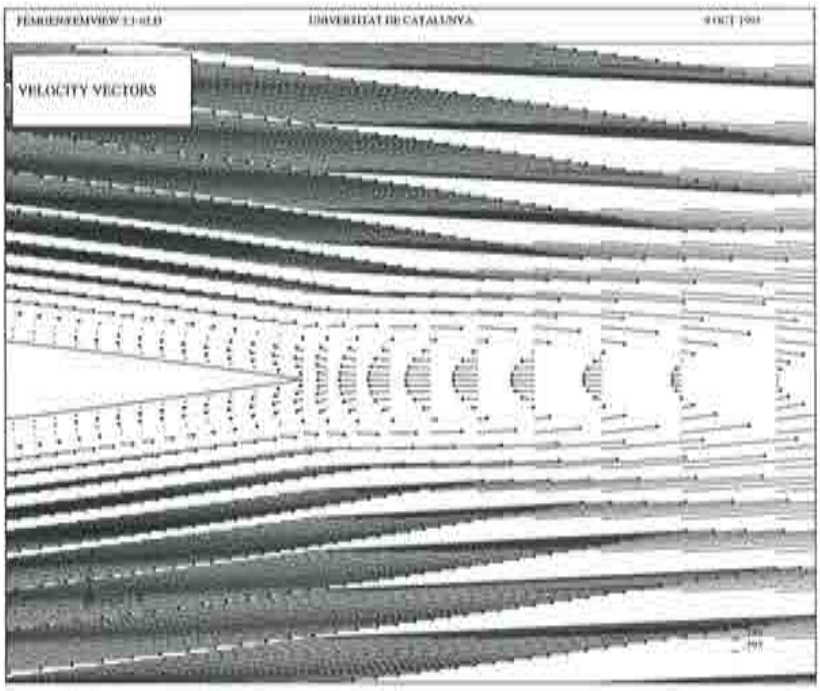


Figure 6.26 : Velocity vectors at the trailing edge for the solution of viscous flow around a NACA0012 airfoil (Re=5000, M_∞=0.5, α=0°) on the common unstructured mesh with 14106 nodes using the TG scheme on the CM-200. Note the recirculation at tip of the trailing edge.

A possible comparison can be the claimed peak performance values of a serial processor, like the MIPS R8000, running at 300 Mflops peak. However, the comparison of peak flop rates is problem and system dependent, therefore the following conclusions can not be generalized.

The CM-200 has a peak performance of about 640 Mflops using 2048 processors. For the present calculations, the communication required between processors consumes nearly 40% of the total computation. So for the reference mesh, we obtain 0.4515 sec. per time iteration. A similar version of the code which was optimized for the R8000 yielded for the reference mesh a value of 0.685 sec per time iteration.

Comparing the results, the losses in performance (ie. due to communication) become visible. The ratio of peak performances for the parallel system over serial computation is 2.13, the ratio of consumed cpu time (parallel over serial) is 1.51. Hence, the parallel version performs less efficient using these reference values than the corresponding serial version. The loss of roughly 30% due to parallelization seems very low, considering that we must communicate information between 2048 processors. As mentioned earlier, the overall performance of the program can be improved by elaborating a better strategy for communication [15], thus reducing that cost.

In [18], a comparison of different parallel computational platforms (CM-200, iPSC-860 and KSR1) using unstructured meshes to explicitly resolve viscous flows is presented. There, the computational performance (75 MFlops) of the scheme runs well

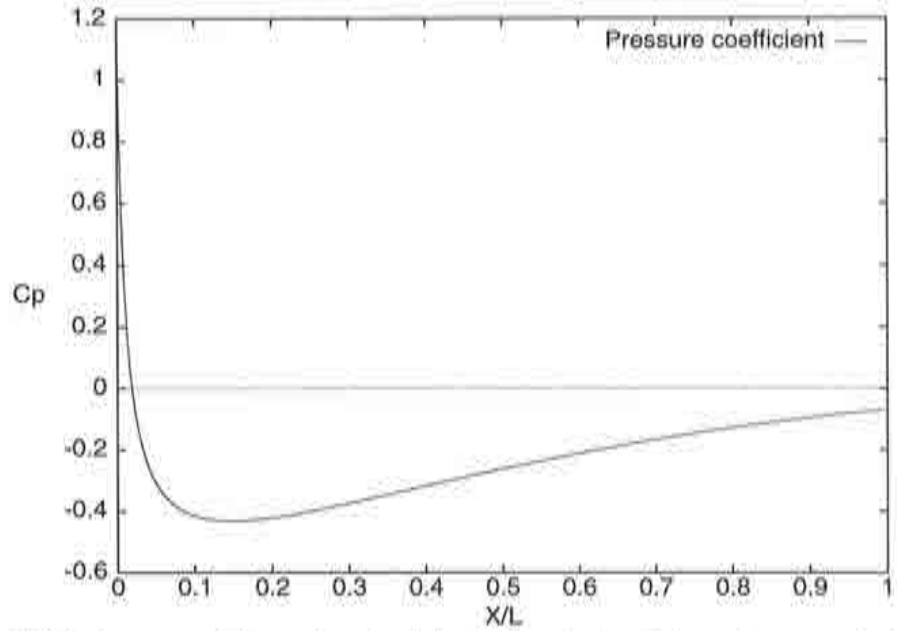


Figure 6.27 : Pressure coefficient along the airfoil for the solution of viscous flow around a NACA0012 airfoil (Re=5000, M_∞=0.5, α=0°) on the common unstructured mesh with 14106 nodes using the TG scheme on the CM-200.

