University of Southampton

Two Dimensional Flow Simulation using Discrete Vortex Methods on MIMD Processor Arrays

Volume 1 of 1

by Nicholas Richard Clarke

Doctor of Philosophy

Department of Aeronautics and Astronautics

September 1992

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING

AERONAUTICS AND ASTRONAUTICS

Doctor of Philosophy

TWO DIMENSIONAL FLOW SIMULATION USING DISCRETE VORTEX METHODS ON MIMD PROCESSOR ARRAYS

by Nicholas Richard Clarke

The discrete vortex method is a Lagrangian technique for solving the two-dimensional Navier-Stokes equations for an incompressible homogeneous Newtonian fluid. The construction of a robust numerical solver, based on the discrete vortex method, is discussed in detail. The viscous effects are modelled using both the random walk and diffusion velocity techniques. The computational cost is reduced by using a zonal decomposition algorithm for the velocity summation and by formulating an implementation suitable for execution on MIMD[1] parallel architecture computers.

The code is validated using the problem of impulsively started flow past a circular cylinder at Reynolds numbers ranging from 300 to 31,700. Short/medium time solutions for both the rotating and non-rotating cases are used to confirm the accuracy of the method by comparison with experimental and numerical results from the literature. Long time solutions for the non-rotating case reveal non-determinism at high Reynolds numbers: the flows can exhibit a multiplicity of vortex shedding patterns.

The low-speed stall process on NACA four-digit wing sections is visualised. A detailed comparison of the flow past a symmetric aerofoil with the flow past a cambered aerofoil shows that the camber delays the onset of the stall.

[1] Multiple Instruction Multiple Data

Acknowledgements

I would like to thank Dr. O. R. Tutty for supervising this work, and Dr. J. S. Reeve for helping to supervise the first year. I am grateful to the Science and Engineering Research Council for funding the first year and providing computing resources. The work would not have been completed without support from Southampton University Computing Services, who have provided me with employment and computing resources for the final four years. Thanks to Prof. A. J. G. Hey for starting me on this project in the first place, Ms. V. L. Pointon for her support and help during the writing of the thesis, R. G. Watts for his help, and B. V. for their encouragement.

Table of Contents

Table of Symbols

ω	vorticity of fluid
$\underline{u}=(u_x,u_y)$	velocity of fluid
ψ	streamfunction of fluid
t	time
Re	Reynolds number
N_V	number of discrete vortices in model
Γ_j	strength of j^{th} vortex
γ_j	vorticity distribution of j^{th} vortex
$\underline{r_j}(t)$	position of j^{th} vortex at time t
σ	core radius of a vortex
δt	time step
$\underline{v}=(v_x,v_y)$	diffusion velocity
λ_j	strength of j^{th} panel element
$\underline{U_e}$	external flow around body
C_D	drag coefficient
C_L	lift coefficient
N_P	number of vortices in a zone
n_t	number of terms in Laurent series for zonal sum
λ	'centre' point of a zone
h	convergence ratio for a zone
τ	non-dimensional time
$\delta\tau$	non-dimensional time step
α	rotational: translational speed ratio for rotating cylinder
$\underline{n}$	unit normal to body boundary
$\underline{s}$	unit tangent to body boundary
d_1,d_2,θ	flow comparison quantities (see p. 96)

1. Introduction

The concept of the vorticity of a fluid has been important to the development of theoretical and computational fluid dynamics. In principle, an incompressible flow can be specified entirely by the combination of a potential flow and the kinetics generated by the local rotation of the fluid. This principle is applied directly when the fundamental equations of fluid motion, the Navier-Stokes equations, are written as vorticity transport equations.

Many theoreticians have found vorticity useful as a mathematical tool. In certain conditions, such as those used throughout this work of a two-dimensional incompressible Newtonian fluid of constant density, the rules for vorticity transport can be significantly simplified. Thus many researchers have favoured analysis using vorticity as a base quantity, rather than using the primitive variables of velocity and pressure. In addition, powerful generalisations such as Kelvin's circulation theorem can be applied to make deductions about the behaviour of the vorticity of a fluid. In numerical studies using vorticity, there are two categories of research which completely dominate the field — the Eulerian and Lagrangian approaches to solving the vorticity transport equations.

The Eulerian approach treats the physical space occupied by the fluid as fundamental and considers the fluid flowing through the space. This typically involves the introduction of a computational grid superimposed on the region containing the fluid. The objective is then to calculate the values of vorticity (and usually another variable, such as the streamfunction) within each cell of the grid. Thus a discrete approximation to the vorticity, at particular points in time and space, is established.

The Lagrangian approach treats the substance of the fluid as fundamental and uses a frame of reference which moves with the fluid. This involves tracking the position and value of quantities embedded in fluid particles. For a vorticity model, these elements are typically localised areas of rotating fluid. Thus the vorticity field is specified by the creation, subsequent positions and possible destruction of a set of fundamental vortex elements.

The discrete vortex method is of the Lagrangian type. The elements are called either discrete vortices (which may include point vortices) or vortex blobs. The simplest discrete vortex method models an infinite inviscid fluid (ie. no boundaries are present). Starting with an initial configuration of discrete vortices to represent a vorticity distribution, the mathematics dictates that the vortices obey a gravity-like body-body interaction law. Thus an

assembly of vortex blobs has a self-convective effect upon itself which models the motion of the fluid.

Solid boundaries can be introduced into the inviscid model by using the standard potential theory methods. Two of the most commonly used alternatives are a panel method or a conformal transformation, both of which have relative advantages and disadvantages. Either can be applied to create a boundary which obeys the inviscid no-penetration condition. We opted to use a panel method, principally because it could be made to model a general body-shape, and developed a new panel element with mathematical properties tailored to give more accurate results in the discrete vortex method.

The effects of viscosity may be added to the model by making two modifications to the inviscid model. Firstly, a mechanism must be added to satisfy the viscous no-slip boundary condition, the physical effect of which is known to be that vorticity is created at the boundaries. Thus, in the discrete vortex model, the no-slip condition is satisfied at the boundaries by introducing new vortices of the correct strength. Secondly, a diffusive component is included in the motion of the vortices in order to model the additional Laplacian term in the Navier-Stokes equations.

The complete Navier-Stokes discrete vortex model has several free parameters, mainly concerning the details of the vortex creation mechanism at the boundaries. These parameters must be adjusted carefully and verified with extensive testing of results, in order to construct a robust numerical solver.

A significant disadvantage of the basic discrete vortex method is its high computational cost. Since each vortex is influenced by all the others, the operation count for updating the flow field rises as the square of the number of vortices. We have tackled this problem in two ways. Firstly, a zonal decomposition/summation algorithm has been developed which significantly reduces the operation count for each time step. Secondly, the software has been designed to run on parallel processing hardware in order to gain the performance benefits these systems offer.

The zonal decomposition/summation algorithm was formulated specifically with the parallel implementation in mind. The basis of the algorithm is to artificially divide the vortices into a hierarchical structure of 'zones'. For each zone, the vortices contained are 'accumulated' using a far field mathematical approximation. Then, when computing the contribution of the entire vortex assembly to the motion of the fluid at a point, much of the sum may be derived from zonal contributions. Since the zonal structure is hierarchical, some of the zones may contain a large number of vortices and hence the operation count is

significantly reduced. The resulting operation count using this technique has a close to linear relationship with the number of vortices used to model the fluid.

Parallel processing, although in its infancy when this work was started in 1987, is now accepted by the computing community as the most cost effective route to building high performance supercomputers. Current tools for developing parallel software are still relatively crude, making the task more time-consuming than software development for sequential architectures. However, the potential performance benefits can quickly recover the added development cost in a suitable computationally intensive project. The aim is to write software which has a performance that scales linearly with the number of processors used. The discrete vortex algorithm was reformulated with this goal and results confirm that a near linear performance relationship has been achieved. The hardware we have used to generate results is relatively inexpensive, which has allowed us to conduct very long runs on both Southampton University Computing Services' and Departmental equipment. Thus the maximum problem size the code has computed is as large as any we have found in the literature. This, combined with a careful mathematical formulation, including some of our own improvements to the model, has produced a solver with predictive power at least as good as those we have found in the literature, for a suitable set of problems.

The initial motivation for using the discrete vortex method was to conduct a several-threaded attempt to evaluate modern computational fluid dynamics techniques using parallel hardware. This work is one thread, others have produced a parallel spectral-element Navier-Stokes code and a parallel finite-volume Euler solver. At the start of the project, the discrete vortex method had produced promising results for basic problems, but seemed to be limited by computational cost. We decided to evaluate the method using large computations and then apply it to generate novel fluid dynamics results.

The Navier-Stokes solver has been applied to the impulsively started flow past an arbitrary shaped bluff body. This problem has many applications in hydrodynamics, such as evaluation of the stress on marine load-bearing structures or examination of laminar mixing behaviour in a fluid. In addition, some low-speed aerodynamics problems, including the low-speed stall process, fall into the flow regime that can be accurately solved.

Following this introductory chapter, a literary survey is presented which gives a brief chronological history of the method and reviews the main papers from which this work has drawn. Some pointers are also given to closely related topics which are outside the scope of this text.

The third chapter details the mathematics of the method. Working from the Navier-Stokes equations, the method is constructed in an intuitive manner. Included are details of the new

panel method, which has been used to satisfy the boundary conditions at the body surface. Explanations are given of the many possible modelling techniques which are used in discrete vortex methods. Some advantages and disadvantages of each of these are revealed and, for those which are used here, implementation details are presented. Finally, several methods for evaluation of force coefficients are detailed.

A description of the computational aspects of the method follows in chapter four. Where possible, the algorithms are presented in a computer-language independent manner. The theory and implementation of the zonal decomposition/summation algorithm are detailed; its accuracy is confirmed numerically. An evaluation of the expected operation count is accompanied by an experimental verification of the algorithmic performance using timings of a long run. The parallel algorithm is explained in a step-by-step manner, including a discussion of performance-related issues and timings. The chapter concludes with a discussion of the flow visualisation aspects and run-time considerations of the code.

The fifth chapter is the most extensive and contains the fluid dynamical results of this work. It opens with an extensive test of the new panel method, giving flow visualisations and demonstrating rapid convergence towards the correct boundary conditions. This is followed by an experimental examination of the convergence of the discrete vortex method, showing how solutions vary with changes in input parameters. The model is then tested using an extensive range of impulsively started flows past circular cylinders. The non-rotating case is considered first. Discrete vortex results are compared to experimental and numerical visualisations for a range of Reynolds numbers at short times and then long term solutions are presented, highlighting non-deterministic phenomena occurring at high Reynolds numbers. The second test case is the rotating and translating circular cylinder, for which the discrete vortex method produces especially close agreement with experimental and finite difference results. Finally, results for flow past various NACA four-digit wing sections are presented in order to visualise the low-speed stall process.

The final chapter is the conclusion; a summary and discussion of the body of the thesis is presented and some ideas for further research are given.

