FINITE ELEMENT SOLUTION OF THE TWO-DIMENSIONAL INCOMPRESSIBLE NAVIER-STOKES EQUATIONS WITH CORIOLIS FORCE

DANIEL DEACU

Finite Element Solution of the Two-Dimensional Incompressible Navier-Stokes Equations with Coriolis Force

by

© Daniel Deacu

A thesis submitted to the
School of Graduate Studies
in partial fulfilment of the
requirements for the degree of
Master of Science

Departments of Physics and Physical Oceanography,
Mathematics and Statistics, and Computer Science
Memorial University of Newfoundland

January, 2002

St. John's Newfoundland

Abstract

A finite-element-based numerical algorithm is developed to solve the two-dimensional incompressible Navier-Stokes equations with Coriolis force, which can be used to simulate the wind-driven barotropic ocean circulation on a beta-plane. The spatial discretization is performed via the standard Galerkin Finite Element Method, by using the classical isoparametric Taylor-Hood serendipity quadrilateral finite element. A variant of the Crank-Nicolson rule is employed for the temporal discretization, and the Picard iteration method deals with the nonlinearity of the advective terms. In an effort to remain faithful to the standard Galerkin Finite Element Method, the consistent mass matrix is used instead of the lumped mass matrix, and least-square best fits are calculated for the quantities derived in the postprocessing phase. The algorithm has been implemented into a program and successfully tested on three benchmark problems, flow past a cylinder, flow over a backward facing step, and mid-latitude wind-driven barotropic ocean circulation for an idealized flat-bottomed ocean.

Acknowledgements

First of all, I would like to thank my co-supervisors Dr. Paul Myers and Dr. Serpil Kocabiyik, who have provided valuable guidance and continuous support throughout my research. I would also like to thank my wife and my son for their support, encouragement and patience.

Special thanks to ...

... Prof. Janet Benger, Director of English as a Second Language Programs, and Dr. Gregory S. Kealey, Dean of the School of Graduate Studies, for their understanding and invaluable help offered in a critical period of my graduate studies,

... Dr. Brad de Young of the Department of Physics and Physical Oceanography for his willingness to give advice whenever requested as well as for allowing me to use the computing facilities of the Physical Oceanography Group,

... my fellow graduate students, Patrick Timko and Paul Simmons, for being great colleagues.

Financial support for this work was provided from NSERC (Natural Sciences and Engineering Research Council) research grants received by Dr. Paul Myers and Dr. Serpil Kocabiyik, and in the form of a graduate fellowship offered by Memorial University of Newfoundland, for which I am grateful.

To my wife, Mariana,

and my son, Lucian

CONTENTS

List of Tables

List of Figures

1. INTRODUCTION

The incompressible Navier-Stokes equations govern many incompressible flows. The difficulty of generating analytical and numerical solutions to these equations, when modelling 'real life' flows, is well known. This is partly due to the non-linearity of the advection terms and to the fact that the incompressibility constraint between the velocity components – the velocity field is divergence-free – does not involve the pressure field. Several approaches to treating these issues within numerical algorithms have been devised, however, without offering a definite conclusion as to what the best numerical treatment would be.

Among the numerical methods employed for solving the incompressible Navier-Stokes equations, the Finite Element Method (FEM) stands out, as it develops numerical algorithms in a precise and consistent manner starting from the underlying partial differential equations, without requiring external 'help' usually consisting of ad hoc and developer-dependent strategies. Moreover, its sound mathematical foundation allows for functional analysis to be used as a powerful tool for error analysis. This has led to the development of adaptive techniques (e.g. mesh refinement) based on error estimates, which aim at improving the reliability and accuracy of numerical solutions.

When modelling oceanic flows in a reference frame fixed to the rotating Earth, an additional (fictitious) force – the Coriolis force – must be included in the incompressible Navier-Stokes equations. The Finite Difference Method (FDM) has been the numerical method of choice for most ocean models for solving these equations. Its relative simplicity as well as the ease of developing numerically efficient code has attracted many ocean modellers. On the other hand, the success of the finite element method in the area of solid mechanics, where it is almost exclusively used, has prompted intensive research on its application in the area of Computational Fluid Dynamics (CFD). The FEM has some clear advantages that recommend it as a powerful alternative numerical method for ocean models. The most important advantage of FEM is that it allows the use of unstructured grids with variable resolution. The grid resolution can be increased in the areas characterized by large gradients in velocity and/or pressure. Thus, flow phenomena concentrated in narrow regions (e.g. western boundary currents, flows through straits) can be resolved by only refining the mesh locally, without changing the resolution for the ocean interior. Moreover, complex coastlines and bottom topography are better represented when using finite element meshes. Another big advantage of FEM is offered by its ability to treat boundary conditions naturally. Furthermore, multi-connected domains are dealt with easily. Compared to the FDM-based algorithms, FEM-based algorithms are more complex and thus more computationally expensive when used to solve the same problem with the same uniformly high resolution, for example. However, the FEM-computation time can be reduced a great deal by decreasing the resolution in the areas where it is unnecessary high.

Several researchers have proposed and studied FEM-based numerical algorithms for ocean modelling (Fix, 1975; Haidvogel et al., 1980; Dumas et al., 1982; Le Provost, 1984 and 1986, Le Provost et al., 1993, Myers et al., 1995; Le Roux et al., 2000). Since the application of the finite element method in this field has not reached maturity, most FEM-based models have been limited to the simulation of two-dimensional barotropic flows.

Fix (1975) was the first to study the properties of FEM applied to a quasigeostrophic formulation of the ocean dynamics. He pointed out some FEM-specific advantages such as the natural treatment of boundary conditions and the very good representation of complex domains offered by meshes based on triangular elements, which would be useful for ocean modelling. Furthermore, he emphasized the accuracy of the method and showed that vorticity, energy and enstrophy were conserved. The interest in FEM for ocean modelling has increased after the comparison study performed by Haidvogel et al. (1980) showed that finite element and spectral models were more accurate and even more efficient than the finite difference models.

A FEM model for the wind-driven barotropic general ocean circulation in a closed basin was proposed and investigated by Dumas et al. (1982). The precision of the results was found advantageous compared to that offered by classical FDM models for the non-linear cases. By using quadratic finite elements for the region characterized by large gradients in velocity, i.e. the western boundary layer, the accuracy has been improved without additional computation time. Le Provost (1984) extended the model of Dumas et al. to a two-layer quasigeostrophic version of a baroclinic model. Local refinement of the triangular finite element mesh was performed in order to resolve baroclinic instabilities. Subsequently, Le Provost et al. (1993) extended the model to a multi-layer stratified flat-bottomed ocean. This model was estimated to be approximately two times more computational expensive than a very efficient finite difference model, for real ocean applications.

Myers et al. (1995) developed a diagnostic barotropic finite-element ocean circulation model using spherical coordinates. The model was used to simulate the barotropic circulation in the North Atlantic Ocean by taking into account realistic coastline and topography. The results compared very well with previous diagnostic calculations. One of the advantages offered by finite element models, i.e. their capability of resolving very dynamic and localized flows, was proved by the prediction of the separation of the Gulf Stream at the correct latitude in the presence of the JEBAR (Joint Effect of Baroclinicity And Relief) term.

High spatial accuracy is required, among others, by the non-linear terms for accurate simulations of slow Rossby modes. Since the Eulerian advection schemes based on higher-order finite elements are computationally expensive, and introduce spurious modes and numerical dispersion, combinations of FEM methods with other types have been proposed. A combination of a finite-element and a semi-Lagrangian method has presented by Le Roux et al. (2000) for a shallow-water ocean model. This model has the advantage of both the flexibility of the finite element mesh and the small numerical dispersion of semi-Lagrangian scheme. Another relatively new numerical method being used in ocean modelling is the Spectral Element Method (Iskandarani et al., 1995). It aims at providing higher spatial accuracy by combining the accuracy of standard spectral methods and the geometric flexibility of finite element methods. One of the salient effects is an improved accuracy of the non-linear terms.

The purpose of the present work is to develop a finite element algorithm for solving the two-dimensional incompressible Navier-Stokes equations with Coriolis force that can be used for simulating wind-driven barotropic ocean circulation on a beta-plane. The algorithm is intended to be used as a foundation for developing a three-dimensional baroclinic finite-element ocean model. The finite element of choice is the classical isoparametric Taylor-Hood serendipity quadrilateral finite element. This element is based on the mixed interpolation of the velocity and pressure; more precisely it offers a continuous piecewise-quadratic variation for velocity components and a continuous piecewise-linear pressure field. Since it is a quadratic isoparametric element, curved boundaries are better represented by this element. Moreover, the quadrilateral elements are superior to triangular elements when used to simulate advection-dominated flows, as they do no exhibit mesh orientation effects.

The chosen approach is to use as much as possible from the field of finite element techniques developed for the simulation of industrial flows, and then add characteristic features of oceanic flows (e.g. Coriolis force). This approach offers the advantage of an easy transfer of many powerful techniques such as adaptive methods, which are already being used for industrial applications, for further development of the ocean model.

The thesis is structured as follows. Chapter 2 presents the incompressible Navier-Stokes equations with Coriolis terms as a set of equations that can be used to model the wind-driven barotropic ocean circulation on a beta-plane. These equations, along with the initial and boundary conditions given in the last section of the same chapter, are the underlying equations of the numerical algorithm developed in the subsequent chapters.

The spatial discretization realized by means of the finite element method, and the resulting semi-discrete equations are described in Chapter 3. The discretization is completed by the temporal discretization, which is presented in Chapter 4. Chapter 5 is devoted to the implementation of the numerical algorithm. The results of testing the finite element program on three benchmark problems are given in Chapter 6. The last chapter, Chapter 7, presents the conclusions and some suggestions for future work needed for improving the numerical algorithm.

2. CONTINUUM EQUATIONS

2.1 Incompressible Navier-Stokes Equations

Laminar flows of viscous and incompressible fluids can be described by the incompressible Navier-Stokes equations, which will simply be referred to as the Navier-Stokes equations hereafter. This is a complete system of partial differential equations consisting of equations corresponding to the conservation of momentum and mass.

The two-dimensional incompressible Navier-Stokes equations in the continuum form, in terms of primitive variables (u, v, P), are

momentum equations:

$$\rho\left(\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}\right)-\mu\nabla^2 u+\frac{\partial P}{\partial x}=F_x, \qquad (2.1.1)$$

$$\rho\left(\frac{\partial v}{\partial t}+u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}\right)-\mu\nabla^2 v+\frac{\partial P}{\partial y}=F_y; \qquad (2.1.2)$$

continuity equation:

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}=0. \qquad (2.1.3)$$

Here u and v represent the components of the velocity in the x and y directions, respectively, P is the pressure, F_x, F_y are the forcing terms, ρ is the density, μ is the dynamic viscosity of the fluid, and t is the time. The presence of the friction terms in

Laplacian form in the momentum equations (2.1.1-2) indicates that the fluid is Newtonian. By definition, these are fluids in which the shear stress is proportional to the velocity gradient.

Assuming that the fluid flows within the region $D \subset \mathbf{R}^2$, during the time interval $[0, t_{final}]$, the velocity components and the pressure are sought in the domain $D \times [0, t_{final}]$. In general, finding analytical solutions to the Navier-Stokes equations is extremely difficult, mainly due to the nonlinearity in the advective terms and the velocity-pressure coupling. For complex problems, numerical solutions are generated instead.

2.2 Wind-Driven Barotropic Ocean Circulation Equations

The momentum equations (2.1.1-2) hold true only if the frame of reference is inertial. Additional terms appear in these equations when they are derived in a rotating frame of reference. These terms correspond to the Coriolis and centrifugal forces, which are two fictitious body forces (Salmon, 1998).

In oceanography, the fluid is normally considered in a frame of reference fixed to the Earth. It is also assumed that the Earth rotates with a constant angular velocity Ω. The governing equations are the three-dimensional Navier-Stokes equations including the additional terms due to the rotation.

A traditional approximation used in oceanography is to ignore most of the effects induced by the curvature of the Earth's surface when modelling flows at regional scale. This is done by using a rectangular Cartesian coordinate system with the *x*-*y* horizontal plane tangent to the Earth at a prescribed point. The *x*- and *y*-axes are oriented eastwards and northwards, respectively (Figure 2.1). Some of the above mentioned effects are retained in this plane, assuming that the vertical component Ω_z of the angular velocity vector $\vec{\Omega}$, varies linearly in the y (latitudinal) direction.

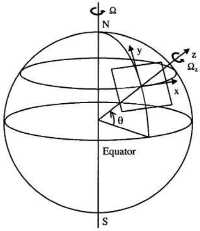


Figure 2.1 Rectangular Cartesian coordinate system used for regional scale flows

Neglecting both the centrifugal force and the horizontal component $\Omega_h = \Omega \cos\theta$ of the Earth's angular velocity vector $\vec{\Omega}$, where θ is the latitude, is another traditional approximation used to simplify the governing equations (Kantha et al., 2000).

With these approximations, the additional terms occurring in the Navier-Stokes equations are generated only by the angular velocity component $\Omega_z = \Omega \sin\theta$. Expressed as Coriolis accelerations, these terms are $-fv$ in the *x*-momentum equation and fu in the *y*-momentum equation, where *f* is the Coriolis parameter defined by $f = 2\Omega \sin\theta$. When

Ω_z is assumed to vary linearly with latitude, f has a linear variation as well. If this variation is $f = f_0 + \beta y$, then the so-called beta-plane approximation is used, where $\beta = \frac{\partial f}{\partial y}$. It is noted that $f_0 = 2\Omega \sin\theta_0$ and $\beta = \frac{2\Omega \cos\theta_0}{R}$, where θ_0 is the latitude of the point where the *x-y* plane is tangent to the Earth surface, and R is the Earth's radius.

The two-dimensional Navier-Stokes equations can be used to model oceanographic flows on a beta-plane, if the following approximations and assumptions are added to those previously presented:

i) the Boussinesq approximation, which allows the use of a constant value ρ_0 for the fluid (seawater) density in all terms, except for the gravitational force term;

ii) the hydrostatic approximation, which assumes that the vertical component of the pressure gradient is balanced by the gravitational force;

iii) vertical velocities are negligible compared to horizontal velocities;

iv) the rigid lid approximation that assumes the sea surface is rigid;

v) the fluid is considered barotropic ($\rho = \rho(P)$ only), therefore the velocity field is uniform with depth.

The hydrostatic approximation reduces the vertical momentum equation in the three-dimensional set of Navier-Stokes equations to a simple relation for calculating the hydrostatic pressure

$$\frac{\partial P}{\partial z} = -\rho g . \qquad (2.2.1)$$

Eliminating this equation from the set of Navier-Stokes equation leads to the removal of the hydrostatic pressure mode. Therefore, the pressure in the resulting two-dimensional Navier-Stokes equations is the sum of the dynamic pressure and the geostrophic pressure, and can be viewed as a deviation from the hydrostatic pressure.

For a flat-bottomed ocean, and with the above approximations, the governing equations are the two-dimensional Navier-Stokes with Coriolis terms

momentum equations:

$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}-A_H\nabla^2 u-fv+\frac{1}{\rho_0}\frac{\partial P}{\partial x}=\frac{1}{\rho_0}F_x, \tag{2.2.2}$$

$$\frac{\partial v}{\partial t}+u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}-A_H\nabla^2 v+fu+\frac{1}{\rho_0}\frac{\partial P}{\partial y}=\frac{1}{\rho_0}F_y; \tag{2.2.3}$$

continuity equation:

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}=0. \tag{2.2.4}$$

Here A_H is the horizontal eddy diffusion coefficient that plays the role of the kinematic viscosity of the fluid $\frac{\mu}{\rho_0}$, and $f = f_0 + \beta y$ in the beta-plane approximation.

Equations (2.2.2-4) can be used to model the wind-driven barotropic ocean circulation at regional scale. In this case, for a flat-bottomed ocean of constant depth D_0, the forcing terms are $\frac{1}{\rho_0}F_x=\frac{\tau_x}{\rho_0 D_0}$ and $\frac{1}{\rho_0}F_y=\frac{\tau_y}{\rho_0 D_0}$, where τ_x and τ_y are the components of the surface wind stress. It is noted that the dissipation of momentum through bottom friction is ignored in this case. Thus, momentum is dissipated only through lateral friction, represented in the equations (2.2.2-3) by the terms containing the Laplacians of velocity components.

The rigid-lid assumption filters out the fast surface gravity waves, which would reduce the magnitude of allowable time steps if considered. However, this assumption affects the accuracy of the representation of the Rossby waves (Dukowicz and Smith, 1994).

2.3 Boundary and Initial Conditions

In order to solve the system of Navier-Stokes equations, boundary and initial conditions are required. There are two types of boundary conditions (BC), essential boundary conditions and natural boundary conditions. Selecting the appropriate boundary conditions from the several formulations available for the Navier-Stokes equations is not an easy task, and depends on the form of the advection and diffusion terms of these equations.

It is well known that many of the problems that occur in numerical simulations are caused by the boundary conditions used (Kreiss et al., 1989). Thus, the type of natural boundary conditions used for the Navier-Stokes equations is crucial when finite element type solutions are sought, as these conditions are incorporated from the beginning in the weak form of the equations.

The essential boundary conditions, also known as Dirichlet boundary conditions, specify the velocity on a portion of the boundary, denoted by Γ_D. In the two-dimensional case this can be expressed as

$$u(x,y,t)=U(x,y,t),\quad v(x,y,t)=V(x,y,t),\quad (x,y)\in\Gamma_D,\ t\in[0,t_{final}], \qquad (2.3.1)$$

where $U(x,y,t)$, and $V(x,y,t)$ are known. If Γ_D is a solid boundary (no penetration) that is fixed and no slip is allowed, equation (2.3.1) becomes

$$u(x,y,t)=0,\quad v(x,y,t)=0,\quad (x,y)\in\Gamma_D,\ t\in[0,t_{final}]. \qquad (2.3.2)$$

The pressure is usually specified at one point only. There is no need for prescribing the pressure at more than one point in the computational domain because it can be determined up to an arbitrary additional constant (the hydrostatic pressure mode) (Gresho et al., 1999, Heinrich et al, 1999). Moreover, it has been proved that no explicit boundary conditions for the pressure are necessary for incompressible flows (Ladyshenskaya, 1969). This is another indicator of the fact that the (dynamic) pressure is strongly coupled with the velocity field and its role is to ensure the flow is divergence-free.

In some cases, the pressure is specified at more than one point. Although this adds more constraints, it has been found to have a stabilizing effect on the solution obtained using the proposed FEM algorithm in some applications where outflow open boundaries were used. It has also a physical meaning as an outflow open boundary is usually used where the flow is considered fully developed and thus the pressure is constant along that boundary.

Equation (2.3.1) can be applied at the inlet, where the fluid enters the computational domain, as an open boundary condition (OBC). When a finite-element-based algorithm generates the solution, the use of the same type of condition at the outlet could lead to numerical oscillations (Cuvelier et al., 1986), and is not recommended. Natural boundary conditions are more appropriate for the outlet in this case.