below the claimed peak performance on the CM-200 and comparisons were made only on the basis of consumed cpu time. The efficiency of the procedure using the CM-200 was the lowest for the given platforms. However, each of the measures which exist for comparing performance on the CM-200 to other platforms is subjective and depends very much on the scheme, the implementation and the programming effort.

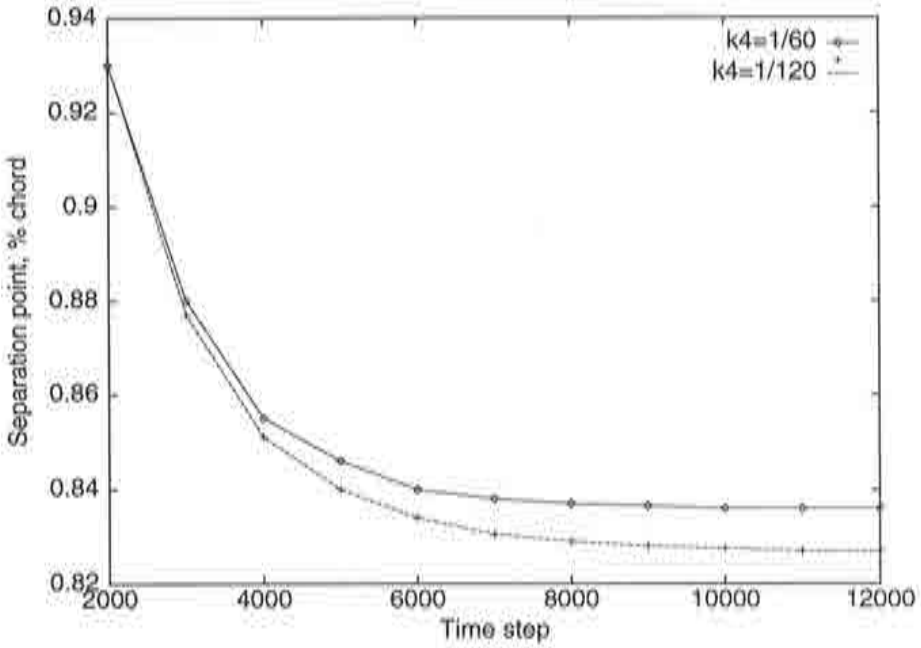


Figure 6.28 : Influence of $\alpha^{(4)}$ on the evolution of the separation point for the solution of viscous flow around a NACA0012 airfoil (Re=5000, M_∞=0.5, α=0°) on the common unstructured mesh with 14106 nodes using the TG scheme on the CM-200.

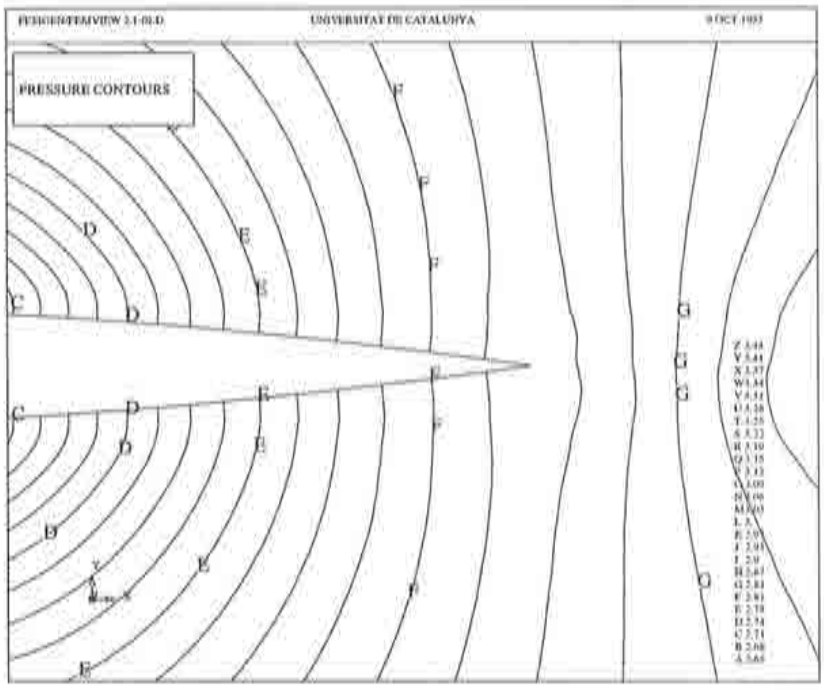


Figure 6.29 : Pressure lines in the wake for the solution of viscous flow around a NACA0012 airfoil (Re=5000, M_∞=0.5, α=0°) on the common unstructured mesh with 14106 nodes using the TG scheme on the CM-200. Note their parallel position to each other in the wake.