2. Existing Research

2.1 Overview

This chapter provides a concise background to the existing research into discrete vortex methods, and gives an outline of the chronological history of the subject. Related topics not followed in detail in this work are also discussed briefly.

The methods are controversial; discrepancies can be found between different research papers. This is due largely to the ad-hoc nature of some of the assumptions which must be made in the formulation of the methods. Supporters of discrete vortex methods praise their intuitive nature and their flexibility; in these papers, adequate solutions are found to all the problems which arise and the conclusions are positive. Critics, often making comparisons with some other favoured method, suggest that the problems encountered in the construction of a discrete vortex method are insurmountable and render the results inaccurate. A newcomer to the subject might presume that the truth lies somewhere between these two extremes. We have found that discrete vortex methods are not a universal replacement for other numerical techniques, but can provide good results for a worthwhile set of problems if constructed carefully.

A completely comprehensive study of the literature available in this subject area would be a sizeable task. Fortunately, as with most well-researched topics, there are several good review papers which make the job more manageable for an individual. To illustrate the scale of material available, one of the most thorough of the reviews, Sarpkaya (1989), cites over 500 references.

Section 2.2 is intended to provide an introduction to the literature available on the subject of discrete vortex methods. Brief summaries are given of the most useful review papers and a few other important papers. Section 2.3 gives references to papers concerning the application of vorticity dynamics to the construction and convergence of discrete vortex methods. A historical overview of early discrete vortex models is included, following the subject from the days of hand calculation of the motion of a few vortices through to the study of the evolution of a vortex sheet using a collection of vortices. Discrete vortex methods have almost exclusively been applied to solving incompressible flows, with the majority solving only the two-dimensional case; the basic model solves the inviscid Euler equations. Section 2.4 gives references for the various models for viscosity which have been devised, allowing extension of the method to give solutions of the unsteady Navier-Stokes equations. The experimental and numerical fluid flow research papers which we have used are reviewed in

section 2.5. These papers provided reliable flow visualisations which allowed the method to be validated and its accuracy to be assessed.

The final three sections of this chapter review research which is not strictly necessary to this work. However, the papers referenced contain discussions and results which provided useful insights during the construction of the method used here. These references are also provided to give an idea of the wider subject area beyond the scope of this thesis. Section 2.6 reviews papers which help provide a comprehensive background to the subject. The topics are closely related to this work, but were not used to develop the techniques presented here. Much of the existing research concerns numerical/computational algorithms to reduce the cost of the calculations involved. In this work, we use a zonal decomposition/summation algorithm (see section 4.5), although a summary of research using the more common technique, vortex-in-cell, is given in section 2.7. Research into three-dimensional discrete vortex methods, which are suitable for solution of relatively simple problems, is discussed in section 2.8.

2.2 First Readings

The original papers on discrete vortex methods were by Chorin (1973, 1978); the review papers we have made most use of are by Sarpkaya (1989), Leonard (1980) and Spalart (1988). The latter provide a comprehensive guide to the subject and all of the commonly used techniques are at least mentioned.

Chorin (1973, 1978) was the first to use the random walk and claim solution of the Navier-Stokes equations. The description of the basic method is clear, with the test case of flow past a circular cylinder. We found that insufficient detail was included to reconstruct his results.

Sarpkaya (1989) is a comprehensive review paper, covering most aspects of discrete vortex methods in great detail. The theory is developed from the relevant mathematical equations, with pragmatic descriptions of pitfalls for the modeller to avoid. Comparisons are made between techniques used by various researchers and the quality of results achieved. Emphasis is placed on engineering applications, as opposed to mathematical tests of the model. Although the author has been involved in a great deal of work using discrete vortex methods, he is critical of much of the research reviewed.

Leonard (1980) is essentially a review paper, but also presents some new theory concentrating on the fundamental mathematics of the model, such as vortex distributions and the accuracy of convection schemes. A particularly clear explanation of other papers dealing with convergence of the method is presented.

Spalart (1988) is a very positive paper including some impressive results of flow over complicated body shapes (see Figure 1). Details of the main techniques are explained in terms of the physics of the flow. The paper is a 'recipe book' for discrete vortex methods, providing instructions on how to quickly prepare good models for a variety of aerodynamics problems.

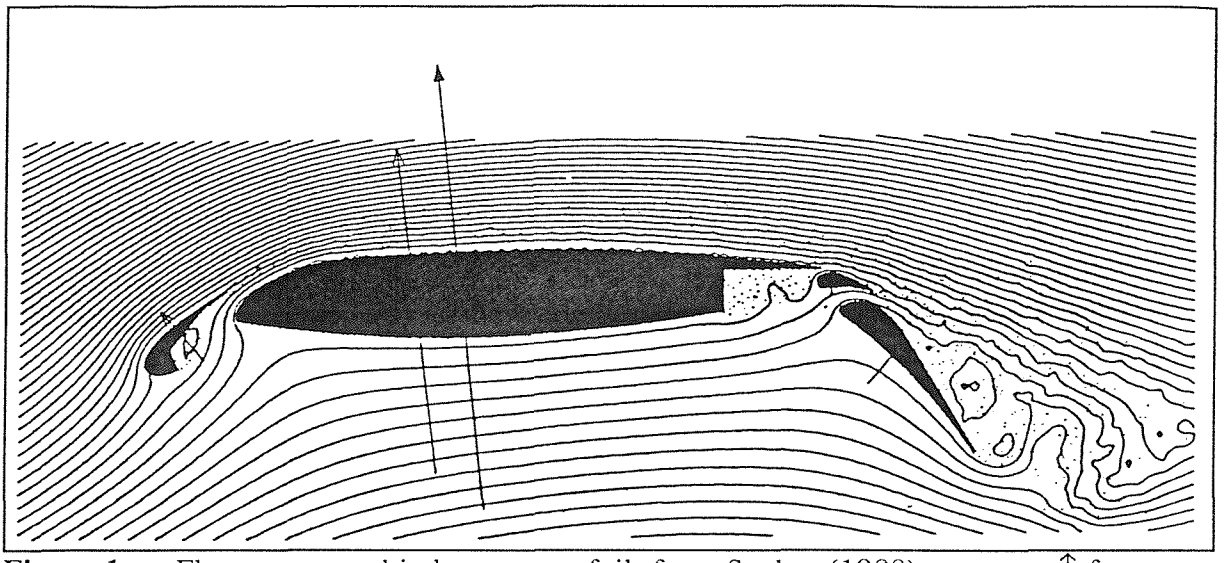


Figure 1 Flow past a multi-element aerofoil, from Spalart (1988). · vortex; ↑ force; — streamline.

Smith and Stansby (1988), using the vortex-in-cell method, present some of the best results for impulsively started flow past a circular cylinder, the test case used in our research. The results for early times match both experiments and other numerical solutions well, although the apparent imposition of transverse symmetry in their model precludes its use beyond the time of symmetry breaking, which occurs in experiments and numerics.

2.3 Vortex Dynamics

Helmholtz (1867) was the first to show that, in an inviscid fluid, vortex lines move with the fluid particles and that some rotational flows can be usefully modelled using vortex lines of appropriate circulation. Following the important work of von Karman, Rosenhead (1931) attempted to model the roll-up of a vortex sheet by treating it as a system of point vortices.

The evolution of a vortex sheet has been the subject of great interest in the literature. This subject is relevant to our work as the trailing vortices behind a bluff body can be modelled with a spiral of rolled-up vortex sheet. The stability of such a spiral has been studied theoretically by Saffman (1974), Moore (1976, 1981, 1984) and experimentally by Pierce (1961), whose conclusion is that Helmholtz waves are formed on the sheet, which can degenerate into turbulence. This turbulence then mixes the nearby parts of the sheet and thus

smooths the core of rolled up regions. This constitutes the justification for the circulation reduction technique used in discrete vortex models (see section 3.10.1). Sheets have also been modelled using discrete vortices by Fink and Soh (1978) and Krasny (1986).

These works led eventually to the idea that continuous regions of vorticity could be modelled by a number of discrete vortices embedded in a potential flow. The objective of such a model is to evolve the Lagrangian (or Lagrangian-Eulerian) description of the discretised vorticity field. Convergence proofs are given by Hald (1979) and Beale and Majda (1982), showing that smooth distributions of vorticity represented by discrete vortices and modelled consistently (see chapter 3) solve the Euler equations.

The application of boundary conditions to a vortex method requires the assignment of values to a number of mathematically arbitrary numerical parameters (eg. Starting positions of vortices, size of vortex cores). These parameters should be adjusted to give good results for a wide range of problems. However, an extensive search for optimum values becomes difficult even with relatively few parameters; in practice sensible values may be assumed for certain physical quantities. A good understanding of the underlying vorticity dynamics is needed to ensure that the reasoning used in these assignments is justified. Introductions to basic vorticity dynamics may be found in Lighthill (1963), Batchelor (1967) and Morton (1984). The most explicit is Morton whose aim is to explain, in terms of the physics, the areas of vorticity dynamics which have traditionally remained mathematical. The previous treatments by Lighthill and Batchelor are discussed and areas for further investigation are highlighted. In particular, Morton feels that the boundary conditions of vorticity dynamics have not been fully explained. Using simple exact solutions, some deductions are made about the nature of the creation and subsequent motion of vorticity.

2.4 Viscosity Models

The random walk viscous diffusion model for discrete vortex methods was proposed by Chorin (1973) and used to simulate slightly viscous flow past a circular cylinder. The basis of the model is to add a 'Brownian motion' component to the motion of the vortices; this has been proven, by Einstein (1956) and Wax (1954), to be statistically equivalent to a diffusion. Chorin states that large numbers of vortices must be used to smooth the noise created by the random walk and that results will become invalid at low Reynolds numbers where the magnitude of the random component of the motion becomes large. The random walk is also used by Sarpkaya and Ihrig (1986), Smith and Stansby (1988), Spalart (1988) and Cheer (1989).

The viscosity model using growing cores is discussed in Leonard (1980). In this model, the viscous diffusion is simulated by allowing the discrete vortices' core sizes to increase with time. Intuitively, this is in an appealing technique as a single spreading Gaussian vortex is an exact solution of the Navier-Stokes equations. However, not many authors have used the technique, possibly because there is some doubt that it converges to solve the correct equation (see section 3.5).

The diffusion velocity method introduced by Ogami and Akamatsu (1991) uses the fundamental principle of diffusion: a diffusive quantity has a positive flux from a region of higher concentration to a region of lower concentration. Mathematical analysis is presented which gives each vortex a 'diffusion velocity' in addition to the convection velocity. An argument is given that the technique solves the correct equation and it does not introduce excess noise into the solutions. Superior convergence is demonstrated in the paper (as compared to the random walk method) for a simple one-dimensional problem. No mention is made of the problem whereby the diffusion modelled by this technique is limited to regions where the vortices are overlapped (see section 3.5.2).