There are different types of natural boundary conditions (traction, Neumann, Robin, total momentum flux, etc.). The simplest natural boundary condition is to specify the normal component of the velocity gradient on the boundary Γ_N (Neumann BC)

$$\frac{\partial u(x,y,t)}{\partial n} = U_n(x,y,t),\ \frac{\partial v(x,y,t)}{\partial n} = V_n(x,y,t),\ (x,y) \in \Gamma_N,\ t \in [0,t_{final}]. \quad (2.3.3)$$

Unlike other types of natural BC (e.g. stress-type natural BC), this condition does not include the pressure values on the boundary. Some finite-element-based numerical simulations (Heinrich et al., 1996) showed that this condition performs even better than those containing the pressure, when used as outflow open boundary condition in the form

$$\frac{\partial u(x,y,t)}{\partial n} = 0,\ \frac{\partial v(x,y,t)}{\partial n} = 0,\ (x,y) \in \Gamma_N,\ t \in [0,t_{final}]. \quad (2.3.4)$$

The initial conditions specify the velocity field at the initial time $t = 0$,

$$u(x,y)=U_0(x,y),\ v(x,y)=V_0(x,y),\ (x,y)\in D\cup\Gamma_D\cup\Gamma_N. \qquad (2.3.5)$$

The initial pressure field is not needed, as the initial velocity field generates it. Furthermore, the initial velocity field must satisfy the incompressibility constraint given by the continuity equation (2.2.4). A compatibility condition involving the initial velocity on Γ_D and the velocity specified by the essential BC at time $t=0$ is also necessary to exclude impulsive changes at start. This is given by

$$U_0(x,y)=U(x,y,0),\ V_0(x,y)=V(x,y,0),\ (x,y)\in\Gamma_D. \qquad (2.3.6)$$

3. SPATIAL DISCRETIZATION

3.1 Introduction

The approach used to find numerical solutions to the problem defined by the equations (2.2.2-4), with the boundary conditions (2.3.1) and (2.3.3), and the initial condition (2.3.5), and described hereafter, is based on the Galerkin Finite Element Method and the Finite Difference Method. The Galerkin Finite Element Method is used for the spatial discretization of the problem over the flow region $D \subset \mathbf{R}^2$, and yields a semi-discrete (the time variable remains continuous) set of equations (ordinary differential equations). The time-integration method for the semi-discrete set of equations over the time interval $[0, t_{final}]$ is the Finite Difference Method. Some authors call this combined method, *the method of lines*.

The Finite Element Method does not aim at solving the original set of PDEs, but integral forms of these, called *weak* or *variational forms*. They are obtained by multiplying the PDEs by some carefully chosen functions, followed by integration over the spatial domain. The natural boundary conditions are naturally incorporated into the weak forms, and this is one of the advantages offered by FEM.

3.2 Weak Formulation

Different weak forms can be obtained for the given problem, depending on the manipulations performed on the original PDEs (e.g. rewriting the advection and/or diffusion terms in some other forms) and/or after integrating over the spatial domain.

Each weak form is associated with the type of natural boundary condition that it can incorporate. Thus, when deriving a weak form for the problem one has to decide on the type of natural boundary condition that is to be used. This is a crucial point in developing a FEM algorithm and a clear answer to what the best weak form is, might not be easy to get.

In order to obtain a weak formulation for the problem defined by the equations (2.2.2-4), with the boundary conditions (2.3.1) and (2.3.3), and the initial condition (2.3.5), the momentum equations are multiplied first by an arbitrary function φ and integrated over the domain D. Furthermore, the continuity equation is multiplied by an arbitrary function ψ and then integrated over D (Taylor, 1981). Thus, we get

$$\int_D \varphi \frac{\partial u}{\partial t} dD + \int_D (\varphi u \frac{\partial u}{\partial x} + \varphi v \frac{\partial u}{\partial y}) dD - \int_D \varphi \nabla (A_H \nabla u) dD$$

$$- \int_D f \varphi v dD + \frac{1}{\rho_0} \int_D \varphi \frac{\partial P}{\partial x} dD = \frac{1}{\rho_0} \int_D \varphi F_x dD, \tag{3.2.1}$$

$$\int_D \varphi \frac{\partial v}{\partial t} dD + \int_D (\varphi u \frac{\partial v}{\partial x} + \varphi v \frac{\partial v}{\partial y}) dD - \int_D \varphi \nabla (A_H \nabla v) dD$$

$$+ \int_D f \varphi u dD + \frac{1}{\rho_0} \int_D \varphi \frac{\partial P}{\partial y} dD = \frac{1}{\rho_0} \int_D \varphi F_y dD, \tag{3.2.2}$$

$$\int_D (\psi \frac{\partial u}{\partial x} + \psi \frac{\partial v}{\partial y}) dD = 0. \tag{3.2.3}$$

The functions φ and ψ, called *test functions*, are time-independent and must satisfy a set of requirements that will be presented during the derivation of the semi-discrete equations. First, because of the essential boundary conditions for velocity (2.3.1), the test functions φ are chosen such that they vanish on the boundary Γ_D

$$\varphi(x,y) = 0, \quad (x,y) \in \Gamma_D. \tag{3.2.4}$$

Invoking Green's theorem and taking into account (3.2.4), we get the following expressions for the diffusion terms in the equations (3.2.1-2)

$$-\int_D \varphi \nabla(A_H \nabla u) dD = \int_D A_H \nabla \varphi \nabla u dD - \int_{\Gamma_N} A_H \varphi \frac{\partial u}{\partial n} d\Gamma, \tag{3.2.5}$$

$$-\int_D \varphi \nabla(A_H \nabla v) dD = \int_D A_H \nabla \varphi \nabla v dD - \int_{\Gamma_N} A_H \varphi \frac{\partial v}{\partial n} d\Gamma, \tag{3.2.6}$$

where $\vec{n}$ is the outward pointing unit normal vector on the boundary Γ_N ($\partial D = \Gamma_D \cup \Gamma_N$). Substituting these terms by the expressions given by the equations (3.2.5-6) into the momentum equations (3.2.1-2) gives

$$\begin{aligned}
&\int_D \varphi \frac{\partial u}{\partial t} dD + \int_D \left(\varphi u \frac{\partial u}{\partial x} + \varphi v \frac{\partial u}{\partial y}\right) dD + \int_D A_H \nabla \varphi \nabla u dD \\
&\quad - \int_D f \varphi v dD + \frac{1}{\rho_0} \int_D \varphi \frac{\partial P}{\partial x} dD = \frac{1}{\rho_0} \int_D \varphi F_x dD + \int_{\Gamma_N} A_H \varphi \frac{\partial u}{\partial n} d\Gamma,
\end{aligned} \tag{3.2.7}$$

$$\int_D \varphi \frac{\partial v}{\partial t} dD + \int_D (\varphi u \frac{\partial v}{\partial x} + \varphi v \frac{\partial v}{\partial y}) dD + \int_D A_H \nabla \varphi \nabla v dD$$

$$+ \int_D f \varphi u dD + \frac{1}{\rho_0} \int_D \varphi \frac{\partial P}{\partial y} dD = \frac{1}{\rho_0} \int_D \varphi F_y dD + \int_{\Gamma_N} A_H \varphi \frac{\partial v}{\partial n} d\Gamma . \qquad (3.2.8)$$

The natural boundary conditions given by (2.3.3) can be readily incorporated into the equations (3.2.7-8). This is done by replacing the normal derivatives of the velocity components u and v with the given functions U_n and V_n, on Γ_N, and this is the reason the boundary integrals are placed along with the data on the right-hand side of the above equations.

In order to formulate a weak form of the given problem, it is necessary to declare the function spaces in which the unknown functions and the test functions lie. These spaces are chosen to ensure the existence of the integrals in the equations (3.2.7-8) and (3.2.3). Thus, it is required that

$$\varphi \in H_0^1(D), \quad \psi \in L^2(D), \quad u \in H_{u,E}^1(D), \quad v \in H_{v,E}^1(D), \quad P \in H^1(D). \qquad (3.2.9)$$

As the test function ψ will be related to the pressure P, it is further required that ψ lie in the more restrictive space $H^1(D)$.

Notice that the Hilbert space $L^2(D)$ is the space of functions f defined over D that are square integrable over D, i.e. $L^2(D) = \left\{ f \mid \int_D f^2 dD < \infty \right\}$. The Sobolev space

$H^1(D)$ is the space of functions f defined over D such that f and all its first-order partial derivatives are in $L^2(D)$. The space $H_0^1(D)$ is the space of all functions in $H^1(D)$ that vanish on the Dirichlet boundary Γ_D. The spaces $H_{u,E}^1(D)$ and $H_{v,E}^1(D)$ are the spaces of all functions in $H^1(D)$ that satisfy the essential (Dirichlet) boundary conditions given by (2.2.1) for the u and v velocities, respectively.

It is now possible to give a weak form of the problem defined by the equations (2.2.2-4), with the boundary conditions (2.3.1) and (2.3.3), and the initial condition (2.3.5).

Find $u \in H_{u,E}^1(D)$, $v \in H_{v,E}^1(D)$, and $P \in H^1(D)$ such that

$$\int_D \varphi \frac{\partial u}{\partial t} dD + \int_D (\varphi u \frac{\partial u}{\partial x} + \varphi v \frac{\partial u}{\partial y}) dD + \int_D A_H \nabla\varphi \nabla u dD$$

$$- \int_D f \varphi v dD + \frac{1}{\rho_0} \int_D \varphi \frac{\partial P}{\partial x} dD = \frac{1}{\rho_0} \int_D \varphi F_x dD + \int_{\Gamma_N} A_H \varphi U_n d\Gamma , \qquad (3.2.10)$$

$$\int_D \varphi \frac{\partial v}{\partial t} dD + \int_D (\varphi u \frac{\partial v}{\partial x} + \varphi v \frac{\partial v}{\partial y}) dD + \int_D A_H \nabla\varphi \nabla v dD$$

$$+ \int_D f \varphi u dD + \frac{1}{\rho_0} \int_D \varphi \frac{\partial P}{\partial y} dD = \frac{1}{\rho_0} \int_D \varphi F_y dD + \int_{\Gamma_N} A_H \varphi V_n d\Gamma , \qquad (3.2.11)$$

$$\int_D (\psi \frac{\partial u}{\partial x} + \psi \frac{\partial v}{\partial y}) dD = 0 , \qquad (3.2.12)$$

for all $\varphi \in H_0^1(D)$ and all $\psi \in H^1(D)$.

Thus, the solution to this weak form is sought, instead of that to the original equations. This solution is called weak or generalized solution (Gresho et al., 1999). The weak solution need not admit second-order spatial derivatives; the replacement involving the diffusion terms weakened the differentiability requirements that u and v must satisfy, because the second-order differential operator is no longer present in the final integral equations.

It is possible to weaken the differentiability for the pressure as well, by integrating by parts the term containing the derivative of the pressure. Thus, the pressure function need not possess first-order spatial derivatives and may be even discontinuous over D. The only requirement for the pressure would be that the pressure function be square integrable over D, i.e. $P \in L^2(D)$. The resulting weak form requires that the natural boundary conditions be prescribed in the form of stresses applied on the boundary, involving the pressure (Reddy et al., 1994, Gresho et al., 1999).

The solution to the original problem, called classical solution, is always a weak solution, whereas a weak solution is a classical solution only when it is sufficiently smooth (Gresho et al., 1999).

3.3 Finite Element Discretization of the Weak Form

The Finite Element Method is a numerical technique used to generate approximate solutions of weak forms associated with ordinary/partial differential equations.

The first step in applying the FEM to the weak form given by the equations (3.2.10-12) is to discretize the domain $\overline{D} = D \cup \Gamma$ into a finite number of subdomains. In the 2D case, these subdomains are triangles or quadrilaterals. Any two such subdomains e_k, e_l $(k \neq l)$, must fall into one of the following situations

(i) $e_k \cap e_l = \varnothing$,

(ii) e_k and e_l have a common edge,

(iii) e_k and e_l have a common vertex.

Following the discretization of the domain, a number of points, called *nodes*, are chosen within each subdomain. Usually, nodes are chosen on the boundary of the subdomain (e.g. vertices, mid-side points) and are shared by adjacent subdomains.

Next, a finite set of piecewise polynomials, called *shape* or *basis functions*, defined on the domain $\overline{D}$, is introduced. Each shape function is associated with a certain node. Its value at that node is *1*, whereas at the rest of the nodes the value is *0*. Furthermore, the shape function has a prescribed variation on each subdomain (e.g. linear, quadratic, cubic, etc.), depending on the number of nodes it contains. The nonzero restriction of the shape function to a subdomain is called *local shape/basis function*. The idea is to seek solutions to the weak form in function spaces that are spanned by the shape functions. These are finite-dimensional subspaces of the underlying function spaces for the unknown functions *u*, *v*, and *P*. By replacing the infinite-dimensional spaces in the weak form with the finite-dimensional spaces, it is assumed a priori that the velocity and pressure functions have prescribed variations over the domain. Thus, approximate solutions in the form of linear combinations of shape functions are sought for a semi-discrete weak form derived from the original weak form. Note that the approximate solutions are strongly dependent on the geometrical representation (i.e. subdomains, nodes) of the domain $\overline{D}$.

A *finite element* consists of a subdomain e_k, the nodes on e_k, and the local shape functions associated with these nodes (Cuvelier et al., 1986). The partition of the domain into finite elements is called *finite element mesh.* The finite-dimensional approximation spaces spanned by the shape functions are called *finite element spaces.* In the case in which both the velocity components and the pressure are linear combinations of the same shape functions, i.e. they lie in the same finite element space, it is said that an *equal-order interpolation* is used. The use of equal-order interpolation leads to the occurrence of spurious pressure solutions, called *spurious pressure modes*, and eventually to instability. However, there are various stabilization techniques that can be used to avoid these problems (Hughes et al., 1986).

Since the friction terms in the Navier-Stokes equations involves a second-order operator applied to the velocity components, namely the Laplacian, whereas only a first-order operator (the gradient) is applied to the pressure, one would consider appropriate the use of a higher-order piecewise polynomial for the velocity than for the pressure. In this case a *mixed interpolation* is used. The mixed interpolation using one-order-lower shape function for pressure than for velocity is the most widely used (Gresho et al., 1999), and is the type of interpolation used hereafter. In this case, the finite element spaces for velocity and pressure satisfy a compatibility condition, called the Babuska-Brezzi condition (Girault et al., 1986), which is a necessary condition for stability.

Let us assume that the domain $\overline{D}$ has been discretized into a number N_e of finite elements, such that $\overline{D} = \bigcup_{k=1}^{N_e} e_k$. Assume that $N_T = N + M$ total nodes are chosen for the velocity, where N is the number of nodes in $D \cup \Gamma_N$ and M the number of nodes on Γ_D. Assume that the nodes on the Dirichlet boundary Γ_D are numbered from $N+1$ to N_T. If the set of shape functions associated with the velocity nodes is $\{\varphi_j(x,y)\}_{j=\overline{1,N_T}}$, the approximate velocity components are:

$$u^h(x,y,t) = u^h_{D\cup\Gamma_N}(x,y,t) + u^h_{\Gamma_D}(x,y,t)$$

$$= \sum_{j=1}^{N} u_j(t)\varphi_j(x,y) + \sum_{j=N+1}^{N_T} u_j(t)\varphi_j(x,y),\ (x,y,t) \in \overline{D}\times[0,t_{final}], \quad (3.3.1)$$

$$v^h(x,y,t) = v^h_{D\cup\Gamma_N}(x,y,t) + v^h_{\Gamma_D}(x,y,t)$$

$$= \sum_{j=1}^{N} v_j(t)\varphi_j(x,y) + \sum_{j=N+1}^{N_T} v_j(t)\varphi_j(x,y),\ (x,y,t) \in \overline{D}\times[0,t_{final}], \quad (3.3.2)$$

where u_j is the nodal value of the velocity u at the node j, and v_j is the nodal value of the velocity v at the same node. The second sum in the above equations interpolates the prescribed velocities $U(x,y,t)$ and $V(x,y,t)$ on Γ_D, given by the essential BC. Thus, only the nodal values of the prescribed velocities are used via $u_j(t) = U_j(t) \equiv U(x_j, y_j, t)$ and $v_j(t) = V_j(t) \equiv V(x_j, y_j, t)$, for $j = N+1, \ldots, N_T$, the rest of the nodal values u_j and v_j, for $j = 1, \ldots, N$, remaining to be determined. Note that the superscript h denotes some measure (e.g. typical, maximum) of element size in the finite element mesh.

Assume that N_P nodes are chosen for the pressure and the set of shape functions associated with them is $\{\psi_j(x,y)\}_{j=\overline{1,N_P}}$. Then, the approximate pressure is

$$P^h(x,y,t) = P^h_{nc}(x,y,t) + P^h_c(x,y,t)$$

$$= \sum_{j=1}^{N_P-1} P_j(t)\psi_j(x,y) + P_{N_P}(t)\psi_{N_P}(x,y),\ (x,y,t) \in \overline{D}\times[0,t_{final}], \quad (3.3.3)$$

where P_j is the nodal value of the pressure P at the node j, and P_{N_P} is the additive constant with respect to the spatial variables, which is a prescribed value for the pressure at a certain pressure node, say N_P.

The unknowns are now the nodal values of the velocity components at the velocity nodes in $D \cup \Gamma_N$ and the nodal values of the pressure at all pressure nodes except for one, where it is prescribed. Hence the total number of unknowns is $2N + N_P - 1$.