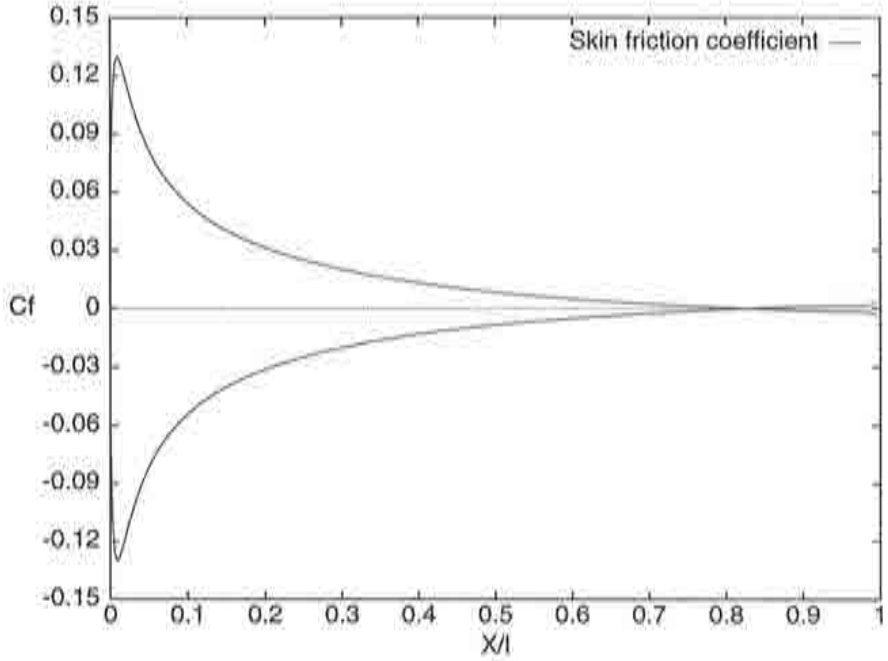


Figure 6.30 : Skin friction coefficient along the airfoil for the solution of viscous flow around a NACA0012 airfoil (Re=5000, M_∞=0.5, α=0°) on the common unstructured mesh with 14106 nodes using the TG scheme on the CM-200.

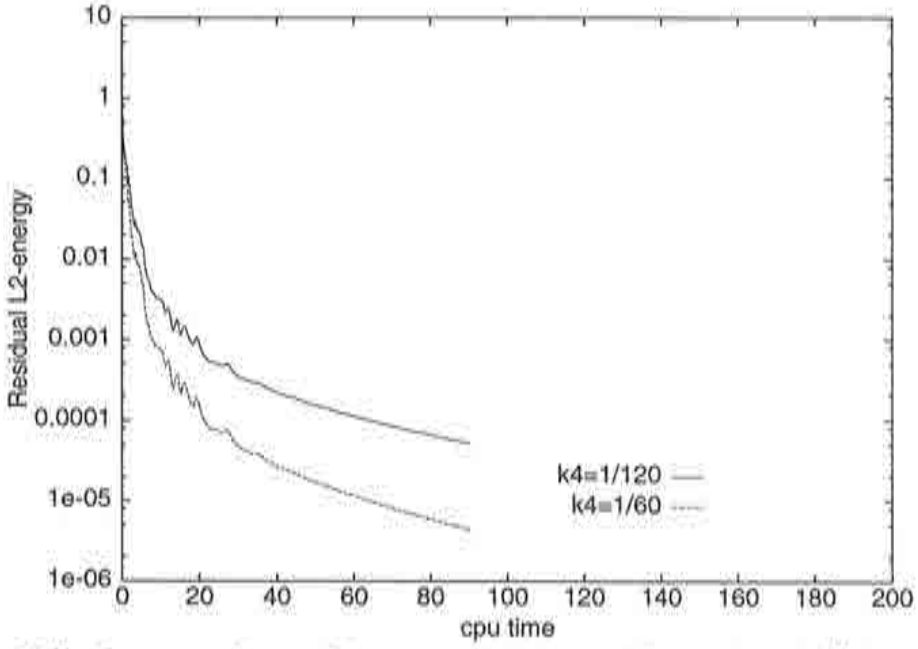


Figure 6.31 : Convergence history of the energy residual for two different values of $\alpha^{(4)}$ for the solution of viscous flow around a NACA0012 airfoil (Re=5000, M_∞=0.5, α=0°) on the common unstructured mesh with 14106 nodes using the TG scheme on the CM-200.