The grid-based diffusion method introduced by Graham (1988) is suitable for use with the vortex-in-cell method (see section 2.7). The diffusion term of the Navier-Stokes equation is taken into account using a conventional finite-difference solution on the vortex-in-cell grid. The vorticity is then reinterpolated onto the discrete vortex field.

2.5 Experimental/Numerical Comparison

In order to compare our results with experiments and numerical simulations, it was necessary to widen our literary search to include some mainstream fluid dynamics papers. Perhaps surprisingly, there are few good visualisations of high Reynolds number flows and there is a similar dearth of numerical work. In particular, visualisations at long times for Reynolds numbers greater than 1,000 are scarce — possibly due to the eventual onset of turbulence in experiments and non-deterministic behaviour in numerical work.

The experimental works of Bouard and Coutanceau, in conjunction with the finite-difference numerical work of Ta Phouc Loc, Dennis and Badr give an extremely clear and comprehensive set of visualisations for the early stages of flow past a circular cylinder at moderate Reynolds numbers. The three key papers used in this work are Bouard and Coutanceau (1980), Ta Phouc Loc and Bouard (1985) and Badr, Coutanceau, Dennis and Menard (1990). In each paper, a discussion is given of the observed fluid structures. Some classification is made of the complex behaviour of the flows into categories for differing Reynolds numbers.

Van der Vegt and Huijsmans (1984) and van der Vegt and de Boom (1985) present an interesting and highly complex numerical scheme. A variational technique is used to solve the Euler equation which claims to smooth the solution and reduce the computation involved by use of cubic spline fitting and fast Fourier transforms. Force results are in qualitative agreement with their own experiments at Reynolds number 31,700 — both papers show an interesting multi-phase/chaotic behaviour in the experiments which we have found also in our calculations, see section 5.4.11.

Ishii, Kuwahara, Ogawa and Chyu (1985), Tamura, Ohta and Kuwahara (1990) and Rao and Kuwahara (1991) present various finite difference results (including some visualisations) for flow past square and circular cylinders up to very high Reynolds numbers. Some results are included for times beyond the symmetry breaking point.

Franke, Rodi and Schönung (1990) use a finite volume technique to solve for flow past square and circular cylinders at Reynolds numbers up to 5,000. Visualisations are given only for the lower Reynolds numbers used, but other results are given for the whole range.

2.6 Related Topics

The references that follow have not provided much direct input to the work undertaken here. However, they are useful as background reading and may be of more relevance to future research.

Sethian and Ghoniem (1987) uses a discrete vortex method to solve flow over a backwards-facing step. Results are presented for several Reynolds numbers in the range 50 to 5,000. Accuracy and convergence are discussed.

Tiemroth (1986), a Ph.D. thesis, describes in great depth the history of vorticity-based fluid models. The primary motivation of this work is to derive the force on cylindrical members (such as the upright supports of an oil platform) in the ocean. After a long and positive introduction, which explains the underlying physics very clearly, the vortex-in-cell method is used, claiming an operation count $O(N_v^{3/2})$ in number of vortices. Analysis is used to derive a formula for force although, unfortunately, it is specific to a circular cylinder. Results are presented for short times only and the flow visualisation presentation is somewhat disappointing.

2.7 Vortex-in-Cell Method

The high computational cost of direct discrete vortex calculations has always been a problem. Reducing the cost allows more computational elements to be used, giving better resolution of the flow characteristics. The cloud-in-cell method was originally developed at Los Alamos (USA) in 1955 (see Harlow (1964)) to help reduce the cost of a Lagrangian problem. The basis of the method is to introduce an Eulerian mesh through which the Lagrangian particles move (see section 4.5.1 for further details).

Christiansen (1973) was the first to apply the method to the interaction of vortices, coining the more common name when applied to discrete vortex methods, ie. the vortex-in-cell method. The vorticity is transferred from the particles to the mesh in order to solve the Poisson equation for the streamfunction, and hence calculate the fluid velocity.

Following Christiansen, the vortex-in-cell method is discussed by Baker (1979), Leonard (1980), Smith and Stansby (1988) and Graham (1988). This numerical scheme has the advantage that the smoothing of the vorticity over a grid at each time step tends to stabilise the problem. Sarpkaya (1989) explains this effect as 'a singular problem is desingularised by an artificial viscosity'. Unfortunately this smoothing is also a major disadvantage — the grid-based artificial viscosity is completely removed only by dispensing with the grid.

2.8 Three-Dimensional Models

Leonard (1980, 1985) describes several three-dimensional flow models and applications. The fluid problems considered are far more fundamental (from an engineering point of view) than those possible in two dimensions. Typical examples are instability of vortex rings (Ashurst (1981)), the evolution of a turbulent vortex (Chorin (1981, 1982)), the evolution of a time-developing round jet (Ashurst (1983)) and interactions of solitons on a rectilinear vortex (Aref and Flinchem (1984)). Practical aerodynamic problems cannot yet be attempted partly because of the large computational cost involved and partly because the mathematical techniques have not reached sufficient maturity.

The methods are broadly grouped into two categories. The most common category use a vortex filament model, in which each vortex is represented by a chain of vortex elements joined to approximate a curve. Unfortunately, these curves have a natural tendency to wrap-up into progressively tighter loops and consequently, methods must employ some form of rediscretisation to accurately represent the increasingly complicated filaments (see Chorin (1982)). The alternative category employs 'vortons', which are three-dimensional discrete vortices that are isolated in space and can stretch independently. The vortons have a

convective influence upon each other in much the same way as do the vortices in the two-dimensional model.

The vorton approach appears to hold more promise of eventually leading to a general purpose three-dimensional Navier-Stokes solver, but is still far from this at present.

2.9 Summary

We have presented a brief introduction to discrete vortex methods and given references to the key papers in the subject. A flavour has been given of some related research which has followed a slightly different course to this thesis. Pointers into the main body of the text have been given in an effort to improve its functionality as a reference material.

3. Mathematics and Modelling

3.1 The Equations

The Navier-Stokes equations, in a streamfunction-vorticity form, for a two-dimensional homogeneous incompressible Newtonian fluid are,

$$\frac{\partial \omega}{\partial t} + (\underline{u}.\nabla)\omega = \frac{1}{Re}\nabla^2\omega \tag{1}$$

$$\nabla^2\psi = -\omega \tag{2}$$

$$\underline{u} = -\nabla\times(\psi\,\underline{k}) \tag{3}$$

where $\underline{u}(\underline{r},t)$ is the velocity of the fluid at position $\underline{r}$ and time t, $\psi(\underline{r},t)$ is the streamfunction, $\omega(\underline{r},t)$ is the vorticity (defined by $\omega\underline{k}=\nabla\times\underline{u}$, where $\underline{k}$ is the unit vector in the direction perpendicular to the plane of the velocity) and Re is the Reynolds number aU/ν (a is a characteristic length, U is a characteristic velocity and ν is the kinematic viscosity of the fluid). These equations form a closed system and hence, with suitable boundary conditions, may in principle be solved to give the motion of the fluid. This implies that the motion of the fluid may be represented solely by the creation/destruction and the motion of the vorticity field.

In the following sub-sections the method of discrete vortex dynamics will be developed as a tool to solve equations (1), (2) and (3). Initially the simpler case of infinite fluids with no fixed boundaries will be dealt with, then the method will be extended to solve flows past bluff bodies placed in an infinite fluid.

3.2 Discrete Vortices

We construct the discrete vortex method starting from the premise that the vorticity field may be partitioned into an assembly of N_V discrete vortices. This assumption is made with little discussion in much of the literature (possibly because the resulting error is difficult to assess) but is thought to be the major source of inaccuracy of the method. Mathematically the vorticity is now

$$\omega(\underline{r},t) = \sum_{j=1}^{N_V} \Gamma_j \gamma_j\left(\underline{r}-\underline{r}_j(t)\right) \tag{4}$$

where γ_j is the vorticity distribution of the discrete vortex located at $\underline{r}_j(t)$, and Γ_j is its strength. It is usual to use distributions where the vorticity depends only upon distance from the centre of a vortex and where all vortices have the same shape (to reduce the cost of computing the velocity field) and core size (to improve convergence, see section 3.4), so we can write

$$\gamma_j(\underline{r}) = f(|\underline{r}|,\sigma) \tag{5}$$

where σ is a measure of the core size of a vortex. Normalising the distribution using

$$2\pi \int_{r=0}^{\infty} r f(r,\sigma)\, dr = 1 \tag{6}$$

we make Γ_j the total circulation of the jth vortex, the most meaningful measure of strength.

The problem of following the motion of a continuous vorticity field has now been reduced to that of following the motion of N_V discrete vortices. In other words the aim of the vortex method is to calculate the quantities $\underline{r}_j(t)$.

3.3 Inviscid Fluids

The relationship $\omega\underline{k}=\nabla\times\underline{u}$ may be inverted using the Biot-Savart law, giving the velocity in terms of the vorticity

$$\underline{u}(\underline{r}) = \frac{1}{2\pi}\underline{k}\times\int_{\mathbf{R}^2}\frac{(\underline{r}-\underline{x})\omega(\underline{x})}{|\underline{r}-\underline{x}|^2}|\underline{dx}| \tag{7}$$

where $|\underline{dx}|$ is the area element at position $\underline{x}$ and the integration is performed over the entire plane. Substituting (4) and (5) gives the velocity induced by the jth discrete vortex as

$$\underline{u}_j(\underline{r},t) = \begin{cases} \underline{k}\times\Gamma_j\dfrac{\underline{r}-\underline{r}_j(t)}{|\underline{r}-\underline{r}_j(t)|^2}F(|\underline{r}-\underline{r}_j(t)|,\sigma) & ;\ \underline{r}\neq\underline{r}_j \\ 0 & ;\ \underline{r}=\underline{r}_j \end{cases} \tag{8}$$

where

$$F(r,\sigma) = \int_{s=0}^{r} s f(s,\sigma)\, ds \tag{9}$$

The total velocity is

$$\underline{u}(\underline{r},t) = \sum_{j=1}^{N_V} \underline{u}_j(\underline{r},t) \tag{10}$$

For an inviscid fluid, equation (1) becomes the Euler equation,

$$\frac{D\omega}{Dt} \equiv \frac{\partial\omega}{\partial t} + (\underline{u}.\nabla)\omega = 0 \tag{11}$$

which is pure convection of vorticity. This is solved exactly for a system of point vortices in an infinite fluid by letting each vortex move at the local fluid velocity (see Batchelor 1967), ie.

$$\frac{d\underline{r}_j}{dt} = \underline{u}(\underline{r}_j(t),t) \ . \tag{12}$$

Using equation (10), equation (12) and a suitable time stepping scheme, it is possible to follow the vortex positions which, as explained in section 3.2, solves the motion of the fluid. Standard time stepping schemes from the literature can be used, for a full discussion of those used here see section 4.6, p. 61.