In the equations (3.3.1-3), the unknown parts of the approximate velocities and pressure belong to the following finite element spaces

$$u^h_{D\cup\Gamma_N}, v^h_{D\cup\Gamma_N} \in V^h_N \equiv span\{\varphi_1,\ldots,\varphi_N\},$$

$$P^h_{nc} \in S^h_{N_P-1} \equiv span\{\psi_1,\ldots,\psi_{N_P-1}\}.$$

These are the solutions to the approximate weak form obtained from the weak form (3.2.10-12) by restricting the original function spaces to the finite element spaces above mentioned. The approximate weak form reads as follows,

Find $u^h_{D\cup\Gamma_N}, v^h_{D\cup\Gamma_N} \in V^h_N$, and $P^h_{nc} \in S^h_{N_P-1}$ such that

$$\begin{aligned}
&\int_D \varphi \frac{\partial}{\partial t}(u^h_{D\cup\Gamma_N} + u^h_{\Gamma_D})dD + \int_D \varphi(u^h_{D\cup\Gamma_N} + u^h_{\Gamma_D})\frac{\partial}{\partial x}(u^h_{D\cup\Gamma_N} + u^h_{\Gamma_D})dD \\
&+ \int_D \varphi(v^h_{D\cup\Gamma_N} + v^h_{\Gamma_D})\frac{\partial}{\partial y}(u^h_{D\cup\Gamma_N} + u^h_{\Gamma_D})dD + \int_D A_H \nabla\varphi\nabla(u^h_{D\cup\Gamma_N} + u^h_{\Gamma_D})dD \\
&- \int_D f\varphi(v^h_{D\cup\Gamma_N} + v^h_{\Gamma_D})dD + \frac{1}{\rho_0}\int_D \varphi\frac{\partial}{\partial x}(P^h_{nc} + P^h_c)dD \\
&\qquad = \frac{1}{\rho_0}\int_D \varphi F_x dD + \int_{\Gamma_N} A_H \varphi U_n d\Gamma, \qquad (3.3.4)
\end{aligned}$$

$$\begin{aligned}
&\int_D \varphi \frac{\partial}{\partial t}(v^h_{D\cup\Gamma_N} + v^h_{\Gamma_D})dD + \int_D \varphi(u^h_{D\cup\Gamma_N} + u^h_{\Gamma_D})\frac{\partial}{\partial x}(v^h_{D\cup\Gamma_N} + v^h_{\Gamma_D})dD \\
&+ \int_D \varphi(v^h_{D\cup\Gamma_N} + v^h_{\Gamma_D})\frac{\partial}{\partial y}(v^h_{D\cup\Gamma_N} + v^h_{\Gamma_D})dD + \int_D A_H \nabla\varphi\nabla(v^h_{D\cup\Gamma_N} + v^h_{\Gamma_D})dD \\
&+ \int_D f\varphi(u^h_{D\cup\Gamma_N} + u^h_{\Gamma_D})dD + \frac{1}{\rho_0}\int_D \varphi\frac{\partial}{\partial y}(P^h_{nc} + P^h_c)dD \\
&\qquad = \frac{1}{\rho_0}\int_D \varphi F_y dD + \int_{\Gamma_N} A_H \varphi V_n d\Gamma, \qquad (3.3.5)
\end{aligned}$$

$$\int_D [\psi \frac{\partial}{\partial x}(u^h_{D\cup\Gamma_N} + u^h_{D\cup\Gamma_N}) + \psi\frac{\partial}{\partial y}(v^h_{D\cup\Gamma_N} + v^h_{D\cup\Gamma_N})]dD = 0. \qquad (3.3.6)$$

for all $\varphi \in V_N^h$ and all $\psi \in S_{N_P-1}^h$. Note that $V_N^h \subset H_0^1(D)$ and $S_{N_P-1}^h \subset H^1(D)$.

Since $V_N^h \equiv span\{\varphi_1,...,\varphi_N\}$ and $S_{N_P-1}^h \equiv span\{\psi_1,...,\psi_{N_P-1}\}$, the equations (3.3.4-6) hold true for all $\varphi \in V_N^h$ and all $\psi \in S_{N_P-1}^h$ if and only if they hold true for the basis functions of these finite element spaces, i.e. for any $\varphi \in \{\varphi_1,...,\varphi_N\}$ and any $\psi \in \{\psi_1,...,\psi_{N_P-1}\}$. In this case, the test functions are the same as the shape functions, and the method is called the *Galerkin Finite Element Method*. The approximate problem can be reformulated as follows,

Find $u_j(t), v_j(t)$ for $j=1,...,N$, and $P_j(t)$ for $j=1,...,N_P-1$, such that

$$
\begin{aligned}
&\sum_{j=1}^{N}\frac{\partial u_j}{\partial t}\int_D \varphi_i\varphi_j dD + \sum_{j=1}^{N} u_j \int_D \left[(\sum_{k=1}^{N_T}\varphi_k u_k)\varphi_i \frac{\partial \varphi_j}{\partial x} + (\sum_{k=1}^{N_T}\varphi_k v_k)\varphi_i \frac{\partial \varphi_j}{\partial y} \right] dD \\
&+\sum_{j=1}^{N} u_j \int_D A_H \nabla\varphi_i \nabla\varphi_j dD - \sum_{j=1}^{N} v_j \int_D f\varphi_i\varphi_j dD + \sum_{j=1}^{N_P-1} P_j \frac{1}{\rho_0}\int_D \varphi_i \frac{\partial \psi_j}{\partial x} dD \\
&= \frac{1}{\rho_0}\int_D \varphi_i F_x dD + \int_{\Gamma_N} A_H \varphi_i U_n d\Gamma \\
&-\left\{ \sum_{j=N+1}^{N_T}\frac{\partial U_j}{\partial t}\int_D \varphi_i\varphi_j dD + \sum_{j=N+1}^{N_T} U_j \int_D \left[(\sum_{k=1}^{N_T}\varphi_k u_k)\varphi_i \frac{\partial \varphi_j}{\partial x} + (\sum_{k=1}^{N_T}\varphi_k v_k)\varphi_i \frac{\partial \varphi_j}{\partial y} \right] dD \right\} \\
&-\left\{ \sum_{j=N+1}^{N_T} U_j \int_D A_H \nabla\varphi_i \nabla\varphi_j dD - \sum_{j=N+1}^{N_T} V_j \int_D f\varphi_i\varphi_j dD \right\} \\
&\qquad\qquad - P_{N_P}\frac{1}{\rho_0}\int_D \varphi_i \frac{\partial \psi_{N_P}}{\partial x} dD, \qquad i = 1,...,N \qquad (3.3.7)
\end{aligned}
$$

$$\sum_{j=1}^{N} \frac{\partial v_j}{\partial t} \int_D \varphi_i \varphi_j dD + \sum_{j=1}^{N} v_j \int_D \left[\left(\sum_{k=1}^{N_T} \varphi_k u_k\right) \varphi_i \frac{\partial \varphi_j}{\partial x} + \left(\sum_{k=1}^{N_T} \varphi_k v_k\right) \varphi_i \frac{\partial \varphi_j}{\partial y} \right] dD$$

$$+ \sum_{j=1}^{N} v_j \int_D A_H \nabla \varphi_i \nabla \varphi_j dD + \sum_{j=1}^{N} u_j \int_D f \varphi_i \varphi_j dD + \sum_{j=1}^{N_P - 1} P_j \frac{1}{\rho_0} \int_D \varphi_i \frac{\partial \psi_j}{\partial y} dD$$

$$= \frac{1}{\rho_0} \int_D \varphi_i F_y dD + \int_{\Gamma_N} A_H \varphi_i V_n d\Gamma$$

$$- \left\{ \sum_{j=N+1}^{N_T} \frac{\partial V_j}{\partial t} \int_D \varphi_i \varphi_j dD + \sum_{j=N+1}^{N_T} V_j \int_D \left[\left(\sum_{k=1}^{N_T} \varphi_k u_k\right) \varphi_i \frac{\partial \varphi_j}{\partial x} + \left(\sum_{k=1}^{N_T} \varphi_k v_k\right) \varphi_i \frac{\partial \varphi_j}{\partial y} \right] dD \right\}$$

$$- \left\{ \sum_{j=N+1}^{N_T} V_j \int_D A_H \nabla \varphi_i \nabla \varphi_j dD + \sum_{j=N+1}^{N_T} U_j \int_D f \varphi_i \varphi_j dD \right\}$$

$$- P_{N_P} \frac{1}{\rho_0} \int_D \varphi_i \frac{\partial \psi_{N_P}}{\partial y} dD, \quad i = 1, \ldots, N, \qquad (3.3.8)$$

$$\sum_{j=1}^{N} u_j \int_D \psi_i \frac{\partial \varphi_j}{\partial x} dD + \sum_{j=1}^{N} v_j \int_D \psi_i \frac{\partial \varphi_j}{\partial y} dD$$

$$= - \sum_{j=N+1}^{N_T} U_j \int_D \psi_i \frac{\partial \varphi_j}{\partial x} dD - \sum_{j=N+1}^{N_T} V_j \int_D \psi_i \frac{\partial \varphi_j}{\partial y} dD, \quad i = 1, \ldots, N_P - 1. \qquad (3.3.9)$$

These equations form a system of non-linear differential equations - the set of equations (3.3.7-8) - with algebraic constraints, given by the set of equations (3.3.9); they are called *differential-algebraic equations* (Gresho et al., 1999). Notice that the algebraic constraints originate from the incompressibility constraint represented by the continuity equation (2.2.4).

Since only the spatial domain has been discretized, while the time variable remained continuous, the equations (3.3.7-9) represent a set of *semi-discrete* equations.

Note that the right-hand side of the equations (3.3.7-8) contains not only given values (data) but also the unknowns u_j, v_j. This is caused by the non-linearity of the advection term. There is a multitude of methods that can be used in finite element algorithms to deal with this non-linearity. The most widely used are the Picard iteration and the Newton-Raphson method.

3.4 Picard Iteration

In order to linearize the non-linear advective terms in the semi-discrete Navier-Stokes equations, the Picard iteration method can be applied. Assuming that the time derivatives are zero (steady-state case) in the equations (3.3.7-8), we get a non-linear system of algebraic equations. The Picard iteration consists of a series of successive iterations, in which the original non-linear system is linearized by using the velocity components obtained in the previous iteration for the transport-velocity components in the advective terms - expressed by the sums $\sum_{k=1}^{N_T} \varphi_k u_k$ and $\sum_{k=1}^{N_T} \varphi_k v_k$ in the equations (3.3.7-8) - at the current iteration. The resulting linear system of algebraic equations is then solved using an appropriate numerical method.

The same method can be applied for the general case, in which the time derivatives are non-zero. Assuming that the time domain is discretized via a finite difference technique, the Picard iteration can be applied at each time step.

There are many variations of the Picard iteration method. The variation used in the following is that presented in Taylor et al. (1981). The transport velocity to be used in the next iteration is obtained by relaxing the velocity solution obtained for the current iteration with the (relaxed) velocity passed from the previous iteration. The resulting linearized semi-discrete equations derived from the equations (3.3.7-9), and written for the m^{th} iteration, are:

$$\sum_{j=1}^{N}\frac{\partial u_j^m}{\partial t}\int_D \varphi_i\varphi_j dD+\sum_{j=1}^{N}u_j^m\int_D\left[(\sum_{k=1}^{N_T}\varphi_k\tilde{u}_k^{m-1})\varphi_i\frac{\partial\varphi_j}{\partial x}+(\sum_{k=1}^{N_T}\varphi_k\tilde{v}_k^{m-1})\varphi_i\frac{\partial\varphi_j}{\partial y}\right]dD$$

$$+\sum_{j=1}^{N}u_j^m\int_D A_H\nabla\varphi_i\nabla\varphi_j dD-\sum_{j=1}^{N}v_j^m\int_D f\varphi_i\varphi_j dD+\sum_{j=1}^{N_P-1}P_j^m\frac{1}{\rho_0}\int_D\varphi_i\frac{\partial\psi_j}{\partial x}dD$$

$$=\frac{1}{\rho_0}\int_D\varphi_i F_x dD+\int_{\Gamma_N}A_H\varphi_i U_n d\Gamma$$

$$-\left\{\sum_{j=N+1}^{N_T}\frac{\partial U_j}{\partial t}\int_D\varphi_i\varphi_j dD+\sum_{j=N+1}^{N_T}U_j\int_D\left[(\sum_{k=1}^{N_T}\varphi_k\tilde{u}_k^{m-1})\varphi_i\frac{\partial\varphi_j}{\partial x}+(\sum_{k=1}^{N_T}\varphi_k\tilde{v}_k^{m-1})\varphi_i\frac{\partial\varphi_j}{\partial y}\right]dD\right\}$$

$$-\left\{\sum_{j=N+1}^{N_T}U_j\int_D A_H\nabla\varphi_i\nabla\varphi_j dD-\sum_{j=N+1}^{N_T}V_j\int_D f\varphi_i\varphi_j dD\right\}$$

$$-P_{N_P}\frac{1}{\rho_0}\int_D\varphi_i\frac{\partial\psi_{N_P}}{\partial x}dD,\qquad i=1,\dots,N\qquad(3.4.1)$$

$$\sum_{j=1}^{N}\frac{\partial v_j^m}{\partial t}\int_D \varphi_i\varphi_j dD+\sum_{j=1}^{N}v_j^m\int_D\left[(\sum_{k=1}^{N_T}\varphi_k\tilde{u}_k^{m-1})\varphi_i\frac{\partial\varphi_j}{\partial x}+(\sum_{k=1}^{N_T}\varphi_k\tilde{v}_k^{m-1})\varphi_i\frac{\partial\varphi_j}{\partial y}\right]dD$$

$$+\sum_{j=1}^{N} v_j^m \int_D A_H \nabla\varphi_i \nabla\varphi_j dD + \sum_{j=1}^{N} u_j^m \int_D f\varphi_i\varphi_j dD + \sum_{j=1}^{N_P-1} P_j^m \frac{1}{\rho_0}\int_D \varphi_i \frac{\partial\psi_j}{\partial y} dD$$

$$= \frac{1}{\rho_0}\int_D \varphi_i F_y dD + \int_{\Gamma_N} A_H \varphi_i V_n d\Gamma$$

$$-\left\{ \sum_{j=N+1}^{N_T} \frac{\partial V_j}{\partial t} \int_D \varphi_i\varphi_j dD + \sum_{j=N+1}^{N_T} V_j \int_D \left[(\sum_{k=1}^{N_T}\varphi_k \tilde{u}_k^{m-1})\varphi_i \frac{\partial\varphi_j}{\partial x} + (\sum_{k=1}^{N_T}\varphi_k \tilde{v}_k^{m-1})\varphi_i \frac{\partial\varphi_j}{\partial y} \right] dD \right\}$$

$$-\left\{ \sum_{j=N+1}^{N_T} V_j \int_D A_H \nabla\varphi_i \nabla\varphi_j dD + \sum_{j=N+1}^{N_T} U_j \int_D f\varphi_i\varphi_j dD \right\}$$

$$- P_{N_P} \frac{1}{\rho_0}\int_D \varphi_i \frac{\partial\psi_{N_P}}{\partial y} dD, \qquad i=1,\dots,N, \qquad (3.4.2)$$

$$\sum_{j=1}^{N} u_j^m \int_D \psi_i \frac{\partial\varphi_j}{\partial x} dD + \sum_{j=1}^{N} v_j^m \int_D \psi_i \frac{\partial\varphi_j}{\partial y} dD$$

$$= -\sum_{j=N+1}^{N_T} U_j \int_D \psi_i \frac{\partial\varphi_j}{\partial x} dD - \sum_{j=N+1}^{N_T} V_j \int_D \psi_i \frac{\partial\varphi_j}{\partial y} dD, \quad i=1,\dots,N_P-1, \quad (3.4.3)$$

where u_j^m ($j=1,\dots,N$), v_j^m ($j=1,\dots,N$), and P_j^m ($j=1,\dots,N_P-1$) are the nodal velocities and pressure values at the m^{th} iteration, whereas $\tilde{u}_j^{m-1}$ ($j=1,\dots,N_T$) and $\tilde{v}_j^{m-1}$ ($j=1,\dots,N_T$) are the relaxed velocities passed from the previous iteration, with $u_j^{m-1}=U_j$ and $u_j^{m-1}=U_j$ for $j=N+1,\dots,N_T$.

The relaxed nodal velocity values to be used in the next iteration (iteration $m+1$) are given by

$$\tilde{u}_j^m = r u_j^m + (1-r)\tilde{u}_j^{m-1}, \qquad j = 1,\dots,N, \tag{3.4.4}$$

$$\tilde{v}_j^m = r v_j^m + (1-r)\tilde{v}_j^{m-1}, \qquad j = 1,\dots,N, \tag{3.4.5}$$

where r is the relaxation coefficient, $r \in (0,1)$.

The velocity values $\tilde{u}_j^0$ and $\tilde{v}_j^0$ for $j = 1,\dots,N$, needed in the first iteration, must be specified. The iterative process stops when a specified tolerance is satisfied by a relative error involving the nodal velocity values obtained at the current iteration and the nodal values of the relaxed velocity passed from the previous iteration, in case it is convergent. The conditions of convergence within a specified tolerance *toler* are

$$\frac{u_j^m - \tilde{u}_j^{m-1}}{u_j^m} < toler, \quad \frac{v_j^m - \tilde{v}_j^{m-1}}{v_j^m} < toler, \quad j = 1,\dots,N. \tag{3.4.6}$$

Notice that, in the steady-state case, the equations (3.4.1-3) form a set of linear system of algebraic equations with the right-hand side containing known values only.

3.5 Matrix Formulation

The linearized semi-discrete Navier-Stokes equations (3.4.1-3) can be expressed in the matrix-vector form as follows

$$\mathbf{M}\frac{\partial \mathbf{u}^m}{\partial t} + [\mathbf{N}(\tilde{\mathbf{u}}^{m-1}) + \mathbf{K} + \mathbf{C}]\mathbf{u}^m + \mathbf{G}\mathbf{P}^m = \mathbf{F}(\tilde{\mathbf{u}}^{m-1}), \tag{3.5.1}$$

$$\mathbf{D}\mathbf{u}^m = \mathbf{H}. \tag{3.5.2}$$

The matrices in the equations (3.5.1-2) are

- the *mass matrix* $\mathbf{M} = \begin{bmatrix} \mathbf{M}^{\mathbf{u}} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}^{\mathbf{v}} \end{bmatrix}$, of dimension $2N \times 2N$; the sub-matrices $\mathbf{M}^{\mathbf{u}}$ and $\mathbf{M}^{\mathbf{v}}$ have the entries

$$\mathbf{M}^{\mathbf{u}}{}_{ij} = \mathbf{M}^{\mathbf{v}}{}_{ij} = \int_D \varphi_i \varphi_j dD \quad \text{for } i, j = 1, \ldots, N ; \tag{3.5.3}$$

- the *advection matrix* $\mathbf{N}(\tilde{\mathbf{u}}^{m-1}) = \begin{bmatrix} \mathbf{N}^{\mathbf{u}}(\tilde{\mathbf{u}}^{m-1}) & \mathbf{0} \\ \mathbf{0} & \mathbf{N}^{\mathbf{v}}(\tilde{\mathbf{u}}^{m-1}) \end{bmatrix}$, of dimension $2N \times 2N$; the submatrices $\mathbf{N}^{\mathbf{u}}(\tilde{\mathbf{u}}^{m-1})$ and $\mathbf{N}^{\mathbf{v}}(\tilde{\mathbf{u}}^{m-1})$ have the entries

$$\mathbf{N}^{\mathbf{u}}(\tilde{\mathbf{u}}^{m-1})_{ij} = \mathbf{N}^{\mathbf{v}}(\tilde{\mathbf{u}}^{m-1})_{ij} = \int_D \left[(\sum_{k=1}^{N_T} \varphi_k \tilde{u}_k^{m-1}) \varphi_i \frac{\partial \varphi_j}{\partial x} + (\sum_{k=1}^{N_T} \varphi_k \tilde{v}_k^{m-1}) \varphi_i \frac{\partial \varphi_j}{\partial y} \right] dD$$

$$\text{for } i, j = 1, \ldots, N ; \tag{3.5.4}$$

- the *diffusion matrix* $\mathbf{K} = \begin{bmatrix} \mathbf{K}^{\mathbf{u}} & \mathbf{0} \\ \mathbf{0} & \mathbf{K}^{\mathbf{v}} \end{bmatrix}$, of dimension $2N \times 2N$; the sub-matrices $\mathbf{K}^{\mathbf{u}}$ and $\mathbf{K}^{v}$ have the entries

$$\mathbf{K}^{\mathbf{u}}{}_{ij} = \mathbf{K}^{v}{}_{ij} = \int_D A_H \nabla \varphi_i \nabla \varphi_j dD \quad \text{for } i, j = 1, \ldots, N ; \tag{3.5.5}$$