6.5 Conclusions

The accurate and robust Taylor-Galerkin scheme described in section 4.2 to solve Euler and laminar Navier-Stokes flow solver in 2D on unstructured meshes has been improved in terms of accuracy and performance by introducing the Runge-Kutta Galerkin procedure. Also, the Taylor-Galerkin scheme has successfully been implemented on a massive parallel computing environment with 2048 and 4096 processors yielding a strong speed up.

For the Runge-Kutta Galerkin scheme, accuracy is demonstrated by solving a typical test case for the Euler equations in 2D for compressible flow conditions. Again, a mesh convergence study using the Runge-Kutta Galerkin scheme is performed for different artificial diffusion constants. The solutions are compared to the original Taylor-Galerkin scheme in terms of accuracy and performance, both for triangles and quadrilaterals. The result is that superior performance for a given accuracy is achieved by using higher order time integration. Also, the use of quadrilaterals improves the cost efficiency provided that meshes (triangles and quadrilaterals) of similar quality are used.

The parallel computations for the test case for the solution of laminar viscous flow were performed on the massive parallel computer, CM-200, and are compared to a proven reference solution [12]. Again, the accuracy is high and performance is very good considering that the CM-200 with 2048 processors is the smallest of this computing system. The disadvantages that were encountered are mainly due to hardware limitations such as availability, service and scalability. Three years ago, when this work was initiated, this contribution to a workshop was to identify the possibilities of exploiting SIMD machines within CFD calculations on unstructured meshes.

The conclusion that can be drawn from this work in the field parallel computing is that the use of SIMD machines is feasible and that the speed up compared to serial machines is impressive, but the scalability of this kind of architecture has its limitations both in cpu performance and problem size, especially when using irregular grids. The communication overhead remains large and convergence acceleration techniques such as multigrid will be difficult to implement on such a system. The complexity of the multigrid strategy will complicate the current data structure and communication.

Another important fact is that the availability and service of CM-200 can not be guaranteed in the future because of problems by the hardware manufacturer. Thus, the author's recommendation must go towards strategies in MIMD environments. This would make necessary the use of domain decomposition methods to divide the load among different processors if a significant performance increase is to be achieved on unstructured meshes in the future. Many tools already exist to help reduce the programming effort in domain decomposition and communication [21, 22, 1]. In addition, these types of programs are less machine dependent, because communication standards like, MPI or PVM are available on many architectures, whereas the environment of the CM-200 uses special communication routines and different data structures.

References

[1] Knight D.D., "Parallel Computing in Computational Fluid Dynamics", AGARD FDP Symposium, Seville, Spain, 2-5 Oct 1995

[2] Jameson, A., Schmidt, W., and Turkel, E. "Numerical simulation of the Euler equations by finite volume methods using Runge-Kutta time stepping schemes" AIAA paper 81-1259. AIAA 5th Computational Fluid Dynamics Conf., 1981

[3] Hirsch, C. "Numerical Computation of Internal and External flows", Volume 2, Wiley, Chichester, 1989

[4] Jameson, A., "Multigrid Algorithms for Compressible Flow Calculations", MAE Report 1743, Lecture given at 2nd European Conference on Multigrid Methods, Cologne, Oct. 4, 1985

[5] Mavripilis, D.J. "Multigrid Solution of the Two-Dimensional Euler Equations on Unstructures Triangular Meshes", AIAA Journal, Vol. 26, July 1988, pp. 824-831

[6] Jameson, A., Baker T.J., "Improvements to the Euler Aircraft Method", AIAA Paper 87-0452, AIAA 25th Aerospace Sciences Meeting, Reno, 1987

[7] Mavripilis P., Jameson A., Martinelli L., "Multigrid Solution of the Navier-Stokes Equations on Triangular Meshes", ICASE Rep. No 89-11, February 1989.

[8] Peraire, J., Peiro, J., Morgan K. "A 3D Finite Element Multigrid Solver for the Euler Equations", AIAA Paper 92-0449, 1992

[9] Agui J. "Introducció a la Supercomputació sota entorn SP2", Short Course, CESCA, Barcelona, 4 May 1995

[10] Peraire, J. "A finite element method for convective dominated flows", PhD Thesis at University College of Swansea, 1986

[11] Zienkiewicz O.C., Taylor 'The Finite Element Method' 4th Edition, Volume 2,Mc Graw Hill, 1991

[12] Fischer, T.,Codina R., Miquel, J. and Oñate E. "Adaptive finite element computations for viscous high speed flows", Proceeding of "Finite Elements in Fluids", Eds. K. Morgan, E. Oñate, J. Périaux, J. Peraire and O.C. Zienkiewicz, 1993

[13] Lapidus A. "A Detached Shock Calculation by Second Order Finite Differences". J. of Computational Physics 2, 154-177, 1967

[14] Jameson A., Baker T.J., Weatherill N.P. "Calculation of Inviscid Transonic Flow over a Complete Aircraft", AIAA Paper 86-0103, AIAA 24th Aerospace Sciences Meeting, Reno, 1986