3.4 Vortex Distributions and Instabilities

Using point vortices gives an exact solution of (11), but numerically they tend to cause instabilities. A point vortex has distribution

$$\gamma_j(\underline{r}) = \delta(\underline{r}) \tag{13}$$

where δ is the two-dimensional Dirac distribution (in effect δ is infinite at the origin, zero elsewhere and its integral over an open domain containing the origin is 1). Calculating the velocity distribution for this vorticity (and implicitly assuming no self-induction as in equation (8)) yields the following

$$\underline{u}(\underline{r},t) = \frac{1}{2\pi}\underline{k} \times \sum_{j=1}^{N_V} \Gamma_j \frac{[\underline{r}-\underline{r}_j(t)]}{|\underline{r}-\underline{r}_j(t)|^2} \tag{14}$$

The vorticity distributions are singular, as are the velocity fields they induce. Consequently, it is no surprise that a numerical time-stepping scheme based on a point vortex system will be become unstable after a finite time (see Krasny (1986)).

Using less singular vorticity distributions stabilises the method numerically (Hald (1979), Beale and Majda (1982) give convergence proofs for smooth distributions in an unbounded fluid). The use of vortex blobs, as they are commonly termed, has one disadvantage: strictly speaking the nonlinearity of the Euler/Navier-Stokes equations forbids superposition of velocity fields induced by non-point vortices. Physically, this error manifests itself as the inability of the blobs to distort with local variations in velocity. They retain their original shape and just convect through the fluid. However, the error has been taken into account in the convergence proofs and, in time, does not become the main source of inaccuracy.

Leonard (1980) shows that the velocity summation converges with a greater degree of accuracy if all vortices have the same core size. Leonard also presents a scheme (which he calls 'scheme B') for calculating the vorticity-weighted average of velocity over a vortex blob. He claims that this scheme gives better convergence and removes the restriction of having a uniform core size. However, the integral to be evaluated is much more complicated than simply taking the velocity at the centre of the vortex, and we were unable to find analytic solutions for any vortex blob distributions. Numerical computation of the integral would be as expensive as using the simpler scheme with many times more blobs. We chose to use as many blobs as possible, in order to be able to resolve the fine flow details necessary to correctly model high Reynolds number or inviscid fluid problems, hence we have not pursued Leonard's scheme B.

Some typical blob distributions are given by the following. The graphs are for distributions with $\sigma=1$.

3.4.1 Rankine Vortex :

$$f(\underline{r},\sigma) = \begin{cases} \dfrac{1}{\pi\sigma^2} & ; \ |\underline{r}| \le \sigma \\ 0 & ; \ |\underline{r}| > \sigma \end{cases} \qquad (15)$$

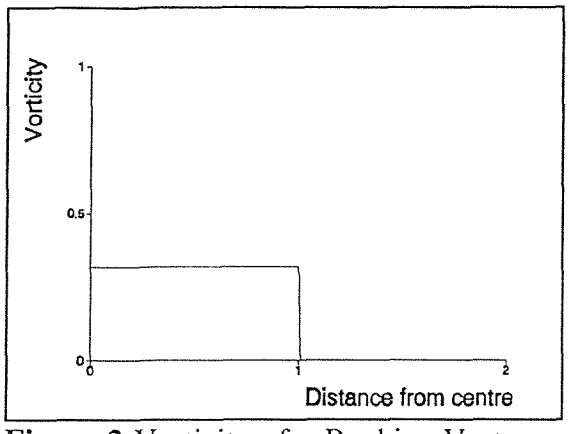


Figure 2 Vorticity of a Rankine Vortex

The vorticity distribution has a constant value inside the core and is zero outside. Physically, the core is rotating as a solid body.

$$F(r,\sigma) = \begin{cases} \dfrac{r^2}{2\pi\sigma^2} & ; \ r \le \sigma \\ \dfrac{1}{2\pi} & ; \ r > \sigma \end{cases} \qquad (16)$$

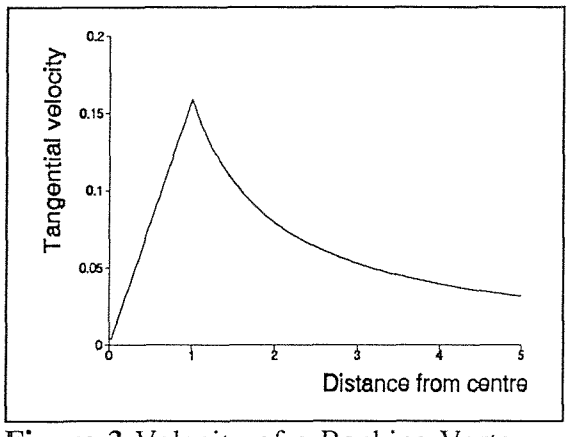


Figure 3 Velocity of a Rankine Vortex

The velocity distribution increases linearly within the core to a maximum value of $1/2\pi\sigma$, then has the same behaviour as a point vortex outside the core.

Rankine vortices are intuitively attractive because they correspond to solidly rotating regions of fluid. However, for discrete vortex methods they have few advantages, being discontinuous in both vorticity and velocity fields. Few papers refer to Rankine vortices other than historically.

3.4.2 Lamb or Gaussian Vortex :

$$f(\underline{r},\sigma) = \frac{1}{\pi\sigma^2}\exp\left(-\frac{|\underline{r}|^2}{\sigma^2}\right) \tag{17}$$

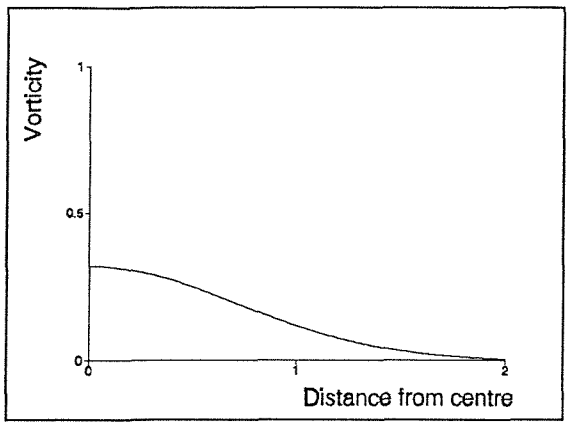


Figure 4 Vorticity of a Gaussian Vortex

The vorticity distribution is smooth everywhere and decays rapidly outside the core, moving away from the centre.

$$F(r,\sigma) = \frac{1}{2\pi}\left[1-\exp\left(-\frac{r^2}{\sigma^2}\right)\right] \tag{18}$$

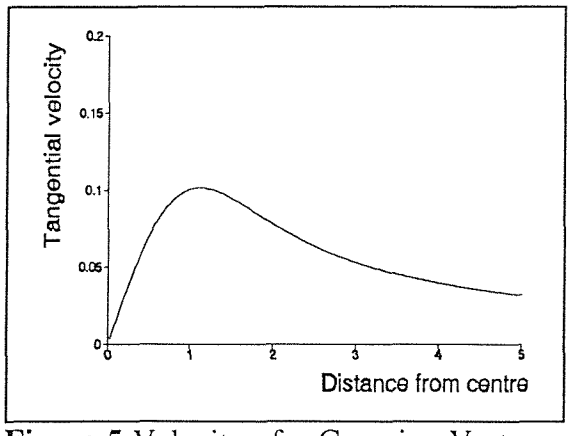


Figure 5 Velocity of a Gaussian Vortex

The velocity distribution is smooth everywhere and has a maximum magnitude near $r=1.12\sigma$ of approximately $0.64/(2\pi\sigma)$.

Gaussian vortices are extremely useful for constructing discrete vortex methods. Many authors use them as the basic computational element; we have found that they lead to good results at reasonable cost. The fact that the vorticity distribution is not 'contained' (in the sense that at an arbitrary distance from the vortex, it is non-zero) is a slight disadvantage as explained in section 3.10.

3.4.3 Chorin Vortex :

$$f(\underline{r},\sigma) = \begin{cases} \dfrac{1}{2\pi\sigma|\underline{r}|} & ; \ |\underline{r}| \le \sigma \\ 0 & ; \ |\underline{r}| > \sigma \end{cases} \tag{19}$$

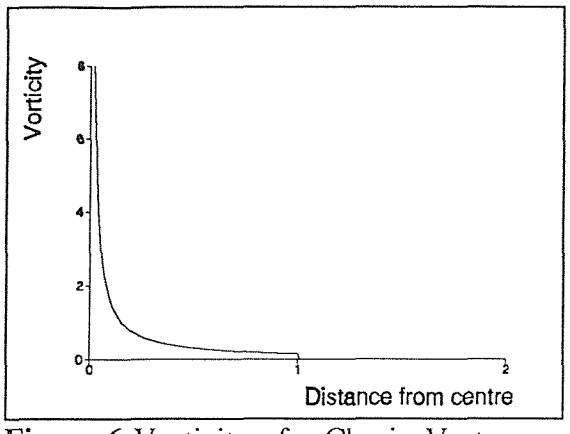


Figure 6 Vorticity of a Chorin Vortex

The vorticity distribution has a singularity at the centre and is zero beyond the core radius.

$$F(r,\sigma) = \begin{cases} \dfrac{r}{2\pi\sigma} & ; \ r \le \sigma \\ \dfrac{1}{2\pi} & ; \ r > \sigma \end{cases} \tag{20}$$

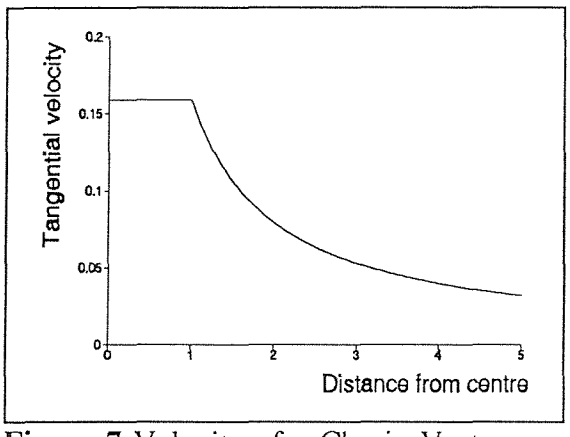


Figure 7 Velocity of a Chorin Vortex

The velocity distribution has a constant radial velocity, of magnitude $1/(2\pi\sigma)$, inside the core and has the same behaviour as a point vortex elsewhere.

Chorin vortices were developed to represent a vortex sheet when placed around a boundary. A velocity jump is experienced in moving across the centre of the vortex — this is similar to the velocity jump across a vortex sheet. This advantage is small compared to the

disadvantage caused by the singularity in vorticity at the centre of the vortex. If the vorticity field is of no interest as such, then reasonable results can be achieved using this distribution.

There are a wide variety of other vortex distributions available in the literature. Some authors conclude that using more complex cores gives better convergence towards the equations. However, the so-called higher order vortex cores are more expensive to use in computations and most authors who advocate their use are only considering errors in the vortex propagation. It is our belief that a significant source of error is the discretisation error incurred by representing the vorticity field by a finite number of discrete vortices. The only way to reduce this error is to use as many vortices as possible. Therefore we have opted to use an inexpensive distribution (Gaussian) so that the limited computing resources available can be employed to maximise the number of blobs.