- the *Coriolis matrix* $\mathbf{C}=\begin{bmatrix}\mathbf{0} & \mathbf{C}^{\mathbf{u}}\\ \mathbf{C}^{\mathbf{v}} & \mathbf{0}\end{bmatrix}$, of dimension $2N\times 2N$; the sub-matrices $\mathbf{C}^{\mathbf{u}}$ and $\mathbf{C}^{\mathbf{v}}$ have the entries

$$\mathbf{C}^{\mathbf{u}}{}_{ij} = -\int_D f\varphi_i\varphi_j dD\,, \quad \text{for } i,j=1,\dots,N\,, \tag{3.5.6}$$

$$\mathbf{C}^{\mathbf{v}}{}_{ij} = \int_D f\varphi_i\varphi_j dD\,, \quad \text{for } i,j=1,\dots,N\,; \tag{3.5.7}$$

- the *gradient matrix* $\mathbf{G}=\begin{bmatrix}\mathbf{G}^{\mathbf{u}}\\ \mathbf{G}^{\mathbf{v}}\end{bmatrix}$, of dimension $2N\times(N_P-1)$; the sub-matrices $\mathbf{G}^{\mathbf{u}}$ and $\mathbf{G}^{\mathbf{v}}$ have the entries

$$\mathbf{G}^{\mathbf{u}}{}_{ij} = \frac{1}{\rho_0}\int_D \varphi_i\frac{\partial\psi_j}{\partial x}dD\,, \quad \text{for } i=1,\dots,N \text{ and } j=1,\dots,N_P-1\,, \tag{3.5.8}$$

$$\mathbf{G}^{\mathbf{v}}{}_{ij} = \frac{1}{\rho_0}\int_D \varphi_i\frac{\partial\psi_j}{\partial y}dD\,, \quad \text{for } i=1,\dots,N \text{ and } j=1,\dots,N_P-1\,; \tag{3.5.9}$$

- the *divergence matrix* $\mathbf{D}=[\mathbf{D}^{\mathbf{u}} \quad \mathbf{D}^{\mathbf{v}}]$, of dimension $(N_P-1)\times 2N$; the sub-matrices $\mathbf{D}^{\mathbf{u}}$ and $\mathbf{D}^{\mathbf{v}}$ have the entries

$$\mathbf{D}^{\mathbf{u}}{}_{ij} = \int_D \psi_i\frac{\partial\varphi_j}{\partial x}dD\,, \quad \text{for } i=1,\dots,N_P-1 \text{ and } j=1,\dots,N\,, \tag{3.5.10}$$

$$\mathbf{D}^{\mathbf{v}}{}_{ij} = \int_D \psi_i\frac{\partial\varphi_j}{\partial y}dD\,, \quad \text{for } i=1,\dots,N_P-1 \text{ and } j=1,\dots,N\,. \tag{3.5.11}$$

The vectors that contain the unknowns in the equations (3.5.1-2) are

- the vector of the nodal values of the velocity components at iteration m,

$$\mathbf{u}^m = [u_1^m \quad \dots \quad u_N^m \quad v_1^m \quad \dots \quad v_N^m]^{\mathrm{T}} \tag{3.5.12}$$

and its derivative with respect to time, $\dfrac{\partial \mathbf{u}^m}{\partial t}$.

- the vector of the nodal values of the pressure at iteration m:

$$\mathbf{P}^m = [P_1^m \quad \dots \quad P_{N_P-1}^m]^{\mathrm{T}}. \tag{3.5.13}$$

The right-hand side vectors in the equations (3.5.1) and (3.5.2) are $\mathbf{F}(\tilde{\mathbf{u}}^{m-1}) = \begin{bmatrix} \mathbf{F}^{\mathbf{u}}(\tilde{\mathbf{u}}^{m-1}) \\ \mathbf{F}^{\mathbf{v}}(\tilde{\mathbf{u}}^{m-1}) \end{bmatrix}$ of length $2N$, and $\mathbf{H}$ of length N_P-1 with the entries

$$\begin{aligned}
\mathbf{F}^{\mathbf{u}}(\tilde{\mathbf{u}}^{m-1})_i &= \frac{1}{\rho_0} \int_D \varphi_i F_x dD + \int_{\Gamma_N} A_H \varphi_i U_n d\Gamma \\
&- \left\{ \sum_{j=N+1}^{N_T} \frac{\partial U_j}{\partial t} \int_D \varphi_i \varphi_j dD + \sum_{j=N+1}^{N_T} U_j \int_D \left[(\sum_{k=1}^{N_T} \varphi_k \tilde{u}_k^{m-1}) \varphi_i \frac{\partial \varphi_j}{\partial x} + (\sum_{k=1}^{N_T} \varphi_k \tilde{v}_k^{m-1}) \varphi_i \frac{\partial \varphi_j}{\partial y} \right] dD \right\} \\
&- \left\{ \sum_{j=N+1}^{N_T} U_j \int_D A_H \nabla \varphi_i \nabla \varphi_j dD - \sum_{j=N+1}^{N_T} V_j \int_D f \varphi_i \varphi_j dD \right\} \\
&- P_{N_P} \frac{1}{\rho_0} \int_D \varphi_i \frac{\partial \psi_{N_P}}{\partial x} dD, \qquad i = 1,\dots,N,
\end{aligned} \tag{3.5.14}$$

$$\mathbf{F}^{\mathrm{v}}(\tilde{\mathbf{u}}^{m-1})_i = \frac{1}{\rho_0}\int_D \varphi_i F_y dD + \int_{\Gamma_N} A_H \varphi_i V_n d\Gamma$$

$$-\left\{\sum_{j=N+1}^{N_T} \frac{\partial V_j}{\partial t}\int_D \varphi_i\varphi_j dD + \sum_{j=N+1}^{N_T} V_j \int_D\left[(\sum_{k=1}^{N_T}\varphi_k \tilde{u}_k^{m-1})\varphi_i \frac{\partial \varphi_j}{\partial x} + (\sum_{k=1}^{N_T}\varphi_k \tilde{v}_k^{m-1})\varphi_i \frac{\partial \varphi_j}{\partial y}\right]dD\right\}$$

$$-\left\{\sum_{j=N+1}^{N_T} V_j \int_D A_H \nabla\varphi_i \nabla\varphi_j dD + \sum_{j=N+1}^{N_T} U_j \int_D f\varphi_i\varphi_j dD\right\}$$

$$-P_{N_P}\frac{1}{\rho_0}\int_D \varphi_i \frac{\partial \psi_{N_P}}{\partial y} dD, \quad i = 1,\dots,N, \qquad (3.5.15)$$

$$\mathbf{H}_i = -\sum_{j=N+1}^{N_T} U_j \int_D \psi_i \frac{\partial \varphi_j}{\partial x} dD - \sum_{j=N+1}^{N_T} V_j \int_D \psi_i \frac{\partial \varphi_j}{\partial y} dD, \quad i = 1,\dots,N_P - 1. \qquad (3.5.16)$$

The vector $\mathbf{F}(\tilde{\mathbf{u}}^{m-1})$ incorporates the forcing terms as well as the natural and essential boundary conditions. Furthermore, it is updated each iteration, and so is the advection matrix.

3.6 Taylor-Hood Serendipity Quadrilateral Finite Element

Although the finite element meshes based on triangles have the advantage of a more accurate representation of very complex boundaries, they induce mesh orientation effects (Figure 3.1), which are more noticeable for rectangular grids (Cuvelier et al., 1986). These effects become important as the Reynolds number of the flow increases (i.e. the flow enters the regime of convection-dominated flows). The use of a quadrilateral finite element mesh leads to more accurate results (Gresho et al., 1999) and a decreased number of elements, when compared to a triangular finite element mesh obtained from it by diving the quadrilaterals as in Figure 3.1.

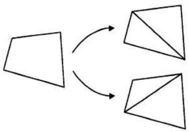


Figure 3.1 Mesh orientation effects are caused by different diagonal directions when a quadrilateral finite element is divided into two triangular finite elements

There are two types of quadrilateral finite elements used in computational fluid dynamics: Taylor-Hood finite elements and Crouzeix-Raviart finite elements. The first type encompasses all the finite elements with continuous pressure on the element boundary, whereas the second includes all those with discontinuous pressure, and requires smoothing of the resulting pressure solution.

The finite element used hereafter is the Taylor-Hood serendipity quadrilateral finite element, and is of Taylor-Hood type. This element offers mixed interpolation for velocity and pressure, and satisfies the Babuska-Brezzi stability condition (Gresho et al. 1999). At the element level, the velocity components are approximated by (incomplete) quadratic polynomials, considered to be sufficient for most simulations, while the pressure is approximated by a linear polynomial. There are eight nodes for velocity and four nodes for pressure. The velocity nodes are located at the corners and at the mid-side points of the element (Figure 3.2). The pressure nodes coincide with the corner velocity nodes. Thus, the corner nodes have 3 degrees of freedom each corresponding to the nodal values of u, v, and P, while the mid-side nodes have 2 degrees of freedom that correspond to the nodal values of u and v. The position of the velocity and pressure nodes in this element assures continuity on the element boundary.

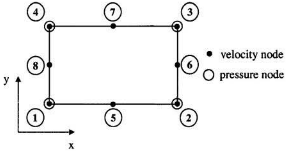


Figure 3.2 Taylor-Hood serendipity quadrilateral finite element

The local shape function φ_i^e corresponding to the velocity node i ($i=1,\dots,8$) of the finite element e, covering the subdomain $D_e \subset D$, is

$$\varphi_i^e(x,y)=c_1+c_2x+c_3y+c_4xy+c_5x^2+c_6x^2y+c_7xy^2+c_8y^2,\ (x,y)\in D_e, \quad (3.6.1)$$

with the constants $c_1,\dots,c_8$ to be determined such that

$$\varphi_i^e(x_j,y_j)=\delta_{ij}\ , \quad (3.6.2)$$

where x_j and y_j are the coordinates of the node j $(j=1,\ldots,8)$. Once these functions are known, the variation of the velocity components u and v over the element e is expressed as a linear combination of the local shape functions φ_i^e, whose coefficients are the nodal values of u and v, respectively

$$u^e(x,y,t)=\sum_{i=1}^{8}u_{(i)}(t)\varphi_i^e(x,y), \qquad v^e(x,y,t)=\sum_{i=1}^{8}v_{(i)}(t)\varphi_i^e(x,y) \tag{3.6.3}$$

where $(x,y,t)\in D_e\times[0,t_{final}]$; the subscript i included in round brackets indicates that the local numbering of nodes is used.

The local shape function ψ_i^e corresponding to the pressure node i $(i=1,\ldots,4)$ of the finite element e is

$$\psi_i^e(x,y)=d_1+d_2x+d_3y+d_4xy\,, \quad (x,y)\in D_e\,, \tag{3.6.4}$$

with the constants $d_1,\ldots,d_4$ to be determined such that

$$\psi_i^e(x_j,y_j)=\delta_{ij}\,, \tag{3.6.5}$$

where x_j and y_j are the coordinates of the pressure node j $(j=1,\ldots,4)$. Once these functions are determined, the variation of the pressure over the element e is expressed as a linear combination of the local pressure shape functions, whose coefficients are the nodal values of the pressure

$$P^{e}(x,y,t)=\sum_{i=1}^{4}P_{(i)}(t)\psi_{i}^{e}(x,y), \qquad (x,y,t)\in D_{e}\times[0,t_{final}]. \qquad (3.6.6)$$

Assuming that the spatial domain D has been discretized into a number N_e of quadrilateral finite elements e_k ($k=1,\ldots,N_e$), such that $\overline{D}=\bigcup_{k=1}^{N_e}D_{e_k}$, the approximate velocity components and pressure over the whole domain can be expressed as:

$$u^{h}(x,y,t)=\sum_{k=1}^{N_e}\left[\sum_{i=1}^{8}u_{(i)}(t)\varphi_{i}^{e_k}(x,y)\right], \qquad (3.6.7)$$

$$v^{h}(x,y,t)=\sum_{k=1}^{N_e}\left[\sum_{i=1}^{8}v_{(i)}(t)\varphi_{i}^{e_k}(x,y)\right], \qquad (3.6.8)$$

$$P^{h}(x,y,t)=\sum_{k=1}^{N_e}\left[\sum_{i=1}^{4}P_{(i)}(t)\psi_{i}^{e_k}(x,y)\right]. \qquad (3.6.9)$$

Although the summation in the equations (3.6.7-9) is over the elements of the finite element mesh, these equations are equivalent to the equations (3.3.1-3), respectively, where the summation is over the nodes of the mesh. The following set of semi-discrete equations, which is equivalent to the equations (3.4.1-3), can be obtained if the sum in the expressions of approximate solutions is over the elements

$$\sum_{k=1}^{N_e}\left[\sum_{j=1}^{8}\frac{\partial u_{(j)}^{m}}{\partial t}\int_{D_{e_k}}\varphi_{i}\varphi_{j}^{e_k}\,dD\right]$$

$$+\sum_{k=1}^{N_e}\left\{\sum_{j=1}^{8}u_{(j)}^{m}\int_{D_{e_k}}\left[\left(\sum_{l=1}^{8}\varphi_l^{e_k}\tilde{u}_{(l)}^{m-1}\right)\varphi_i\frac{\partial\varphi_j^{e_k}}{\partial x}+\left(\sum_{l=1}^{8}\varphi_l^{e_k}\tilde{v}_{(l)}^{m-1}\right)\varphi_i\frac{\partial\varphi_j^{e_k}}{\partial y}\right]dD\right\}$$

$$+\sum_{k=1}^{N_e}\left[\sum_{j=1}^{8}u_{(j)}^{m}\int_{D_{e_k}}A_H\nabla\varphi_i\nabla\varphi_j^{e_k}\,dD\right]-\sum_{k=1}^{N_e}\left[\sum_{j=1}^{8}v_{(j)}^{m}\int_{D_{e_k}}f\varphi_i\varphi_j^{e_k}\,dD\right]$$

$$+\sum_{k=1}^{N_e}\left[\sum_{j=1}^{4}P_{(j)}^{m}\frac{1}{\rho_0}\int_{D_{e_k}}\varphi_i\frac{\partial\psi_j^{e_k}}{\partial x}dD\right]$$

$$=\sum_{k=1}^{N_e}\frac{1}{\rho_0}\int_{D_{e_k}}\varphi_i F_x\,dD+\int_{\Gamma_N}A_H\varphi_i U_n\,d\Gamma,\qquad i=1,\dots,N_T\ ,\qquad(3.6.10)$$

$$\sum_{k=1}^{N_e}\left[\sum_{j=1}^{8}\frac{\partial v_{(j)}^{m}}{\partial t}\int_{D_{e_k}}\varphi_i\varphi_j^{e_k}\,dD\right]$$

$$+\sum_{k=1}^{N_e}\left\{\sum_{j=1}^{8}v_{(j)}^{m}\int_{D_{e_k}}\left[\left(\sum_{l=1}^{8}\varphi_l^{e_k}\tilde{u}_{(l)}^{m-1}\right)\varphi_i\frac{\partial\varphi_j^{e_k}}{\partial x}+\left(\sum_{l=1}^{8}\varphi_l^{e_k}\tilde{v}_{(l)}^{m-1}\right)\varphi_i\frac{\partial\varphi_j^{e_k}}{\partial y}\right]dD\right\}$$

$$+\sum_{k=1}^{N_e}\left[\sum_{j=1}^{8}v_{(j)}^{m}\int_{D_{e_k}}A_H\nabla\varphi_i\nabla\varphi_j^{e_k}\,dD\right]+\sum_{k=1}^{N_e}\left[\sum_{j=1}^{8}u_{(j)}^{m}\int_{D_{e_k}}f\varphi_i\varphi_j^{e_k}\,dD\right]$$

$$+\sum_{k=1}^{N_e}\left[\sum_{j=1}^{4}P_{(j)}^{m}\frac{1}{\rho_0}\int_{D_{e_k}}\varphi_i\frac{\partial\psi_j^{e_k}}{\partial y}dD\right]$$

$$=\sum_{k=1}^{N_e}\frac{1}{\rho_0}\int_{D_{e_k}}\varphi_i F_y\,dD+\int_{\Gamma_N}A_H\varphi_i V_n\,d\Gamma,\qquad i=1,\dots,N_T\ ,\qquad(3.6.11)$$

$$\sum_{k=1}^{N_e}\left[\sum_{j=1}^{8}u_{(j)}^{m}\int_{D_{e_k}}\psi_i\frac{\partial\varphi_j^{e_k}}{\partial x}dD\right]+\sum_{k=1}^{N_e}\left[\sum_{j=1}^{8}v_{(j)}^{m}\int_{D_{e_k}}\psi_i\frac{\partial\varphi_j^{e_k}}{\partial y}dD\right]=0,\ i=1,\dots,N_P. \quad (3.6.12)$$

The matrix-vector form of the set of equations (3.6.10-12) is

$$\mathbf{M_g}\frac{\partial\mathbf{u_g}^m}{\partial t}+[\mathbf{N_g}(\tilde{\mathbf{u}}_\mathbf{g}^{m-1})+\mathbf{K_g}+\mathbf{C_g}]\mathbf{u_g}^m+\mathbf{G_g}\mathbf{P_g}^m=\mathbf{F_g}, \quad (3.6.13)$$

$$\mathbf{D_g}\mathbf{u_g}^m=\mathbf{0}. \quad (3.6.14)$$

The matrices in the above equations are similar to those in the matrix-vector form (3.5.1-2), the only difference consisting in their dimensions. The vectors $\mathbf{u_g^m}$ and $\frac{\partial\mathbf{u_g}^m}{\partial t}$ contain all nodal values for the velocity components and their derivatives with respect to time, respectively; therefore their size is $2N_T$. Similarly, the size of the $\mathbf{P_g^m}$ vector is N_P, as it contains all the nodal pressure values. Consequently, the dimensions of the matrices in (3.6.13-14) are greater than those of their correspondents in (3.5.1-2). The subscript g indicates matrices and vectors that are *global*, i.e. they are generated taking into account all nodes in the finite element mesh (including those on the Dirichlet boundary Γ_D).