[15] Farhat, C., Sobh, N., and Park, K.C. "Transient finite element computations on 65536 Processors: The Connection Machine", Int. J. for Num. Meth. in Eng., Vol. 30, 27-55, 1990

[16] Farhat, C., Fouzi, L. and Lanteri, S. "Two-dimensional viscous flow computations on the Connection Machine: Unstructured meshes, upwind schemes and massively parallel computations", Comp. Meth. in Appl. Mech. and Eng., 102, 61-88, 1993

[17] "CMSSL Release Notes for the CM-200", Version 3.0, Thinking Machines Corporation, Cambridge, Massachusetts, 1992

[18] Lanteri S., Farhat, C. "Viscous Flow Computations on MPP Systems: Implementation Issues and Performance Results", Society for Industrial and Applied Mathematics, 1993

[19] Cante J.C., Joannas D., Oliver J., Oller S. "Experiences in massive parallel computations in a finite element context", IT-95 CIMNE, May 1993

[20] Fischer, Oñate E. and Miquel, J. "Finite element analysis of high speed viscous flows in a massive parallel computer", Proceedings of "Computational Fluid dynamics '94, Stuttgart", Eds. S. Wagner, E.H. Hirschel, J. Périaux, J. Peraire and E. Piva, 1994

[21] Farhat, C., Lesoinne M. "Automatic Partitioning of Unstructured Meshes for the Parallel Solution of Problems in Computational Mechanics", Int. J. for Num. Meth. in Eng., Vol. 36, 745-764, 1993

[22] Farhat, C., Simon H.D. "TOP/DOMDEC: A Software Tool for Mesh Partitioning and Parallel Processing", CU-CSSC-93-11, College of Engineering at University of Colorado, May, 1993

Chapter 7

Conclusions and Future Work

The objective of this monograph was to present a methodology which addresses two of today's CFD problems, 1) the cost efficient resolution of flow problems and 2) the difficulties in mesh generation. A strong emphasis was put on the accuracy of the scheme and the presented results.

1) Cost efficiency relates assured accuracy to performance. In this context, this monograph has provided solutions for a wide range of compressible flow applications yielding precise results using the Taylor-Galerkin procedure. The basic procedure has been successfully enhanced to cover different types of flow constellations in 2D, 3D and axisymmetric flows for subsonic, transonic and supersonic flows. Validation of the program was done by comparing results to analytical, numerical as well as experimental data. Performance was enhanced by incorporating the Runge-Kutta time integration procedure, together with convergence acceleration techniques such as enthalpy damping and residual smoothing. Moreover, the Taylor-Galerkin scheme was implemented on a parallel computing environment, the Connection Machine CM-200, yielding an additional speed increase.

2) The difficulties of mesh generation programs have lead to investigate a solution algorithm based only on clouds of points, in which arbitrary points are locally selected to form the interpolation domain. This relieves the generation of fixed connectivities forming non overlapping elements. The initial scheme has been compared to the classical solution of the steady 1D convection-diffusion equation using an explicit time marching scheme. Exact nodal values were obtained as the scheme is equivalent to the Taylor-Galerkin scheme for three noded clouds. Nearly exact nodal results resulted for use with Gauss weighting functions for $n > 3$. Using weighting functions, the order of the scheme can also be retained if $n > 3$, whereas a deterioration of accuracy is reached if no weighting functions are employed.

This so called finite point method has been extended to the solution of some compressible flow problems in 2D. The results for subsonic inviscid and viscous flows showed very good results. Especially, the need for weighting functions, also for the diffusion terms, is vital for the guarantee of accuracy. Many possibilities using linear and quadratic basis functions have been also tested and compared in 1D and 2D examples. Difficulties were encountered when solving transonic and supersonic flows as the scheme

does not satisfy the Rankine-Hugeniot condition. Hence, this problem of conservation must be resolved in future research.

Future work

This work may be the initiative to extend some of the ideas presented.

1) It was shown that the algorithm is second order accurate, but only for smooth meshes. However, especially using quadrilateral elements in 2D or tetrahedral elements in three dimensions, smooth unstructured meshes are difficult to control. Therefore, a line of future work may be the presentation of a scheme which is second order accurate no matter what mesh is used. Another form of improving the accuracy is the better control of the artificial dissipation which is added to the scheme.

2) Performance enhancement techniques such as multigrid or parallel computations can be implemented. The multigrid method achieves high convergence properties and the use of MIMD parallel computation including domain decomposition is also a way to improve performance of the scheme.

3) The concept of clouds of points needs the modification to guarantee conservation especially in those parts of the flows where strong gradients are present. Also, the selection of points can be optimized, because the criteria of quadrants selects more points than necessary or is not sufficient for quadratic basis functions in 2D.