3.5 Viscosity

In a fluid with no solid boundaries, the introduction of viscosity into the Euler equations has only one effect: the right-hand side of equation (1) introduces a diffusion of vorticity. There are a number of methods for extending the inviscid model to account for the effect of the extra term.

3.5.1 Operator Splitting : Random Walk / Growing Core

The most commonly used technique is the random walk method, where equation (1) is split into two parts

$$\frac{\partial \omega}{\partial t} = -(\underline{u}.\nabla)\,\omega \tag{21}$$

$$\frac{\partial \omega}{\partial t} = \frac{1}{Re}\nabla^2 \omega \tag{22}$$

This is called fractional-step and is an example of operator splitting. Equations (21) and (22) are solved consecutively, rather than simultaneously as they appear in (1). Equation (21) is the Euler equation (cf. equation (11)), the solution of which using vortices has already been discussed, leading to equation (12). Equation (22) is the viscous diffusion equation, which can be modelled by adding a random walk of variance $2\delta t/Re$ (where δt is the time step used for the convective motion of the vortices) to the position of each blob at every time step (see Wax (1954)). The viscous diffusion induced by the random walk is statistically equivalent to that caused by equation (22). The disadvantage is that large numbers of vortices are needed

to converge to the analytic diffusive solution. The result of using smaller numbers of vortices is to introduce excess noise into the solution.

The splitting technique has been shown to converge for an infinite fluid, provided each sub-step converges, by Hald and Del Prete (1978), Hald (1979, 1985, 1986) and Beale and Majda (1981). Exact Navier-Stokes solutions are approached as the number of vortices, N_V, tends to infinity, core size, σ, tends to zero and average vortex spacing tends to zero (more quickly than core size). These conditions can be explained by reasoning that, to approximate an arbitrary vorticity distribution accurately using a sum of vortex blobs, it would be necessary to use many small blobs and they must overlap.

Alternatively, the viscous component in equation (22) can be simulated by allowing the core sizes of the blobs to grow with time. This is commonly applied using Gaussian vortices (equations (17) and (18)) since, with a core size growing at the rate $2(t/Re)^{\frac{1}{2}}$, a single one gives an exact solution of the Navier-Stokes equations. The problem with the growing core size approach is that it demands the presence of a variety of core sizes which, as discussed in section 3.4, adversely affects the convergence of the Biot-Savart law to a solution of the Euler equation. The viscous convergence conditions given above illustrate another advantage of the random walk model over the spreading core approach, this advantage being the ability to control the *maximum* core size and hence ensure the solution converges. Spalart references Greengard (1985) who claims that, whilst the growing core approach solves the viscous equation correctly, the convective terms converge to solve the *wrong* equation.

3.5.2 Diffusion Velocity

A completely different approach is employed by Ogami and Akamatsu (1991), who construct a diffusion component for the velocity of each vortex as follows. The equation for a scalar function $F(x,y,t)$ moving with velocity $\underline{u}(x,y,t)=(u_x(x,y,t),u_y(x,y,t))$ in the (x,y)-plane is easily shown to be

$$\frac{\partial F}{\partial t} + \frac{\partial (u_x F)}{\partial x} + \frac{\partial (u_y F)}{\partial y} = 0. \tag{23}$$

The Navier-Stokes equation (1) may be rewritten in this form as follows

$$\frac{\partial \omega}{\partial t} + \frac{\partial}{\partial x}\left[\left(u_x - \frac{1}{Re\,\omega}\frac{\partial \omega}{\partial x}\right)\omega\right] + \frac{\partial}{\partial y}\left[\left(u_y - \frac{1}{Re\,\omega}\frac{\partial \omega}{\partial y}\right)\omega\right] = 0 \tag{24}$$

where u_x and u_y are the usual convective velocities. Hence the effect of the viscosity is to add a diffusion velocity component, $\underline{v}=(v_x,v_y)$ to the motion of each vortex, with magnitude given by

$$v_x = -\frac{1}{Re\,\omega}\frac{\partial \omega}{\partial x} \tag{25}$$

$$v_y = -\frac{1}{Re\,\omega}\frac{\partial \omega}{\partial y}. \tag{26}$$

So each vortex moves with total velocity $\underline{u}+\underline{v}$. In practice we have calculated the diffusion velocity by dividing by the strength of the vortex under consideration, rather than summing the vorticity at that point. This is simpler to implement and gives the correct derivative when multiplied by the local vortex strength in equation (23). The problem of a possible division by zero (where the vorticity is zero) may be conveniently circumvented by simply deleting vortices of zero strength.

This has all the advantages of the random walk method, it does not involve the introduction of a grid and allows control over the maximum core size. In addition, it does not incur an error due to operator splitting and does not add excess noise to the solution as does the random walk. The solutions will converge as more vortices are used, but it should only be necessary to use enough blobs to resolve the detail of the viscous diffusion process, rather than enough to statistically model a particle diffusion problem.

A significant disadvantage of this technique is that very little diffusion is induced between vortices which become too far separated. This is because the diffusion velocity formulae decay rapidly moving away from the vortex centre. Unfortunately, this means that large scale eddies in the wake behind a body will stop spreading once the component vortices have moved sufficiently far apart. The random walk will continue to spread such eddies indefinitely.

Our findings have confirmed that the vorticity diffusion method gives smoother solutions than the method using the random walk. However, the eddies behind the body *were* observed not to spread to the same extent. This problem was eventually rectified by using a bilinear combination of the two methods. Pure diffusion velocity is used near the body surface where there are many overlapping vortices; this is graded to pure random walk at a distance from the body where the vortices have become more sparse, but where they are far enough from the body that the noise introduced cannot affect the boundary layer to the same extent.

3.6 Boundary Conditions

In the early stages of this project it was decided to concentrate on flow past bluff bodies due to our aerodynamical interests and the wealth of experimental and reliable numerical data available for verification of results (see eg. Collins and Dennis (1973), Ta Phouc Loc and Bouard (1985) and Bouard and Coutanceau (1980)). In such flows we consider a body in an unbounded region of fluid. The Navier-Stokes equations for this situation can be solved given the velocity at the body surface, the flow 'at infinity' and the starting flow everywhere.

In order to study the case of an impulsively started body, the initial condition is zero flow everywhere then uniform flow past the body an instant later (leading to the non-lifting inviscid solution for the instant after motion has started). This is the most common starting situation in mathematical models and there are many experiments where the initial acceleration of the body is extremely rapid.

One of the main advantages of vortex methods is that the boundary condition far from the body is solved exactly. The flow induced by the vortex elements decays to zero in the far field, so the boundary condition at infinity is automatically set to be uniform flow. Also, all the flow elements used are mathematically valid at any distance from the body and, because no grid is used, calculation of the flow far from the body is not especially expensive.

The flow at the body surface is zero due to the inviscid no-penetration condition (zero normal velocity) and the viscous no-slip condition (zero tangential velocity). The no-penetration condition can be satisfied either approximately — using a panel method — or exactly — using a conformal transformation to a circle and an image system inside the circle. The no-slip condition is satisfied by generating new vorticity on the surface of the body at each time-step, in an attempt to model the physical process of vorticity creation in a boundary layer. The quality of the results produced by this method is highly dependent on accurate modelling of the boundary layer around the body. A noisy boundary model can fail to resolve important features, such as separation points, correctly. Hence a considerable effort has been devoted to improving this aspect of the model.

Dealing with the inviscid condition first, a comparison of the two alternatives is required. Conformal transformations are exact (ie. no fluid leaks into the body), but require an image system within the body. Since the operation count for the velocity calculation is greater than linear in N_v (see section 4.5) this more than doubles the problem size. Also, Spalart (1988) maintains that an image system can introduce a large error in the flux of vorticity along the boundary layer, because a vortex and its image can approach closely and convect each other strongly. However, it is not clear whether this argument is limited to point vortices, with

singularities in their velocity distributions, or whether it includes smoother distributions with finite velocity distributions.

Panel methods require one linear matrix decomposition at the start of the calculation, but can be applied at a fixed cost per vortex thereafter. Spalart believes that the use of vortex panels reconstructs the physical processes of a viscous boundary layer, causing vortices to separate correctly from sharp corners. Our results are consistent with this, as we have observed boundary layer separation in regions of very high curvature (see section 5.6). If a panel method is used, a certain amount of leakage into/from the body will occur and should be assessed as a source of error (see section 5.1).

On the grounds of computational cost and the relative simplicity of writing the code to deal with arbitrary body shapes, it was decided to use panel methods rather than conformal transformations. A new panel method has been developed which is suitable for use with vortex methods. We have concentrated on vortex panel methods, as these extend more naturally than source panel methods to the full discrete vortex model. Hence results are presented for vortex panels only, but the mathematics involved is easily modified to construct a source panel method.

As a general description of a vortex panel method, suppose we have a vortex element (eg. blob, sheet) of unit strength with velocity distribution $\underline{v}(\underline{r},P)$ (P is a set of parameters defining the position and orientation of the element). Choose a set of M positions P_j for elements, just outside the boundary. As a mathematical tool, the elements could equally well be just inside the boundary, but later the panels are used as a source of vorticity in the flow; placing the elements outside gives the correct sign for the generated vorticity. Next choose M control points $\underline{x}_i$ on the body contour, with corresponding unit tangents $\underline{s}_i$. Suppose there is an external flow $\underline{U}_e(\underline{r})$. Form a set of linear equations to determine element strengths λ_j which will exactly satisfy the boundary condition at the control points. For example, to satisfy the tangential condition,

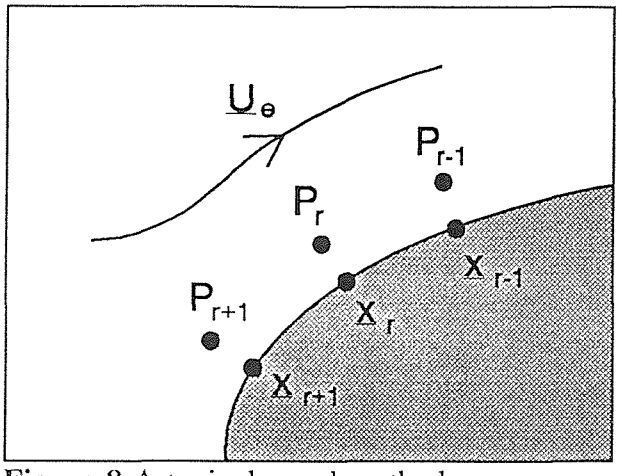


Figure 8 A typical panel method

$$\sum_{j=1}^{M} \lambda_j \underline{v}(\underline{x}_i, P_j).\underline{s}_i = -\underline{U}_e(\underline{x}_i).\underline{s}_i \qquad \text{for} \quad i=1,\ldots M \tag{27}$$

Solve these equations for λ_j and form a new velocity, which models flow around the body

$$\underline{w}(\underline{r}) = \underline{U}_e(\underline{r}) + \sum_{j=1}^{M} \lambda_j \underline{v}(\underline{r}, P_j) \tag{28}$$

Due to the nature of the Laplace equation, which is being solved here (considering the problem in terms of streamfunction), satisfying the tangential no-slip condition has the beneficial side-effect of satisfying the zero normal velocity condition, provided that the vorticity inside the body is zero (See Spalart (1988) pp. 29). Note that the physical interpretation of this method is one of modelling a thin boundary layer on the body surface by a vortex sheet, with the velocity jump in passing over this sheet corresponding to going from the stationary fluid at the surface to the moving fluid in the exterior flow.