The equations (3.6.10-12) show that the global matrices can be written as sums of local matrices whose entries are calculated at the finite element level as follows:

- the *global mass matrix* $\mathbf{M_g}$, of dimension $2N_T \times 2N_T$, can be expressed as

$\mathbf{M_g} = \begin{bmatrix} \mathbf{M_g^u} & \mathbf{0} \\ \mathbf{0} & \mathbf{M_g^v} \end{bmatrix}$, with $\mathbf{M_g^u} = \mathbf{M_g^v} = \sum_{k=1}^{N_e} \mathbf{M}^{e_k}$. The matrix $\mathbf{M}^{e_k}$ $(k=1,...,N_e)$ is the *local mass matrix* corresponding to the finite element e_k and has the entries

$$\mathbf{M}^{e_k}{}_{ij} = \int_{D_{e_k}} \varphi_i^{e_k} \varphi_j^{e_k} dD, \quad i,j = 1,...,8; \qquad (3.6.15)$$

- the *global advection matrix* $\mathbf{N_g}(\tilde{\mathbf{u}}^{m-1})$, of dimension $2N_T \times 2N_T$, can be expressed as

$\mathbf{N_g}(\tilde{\mathbf{u}}^{m-1}) = \begin{bmatrix} \mathbf{N_g^u}(\tilde{\mathbf{u}}^{m-1}) & \mathbf{0} \\ \mathbf{0} & \mathbf{N_g^v}(\tilde{\mathbf{u}}^{m-1}) \end{bmatrix}$, with $\mathbf{N_g^u}(\tilde{\mathbf{u}}^{m-1}) = \mathbf{N_g^v}(\tilde{\mathbf{u}}^{m-1}) = \sum_{k=1}^{N_e} \mathbf{N}^{e_k}(\tilde{\mathbf{u}}^{m-1})$. The matrix $\mathbf{N}^{e_k}(\tilde{\mathbf{u}}^{m-1})$ $(k=1,...,N_e)$ is the *local advection matrix* corresponding to the finite element e_k and has the entries

$$\mathbf{N}^{e_k}(\tilde{\mathbf{u}}^{m-1})_{ij} = \int_{D_{e_k}} \left[(\sum_{l=1}^{8} \varphi_l^{e_k} \tilde{u}_{(l)}^{m-1}) \varphi_i^{e_k} \frac{\partial \varphi_j^{e_k}}{\partial x} + (\sum_{l=1}^{8} \varphi_l^{e_k} \tilde{v}_{(l)}^{m-1}) \varphi_i^{e_k} \frac{\partial \varphi_j^{e_k}}{\partial y} \right] dD$$

$$i,j = 1,...,8; \qquad (3.6.16)$$

- the *global diffusion matrix*, of dimension $2N_T \times 2N_T$, can be expressed as

$\mathbf{K_g} = \begin{bmatrix} \mathbf{K_g^u} & \mathbf{0} \\ \mathbf{0} & \mathbf{K_g^v} \end{bmatrix}$, with $\mathbf{K_g^u} = \mathbf{K_g^v} = \sum_{k=1}^{N_e} \mathbf{K}^{e_k}$. The matrix $\mathbf{K}^{e_k}$ $(k=1,...,N_e)$ is the *local diffusion matrix* corresponding to the finite element e_k and has the entries

$$\mathbf{K}^{e_k}{}_{ij} = \int_{D_{e_k}} A_H \nabla \varphi_i^{e_k} \nabla \varphi_j^{e_k} dD, \quad i,j = 1,...,8; \qquad (3.6.17)$$

- the *global Coriolis matrix*, of dimension $2N_T \times 2N_T$, can be expressed as $\mathbf{C_g} = \begin{bmatrix} \mathbf{0} & \mathbf{C_g^u} \\ \mathbf{C_g^v} & \mathbf{0} \end{bmatrix}$, with $\mathbf{C_g^v} = \sum_{k=1}^{N_e} \mathbf{C}^{e_k}$ and $\mathbf{C_g^u} = -\mathbf{C_g^v}$. The matrix $\mathbf{C}^{e_k}$ (*k=1,...,N*$_e$) is the *local Coriolis matrix* corresponding to the finite element e_k and has the entries

$$\mathbf{C}^{e_k}{}_{ij} = \int_{D_{e_k}} f \varphi_i^{e_k} \varphi_j^{e_k} \, dD, \qquad i,j = 1,...,8; \tag{3.6.18}$$

- the *global gradient matrix*, of dimension $2N_T \times N_P$, can be expressed as $\mathbf{G_g} = \begin{bmatrix} \mathbf{G_g^u} \\ \mathbf{G_g^v} \end{bmatrix}$, with $\mathbf{G_g^u} = \sum_{k=1}^{N_e} \mathbf{G}^{\mathbf{u},e_k}$ and $\mathbf{G_g^v} = \sum_{k=1}^{N_e} \mathbf{G}^{\mathbf{v},e_k}$; the *local gradient matrices* $\mathbf{G}^{\mathbf{u},e_k}$ and $\mathbf{G}^{\mathbf{v},e_k}$ have the entries

$$\mathbf{G}^{\mathbf{u},e_k}{}_{ij} = \frac{1}{\rho_0} \int_{D_{e_k}} \varphi_i^{e_k} \frac{\partial \psi_j^{e_k}}{\partial x} dD, \quad i = 1,...,8, \;\; j = 1,...,4, \tag{3.6.19}$$

$$\mathbf{G}^{\mathbf{v},e_k}{}_{ij} = \frac{1}{\rho_0} \int_{D_{e_k}} \varphi_i^{e_k} \frac{\partial \psi_j^{e_k}}{\partial y} dD, \quad i = 1,...,8, \;\; j = 1,...,4; \tag{3.6.20}$$

- the *divergence matrix*, of dimension $N_P \times 2N$, can be expressed as $\mathbf{D_g} = [\mathbf{D_g^u} \;\; \mathbf{D_g^v}]$, with $\mathbf{D_g^u} = \sum_{k=1}^{N_e} \mathbf{D}^{\mathbf{u},e_k}$ and $\mathbf{D_g^v} = \sum_{k=1}^{N_e} \mathbf{D}^{\mathbf{v},e_k}$; the *local divergence matrices* $\mathbf{D}^{\mathbf{u},e_k}$ and $\mathbf{D}^{\mathbf{v},e_k}$ have the entries

$$\mathbf{D}^{\mathbf{u},e_k}{}_{ij} = \int_{D_{e_k}} \psi_i^{e_k} \frac{\partial \varphi_j^{e_k}}{\partial x} dD, \quad i = 1,\ldots,4 \text{ and } j = 1,\ldots,8, \tag{3.6.21}$$

$$\mathbf{D}^{\mathbf{v},e_k}{}_{ij} = \int_{D_{e_k}} \psi_i^{e_k} \frac{\partial \varphi_j^{e_k}}{\partial y} dD, \quad i = 1,\ldots,4 \text{ and } j = 1,\ldots,8. \tag{3.6.22}$$

It should be noted that, there is a general numbering scheme that assigns to every node of the mesh a certain number from *1* to N_T, as well as a local numbering scheme that assigns to the same node a number between *1* and *8* that depends on the finite element it belongs to.

The previous equations show how the global matrices of semi-discrete equations can be obtained by summing the local matrices generated at the finite element level. This procedure is called *assembly of local/element matrices* and is the common approach used in the implementation of finite element algorithms. The implementation based on element-level calculations followed by assembly consists of the following general steps:

Step 1. Generate the local/element matrices for each finite element in the mesh,

Step 2. Assemble the local matrices to obtain the global matrices,

Step 3. Solve the resulting system of equations.

Any of the equations (3.6.13-14) corresponds to a degree of freedom that is a nodal value of u, v or P, via the test function used in that equation. In order to solve the system given by (3.6.13-14), the known values of the velocity and pressure are substituted into the vectors $\mathbf{u}_\mathbf{g}^\mathbf{m}$ and $\mathbf{P}_\mathbf{g}^\mathbf{m}$, then the equations corresponding to those degrees of freedom are eliminated from the system. The resulting system of equations is exactly the same as the system given by the equations (3.5.1-2).

3.7 Isoparametric Finite Elements and Numerical Integration

All finite elements of the same type, for example the Taylor-Hood serendipity quadrilaterals, can be derived from a particular element of the same type called *reference element*, for example a square of prescribed side length. If the reference element is given in a rectangular system of coordinates (ξ, η), then the elements of the same type, in the (x, y) plane, can be obtained via appropriate mappings (Figure 3.3).

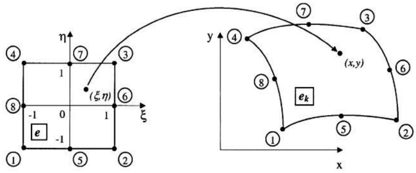


Figure 3.3 Isoparametric serendipity quadrilateral finite element mapped from the square reference element

One way to derive the finite element e_k from the reference element e is to use the following mapping

$$(x, y) = (\sum_{i=1}^{8} \varphi_i^e (\xi,\eta) x_i , \sum_{i=1}^{8} \varphi_i^e (\xi,\eta) y_i), \tag{3.7.1}$$

where the function φ_i^e $(i = 1,\ldots,8)$ is the shape function associated with the i^{th} node of the reference element *e*, and (x_i , y_i) are the coordinates of the corresponding node of the element e_k.

The velocity components *u* and *v* on the element e_k, can now be interpolated over the reference element *e* using the same shape functions

$$u^{e_k} (\xi,\eta,t) = \sum_{i=1}^{8} u_{(i)}(t)\varphi_i^{e_k} (x(\xi,\eta), y(\xi,\eta)) = \sum_{i=1}^{8} u_{(i)}(t)\varphi_i^e (\xi,\eta), \tag{3.7.2}$$

$$v^{e_k} (\xi,\eta,t) = \sum_{i=1}^{8} v_{(i)}(t)\varphi_i^{e_k} (x(\xi,\eta), y(\xi,\eta)) = \sum_{i=1}^{8} v_{(i)}(t)\varphi_i^e (\xi,\eta), \tag{3.7.3}$$

while the pressure can be interpolated using the linear shape functions corresponding to the corner nodes of the reference element

$$P^{e_k} (\xi,\eta,t) = \sum_{i=1}^{4} P_{(i)}(t)\psi_i^{e_k} (x(\xi,\eta), y(\xi,\eta)) = \sum_{i=1}^{4} P_{(i)}(t)\psi_i^e (\xi,\eta). \tag{3.7.4}$$

The shape functions defined on the square reference element in Figure 3.3 have the following expressions (Taylor et al., 1981)

$$\varphi_i^e (\xi,\eta) = \tfrac{1}{4} (1 + \xi_i\xi)(1 + \eta_i\eta)(\xi_i\xi + \eta_i\eta - 1), \quad i = 1,\ldots,4 , \tag{3.7.5}$$

$$\varphi_i^e(\xi,\eta)=\tfrac{1}{2}(1-\xi^2)(1+\eta_i\eta),\quad i=5,7, \tag{3.7.6}$$

$$\varphi_i^e(\xi,\eta)=\tfrac{1}{2}(1+\xi_i\xi)(1-\eta^2),\quad i=6,8, \tag{3.7.7}$$

$$\psi_i^e(\xi,\eta)=\tfrac{1}{4}(1+\xi_i\xi)(1+\eta_i\eta),\quad i=1,\dots,4. \tag{3.7.8}$$

The finite elements mapped from the reference element by using the same shape functions that are used to interpolate the unknown function (in this case the velocity) over that element, are called *isoparametric finite elements.*

The main advantage of using the reference element is that any finite element in the mesh can be mapped back into it. Thus, all functions defined on an arbitrary element e_k can be defined on the reference element e. Moreover, all integrals on e_k can be calculated on the reference element using the transformation of coordinates (3.7.1). For example, if $f(x,y)$ is a function defined on e_k, then by using the change of variables formula we get

$$\int_{D_{e_k}} f(x,y)\,dD=\int_{-1}^{1}\int_{-1}^{1} f(x(\xi,\eta),y(\xi,\eta))\,|J|\,d\xi\,d\eta, \tag{3.7.9}$$

where $|J|$ is the Jacobian determinant of the transformation of coordinates.

The integrals over the domain of the reference element can be calculated numerically by means of quadrature formulae. The most widely used quadrature formula in the field of the Finite Element Method is the Gauss quadrature formula. In two dimensions, the Gauss quadrature formula of order m – also called the $m\times m$ *Gauss scheme/rule* – is,

$$\int_{-1}^{1}\int_{-1}^{1} g(\xi,\eta)d\xi d\eta \cong \sum_{i=1}^{m}\sum_{j=1}^{m} w_i w_j g(\xi_i,\eta_j). \tag{3.7.10}$$

This rule has the degree of precision *2m-1*, which means that the evaluation of the integral in (3.7.10) is exact if *g* is a polynomial of degree up to *2m-1*. For the integrals that appear in the semi-discrete (3.6.10-12), the 3×3 Gauss rule is considered to be sufficient (Heinrich et al., 1999). The integration points (ξ_i,η_j) and the weights w_i of this scheme are given in table 3.1.

Table 3.1 The coordinates of the integration points and the weights of the 3×3 Gauss rule

i	ξ_i	η_i	w_i
1	0.7745966692414833	0.7745966692414833	0.5555555555555556
2	0.0	0.0	0.8888888888888889
3	-0.7745966692414833	-0.7745966692414833	0.5555555555555556

Another advantage of the reference element is that it can be mapped into finite elements with curved sides if the shape functions are polynomials of second-order or higher. Thus, the mapping defined by (3.7.1) enables the generation of finite elements with parabolic-shaped sides, which are very useful when discretizing domains with complex geometry.

4. TIME INTEGRATION

4.1 Introduction

In order to complete the discretization of the system of equations (3.5.1-2), a time integration rule is required. There are two types of such rules, explicit and implicit. The explicit integration rules can be easily implemented and are computationally inexpensive. However, they impose severe restrictions on the time step size (i.e. constant and small) in order to ensure stability, and are not appropriate given the implicit character of the pressure (Reddy et al., 1994). The implicit integration rules do not display these inconveniencies, but are computationally expensive because a linear/nonlinear algebraic system must be solved at each time step. The main advantage of the implicit rules is their unconditional stability that allows for the use of larger and even variable time steps. These rules are often used for the time integration of the Navier-Stokes equations.

The numerical integration rule used and described hereafter is the Crank-Nicolson/trapezoidal rule. This is an implicit rule that is second-order accurate, unconditionally stable, and does not introduce extraneous solutions or spurious damping (Gresho et al., 1999).

4.2 Crank-Nicolson Rule

The Crank-Nicolson rule applied to the system of ordinary differential equations,

$$\frac{d\mathbf{u}}{dt} = \mathbf{F}(\mathbf{u},t) \qquad (4.2.1)$$

with the initial condition $\mathbf{u}(0)=\mathbf{u}_0$ yields

$$\frac{\mathbf{u}_{n+1}-\mathbf{u}_n}{\Delta t}=\frac{\mathbf{F}(\mathbf{u}_{n+1},t_{n+1})+\mathbf{F}(\mathbf{u}_n,t_n)}{2}, \tag{4.2.2}$$

where t_n stands for the n^{th} time level, $\mathbf{u}_n$ is the numerical solution of (4.2.1) at time t_n, and Δt is the time step size.

The linearized semi-discrete Navier-Stokes equations in the matrix-vector form (3.5.1-2) are

$$\mathbf{M}\frac{d\mathbf{u}}{dt}+[\mathbf{N}(\tilde{\mathbf{u}})+\mathbf{K}+\mathbf{C}]\mathbf{u}+\mathbf{GP}=\mathbf{F}(\tilde{\mathbf{u}}), \tag{4.2.3}$$

$$\mathbf{Du}=\mathbf{H}, \tag{4.2.4}$$

in which the superscript indicating the iteration number has been omitted.

When applied to (4.2.3-4) the Crank-Nicolson rule reads

$$\mathbf{M}\frac{\mathbf{u}_{n+1}-\mathbf{u}_n}{\Delta t}+\frac{1}{2}\{[\mathbf{N}(\tilde{\mathbf{u}}_n)+\mathbf{K}+\mathbf{C}]\mathbf{u}_n+[\mathbf{N}(\tilde{\mathbf{u}}_{n+1})+\mathbf{K}+\mathbf{C}]\mathbf{u}_{n+1}\}$$

$$+\frac{1}{2}(\mathbf{GP}_n+\mathbf{GP}_{n+1})=\frac{1}{2}[\mathbf{F}(\tilde{\mathbf{u}}_n)+\mathbf{F}(\tilde{\mathbf{u}}_{n+1})], \tag{4.2.5}$$

$$\frac{1}{2}(\mathbf{Du}_n+\mathbf{Du}_{n+1})=\frac{1}{2}(\mathbf{H}_n+\mathbf{H}_{n+1}). \tag{4.2.6}$$

These equations can be reformulated as,

$$\frac{2}{\Delta t}\mathbf{M}\mathbf{u}_{n+1} + [\mathbf{N}(\tilde{\mathbf{u}}_{n+1}) + \mathbf{K} + \mathbf{C}]\mathbf{u}_{n+1} + \mathbf{G}\mathbf{P}_{n+1}$$

$$= \mathbf{F}(\tilde{\mathbf{u}}_{n+1}) + \frac{2}{\Delta t}\mathbf{M}\mathbf{u}_n + \mathbf{F}(\tilde{\mathbf{u}}_n) - [\mathbf{N}(\tilde{\mathbf{u}}_n) + \mathbf{K} + \mathbf{C}]\mathbf{u}_n - \mathbf{G}\mathbf{P}_n, \tag{4.2.7}$$

$$\mathbf{D}\mathbf{u}_n + \mathbf{D}\mathbf{u}_{n+1} = \mathbf{H}_n + \mathbf{H}_{n+1}. \tag{4.2.8}$$

At the time level t_n, (4.2.3) becomes

$$\mathbf{M}\frac{d\mathbf{u}}{dt}(t_n) = \mathbf{F}(\tilde{\mathbf{u}}(t_n)) - [\mathbf{N}(\tilde{\mathbf{u}}(t_n)) + \mathbf{K} + \mathbf{C}]\mathbf{u}(t_n) - \mathbf{G}\mathbf{P}(t_n). \tag{4.2.9}$$

Let $\left(\frac{d\mathbf{u}}{dt}\right)_n$ be such that

$$\mathbf{M}\left(\frac{d\mathbf{u}}{dt}\right)_n = \mathbf{F}(\tilde{\mathbf{u}}_n) - [\mathbf{N}(\tilde{\mathbf{u}}_n) + \mathbf{K} + \mathbf{C}]\mathbf{u}_n - \mathbf{G}\mathbf{P}_n, \tag{4.2.10}$$

then (4.2.7) becomes

$$[\frac{2}{\Delta t}\mathbf{M} + \mathbf{N}(\tilde{\mathbf{u}}_{n+1}) + \mathbf{K} + \mathbf{C}]\mathbf{u}_{n+1} + \mathbf{G}\mathbf{P}_{n+1} = \mathbf{F}(\tilde{\mathbf{u}}_{n+1}) + \mathbf{M}[\frac{2}{\Delta t}\mathbf{u}_n + \left(\frac{d\mathbf{u}}{dt}\right)_n]. \tag{4.2.11}$$

If $\mathbf{D}\mathbf{u}_0 = \mathbf{H}_0$ (i.e. the initial velocity field is divergence free) then (4.2.8) reduces to

$$\mathbf{D}\mathbf{u}_{n+1} = \mathbf{H}_{n+1}. \tag{4.2.12}$$

The matrix-vector form of the linear system of discrete equations (4.2.11-12) is,

$$\begin{bmatrix} \frac{2}{\Delta t}\mathbf{M}+\mathbf{N}(\tilde{\mathbf{u}}_{n+1})+\mathbf{K}+\mathbf{C} & \mathbf{G} \\ \mathbf{D} & \mathbf{0} \end{bmatrix}\begin{pmatrix} \mathbf{u}_{n+1} \\ \mathbf{P}_{n+1} \end{pmatrix}=\begin{pmatrix} \mathbf{F}(\tilde{\mathbf{u}}_{n+1})+\mathbf{M}[\frac{2}{\Delta t}\mathbf{u}_n+\left(\frac{d\mathbf{u}}{dt}\right)_n] \\ \mathbf{H}_{n+1} \end{pmatrix}. \quad (4.2.13)$$

and is called the 'shortened' version of the Crank-Nicolson rule (Gresho et al., 1999).