4) Finally, the proposed methodology for the solution of flows using points can only exploit the advantages in mesh generation if a point generator, adaptivity and a 3D version is operative. Still a lot of open topics and questions exist in that field.

Appendix A

A.1 Quadratic basis functions in 1D

In chapter 3 the equivalence of the finite point method for three noded clouds ($n = 3$) to a central difference approximation has been shown using linear basis functions $\mathbf{p} = [1, x]$. Here the proof of equivalence is extended to quadratic basis funtions in 1D $\mathbf{p} = [1, x, x^2]$.

Consider again the three noded cloud with the points $(i-1,i,i+1)$ (as shown in Figure 3.8) and the coordinates (x_{i-1},x_i,x_{i+1}) and at equal distance h from each other. Their unknown values are u_{i-1}, u_i, u_{i+1}, respectively. The unknown polynomial coefficients a_1, a_2 and a_3 for the local interpolation domain are given by equation 3.52 and is for $m = 3$:

$$\begin{Bmatrix} a_1 \\ a_2 \\ a_3 \end{Bmatrix} = \mathbf{A}^{-1}\mathbf{B}\mathbf{u} \tag{A.1}$$

Matrix $\mathbf{A}$ can be calculated from equation 3.53 using eq. 3.42 as

$$\begin{aligned} \mathbf{A} &= \begin{bmatrix} n & \sum_{j=1}^{n} x_j & \sum_{j=1}^{n} x_j^2 \\ \sum_{j=1}^{n} x_j & \sum_{j=1}^{n} x_j^2 & \sum_{j=1}^{n} x_j^3 \\ \sum_{j=1}^{n} x_j^2 & \sum_{j=1}^{n} x_j^3 & \sum_{j=1}^{n} x_j^4 \end{bmatrix} \\ &= \begin{bmatrix} 3 & 3x_i & 3x_i^2 + 2h^2 \\ 3x_i & 3x_i^2 + 2h^2 & 3x_i^3 + 6x_i h^2 \\ 3x_i^2 + 2h^2 & 3x_i^3 + 6x_i h^2 & 3x_i^4 + 12x_i^2 h^2 + 2h^4 \end{bmatrix} \end{aligned} \tag{A.2}$$

Inversion of $\mathbf{A}$ leads to

$$\mathbf{A}^{-1} = \begin{bmatrix} \frac{3}{2}\frac{x_i^4}{h^4} - \frac{3}{2}\frac{x_i^2}{h^2} + 1 & -3\frac{x_i^3}{h^4} + \frac{3}{2}\frac{x_i}{h^2} & \frac{3}{2}\frac{x_i^2}{h^4} - \frac{1}{h^2} \\ -3\frac{x_i^3}{h^4} + \frac{3}{2}\frac{x_i}{h^2} & 6\frac{x_i^2}{h^4} + \frac{1}{2h^2} & 3\frac{x_i}{h^4} \\ \frac{3}{2}\frac{x_i^2}{h^4} - \frac{1}{h^2} & -3\frac{x_i}{h^4} & \frac{3}{2}\frac{1}{h^4} \end{bmatrix} \tag{A.3}$$

The missing array $\mathbf{Bu}$ is obtained by multiplication of eq. 3.54 with $\mathbf{u}$ and is for quadratic basis functions

$$\mathbf{Bu} = \begin{Bmatrix} u_{i-1} + u_i + u_{i+1} \\ u_{i-1}x_{i-1} + u_i x_i + u_{i+1}x_{i+1} \\ u_{i-1}x_{i-1}^2 + u_i x_i^2 + u_{i+1}x_{i+1}^3 \end{Bmatrix} \tag{A.4}$$

Recall that a function $\hat{u}(x)$ can be approximated by quadratic polynomials from eq. 3.38 as (again omitting the hat over the function $u(x)$ for clarity)

$$\hat{u}(x) = u(x) = \mathbf{p}^T\mathbf{a} = a_1 + a_2 x + a_3 x^2 \tag{A.5}$$

The first derivative of $u(x)$ remains a function of x and becomes at point x_i

$$\frac{\partial u}{\partial x}(x_i) = a_2 + 2a_3 x_i \tag{A.6}$$

Using eqs. A.1, A.3 and A.4, the polynomial coefficients a_2 and a_3 are obtained and can be replaced in eq. A.6. Regrouping leads to

$$\begin{aligned} \frac{\partial u}{\partial x}(x_i) &= a_2 + 2a_3 x_i & \text{(A.7)} \\ &= (u_{i-1} + u_i + u_{i+1})\left(-\frac{3x_i^3}{h^4} + \frac{3x_i}{2h^2} + \frac{3x_i^3}{h^4} - \frac{2x_i}{h^2}\right) + & \text{(A.8)} \\ &\quad (u_i x_i + u_{i-1}(x_i - h) + u_{i+1}(x_i + h))\left(\frac{6x_i^2}{h^4} + \frac{1}{2h^2} - \frac{6x_i^2}{h^4}\right) + \\ &\quad \left(u_i x_i^2 + u_{i-1}(x_i - h)^2 + u_{i+1}(x_i + h)^2\right)\left(-\frac{3x_i}{h^4} + \frac{3x_i}{h^4}\right) \end{aligned}$$