The most basic element used (see eg. Anderson (1984), pp. 160) is a constant vorticity along a line from η_0 to η_1 in the complex plane, and this has velocity distribution

$$u^*(z) = \frac{-i}{2\pi(\eta_1 - \eta_0)} \log\left(\frac{z-\eta_1}{z-\eta_0}\right) \tag{29}$$

where the notation is in the complex plane, and * denotes the complex conjugate. This element has singularities at η_0 and η_1. In the vortex method, a vortex can approach close to one of these points leading to large velocities and consequently inaccurate solutions.

Chorin avoids this problem by using a smoothed point source (with a source panel method), with distribution

$$u^*(z) = \begin{cases} \dfrac{1}{2\pi(z-\eta_0)} & ; \ |z-\eta_0| > h/2 \\ \dfrac{n^*}{2h} & ; \ |z-\eta_0| \le h/2 \end{cases} \tag{30}$$

where h is the diameter of the smoothed core and n is the (complex) unit normal to the body contour near the element. This has no singularities but has discontinuities, which are not desirable (functions with discontinuities have delta functions in the first derivative).

We have developed a new element with better mathematical properties. The element is a linear distribution of vorticity on a quadratic curve (the element has zero vorticity off the curve) and has velocity distribution

$$u^*(z) = \frac{-i}{2\pi\zeta}\left[(s_0-1)\log\left(\frac{s_0-1}{s_0}\right) + (s_1-1)\log\left(\frac{s_1-1}{s_1}\right)\right]$$

$$\text{where}\begin{cases}\zeta = \left(\beta_0^2 - 4(\eta_1-\beta_0-\eta_0)(\eta_0-z)\right)^{\frac{1}{2}} \\ s_0 = \dfrac{-\beta_0+\zeta}{2(\eta_1-\beta_0-\eta_0)} \\ s_1 = \dfrac{-\beta_0-\zeta}{2(\eta_1-\beta_0-\eta_0)}\end{cases} \qquad (31)$$

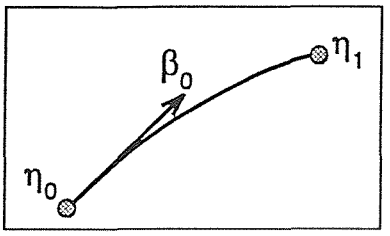


Figure 9 A new element

This element runs from η_0 to η_1 and has first (complex) derivative β_0 at η_0. An example of such a curve is shown in Figure 9, the equation of the curve and the vorticity distribution is given below in equation (32). The vorticity distribution is linear, from zero strength at η_1 to unit strength at η_0.

$$\begin{aligned}&\text{position, } \eta(t) = (\eta_1-\eta_0-\beta_0)t^2+\beta_0 t+\eta_0 \\ &\text{vorticity, } \omega(t) = 1-t \qquad ;0\le t<1\end{aligned} \qquad (32)$$

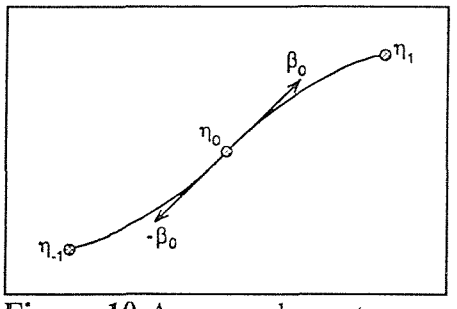


Figure 10 A super-element

This distribution has a singularity at η_0, but this can be removed by forming a super-element from two elements placed 'back-to-back'. Place one at position $P=(\eta_0,-\beta_0,\eta_{-1})$ and the other at $P=(\eta_0,\beta_0,\eta_1)$ to make the super-element run from η_{-1} to η_1 through η_0, with gradient β_0 there (see Figure 10). These super-elements have no singularities, since the limiting velocities of the two sub-elements, approaching η_0, are equal and opposite. In our model, we place each element through three consecutive position points around the body, so that neighbouring elements overlap. Thus each super-element curve matches the first derivative of the body contour at its middle point, where the vorticity distribution is at a maximum. This placement scheme creates a singularity-free vorticity distribution on the body surface, second order in space and first order (ie. linear) in vorticity.

In the form given above, the velocity distribution of an element has been normalised, in the sense that,

$$\text{as} \quad |z-\eta_0| \to \infty, \quad u^*(z) \to \frac{-i}{2\pi(z-\eta_0)}, \tag{33}$$

the same functional form as for a point vortex at η_0. Thus, in the form given, the total circulation of an element is one (other terms in the expansion will not contribute to a circulation integral). This is an equivalent normalisation process to integrating for the total strength as in equation (6), but examining the asymptotic form is much simpler for an element with such a complicated velocity distribution.

As mentioned above, new vorticity is introduced to the flow via the no-slip boundary condition. There are several ways of doing this. Initially we followed Chorin's method based on the fact that the velocity jump over a vortex sheet is given by the strength of the sheet. Hence, if the fluid is stationary at the body surface, the strength of the vortex sheet must be determined by the tangential velocity just outside. Hence the strength of the surface sheet at any position may be calculated, allowing the sheet to be partitioned into discrete vortices. This works in a sense, but takes no account of the self-interaction of the sheet as it develops, and so there is no point in the model at which both normal and tangential boundary conditions are satisfied. To clarify this, the no-slip condition is only solved in a 'local' fashion at each control point, and the effect of a created vortex is not taken into account other than at its own control point.

Another approach is to use a vortex panel method to satisfy the boundary conditions and then at the next time step propagate the sheets as blobs with the same total circulation. This at least gives one state where, since both boundary conditions are approximately satisfied, forces can be measured. Certain details of the creation mechanism are ad-hoc, and are given in section 3.10.

A significant problem with the discrete vortex method is that vortices with large circulations can be formed, which can then induce large velocities on close neighbouring vortices leading to large time-stepping errors. We have used dynamic vortex creation which helps to reduce this problem. The basic idea is simple — impose a maximum circulation and do not allow any vortices to be formed with strengths which exceed this upper bound. If more circulation than the maximum must be injected into the flow at a boundary point, do so using multiple vortices whose total circulation matches the required value but who, individually, do not exceed the maximum. This has the beneficial side-effect of injecting many more vortices in regions of high vorticity generation, which helps to resolve the fine details. It was found the results were significantly improved by the addition of this technique into the model.

An example of dynamic vortex creation is illustrated in Figure 11. The multiple vortices generated from a given boundary point are arranged evenly along the line segments between the associated vortex creation point and its two neighbours. Vortices from a point are placed up to half way along these two line segments. This scheme causes the vortices to overlap, which is necessary for rapid convergence (see section 3.5). Also, the transition between panels and blobs is smoothed by using several blobs arranged in a straight line, as opposed to one strong blob at the creation point. For details see section 3.10.

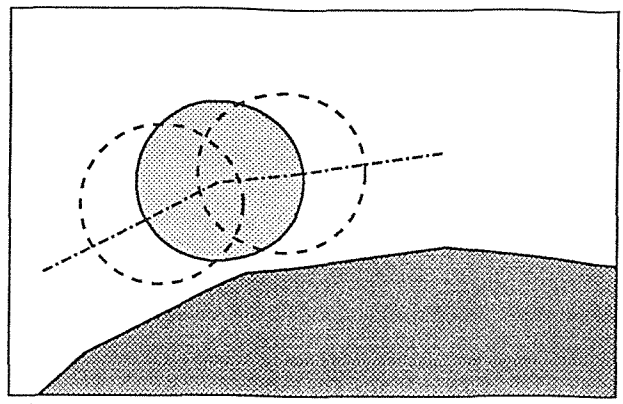


Figure 11 Dynamic vortex creation

In a two-dimensional flow, the total circulation around a contour far from the body remains constant, the same condition as Kelvin's Circulation theorem. The condition leads to the following relationship between the panel and vortex strengths (for a non-rotating body),

$$\sum_{j=1}^{M} \lambda_j = -\sum_{j=1}^{N_V} \Gamma_j \tag{34}$$

which can be used as an extra equation in the panel method. Spalart gives a proof of this condition based on keeping the pressure (which can be evaluated by integration of the rate of creation of new vorticity around the body, see section 3.11) single-valued. Many authors advocate dropping one control point equation and using the circulation condition as a replacement. The problem here is that it can cause large leakage at the control point which has been dropped. We have achieved good results by using equation (34) to remove one unknown from equations (27), giving us M equations for $M-1$ unknowns. We then use a least squares solution to minimise the errors from the remaining equations. Thus each equation is treated in the same manner, leading to a 'symmetrical' solution at the control points and satisfying (34) precisely. A general formulation of a least squares solution is as follows. Suppose the errors are to be minimised in the over-specified system

$$\sum_{j=1}^{R} A_{ij}\lambda_j = v_j \qquad \text{for } i=1,\ldots S \tag{35}$$

where S>R (more equations than unknowns). Define a new matrix B by

$$B_{ij} = (A^T A)_{ij} = \sum_{k=1}^{S} A_{ki}A_{kj} \tag{36}$$

Then the errors are minimised by solving

$$\sum_{j=1}^{R} B_{ij}\lambda_j = (A^T v)_i = \sum_{j=1}^{S} A_{ji} v_j \qquad \text{for } i=1,\ldots R \tag{37}$$

which is simply R equations for R unknowns.

In our panel method formulation it would then be necessary to use equation (34) to calculate the strength of the removed unknown (λ_{R+1} in the notation used above).

3.7 Modelling the Surface of the Body

Since the boundary is only 'virtual', it is possible that vortices cross into the interior of the body. This is due to the movement across the boundary of vortices near the body surface (essentially caused by time stepping errors) or boundary conditions not being satisfied exactly (fluid 'leaking' across the boundary). There is some debate over whether these vortices should be reflected back into the exterior flow (there are several different ways of doing this) or removed from the model completely.