Equation (4.2.9) applied for $t_0 = 0$ yields

$$\mathbf{M}\left(\frac{d\mathbf{u}}{dt}\right)_0+\mathbf{G}\mathbf{P}_0=\mathbf{F}(\mathbf{u}_0)-[\mathbf{N}(\mathbf{u}_0)+\mathbf{K}+\mathbf{C}]\mathbf{u}_0, \quad (4.2.14)$$

taking into account that $\tilde{\mathbf{u}}_0 = \mathbf{u}_0$.

After taking the time derivative with respect to time at $t_0 = 0$ of both sides, the equation (4.2.4) becomes

$$\mathbf{D}\left(\frac{d\mathbf{u}}{dt}\right)_0=\left(\frac{d\mathbf{H}}{dt}\right)_0. \quad (4.2.15)$$

The matrix-vector form of the previous two equations is

$$\begin{bmatrix} \mathbf{M} & \mathbf{G} \\ \mathbf{D} & \mathbf{0} \end{bmatrix}\begin{pmatrix} \left(\frac{d\mathbf{u}}{dt}\right)_0 \\ \mathbf{P}_0 \end{pmatrix}=\begin{pmatrix} \mathbf{F}(\mathbf{u}_0)-[\mathbf{N}(\mathbf{u}_0)+\mathbf{K}+\mathbf{C}]\mathbf{u}_0 \\ \left(\frac{d\mathbf{H}}{dt}\right)_0 \end{pmatrix}. \quad (4.2.16)$$

Since the initial acceleration vector $\left(\frac{d\mathbf{u}}{dt}\right)_0$ and the initial pressure field $\mathbf{P}_0$ can be obtained from (4.2.16), the rule given by (4.2.13) is self-starting.

The rule presented above uses the original mass matrix, called the *consistent* mass matrix. In order to reduce the computational cost of the algorithm the so-called *lumped* mass matrix could be used instead. This matrix is a diagonal matrix whose diagonal element m_{ii}^L is the sum of the elements on the row i in the consistent mass matrix. The ODEs in (3.5.1) become decoupled when the lumped mass matrix is used, and this has a negative impact on the accuracy of the numerical solution. Due to this disadvantage, the lumped mass matrix is not used hereafter.

4.3 Pressure Oscillation

Many numerical solutions to the Navier-Stokes equations exhibit non-physical oscillations/wiggles. Some numerical algorithms try to eliminate the wiggles by using an increased diffusion/damping or some other techniques. It is considered that these wiggle-free methods hide their deficiencies (Gropp et al., 1992), and actually not the original problem is solved but a modified one. On the contrary, wiggles may indicate, for example, that a flow region is poorly resolved and mesh refinement is needed in order to capture the physical phenomenon accurately, or that the initial conditions are ill-posed. They may also flag some limitations of the numerical algorithm. That is the reason some experts in the field recommend that the wiggles should not be a priori suppressed (Gresho et al., 1979).

When the nonlinear, diffusion and Coriolis terms are dropped in (4.2.5), the latter becomes equivalent to the equation

$$\mathbf{GP}_{n+1} - \mathbf{F}(\tilde{\mathbf{u}}_{n+1}) = -[\mathbf{GP}_n - \mathbf{F}(\tilde{\mathbf{u}}_n) + \frac{2}{\Delta t}\mathbf{M}(\mathbf{u}_{n+1} - \mathbf{u}_n)]. \quad (4.3.1)$$

Multiplying the above equation by $\mathbf{DM}^{-1}$, where $\mathbf{D}$ is the divergence matrix and $\mathbf{M}^{-1}$ is the inverse of the mass matrix, it is obtained

$$(\mathbf{DM}^{-1}\mathbf{G})\mathbf{P}_{n+1} - (\mathbf{DM}^{-1})\mathbf{F}(\tilde{\mathbf{u}}_{n+1})$$
$$= -[(\mathbf{DM}^{-1}\mathbf{G})\mathbf{P}_n - (\mathbf{DM}^{-1})\mathbf{F}(\tilde{\mathbf{u}}_n) + \frac{2}{\Delta t}(\mathbf{Du}_{n+1} - \mathbf{Du}_n)]. \quad (4.3.2)$$

By writing the above equation for all previous time levels, followed by successive substitution of the first two terms on the right hand side of the equation, we get

$$(\mathbf{DM}^{-1}\mathbf{G})\mathbf{P}_{n+1} - (\mathbf{DM}^{-1})\mathbf{F}(\tilde{\mathbf{u}}_{n+1}) = (-1)^{n+1}[(\mathbf{DM}^{-1}\mathbf{G})\mathbf{P}_0$$
$$-(\mathbf{DM}^{-1})\mathbf{F}(\tilde{\mathbf{u}}_0) + (-1)^n \frac{2}{\Delta t}(\mathbf{Du}_{n+1} - \mathbf{Du}_n) + \ldots + \frac{2}{\Delta t}(\mathbf{Du}_1 - \mathbf{Du}_0)]. \quad (4.3.3)$$

Taking into account (4.2.12) we get

$$\mathbf{Du}_{n+1} - \mathbf{Du}_n = \mathbf{H}_{n+1} - \mathbf{H}_n \quad (4.3.4)$$

for all values of n. We also have $\mathbf{Du}_0 = \mathbf{H}_0 + (\mathbf{Du}_0 - \mathbf{H}_0)$. Hence (4.3.3) is equivalent to

$$(\mathbf{DM}^{-1}\mathbf{G})\mathbf{P}_{n+1} = (\mathbf{DM}^{-1})\mathbf{F}(\tilde{\mathbf{u}}_{n+1}) + (-1)^{n+1}[-\left(\frac{d\mathbf{H}}{dt}\right)_0$$

$$+(-1)^n \frac{2}{\Delta t}(\mathbf{H}_{n+1} - \mathbf{H}_n) + \ldots + \frac{2}{\Delta t}(\mathbf{H}_1 - \mathbf{H}_0)] + (-1)^n \frac{2}{\Delta t}(\mathbf{Du}_0 - \mathbf{H}_0). \quad (4.3.5)$$

It should be noticed that, in case the initial velocity field is not divergence free, i.e. $\mathbf{Du}_0 - \mathbf{H}_0 \neq 0$, then the pressure given by the "shortened" Crank-Nicolson rule exhibits non-physical oscillations due to the presence of the last term in (4.3.5). If the "long" Crank-Nicolson rule that involves (4.2.8) rather than (4.2.12) is used, then the oscillations of the pressure are amplified in time (Gresho et al., 1999). In both cases the wiggles are due to ill-posed initial conditions. Therefore, in order to remove this noise, the pressure that is written out should be the average of the pressure values calculated at two consecutive time levels

$$\mathbf{P}_{n+1/2} = \frac{\mathbf{P}_n + \mathbf{P}_{n+1}}{2}. \quad (4.3.6)$$

5. IMPLEMENTATION

The numerical techniques presented in the previous chapters have been implemented into a finite element program written in FORTRAN 90. The input file for this program is generated by the GID preprocessor (GID, 2001), whereas the output file is processed by means of a set of MATLAB scripts.

5.1 Preprocessor

The preprocessor provided by the GID software has been used to generate the finite element mesh and input file for the finite element program. It employs the advancing-front technique to generate unstructured meshes based on the domain geometry and mesh parameters (for example, element size, size transition) provided by the user. Moreover, the GID preprocessor allows for the specification of general problem data, fluid properties, initial and boundary conditions, and forcing. All these must be consistent with the type of problem to be solved as well as with the finite element algorithm that is to be used, and can be set by means of a group of configuration files. The concept of problem type employed by GID refers to such a group of configuration files.

In order to further customize the preprocessor, another configuration file containing the format of the input file to be used by the finite element program can be set up. As well, a batch file can be written to define domain geometry, set values for parameters and conditions, and control the preprocessing phase.

5.2 Finite Element Program

The finite element program has been written in FORTRAN 90 and consists of a main program and a set of external subroutines. The main program (Figure 5.1.a) starts by calling three subroutines for reading problem parameters (`InputParam`), mesh data (`InputMeshData`) as well as initial and boundary conditions (`InputIBV`).

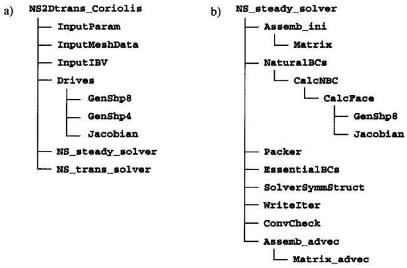


Figure 5.1 a) Main program structure; b) Structure of the solver for steady-state solutions.

Subsequently, it calls the `Drives` subroutine to generate the shape functions, their derivatives, and Jacobian determinant values at Gauss integration points for each finite element. Next, it calls either the subroutine `NS_steady_solver` (Figure 5.1.b) or the subroutine `NS_trans_solver` (Figure 5.2). The former contains the Navier-Stokes solver for steady-state solutions. The latter contains the Navier-Stokes solver for transient solutions and is much more important as most flows are transient. This solver is briefly described in the following.

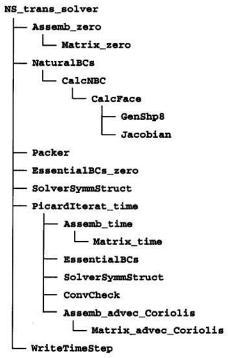


Figure 5.2 Structure of the solver for transient solutions

The solver for transient solutions starts by calling `Assemb_zero`, which assembles the local matrices and right-hand side (RHS) vectors calculated for each finite element by `Matrix_zero`. The goal is to obtain the global system matrix and RHS vector of the linear system in equation (4.2.16). The solution of this system provides the initial acceleration and pressure vectors needed for start-up. All elements of the global system matrix and RHS vector are stored in external files, as their number is very large - of order N^2 for N unknown nodal values of velocity and pressure. Next, the contributions of the natural boundary conditions are calculated and added to the global RHS vector by the subroutine `NaturalBCs`. To enable the use of a solver for sparse linear systems of algebraic equations, the subroutine `Packer` is invoked. It reads the above-mentioned files and compresses both the system matrix and the RHS vector by only storing the non-zero values in linear arrays; additional linear arrays are created to hold the positions of the non-zero entries in the global matrix and vector. Before solving the linear system, the subroutine `EssentialBCs_zero` is called to replace the entries in the RHS vector corresponding the known nodal values of the initial acceleration and pressure with these values. Furthermore, all non-zero entries on a row of the system matrix associated with a known nodal value are replaced with zero, except for the entry on the main diagonal that is replaced by one.

The direct sparse solver from the Compaq Extended Math Library (Compaq, 2001) is used as solver for linear systems. The `SolverSymmStruct` subroutine sets the options and calls the direct sparse solver subroutines. An interesting fact is that the system matrices provided by the finite element algorithm, although non-symmetric, have a symmetric structure (i.e. the locations of the non-zero entries are symmetric about the main diagonal), which allows for the use of an option provided for such cases to optimize the solution process. The direct sparse solver is based on the factorization of the system matrix into triangular matrices (*LU* factorization). In order to increase the sparsity of the resulting triangular matrices and thus reduce the storage and computation time, the rows and columns of the system matrix need to be permuted before factorization. The permutation vector is computed automatically by the `dss_reorder` subroutine of the sparse solver when the `CXML_DSS_AUTO_ORDER` option is used. When Coriolis matrices are added to the global system matrix, as a result of considering Coriolis forces, the factorization fails due to causes unexplained by now. Noticing that the factorization of the permuted matrix fails, not that of the original matrix, the sparse solver has been instructed not to permute the latter. Thus, replacing the option `CXML_DSS_AUTO_ORDER` with `CXML_DSS_MY_ORDER`, followed by providing the identical permutation vector so that the rows and columns of the original matrix are not interchanged, has solved the problem. However, this affects considerably the efficiency of the sparse solver and has been accepted as a temporary solution only.

After the initial acceleration and pressure vectors have been calculated, the subroutine `PicardIterat_time` is called at every time step. This subroutine starts by calling `Assemb_time` to assemble the local matrices and RHS vectors provided for each finite element by `Matrix_time`. The resulting global system matrix and RHS vector are those given in equation (4.2.13). The global system matrix calculated during start-up can be used in its compressed form hereafter. Thus, `Assemb_time` first multiplies the existent entries corresponding to the mass matrix by $\frac{2}{\Delta t}$, then adds the contributions of the advection, diffusion and Coriolis matrix (if the value of the `Coriolis` boolean variable is `.TRUE.`). At the beginning of every time step, the velocity vector used in the calculation of the advection matrix is that obtained at the end of the previous time step. After obtaining the first solution in a time step, by using `SolverSymmStruct` preceded by `EssentialBCs`, the Picard iteration proceeds by using it in the next iteration. The advection matrix is recalculated with the newly calculated velocity vector by calling `Assemb_advec_Coriolis`, which invokes `Matrix_advec_Coriolis`. The iterative process continues until the relative error defined in equation (3.4.6) is within a prescribed tolerance. The convergence is checked by `ConvCheck` subroutine. At the end of the time step the velocity and pressure vectors are written out by `WriteTimeStep` subroutine.

5.3 Postprocessor

The postprocessor consists of a set of MATLAB scripts. One of them is used to read the mesh data from the input file used by the finite element program. Another script is employed to read the nodal results for velocity components and pressure at a specified iteration (for steady-state case) or time step (for transient case) from the output file generated by the finite element program. Two scripts that calculate the derived quantities, vorticity and streamfunction, further exploit these results. The last script plots the finite element mesh, velocity vectors at velocity nodes, as well as velocity, pressure, vorticity and streamfunction fields.

The derived quantities, vorticity and streamfunction, are calculated by determining their least-squares best fit in spaces spanned by pressure basis functions and velocity basis functions, respectively.

5.3.1 Vorticity calculation

The finite element program yields the vectors of nodal values of velocity components and pressure. Thus, the approximate velocity components u^h and v^h become completely determined by virtue of equations (3.3.1-2). A vorticity field corresponding to this approximate velocity field can be obtained by using the equation

$$\omega = \frac{\partial v^h}{\partial x} - \frac{\partial u^h}{\partial y} . \qquad (5.3.1)$$

In order to obtain the vorticity field, rather than using (5.3.1), a least-square best fit, ω^h, to this field is sought in the linear space spanned by the pressure basis functions, $S^h_{N_P} \equiv span\{ \psi_1, ..., \psi_{N_P} \}$. Hence the approximate vorticity field is a linear combination of these functions

$$\omega^h = \sum_{j=1}^{N_P} \omega_j \psi_j , \qquad (5.3.2)$$

where ω_j values are vorticity values at pressure nodes. Notice that the use of one-order-lower basis functions than those used for velocity is suggested by the equation (5.3.1) (Gresho, 1999).

The function ω^h is the least-square best fit to the vorticity function, in $S^h_{N_P}$, if and only if

$$\int_D \psi_i \omega^h dD = \int_D \psi_i \left(\frac{\partial v^h}{\partial x} - \frac{\partial u^h}{\partial y} \right) dD, \quad i = 1,\dots,N_P. \tag{5.3.3}$$

Therefore, the nodal vorticity values form the solution to the linear system

$$\sum_{j=1}^{N_P} \omega_j \int_D \psi_i \psi_j dD = \sum_{k=1}^{N} \int_D \psi_i \left(v_k \frac{\partial \varphi_k}{\partial x} - u_k \frac{\partial \varphi_k}{\partial y} \right) dD, \quad i = 1,\dots,N_P. \tag{5.3.4}$$

Notice that ω^h in equation (5.3.3) is an approximate solution to a weak form of the equation (5.3.1), which makes this approach consistent with the FEM theory.

5.3.2 Streamfunction calculation

The streamfunction, Ψ, is a function such that $\frac{\partial \Psi}{\partial y} = u$ and $-\frac{\partial \Psi}{\partial x} = v$. It satisfies the equation $\nabla^2 \Psi = -\omega$, which is equivalent to

$$\nabla^2 \Psi = -\left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \right). \tag{5.3.5}$$

The approach used to determine the streamfunction field is similar to that employed for vorticity. A least-square best fit, Ψ^h, to this field is sought in the linear space spanned by the velocity basis functions, $V^h_{N_T} \equiv span\{\varphi_1,...,\varphi_{N_T}\}$ (Gresho, 1999). Hence the approximate streamfunction field is a linear combination of these functions

$$\Psi^h = \sum_{j=1}^{N_T} \Psi_j \varphi_j , \tag{5.3.6}$$

where Ψ_j values are streamfunction values at velocity nodes.

The function Ψ^h is the least-square best fit to the streamfunction function, in $V^h_{N_T}$, if and only if

$$\int_D \varphi_i \nabla^2 \Psi^h dD = -\int_D \varphi_i \left(\frac{\partial v^h}{\partial x} - \frac{\partial u^h}{\partial y} \right) dD, \quad i = 1,...,N_T . \tag{5.3.7}$$

Invoking the Green's theorem, we get

$$\int_D \nabla \varphi_i \nabla \Psi^h dD = \int_D \varphi_i \left(\frac{\partial v^h}{\partial x} - \frac{\partial u^h}{\partial y} \right) dD + \int_\Gamma \varphi_i \frac{\partial \Psi^h}{\partial n} d\Gamma, \quad i = 1,...,N_T \tag{5.3.8}$$

where $\vec{n}$ is the outward pointing unit normal vector on the boundary $\Gamma = \partial D$.

Notice that the equation (5.3.8) is actually a weak form of the Poisson-type equation obtained from (5.3.5) by replacing u and v with u^h and v^h, respectively.

6. RESULTS

The program based on the algorithm presented in the previous chapters has been tested on three problems. The first two, flow past a cylinder and flow over a backward-facing step are classical benchmark problems for flow simulation programs based on a variety of numerical methods. The third problem, mid-latitude wind-driven barotropic ocean circulation in a closed basin, is classical in the field of ocean modelling.

6.1 Flow Past a Cylinder

6.1.1 Introduction

Experimental studies have shown that the flow past a circular cylinder immersed in a channel exhibits a sequence of different structures as the Reynolds number increases. When $Re << 1$, the upstream pattern of the flow is symmetric to the downstream pattern. As Re increases, two steady symmetric eddies, called *attached eddies*, develop behind the cylinder. These eddies grow with Re, while the wake of the cylinder remains steady. When $Re \approx 40$, this structure undergoes a spontaneous change to an unsteady and non-symmetric wake characterized by a periodic shedding of eddies from the upper and lower parts of the cylinder. The resulting flow structure is called the *Karman vortex street* (Tritton, 1988).