Simplification of eq. A.8 results in

$$\begin{aligned} \frac{\partial u}{\partial x}(x_i) &= (u_i + u_{i-1} + u_{i+1})\left(-\frac{x_i}{2h^2}\right) + \\ &\quad (u_i x_i + u_{i-1}(x_i - h) + u_{i+1}(x_i + h))\left(\frac{1}{2h^2}\right) & \text{(A.9)} \end{aligned}$$

Further simplification cancels the contributions of x_i and leads to

$$\frac{\partial u}{\partial x}(x_i) = \frac{u_{i+1} - u_{i-1}}{2h} \tag{A.10}$$

which is equal to a second order central difference approximation at point x_i as in eq. 3.1 which in turn is equal to a linear Galerkin finite element discretization.

The second derivative of function $u(x)$ using quadratic basis functions from eq. A.5 is constant

$$\frac{\partial^2 u}{\partial x^2} = 2a_3 \tag{A.11}$$

Again inserting eqs. A.1, A.3 and A.4 in eq. A.6 the expression for a_3 is obtained. Thus, we get for the second derivative

$$\begin{aligned}\frac{\partial^2 u}{\partial x^2} &= 2a_3 && \text{(A.12)}\\ &= (u_i + u_{i-1} + u_{i+1})\left(\frac{3x_i^2}{h^4} - \frac{2}{h^2}\right) + \\ &\quad (u_i x_i + u_{i-1}(x_i - h) + u_{i+1}(x_i + h))\left(-\frac{6x_i}{h^4}\right) + && \text{(A.13)}\\ &\quad \left(u_i x_i^2 + u_{i-1}(x_i - h)^2 + u_{i+1}(x_i + h)^2\right)\left(\frac{3}{h^4}\right)\end{aligned}$$

Simplification of eq. A.14 reduces to

$$\frac{\partial^2 u}{\partial x^2} = -\frac{2}{h^2}(u_i + u_{i-1} + u_{i+1}) + \frac{3u_{i-1}}{h^2} + \frac{3u_{i+1}}{h^2} \qquad \text{(A.14)}$$

The final term for the second derivative within a three noded cloud using quadratic basis function then becomes

$$\frac{\partial^2 u}{\partial x^2} = \frac{u_{i-1} - 2u_i + u_{i+1}}{h^2} \qquad \text{(A.15)}$$

This result is what one obtains from a central difference approximation (eq. 3.2) or a Galerkin finite element space discretization for the point x_i in one space dimension.

A.2 Accuracy Estimation for $n > 3$

A.2.1 Without Weighting Functions

The order of approximation is estimated for the use of 4 noded clouds but can be performed in a similar fashion for more than 4 nodes. The following proof describes the use of non symmetric clouds of points with 4 nodes per Ω_i ($n = 4$) for demonstrating that second order accuracy is lost for equally spaced points as shown in figure 3.9.

The matrix $\mathbf{A}$ can be assembled from eq. 3.53 as

$$\mathbf{A} = \begin{bmatrix} 4 & 4x_i + 2h \\ 4x_i + 2h & 4x_i^2 + 4x_i h + 6h^2 \end{bmatrix} \qquad \text{(A.16)}$$

Inversion of $\mathbf{A}$ leads to

$$\mathbf{A}^{-1} = \begin{bmatrix} \frac{2x_i^2}{h^2} + \frac{2x_i}{h} + 3 & -\frac{2x_i}{h^2} - \frac{1}{h} \\ -\frac{2x_i}{h^2} - \frac{1}{h} & \frac{2}{h^2} \end{bmatrix} \qquad \text{(A.17)}$$

From eq. 3.54, the matrix $\mathbf{B}$ is obtained and when multiplied by $\mathbf{u}$ we get:

$$\mathbf{Bu} = \begin{Bmatrix} u_{i-1} + u_i + u_{i+1} + u_{i+2} \\ u_{i-1}(x_i - h) + u_i x_i + u_{i+1}(x_i + h) + u_{i+2}(x_i + 2h) \end{Bmatrix} \tag{A.18}$$

The equation for the polynomial coefficients a_1 and a_2 is:

$$\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \mathbf{A}^{-1}\mathbf{Bu} \tag{A.19}$$

The derivatives of the function u of eq. 3.79 can now be calculated:

$$\begin{aligned} \frac{\partial u}{\partial x} = a_2 \quad = \quad & -\left(\frac{2x_i}{h^2} + \frac{1}{h}\right)(u_{i-1} + u_i + u_{i+1} + u_{i+2}) + \\ & \frac{2}{h^2}(u_{i-1}(x_i - h) + u_i x_i + u_{i+1}(x_i + h) + u_{i+2}(x_i + 2h)) \end{aligned} \tag{A.20}$$

Simplification leads to the final term for which the order of approximation can be estimated.

$$\frac{\partial u}{\partial x} = a_2 = \frac{1}{h}(-3u_{i-1} - u_i + u_{i+1} + 3u_{i+2}) \tag{A.21}$$

Then, the condition 3.97 for second order accuracy is not satidfied:

$$a + c + 4d = -3 + 1 + 12 \neq 0 \tag{A.22}$$

So only first order accuracy is obtained for $n = 4$ and for equally spaced points.