In fact, while it is accepted that vorticity can only be created at solid boundaries in a homogeneous fluid, there is considerable debate over the fundamental mechanism of vorticity destruction. Batchelor (1967) pp. 266, writes "vorticity cannot be created or destroyed in the interior of a homogeneous fluid under normal circumstances, and is produced only at boundaries" implying that vorticity is lost by diffusion to the boundaries. Morton (1984) is felt (by Sarpkaya) to have clarified the situation, he claims "vorticity once generated cannot subsequently be lost by diffusion to the boundaries", "walls play no direct part in the decay or loss of vorticity", "vorticity decay results from cross-diffusion of two fluxes of opposite sense and takes place in the fluid interior", "vorticity generation results from tangential acceleration of a boundary, from tangential initiation of a boundary and from tangential pressure gradients along a boundary" and "reversal of the sense of acceleration or of the sense of the pressure gradient results in reversal of the sense of vorticity generated".

Given this confusion, a pragmatic approach was adopted; both deletion and reflection were tested and the results compared. Deletion is easy to implement, simply remove vortices from the calculation if they enter the body. Reflection is less trivial — a discussion of how this can be implemented is given below. An algorithm for detection of a vortex crossing the boundary is given in appendix B; it is suitable for implementation of either reflection or deletion.

The precise details of the reflection mechanism are not obvious — most authors do not include any details and some merely state that they use 'elastic reflection'. The obvious reflection will here be called 'mirror reflection' and has the same properties as a reflected beam of light in a mirror. This technique is shown at point A in Figure 12: the velocity component normal to the body surface is reversed after contact with the surface. In terms of the physics of the fluid, this seems a strange choice. Consider an example where two vortices have the same velocity, directly towards the wall. Using mirror reflection the vortex initially closer to the wall is further away at the end of the time-step, implying that fluid particles can exchange positions — a rather unusual phenomenon. Improved results were obtained from either zeroing the normal component of velocity (as shown at point C in Figure 12) upon contact with the wall, or stopping the vortex completely (as shown at point B in Figure 12) at the wall. In terms of results, there is little difference between these two techniques, but the stopping technique is slightly easier to implement (using the zero normal method, a vortex may need multiple reflections at a concave surface, as shown in Figure 12). Hence, we have used the scheme illustrated at point B where reflection is required in our calculations.

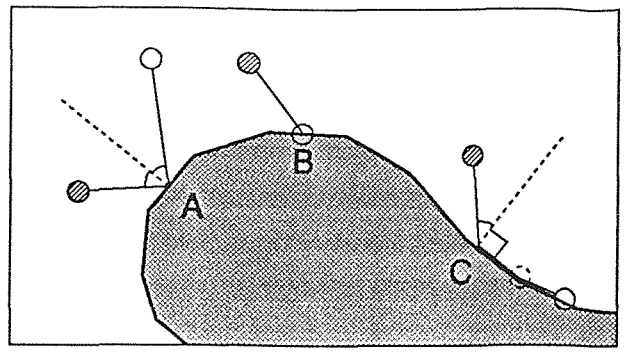


Figure 12 Reflection methods

An important point is that for consistency reflection, and *not* deletion, should be used for the sub-step in a high order time stepping scheme (see p. 61 for a discussion of time stepping). Therefore any vortex which enters the body in a predictor step is reflected, ie. with deletion, only vortices located inside the body at the *end* of a time step are deleted, not those taken across the boundary by a predictor step.

After extensive testing and comparison, it became apparent that deletion of vortices crossing the boundary was giving better results (both in terms of flow visualisation and body force coefficients). After further investigation it appeared that this was mainly due to the fact that many of the vortices diffuse inside the body just after they have been created. This means that, when employing the method of deletion, much of the vorticity is lost immediately causing a much weaker rolling-up effect of the wake behind the body. This weaker roll-up seemed to be necessary: using reflection caused the initial wake eddies to be too small and concentrated, as compared to experimental visualisations. Also, the reflection technique caused excessive oscillation in the early time values of force coefficients. From observations of the vortices plotted on a graphics monitor at these early times, there seemed to be an alternating oscillatory 'overshoot' in the amount of vorticity shed from the body surface. The deletion

mechanism consumes a significant number of vortices, making runs that use this surface model less expensive.

In conclusion, the results presented here use deletion of vortices which are inside the body at the end of a time step. Reflection scheme B (see Figure 12) is applied to any vortices entering the body in a predictor sub-step. It is worth noting that, due to the enforcement of equation (34), any net circulation removed by deletion of vortices will be replaced on the next time step.

3.8 Vortex Core Size

The radius of a vortex core (σ in Figure 13) is set to satisfy a number of heuristic conditions. It is important to ensure that the convergence conditions (see section 3.5) are satisfied in the limit of increasing numbers of vortices. In practice this means that as the number of vortices created per time step increases, the core size should tend to zero and some tendency should be given towards overlapping the cores. Within these constraints the choice of the actual size is arbitrary. We use a core radius of a quarter of the average distance between boundary points and rely upon the multiple vortices created using the dynamic vortex creation to satisfy the overlap criterion (see section 3.9). A suitable value was found through experimenting with different core sizes. For a given number of boundary points, solutions are reasonably insensitive to the choice of core size, although using much larger core sizes fails to capture the finer flow details and tends to make the layer of vortices near the boundary too fat; using much smaller core sizes leaves too many gaps in the vorticity distribution near the boundary and fails to overlap the cores sufficiently to converge the method. Note that as the number of boundary points increases, and hence as the number of vortices introduced into the flow increases, the condition that the core size must tend to zero is automatically satisfied.

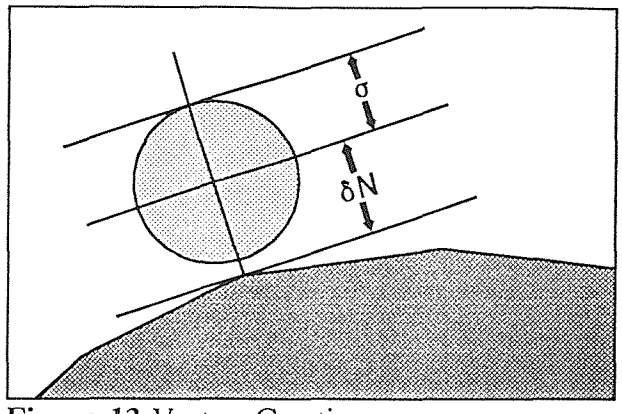


Figure 13 Vortex Creation

An error arises in the transition from boundary panel elements to vortex blobs, because the vorticity distributions can never be matched exactly. This error is related to the choice of core size and could lead to numerical instability when using a small time step (see section 5.3).

3.9 Initial Vortex Positions

In the usual, non-dynamic, creation scheme the vortices are created slightly off the boundary — one at each point of normal distance δN from a boundary point, as shown in Figure 13. The distance δN is chosen to balance the following criteria. Each vortex must be placed so that the effect of moving it away from the boundary along the normal is that its influence on the boundary *decreases*. For instance, a Gaussian vortex must be placed at a minimum distance of 1.12σ from the boundary, as the velocity distribution decreases on either side of this point (see Figure 5). If this condition is not met, the method can become unstable using a small time step. The observed result of this instability is that the effect of the newly created vortices on the boundary increases immediately. At a subsequent time step the effect of the original vortices has increased and, by now, more vorticity has been created. Hence the system creates vorticity of the opposite sign to compensate and in this way oscillations can be established. The second factor is that the no-penetration condition is only satisfied as a result of solving a Laplace equation inside the body (see section 3.6). When using vortices with unbounded cores, such as Gaussian vortices, there is some residual vorticity inside the body leading to a weak Poisson equation. Spalart (1988) states that the smoothness of solutions to the Laplace equation ensures that this error will not cause instability and this is in our results confirm this. However, we have found that if vortices are created too close to the boundary the amount of fluid leaking across the body contour increases significantly, possibly due to the existence of interior vorticity. The third factor is the desire to keep the minimum representable boundary layer as thin as possible in order to accurately represent the experimentally observed physics at high Reynolds numbers. The value $\delta N=1.12\sigma$ has given good results with Gaussian vortices, ie. with the maximum velocity on the boundary.

The other aspect of vortex positioning which is not prescribed by the model is the positioning of the 'extra' vortices created by the dynamic vortex creation scheme. We have tried two different schemes and found results to be insensitive to the implementation details. With a random walk it is adequate to initially place all extra vortices at the creation point described above and allow the random viscous motion to take each vortex to a different position. Using the diffusion velocity the extra vortices must be placed at different initial positions or they will remain coincident for all time. We chose to use the simple scheme of distributing the extra vortices at equally spaced points along the straight line segments joining the normal creation points. Then the maximum allowable circulation of an individual vortex is set to be proportional to the core size. If the strength of a boundary panel exceeds this maximum, it is split into several vortices of equal strengths, less than the maximum. Thus increasing the number of boundary points has the effect of confining an increasing number of 'multiply created' vortices to a smaller physical space. This should help to satisfy the overlap condition required for convergence of the convective equation (see section 3.5).

3.10 'Ad-Hoc' Modelling

There are a wide variety of less rigorous adjustments which can be made to the model. Care must be taken to implement these judiciously and to test them properly. The techniques are mainly intuitive and often involve retrofitted parameters. However, they are reported to improve results and warrant investigation. To broadly categorise these techniques, they all fall under the following titles :-

3.10.1 Circulation Reduction

In this addition to the model, the strength of each vortex decays with time. This improves drag and lift figures, see Sarpkaya and Shoaff (1979), and is thought to work because it is a crude model of the three-dimensional vortex stretching which occurs in real fluids. The rate of decay is purely arbitrary and is set to improve drag and lift results. See Sarpkaya and Shoaff (1979). The results presented in this paper do not use circulation reduction — we wanted to apply the model with as few arbitrary choices as possible.

3.10.2 Vortex Amalgamation

When using a vortex amalgamation scheme, if two vortices of opposite sign approach closely they are removed and replaced with one 'averaged' vortex (see Ham (1968)), modelling the process of opposite signs of vorticity cross-diffusing and decaying, which is observed in real fluids. The problem is that there are many different ways of performing the averaging. Sarpkaya (1989) claims that it is important to average only far from the body, to ensure that a previously amalgamated pair does not recirculate close to the boundary, where the high velocity gradients could draw a close pair of vortices apart. This technique was not tested as it would have been difficult to implement on top of the parallel algorithm used here (see section 4.6). It would be important to implement amalgamation so that vortices are amalgamated depending only on their physical position, not on which processor they reside.

3.10.3 Turbulence Modelling

Above Reynolds numbers of O(5000) the wake behind a circular cylinder becomes turbulent, although the boundary layer remains laminar until $O(10^5)$, see Schlichting (1960). Various attempts have been made to add vorticity diffusion via the random walk, along the lines of an eddy-viscosity model. Since turbulence is fundamentally three-dimensional in nature, this approach could only provide a very crude correction factor. We have not used turbulence modelling in order to minimise the number of arbitrary factors in the model.

3.11 Body Force Evaluation

Body forces are extremely important results for practical use of the method. Unfortunately they are rather difficult to calculate as there is no explicit pressure variable. There are several techniques suggested in the literature.