The program has been run to simulate flows past a cylinder for two different values of the Reynolds number, $Re = 40$ and 100. The results, in terms of flow pattern, reattachment length of eddies (for $Re = 40$), and periodicity of vortex shedding (for $Re = 100$) are compared with results of numerical simulations provided by FLUENT software (based on the Finite Volume Method) and published by FLUENT Inc. (1999), as well as with experimental results obtained by Braza, M. et al. (1986) and included in the same publication.

6.1.2 Spatial Discretization

A circular cylinder of diameter $d = 2.0m$ is immersed in a uniform flow of speed $u_\infty = 1.0ms^{-1}$. The geometry and unstructured finite element mesh of the computational domain around the cylinder are depicted in Figure 6.1. The mesh contains *578* isoparametric Taylor-Hood serendipity quadrilateral finite elements and *1794* nodes. Finer mesh is required in the vicinity of the cylinder and behind it, along the centreline (where the wake occurs). Since the mesh is rather coarse overall, it should be as close to a symmetric mesh as possible in order that the influence on the flow be insignificant.

6.1.3 Boundary and Initial Conditions

The essential boundary condition $u = 1.0ms^{-1}$, $v = 0.0ms^{-1}$ is applied at the inlet (the left-hand side vertical boundary in Figure 6.1) as well as on the lateral boundaries. No-slip boundary condition $u = 0.0ms^{-1}$, $v = 0.0ms^{-1}$ is applied to the cylinder wall. The outflow open boundary condition $\frac{\partial u}{\partial x} = 0$, $\frac{\partial v}{\partial x} = 0$ is applied at the outlet. A reference value for the pressure of $100Pa$ (this value is arbitrary and has no physical meaning) is assigned to the nodal pressure at the node located at the upper right-hand side corner of the domain.

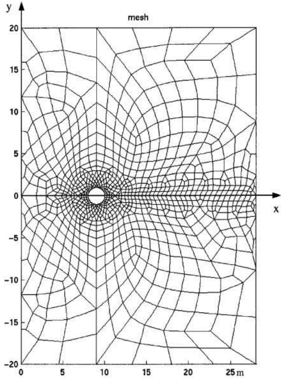


Figure 6.1 Flow past a cylinder – Geometry and unstructured finite element mesh of the computational domain

The initial conditions specify a velocity field given by $u_0 = 0.0ms^{-1}$, $v_0 = 0.0ms^{-1}$ at all nodes except for those on the inflow and lateral boundaries where it should match the boundary condition $u_0 = 1.0ms^{-1}$, $v_0 = 0.0ms^{-1}$. A mismatch would lead to an ill-posed problem. Even if this requirement is satisfied, it is very likely to obtain an interpolated initial velocity field that is not divergence-free, as the type of finite element used does not ensure the generation of a divergence-free velocity field. Thus, unless a divergence-free initial velocity field that matches the boundary conditions is specified, the initial condition is ill-posed. In practice, this situation occurs frequently and different methods to deal with it have been devised (see pressure oscillation, Sec. 4.3)

6.1.4 Results for *Re=40*

In order to get the Reynolds number $Re = 40$ for the given diameter of the cylinder, the fluid properties have been set as follows:

- fluid density: $\rho = 1.0 kgm^{-3}$,
- dynamic viscosity: $\mu = 0.05 Nsm^{-2}$.

The program has been run for a steady-state solution. Results of the simulation are illustrated in Figures 6.2-7.

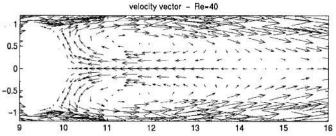


Figure 6.2 Flow past a cylinder - Velocity vectors behind the cylinder (*Re=40*, partial computational domain)

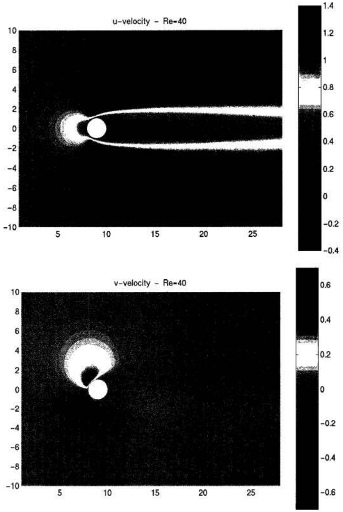


Figure 6.3 Flow past a cylinder - Velocity components *u* and *v* plots (in ms^{-1})

(*Re=40*, partial computational domain)

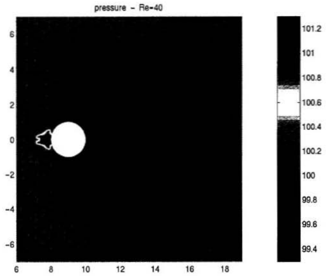


Figure 6.4 Flow past a cylinder – Pressure plot (in *Pa*)

(*Re=40*, partial computational domain)

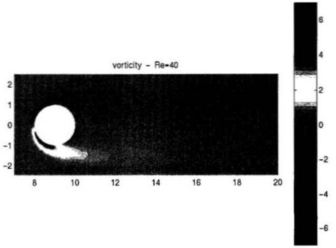


Figure 6.5 Flow past a cylinder – Vorticity plot (in s^{-1})

(*Re=40*, partial computational domain)

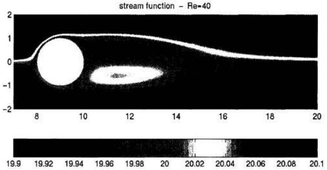


Figure 6.6 Flow past a cylinder – Streamfunction plot (in m^2s^{-1})
(*Re=40*, partial computational domain, partial streamfunction range)

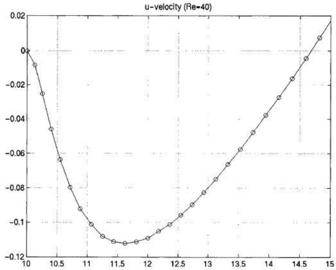


Figure 6.7 Flow past a cylinder – *u*-velocity along the *x*-axis (in ms^{-1})
vs. distance from the inlet (*Re=40*)

Figures 6.2-6 show a symmetric flow about the centreline (x-axis) with a pattern very similar to that simulated by FLUENT. The attached eddies can be clearly seen in Figures 6.2 and 6.6.

Figure 6.7 depicts the variation of the *u*-velocity along the centreline behind the cylinder. The *u*-velocity, which starts from zero on the cylinder wall, first takes negative values where the flow is reversed, then takes positive values after the reattachment has occurred. The reattachment length is the distance along the centreline from the backside of the cylinder to the point where *u*-velocity changes its sign. Its value extracted from Figure 6.7 is $L \cong 4.7m$. The dimensionless reattachment length (L_A) is obtained after dividing this by the cylinder radius ($1m$), $L_A = 4.7$. The FLUENT result is $L_A = 4.27$, which is very close to the values obtained experimentally. The result is considered fairly good given the coarse mesh of the domain compared to the mesh used by FLUENT that had 2601 quadrilateral elements.

6.1.5 Results for *Re=100*

By reducing the dynamic viscosity to $\mu = 0.02Nsm^{-2}$, the Reynolds number is increased to $Re = 100$, assuming the cylinder diameter and fluid density remain unchanged. In this case, the flow is unsteady and can be simulated only if the transient Navier-Stokes equations are considered. This brings the time integration scheme into play.

The time integration has been performed with a time step of $0.2s$. A period of approximately $12s$ was determined for vortex shedding. Velocity vector fields and streamfunction plots corresponding to five instants during one cycle are presented in Figures 6.8 and 6.9, respectively.

The non-dimensional parameter used to characterize the periodicity of vortex shedding is the Strouhal number, *S*, given by:

$$S = \frac{d}{\tau U} \tag{6.1.1}$$

where d is the cylinder diameter, τ is the period of vortex shedding, and U is the characteristic value of velocity. For $\tau = 12s$, $d = 2m$, and $U = 1ms^{-1}$, the Strouhal number is $S = \frac{1}{6} \cong 0.166$. The Strouhal numbers obtained from physical experiments are about 0.16, for $Re = 100$ (Braza et al., 1986). Hence a very good agreement exists between the Strouhal number calculated by the program and those from experiments. As for the FLUENT result, it is $S = 0.165$. It should be mentioned here that part of the difference between the calculated and experimental values originates from simulating three-dimensional flows using two-dimensional models.

Plots of velocity components, pressure and vorticity at time $t = 50s$ are presented in Figures 6.10-12.

6.1.6 Conclusions

The simulation of the flow past a circular cylinder by means of the finite element program has provided good results (given the coarse mesh used) for $Re = 40$, in terms of

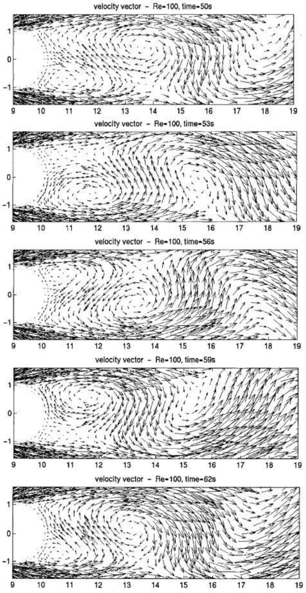


Figure 6.8 Flow past a cylinder - Velocity vectors behind the cylinder at different instants (*Re=100*, partial computational domain)

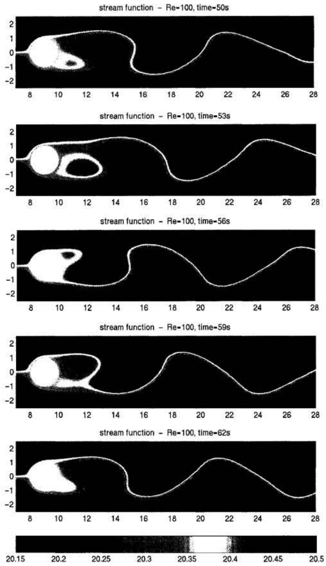


Figure 6.9 Flow past a cylinder – Streamfunction plots (in $m^2 s^{-1}$) at different instants (*Re=40*, partial computational domain, partial streamfunction range)

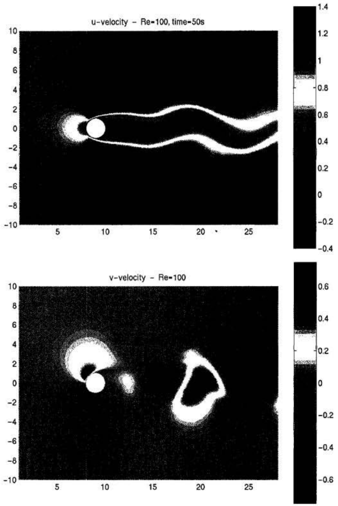


Figure 6.10 Flow past a cylinder - Velocity components *u* and *v* plots (in ms^{-1}) at $t = 50s$ ($Re=100$, partial computational domain)

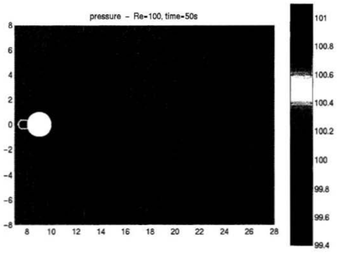


Figure 6.11 Flow past a cylinder – Pressure plot (in *Pa*) at $t = 50s$ (*Re=100*, partial computational domain)

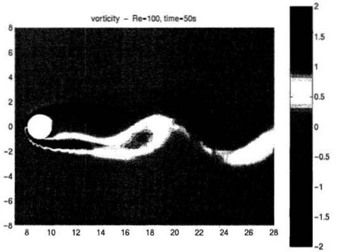


Figure 6.12 Flow past a cylinder – Vorticity plot (in s^{-1}) at $t = 50s$ (*Re=100*, partial computational domain)

flow pattern and reattachment length. Very good results have been obtained for $Re = 100$, the calculated Strouhal number being very close to that obtained from experiments.

The pressure field obtained for the two cases is not very accurate, especially in front of the cylinder. It can be seen (Figures 6.4 and 6.11) that the variation of the pressure depends on the finite element mesh. As mentioned before, pressure is the most sensitive variable in finite element simulations of incompressible flows. The same finite element program may provide an accurate pressure field for one problem and a poor representation of the pressure when the problem is slightly changed. Here it is worth citing Gresho's remark on the pressure (Gresho et al., 1999, p.845):

"It is by far the most 'sensitive' variable to any change in any 'parameter'; e.g. μ, *h*, Δt, IC, BC, *D*, ∂D, ..."

Another important aspect that is worth mentioning is that the boundary of the circular cylinder has been exactly represented by the curved edges of the finite elements employed. As well, the multi-connected domain did not create any difficulties in the generation of the solution or in the computation of the streamfunction, despite the absence of any special treatment.

6.2 Flow Over a Backward-Facing Step

6.2.1 Introduction

The flow in a channel with a backward-facing step is another flow that has been extensively studied both experimentally and numerically. The sudden increase of width on one side of the channel separates the flow from the lower boundary. The flow widens and an eddy is formed immediately behind the step where fluid gets recirculated. Studies involving this type of flow are usually done in the regime of turbulent flows. Thus, for $Re \geq 250$, the flow separates from the upper boundary downstream the step, and another eddy is formed (Armaly et al., 1983). The objective is to determine the lengths of the recirculation regions corresponding to the eddies.

The finite element program has been used to simulate the flow over a backward-facing step for $Re = 73$, which is a steady and laminar flow. The length of the recirculation zone immediately after the step is compared to the length corresponding to $Re = 73$ in Figure 3 from Atkins et al. (1980); this figure presents measured recirculation length against Re number. It should be mentioned that in order to eliminate the hydrostatic pressure mode, the gravitational acceleration is not taken into account during the calculations. The hydrostatic pressure can be added to the calculated pressure at the end.

6.2.2 Spatial Discretization

The computational domain is that of a *22m*-long channel whose initial width of *1m* increases suddenly to *1.5m* at *3m* from the inlet. Hence the height of the step is $h = 0.5m$. The unstructured finite element mesh of the computational domain is depicted in Figure 6.13. The mesh contains *775* isoparametric Taylor-Hood serendipity quadrilateral finite elements and *2548* nodes. The mesh is finer in the vicinity of the step to allow for a better representation of flow in this region.

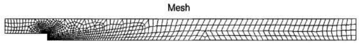


Figure 6.13 Flow over a backward-facing step – Unstructured finite element mesh of the computational domain

6.2.3 Boundary and Initial Conditions

The no-slip boundary condition $u = 0.0ms^{-1}$, $v = 0.0ms^{-1}$ is applied to the channel lateral walls. The *u*-velocity prescribed at the inlet (the left-hand side vertical boundary in Figure 6.13) has a parabolic variation with a maximum value $u_{max} = 1.0ms^{-1}$ at the centre, and matches the no-slip condition at the walls. The *v*-velocity at the inlet is $v = 0.0ms^{-1}$. The outflow open boundary condition $\frac{\partial u}{\partial x} = 0$, $\frac{\partial v}{\partial x} = 0$ is applied at the outlet. The reference value chosen for the pressure is $0.0Pa$ and is assigned to the nodal pressure at the node located at the upper end of the outlet.

The initial conditions specify a velocity field given by $u_0 = 0.0ms^{-1}$ and $v_0 = 0.0ms^{-1}$ at all nodes, except for those on the inflow boundary where it should match the prescribed boundary condition.

6.2.4 Results and Conclusions

The fluid density and its dynamic viscosity have been set to $1kgm^{-3}$ and $0.00457Nsm^{-2}$, respectively. The Reynolds number based on the height of the step is

$Re = \frac{\rho U h}{\mu}$. The average velocity at the inlet, $U = \int_0^1 u dy = \frac{2}{3} ms^{-1}$, has been considered as the characteristic velocity of the flow. Thus, the corresponding Reynolds number is approximately *73*.

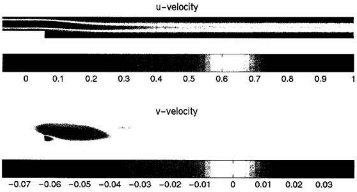


Figure 6.14 Flow over a backward-facing step – Velocity components u and v plots (in ms^{-1}) (*Re=73*)

The program was run tok a steady-state solution. Results of the simulation are illustrated in Figures 6.14-18. From Figure 6.15, which shows the reverse flow behind the step, it has been estimated the length of the recirculation region, $x_L = 2.5m$, which gives a ratio $\frac{x_L}{h} = 5$. Figure 3 in Atkins et al. (1980) provides a value of approximately *4.8*. It can be concluded that a fairly good agreement exists between the calculated and the experimental value.

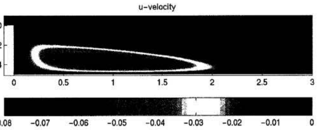


Figure 6.15 Flow over a backward-facing step – *u*-velocity plot (in ms^{-1}) (*Re=73*, zoom in behind the step, negative values of velocity)

Pressure, vorticity and streamfunction plots are presented in Figures 6.16-18.

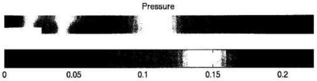


Figure 6.16 Flow over a backward-facing step– Pressure plot (in *Pa*) (*Re=73*)

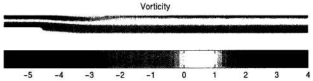


Figure 6.17 Flow over a backward-facing step – Vorticity plot (in s^{-1}) (*Re=73*)

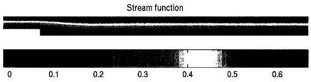


Figure 6.18 Flow over a backward-facing step – Streamfunction plot (in $m^2 s^{-1}$) (*Re=73*)

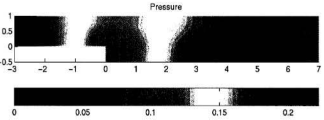


Figure 6.19 Flow over a backward-facing step– Pressure plot (in *Pa*) (*Re=73*, short computational domain)

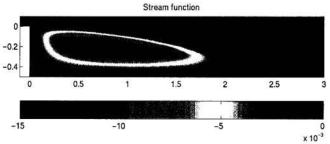


Figure 6.20 Flow over a backward-facing step– Streamfunction plot (in $m^2 s^{-1}$) (*Re=73*, short computational domain, values of the streamfunction inside the recirculation region)

It can be noticed that a smooth pressure field (Figure 6.16 and 6.19) has been obtained for this application, without using any 'smoothing' techniques in the post-processing phase. This shows that fairly accurate pressure distributions can be obtained using the same program. Figure 6.20 illustrates the recirculation region behind the step.

6.3 Mid-Latitude Wind-Driven Barotropic Ocean Circulation

6.3.1 Introduction

Models of wind-driven barotropic ocean circulation based on the Navier-Stokes equation must take into account the effect of the rotation of the Earth by including the Coriolis force/acceleration. In section 2.2, a set of approximations and assumptions are given, which allow for the two-dimensional incompressible Navier-Stokes equations (2.2.2-4) to be used for modelling mid-latitude wind-driven barotropic ocean circulation on a beta-plane, at regional scale. For a flat-bottomed ocean of depth h, driven by a wind-stress field (τ_x, τ_y), these equations become

momentum equations:

$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}-A_H\nabla^2u-fv+\frac{1}{\rho_0}\frac{\partial P}{\partial x}=\frac{\tau_x}{\rho_0 h}, \qquad (6.3.1)$$

$$\frac{\partial v}{\partial t}+u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}-A_H\nabla^2v+fu+\frac{1}{\rho_0}\frac{\partial \mathrm{P}}{\partial y}=\frac{\tau_y}{\rho_0 h}; \qquad (6.3.2)$$

continuity equation:

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}=0. \qquad (6.3.3)$$

The dissipation of momentum through bottom friction is neglected in the above equations; momentum is dissipated through lateral friction only.