A.2.2 Gauss Weighting Functions

The order of approximation is estimated for the use of weighting functions in a given cloud of points. The proof is performed for 4 noded clouds but can be performed in a similar fashion for more than 4 nodes.

Let the weighting functions be defined by the Gauss curve from eq. 3.64. This means that the value decreases quickly as the distance from the central node r_j increases. The following proof describes the use of non symmetric clouds of points with 4 nodes per Ω_i ($n = 4$) for demonstrating the second order accuracy for equally spaced points as shown in figure 3.9. The associated weights are here 1 for the central node i, w_1 for the nodes $i - 1$ and $i + 1$ and w_2 for the node $i + 2$.

Then, the weighted matrix $\mathbf{A}$ can be assembled from eq. 3.61 as

$$\mathbf{A} = \begin{bmatrix} 1 + 2w_1 + w_2 & x_i + 2w_1 x_i + w_2 x_i + 2w_2 h \\ x_i + 2w_1 x_i + w_2 x_i + 2w_2 h & x_i^2 + 2w_1\left(x_i^2 + h^2\right) + w_2\left(x_i + 2h\right)^2 \end{bmatrix} \tag{A.23}$$

Inversion of $\mathbf{A}$ leads to

$$\mathbf{A}^{-1} = \frac{1}{\det(\mathbf{A})} \begin{bmatrix} x_i^2 + 2w_1\left(x_i^2 + h^2\right) + w_2\left(x_i + 2h\right)^2 & -\left(x_i + 2w_1 x_i + w_2 x_i + 2w_2 h\right) \\ -\left(x_i + 2w_1 x_i + w_2 x_i + 2w_2 h\right) & 1 + 2w_1 + w_2 \end{bmatrix} \tag{A.24}$$

where $\det(\mathbf{A})$ is a constant and is approximated ($w_2 \ll w_1$) by

$$\det(\mathbf{A}) \simeq 2h^2 w_1 \left(1 + 2w_1\right) \tag{A.25}$$

From eq. 3.62, the weighted matrix $\mathbf{B}$ can be calculated and multiplied with $\mathbf{u}$ leading to:

$$\mathbf{Bu} = \begin{Bmatrix} w_1 u_{i-1} + u_i + w_1 u_{i+1} + w_2 u_{i+2} \\ w_1 u_{i-1}(x_i - h) + u_i x_i + w_1 u_{i+1}(x_i + h) + w_2 u_{i+2}(x_i + 2h) \end{Bmatrix} \tag{A.26}$$

Then the polynomial coefficients a_1 and a_2 become:

$$\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \mathbf{A}^{-1}\mathbf{Bu} \tag{A.27}$$

which is inserted into eq. 3.79 so that we can now write:

$$\begin{aligned} &\frac{\partial u}{\partial x} = a_2 \simeq \\ &\frac{1}{2h^2 w_1\left(1 + 2w_1\right)}\left[-\left(w_1 u_{i-1} + u_i + w_1 u_{i+1} + w_2 u_{i+2}\right)\left(x_i + 2w_1 x_i + w_2 x_i + 2w_2 h\right) + \right. \\ &\left.\left(w_1 u_{i-1}(x_i - h) + u_i x_i + w_1 u_{i+1}(x_i + h) + w_2 u_{i+2}(x_i + 2h)\right)\left(1 + 2w_1 + w_2\right)\right] \end{aligned} \tag{A.28}$$

The approximation $w_2 \ll w_1$ can be used to cancel the terms involving $w_1 w_2$ and w_2^2 and simplifying gives

$$\frac{\partial u}{\partial x} = a_2 \simeq \frac{1}{2h^2 w_1\left(1 + 2w_1\right)}\left[\left(-w_1 u_{i-1} h + w_1 u_{i+1} h\right)\left(1 + 2w_1\right) - 2w_2 h u_i + 2w_2 h u_{i+2}\right] \tag{A.29}$$

Further simplification leads to the final term for which the order of approximation can be estimated.

$$\frac{\partial u}{\partial x} = a_2 \simeq \frac{1}{2h}\left(u_{i+1} - u_{i-1}\right) - \frac{w_2}{h w_1\left(1 + 2w_1\right)}\left(u_{i+2} - u_i\right) \tag{A.30}$$

For the condition $w_2 \ll w_1$, the condition 3.97 becomes

$$a + c + 4d \simeq 0 \tag{A.31}$$

which yields second order accuracy for equally spaced points.