Bryan (1963) has performed numerical simulations for wind-driven ocean circulation using a numerical model based on a finite-difference discretization of the Navier-Stokes equations in stream-function formulation. This model was used to study the influence of the non-linear terms on the flow patterns, especially in the area of the western boundary current.

The application presented in this section is concerned with the simulation of a mid-latitude wind-driven barotropic ocean circulation using the spatial domain, wind-stress field and parameter values from Bryan (1963). Although the numerical model based on the equations (6.3.1-3) is given in primitive variables and the finite element method is used for spatial discretization, it is expected to obtain similar results.

6.3.2 Spatial Discretization

The two-dimensional computational domain is a rectangle that corresponds to an idealized ocean gyre bounded by continents on the eastern and western sides, and by other two gyres on the northern and southern sides. The length of the rectangle in the east-west direction is $L = 2.5 \times 10^6\, m$, and the length in the north-south direction is $2L = 5 \times 10^6\, m$. It is considered that most of the mass transport takes place above the main thermocline for large-scale and time-averaged flow (Bryan, 1963). Therefore, the depth of the idealized ocean is taken as the depth of the main thermocline, in this case $h = 200m$.

The unstructured finite element mesh of the computational domain is illustrated in Figure 6.21. The mesh contains *580* isoparametric Taylor-Hood serendipity quadrilateral finite elements and *1833* nodes. Since the circulation is most dynamic near the western boundary – where the western boundary current occurs, and all quantities (i.e. velocity, pressure, vorticity, streamfunction) have high gradients – the mesh is finer in this region.

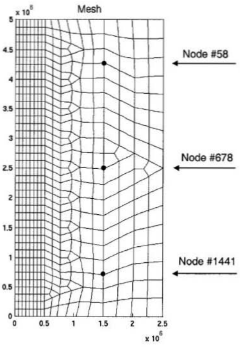


Figure 6.21 Wind-driven ocean circulation – Finite element mesh of the computational domain

6.3.3 Boundary and Initial Conditions

The no-slip boundary condition $u = 0.0ms^{-1}$, $v = 0.0ms^{-1}$ is applied to the west and east boundaries, where the domain is bounded by continents. Since the ocean gyre is contained in the computational domain, there is no outflow/inflow through the north and south boundaries. Hence v-velocity is $v = 0.0ms^{-1}$ here. Furthermore, slip boundary conditions are applied at these boundaries, i.e. the vorticity $\omega = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}$ is set equal to zero. This leads to $\frac{\partial u}{\partial y} = 0$, as $\frac{\partial v}{\partial x} = 0$ is brought about by the constant value assigned to the v-velocity.

The reference value chosen for the pressure is $0.0Pa$ and is assigned to the nodal pressure at the node located at the upper corner of the eastern boundary.

The initial velocity field is given by specifying $u_0 = 0.0ms^{-1}$ and $v_0 = 0.0ms^{-1}$ at all nodes, which means that the initial velocity field is divergence free. Thus, the ocean is considered initially at rest.

6.3.4 Wind Stress and Physical Parameters

The applied wind stress is taken to be constant in time, with the x- and y-direction components given by $\tau_x = -w_0 \cos\frac{\pi y}{2L}$ and $\tau_y = 0$, respectively. The value assigned to the amplitude of the wind stress is $w_0 = 0.2Nm^{-2}$.

The value of β at midlatitude, i.e. for $\theta = 45°$, is $\beta \cong 1.72\times10^{-11} m^{-1}s^{-1}$, and the value used for the density of the seawater is $\rho_0 = 1.025\times10^{3}\, kgm^{-3}$.

In order to calculate the Reynolds number (*Re*) and the Rossby number (ε) of the flow, a scale velocity is needed. This velocity is obtained assuming that an approximate geostrophic balance exists over most of the interior of the basin. Over this area, the nonlinear effects are negligible and the expression obtained for the scale velocity from Sverdrup's formula is (Bryan, 1963)

$$V_{Sverdrup} = \frac{w_0 \pi}{2\beta L \rho_0 h}. \tag{6.3.4}$$

For the values used in this application the scale velocity is $V_{Sverdrup} \cong 3.6\times10^{-2}\, ms^{-1}$. This velocity is characteristic for the interior flow and not for the western boundary current where the velocity can exceed $1ms^{-1}$. The value of the Rossby number is $\varepsilon = \dfrac{V_{Sverdrup}}{\beta L^2} \cong 0.33\times10^{-3}$.

The $Re = 100$ case has been analysed in this application. This value is obtained by applying the formula $Re = \dfrac{V_{Sverdrup} L}{A_H}$, for a large-scale horizontal eddy diffusion coefficient $A_H = 0.895\times10^{3}\, m^2 s^{-1}$. Runs have been performed for lower degrees of nonlinearity, namely for $Re = 5$, 30 and 60, but the results are not presented here.

6.3.5 Results and Conclusions

The time step used in all four cases is $\Delta t = 0.2328 \times 10^5 s$, which is equal to $(\beta L)^{-1}$.

The primitive variables (*u*, *v*, *P*) have been monitored at three nodes with the numbers *58*, *678*, and *1441*, whose locations can be seen in Figure 6.21. The velocity components and pressure exhibit oscillations characteristic to the geostrophic adjustment phenomenon (Salmon, 1998) that occurs due to impulsive application of wind forcing to the ocean initially at rest (Figure 6.22).

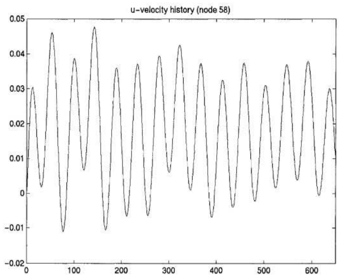


Figure 6.22 Wind-driven ocean circulation ($Re = 100$) – Variation in time of u-velocity at node #58 (the units are ms^{-1} on the y-axis and $(\beta L)^{-1} s$ on the x-axis)

The geostrophic adjustment is associated with the generation of westward propagating Rossby waves. A basic period of about *45* units of $(\beta L)^{-1} s$ has been obtained from the variations in time of the velocity components and pressure. For example, the basic period extracted from the pressure variation at the three nodes (Figures 6.23-24) is approximately equal to *44.75* units (i.e. *12.06* days), which is very close to that of a free wave for an equivalent barotropic model in the same rectangular domain (Bryan, 1963), namely *44.85* units. It can also be seen that there is a higher mode of oscillation with a period almost twice longer, which agrees very well with the results presented by Bryan.

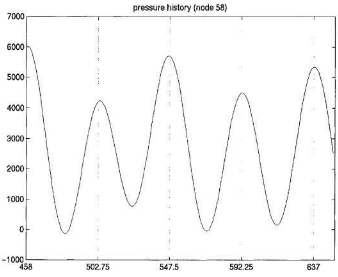
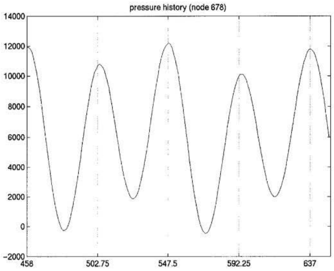


Figure 6.23 Wind-driven ocean circulation ($Re = 100$) – Variation in time of pressure at node #58 (the units are Pa on the y-axis and $(\beta L)^{-1} s$ on the x-axis)

a)



b)

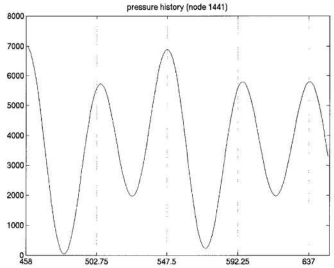


Figure 6.24 Wind-driven ocean circulation ($Re = 100$) – Variation in time of pressure: a) at node #678; b) at node #1441;

(the units are Pa on the y-axis and $(\beta L)^{-1}\,s$ on the x-axis)

The plot of the normalized stream function, averaged over the interval *458*-637 units (i.e. over *4* periods) is presented in Figure 6.25.a. The pattern of the stream function is very similar to that in Figure 8.b from Bryan's article. The averaging has been done in order to eliminate the effects of the transient Rossby waves. At $Re = 100$, the boundary current is unstable and a gyre of recirculation develops in the northwest corner of the domain (Figure 6.25.a-26). Another interesting feature is the counter-current that appears in upper-half of the domain, next to the boundary layer. Figure 6.25.b shows, for comparison purposes, the symmetric pattern of the normalized streamfunction when the nonlinear terms are dropped and a steady-state solution is obtained.

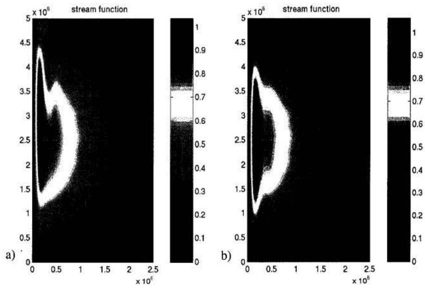


Figure 6.25 Wind-driven ocean circulation – Normalized streamfunction plot: a) (averaged) unsteady and nonlinear ($Re = 100$); b) steady and linear; (the units are Pa on the y-axis and $(\beta L)^{-1}\, s$ on the x-axis)

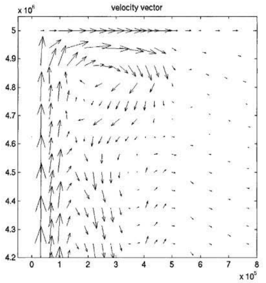


Figure 6.26 Wind-driven ocean circulation (*Re = 100*) – Velocity vectors in the northwest corner of the domain

Figure 6.27 depicts the plot of the pressure averaged over the same *4* periods. The patterns of the streamfunction and pressure show that an approximate geostrophic balance is achieved over most of the ocean basin.

The very good match of the predicted basic period of the Rossby waves and pattern of streamfunction with those presented by Bryan (1963), shows that the program is able to simulate with very good accuracy (given the coarse mesh used) geophysical flows, at least for ranges of Rossby and Reynolds numbers close to the values used in this application.

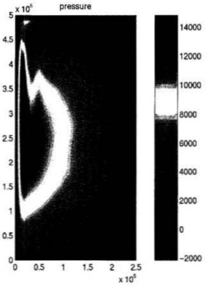


Figure 6.27 Wind-driven ocean circulation – Pressure plot (in *Pa*) ($Re = 100$)

7. CONCLUSIONS AND FUTURE WORK

The main objective of the present work was to develop a finite-element algorithm for solving the two-dimensional incompressible Navier-Stokes equations with Coriolis force, based on standard finite element techniques that have been used in the field of industrial flows simulation. The classical isoparametric Taylor-Hood serendipity quadrilateral finite element has been chosen for spatial discretization, as it performs better than the triangular elements for advection-dominated flows and can represent with a better accuracy curved boundaries. A version of the Crank-Nicolson scheme, which is unconditionally stable, has been used for time integration in order to circumvent the time step restrictions imposed by the explicit schemes. The use of the GID preprocessor for the generation of the finite element meshes and the preparation of the input files has proven to be successful and showed the robustness of the software. The numerical algorithm has been implemented into a program written in FORTRAN 90, and the sparse solver from the Compaq Extended Math Library has been used for solving linear systems. The presence of the entries corresponding to the Coriolis terms in the system matrix has led to the failure of the automatic optimization of the factorization performed by the sparse solver, which affected considerably the efficiency of this solver. The MATLAB environment has been chosen for postprocessing. The vorticity and streamfunction are calculated by determining their least-square best fit in spaces spanned by pressure basis functions and velocity basis functions, respectively, in a manner consistent with the FEM methodology.

The numerical algorithm has performed fairly well when tested on three benchmark problems, given the relative coarse meshes used. Since the quality of the results depends not only on the algorithm utilised but also on the skills of the finite element analyst, one could obtain better results by applying some practical finite element analysis techniques. These include the extension of the computational domain where open boundaries are present, mesh refinement of the regions of interest and avoidance of excessively distorted elements.

There is definitely much to be done for improving the algorithm before extending it to the three-dimensional case. Here are some suggestions:

1) Some other types of finite elements should be implemented to study their behaviour when the Coriolis force is present, in the search of the 'best' finite element for this case;
2) Newton-like methods should be used to deal with non-linearity, as they provide higher rates of convergence than the Picard iteration method;
3) Iterative solvers for the linear systems should be also implemented, as they are more efficient than direct solvers for large numbers of unknowns;
4) Implementation of a variable-step time integrator as a first step in the application of adaptive methods.

BIBLIOGRAPHY

Armaly, B., Durst, F., Pereira, J., Schonung, B. (1983): Experimental and Theoretical Investigation of Backward-Facing Step Flow. *J. Fluid Mech.*, **127**, 473-496.

Atkins, D.J., Maskell, S.J., Patrick, M.A. (1980): Numerical Prediction of Separated Flows. *Int. J. Numer. Meth. Eng.*, **15**, 129-144.

Braza, M., Chassaing, P., Minh, H.H. (1986): Numerical Study and Physical Analysis of the Pressure and Velocity Fields in the Near Wake of a Circular Cylinder. *J. Fluid Mech.*, **165**, 79-130.

Bryan, K. (1963): A Numerical Investigation of a Nonlinear Model of a Wind-Driven Ocean. *Journal of the Atmospheric Sciences*, **20**, 594-606.

Cuvelier, C., Segal, A., van Steenhoven, A.A. (1986): *Finite Element Methods and Navier–Stokes Equations*. D. Reidel Publishing Company, Dordrecht, Holland.

Dukowicz, J.K., Smith, R.D. (1994): Implicit free-surface method for the Bryan-Cox-Semtner ocean model. *J. Geophys. Res.*, **99**(C4), 7991-8014.

Dumas, E., Le Provost, C., Poncet, A. (1982): Feasibility of Finite-Element Methods for Oceanic General Circulation Modelling. Vol. 5, *4th International Conference on Finite Element in Water Resources*, Spinger-Verlag, 43-55.

Fix, G.J. (1975): Finite Element Models for Ocean Circulation Problems. *SIAM, J. Appl. Math.*, **29**, 371-387.

Girault, V., Raviart, P.A. (1986): *Finite Element Methods for Navier-Stokes Equations.* Springer Verlag, Berlin-Heidelberg, Germany.

Gresho, P.M., Lee, R.L. (1979): Don't suppress the wiggles – they are telling you something!. *Finite Element Methods for Convection Dominated Flows*, Chap. 3, 37-61, T.J.R. Hughes (Ed.), The American Society of Mechanical Engineers, New York, USA.

Gresho, P.M., Sani, R.L. (1999): *Incompressible Flow and the Finite Element Method.* John Wiley and Sons Ltd., Chichester, UK.

Gropp, W.D., Keyes, D.E. (1992): Domain Decomposition Methods in Computational Fluid Dynamics, *Int. J. Numer. Meth. Fluids*, **14**, 147-165.

Haidvogel, D.B., Robinson, A.R., Schulman E.E. (1980): The Accuracy, Efficiency and Stability of Three Numerical Models with Application to Open Ocean Problems. *J. Comput. Physics*, **34**, 1-53.

Heinrich, J.C., Idelsohn, S.R., Onate, E., Vionet, C.A. (1996): Boundary Conditions for Finite Element Simulations of Convective Flows with Artificial Boundaries, *Int. J. Num. Mech. Engr.*, **39**, 1053-1071.

Heinrich, J. C., Pepper, D.W. (1999): *Intermediate Finite Element Method: Fluid Flow and Heat Transfer Applications.* Taylor & Francis, Philadelphia, USA.

Hughes, T.J.R., Franca, L.P., Balestra, M. (1986): A New Finite Element Formulation for Computational Fluid Mechanics: V. Circumventing the Babuska-Brezzi condition: A Stable Petrov-Galerkin formulation of the Stokes Problem accommodating equal order interpolation, *Comp. Meth. Appl. Mech. Eng.*, **59**, 85-99.

Iskandarani, M., Haidvogel, D., Boyd, J. (1995): A Staggered Spectral Finite Element Model for the Shallow-Water Equations. *Int. J. Numer. Methods Fluids*, **20**, 393-414.

Kantha, L. H., Clayson, C.A. (2000): *Numerical models of oceans and Oceanic Processes*. Academic Press, San Diego, USA.

Kreiss, H.-O., Lorentz, J. (1989): *Initial-Boundary Value Problems and the Navier-Stokes Equations*. Academic Press, Boston, USA.

Ladyshenskaya, O.A. (1969): *The Mathematical Theory of Viscous Incompressible Flow*. Gordon and Breach, New York, USA.

Le Provost C. (1984): An Application of Finite Element Methods for Modelling Wind-Driven Circulations in a Stratified Ocean. *5th International Conference on Finite Element in Water Resources*, Spinger-Verlag, 567-576.

Le Provost C. (1986): On the Use of Finite Element Methods for Ocean Modelling. *Advanced Physical Oceanography Numerical Modelling*, NATO-ASI Series, Vol. 186, D. Reidel, 557-580.

Le Provost C., Bernier, C., Blayo E. (1993): An Intercomparison of Two Numerical Methods for Integrating a Quasi-Geostrophic Multilayer Model of Ocean Circulations. *J. Comput. Phys.*, **110**, 341-359.

Le Roux, D.Y., Lin, C.A., Staniforth, A. (2000): A Semi-implicit Semi-Lagrangian Finite-Element Shallow-Water Ocean Model, *Mon. Wea. Rev.*, **128**, 1384-1401.

Myers, P.G., Weaver A.J. (1995): A Diagnostic Barotropic Finite-Element Ocean Circulation Model. *Journal of Atmospheric and Oceanic Technology* , **12** , 511-526.

Reddy, J.N., Gartling, D.K. (1994): *The Finite Element Method in Heat Transfer and Fluid Dynamics*. CRC Press, Boca Raton, USA.

Salmon, R. (1998): *Lectures on Geophysical Fluid Dynamics*. Oxford University Press, New York, USA.

Taylor, C., Hughes T.G. (1981): *Finite Element Programming of the Navier-Stokes Equations*. Pineridge Press Ltd., Swansea, UK.

Tritton, D.J. (1988): *Physical Fluid Dynamics*. Oxford University Press, Oxford, UK.

*** (2001): *Compaq Extended Math Library Documentation*. Compaq, USA, at: http://www.compaq.com/math/documentation/cxml

*** (1999): *FLUENT 5.5 Documentation*. FLUENT Inc., USA, at: http://sp81.msi.umn.edu:999/fluent/fluent5/index.htm

*** (2001): *GID Reference Manual*. International Centre for Numerical Methods in Engineering (CIMNE), Barcelona, Spain, at: http://gid.cimne.upc.es/support/gid_toc.subst