Chorin (1973) suggested the use of numerical integration of vorticity near the body contour to calculate the pressure on the boundary and the skin friction. He introduced a special grid just off the boundary and calculated properties on the grid to use in the integration. Our attempts to compute forces in this manner produced poor results.

Sarpkaya (1963) gave the following formula for the body force on a cylinder in a uniform flow with discrete vortices,

$$D + iL = -i\rho \frac{\partial}{\partial t} \sum_{j=1}^{N_V} \Gamma_j [\underline{r}_j(t) - \underline{r}^I_j(t)] \tag{38}$$

where $\mathbf{r}^I_j$ is the position of the 'image vortex' associated with the j^{th} vortex. The formula is a modified form of Blasius' theorem. The drawback with this is that, although it is easy to apply for geometrically simple shapes, in general the position of an image vortex is not easy to determine.

Quartapelle and Napolitano (1983) give a general formula for the pressure force acting on a rigid body in an incompressible flow. This simplifies slightly for high Reynolds numbers (see Smith and Stansby (1988)) to

$$f_y = \int_V (\underline{u} \times \underline{\omega}).\nabla \eta_y \, dV \tag{39}$$

where $\nabla^2 \eta_y = 0$

with $\underline{n}.\nabla \eta_y|_{S_1} = -\underline{n}.\hat{\underline{y}}$ and $\underline{n}.\nabla \eta_y|_{S_2} = 0$

V is the volume containing the fluid, S_1 is the surface of the body, S_2 is an arbitrary surface enclosing the body and delimiting the considered flow domain ($S_1 \cup S_2$ encloses V), and $\hat{\mathbf{y}}$ is the unit vector in the direction of the force component f_y we are calculating. This formula is very useful but suffers from a lack of generality in that the function η_y can only be calculated analytically for fairly simple geometries. For example, the case of a circular cylinder leads to the following formulae for drag and lift,

$$C_D = \sum_{j=1}^{N_V} \frac{\Gamma_j a^2 \left[(x_j^2 - y_j^2) v_j - 2 x_j y_j u_j \right]}{r_j^4} \tag{40}$$

$$C_L = \sum_{j=1}^{N_V} \frac{\Gamma_j a^2 \left[(x_j^2 - y_j^2) u_j + 2 x_j y_j v_j \right]}{r_j^4} \tag{41}$$

where, as before, Γ_j, (x_j,y_j) and (u_j,v_j) are the vortex strengths, positions and velocities, $r_j^2=x_j^2+y_j^2$, and a is the cylinder radius. This formula has given better results for the circular cylinder than those which generalise to arbitrary shapes, as found by Smith and Stansby (1988).

Smith and Stansby (1988) and Spalart (1988) give an alternative formula based on the rate of creation of vorticity at the body surface, which is applicable to arbitrary body shapes. To derive the equation, first write equation (1) in the form,

$$\frac{\partial \omega}{\partial t} = -\nabla . \left(\underline{u}\,\omega - \frac{1}{Re} \nabla \omega \right) \tag{42}$$

which is a flux equation for the vorticity, the term in the large bracket on the right-hand side being the flux of vorticity at a point. At the wall $\underline{u}=\underline{0}$, so the flux of vorticity normal to the wall (equivalently the rate at which the wall is emitting or 'creating vorticity') is

$$-\frac{1}{Re} \frac{\partial \omega}{\partial n} \equiv \frac{\partial^2 \Gamma}{\partial s \partial t} \tag{43}$$

Starting from the velocity Navier-Stokes equations (see Batchelor) it is simple to prove for a two-dimensional fluid that, on the body boundary,

$$\frac{\partial p}{\partial s} = -\frac{1}{Re} \frac{\partial \omega}{\partial n} \tag{44}$$

Hence the pressure force on the body may be calculated by integration of the rate of creation of new vorticity at the boundary. Using the notation for panel strengths from section 3.6, the formula

$$p_{j+1}(t) - p_j(t) = \frac{\lambda_j(t) + \lambda_{j+1}(t)}{2\delta t} \tag{45}$$

gives the pressure on the body at the j^{th} control point. It was discovered that slightly less noise was produced by using this formula within the sub-step phase of the second order time

stepping scheme (possibly because no vortices can be deleted in this sub-step). See p. 61 for a discussion of time stepping schemes.

Notice that by integration of equation (43) around the whole boundary and use of the fact that the pressure must be single valued, we arrive at equation (34). Strictly this should be zero circulation at each step, but if we allow deletion of vortices as part of the model it is important to ensure that the lost vorticity is regenerated in the next panel calculation.

The skin friction force on the boundary is most easily derived from first-principles. The fluid speed just off the surface, u_s is available as a result of the panel method. The speed at the surface, u_0 is usually zero, but in any case is known (for example, the case of the rotating cylinder, see section 5.5). The shear stress at a point on the surface is given by,

$$F_s = \mu \frac{du}{dn} \equiv \frac{1}{Re} \frac{u_s - u_0}{l_t} \qquad \text{(assuming non-dimensionality of quantities)} \qquad (46)$$

where l_t is a measure of the thickness of the panel layer. In the model used in our code, the actual panel layer has zero thickness, but is continually being transformed to discrete vortices just off the surface. Hence the thickness of the vortex layer is used for l_t.

It is worth noting that the skin friction force only contributes a significant proportion of the total force at the lower values of Reynolds number for which the method is valid. As Reynolds number increases beyond approximately 5000 this force becomes negligible.

In the computations which follow, we have used equation (45) to calculate the pressure force, since it is the only technique we have found which is applicable to arbitrary body shapes and which produces meaningful results. Equations (40) and (41) are used as a trusted reference formula for the pressure forces in the test case of flow past a circular cylinder. The skin friction is calculated using equation (46).

4. Computing

4.1 Introduction to Parallel Processing

4.1.1 What is Parallel Processing ?

The essential idea behind parallel processing is simple - many processors working simultaneously on a task will complete the task more quickly than one processor could. The job must first be broken down into pieces which may be completed semi-independently. Applied to a computer program, this procedure is known as parallelisation.

This is a similar approach to that used in a working environment to divide a single job amongst several employees. Take, for example, the task of several people writing a newspaper article. The article would be divided up into sections which could be written simultaneously. The writers must communicate with each other as they write to avoid duplication and to give the article overall continuity. The flow of work and communication involved is illustrated in Figure 14. Notice that the whole article cannot be printed until the writer who takes longest has finished. This illustrates the very important concept of 'load balancing' a parallel program, to ensure that each worker has similar amounts of work to do. It is also of interest to consider the extreme case where the article is to be two thousand words long and an attempt is made to divide the job between two thousand people. Each of the writers would need to speak to so many others, to ensure that their particular word made sense in the article, that (fairly obviously) the job could be done more quickly using less workers. In parallel processing parlance, this process of breaking the problem into useful sized chunks is called achieving the correct 'granularity' for the problem.

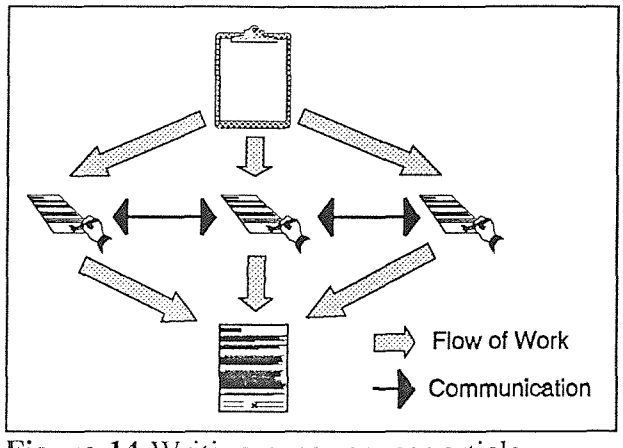


Figure 14 Writing a newspaper article

4.1.2 Parallel Computer Architectures

A traditional computer executes a single sequence of instructions which operate on data stored in its memory. This is known as a Single Instruction Single Data (SISD) machine.

The first generation of parallel computers were nearly all based on Single Instruction Multiple Data (SIMD) architectures. In these machines several processing units execute the same sequence of instructions on different pieces of data. This can be applied extremely efficiently to very repetitive algorithms, such as matrix calculations, but there remains a large set of problems which are not suited to SIMD parallelism. Examples of SIMD machines are the Distributed Array Processor (DAP) and the vector processors within some supercomputers.

Although progress continues with SIMD technology, recent developments in parallel computing have followed a trend towards Multiple Instruction Multiple Data (MIMD) architectures. The current machines have multiple independent processors which have some means of communicating with each other. The communication is performed either by a shared global memory or by special communications hardware which is used to send messages from one processor to another. Most algorithms can be reformulated to execute as several sequential threads which communicate as necessary to complete a problem. Several commercial MIMD machines are based on the Transputer, which was designed specifically as a building block for such architectures and has special in-built communication links as standard. Other machines are available based around off-the-shelf microprocessors (eg. 80386, i860) with additional hardware for communications.

4.1.3 Why Use Parallel Computers ?

The principle reason for using parallel computers is economy. Doubling the performance of a SISD machine means moving to a faster technology. This invariably requires an increase of the clock speed, leading to expensive cooling systems and critical timing designs. These are very expensive additions to make, so twice the performance tends to cost more than twice the price, at any particular time. For a MIMD machine, all that is necessary to achieve a twofold increase in potential performance is to use twice as many processors. The individual processors which have the best price/performance ratio can be used to build the machine, giving a computer which is cheap, powerful and easy to expand.

This argument may seem to contradict the current trend of high performance workstations, which are doubling in performance over an ever decreasing time period. However, the same argument applies — performance and price may in theory still be increased linearly by a linear increase in the number of processors. Workstations are starting to become available containing more than one processor and clusters can be used as a single parallel computer.

4.2 Introduction to Transputers

The range of Transputers form a family of microprocessors designed and manufactured by Inmos as building blocks for MIMD parallel processing systems. The first floating-point transputer (the T800) has four serial communications links which allow a number of processors to be connected together to form one parallel machine. The T800 also has 4K of fast internal memory, and has performance of 10 MIPS[2] / 1.5 MFLOPS[3]. For technical details of the transputer family see 'The Transputer Reference Manual' (Prentice Hall).

The latest transputer (the T9000) promises a tenfold increase in performance and communication bandwidth. Inmos claim a peak performance of 200 MIPS / 25 MFLOPS. In addition a separate routing processor (the C104) will automatically route messages across a network of T9000s, whereas routing on T800s must be done in software. This simplifies parallel codes and allows more flexible mapping of software processes onto hardware processors. The T9000 is currently under development and is due for release some time in 1992. It is instruction set compatible with the T800 so all the software developed under this project should easily port to architectures based on the T9000.

4.3 The Occam Programming Language

Occam was developed by Inmos and the Programming Research Group at Oxford University. It implements a subset of the Communicating Sequential Processes (CSP) language defined by Hoare (1978) and provides a simple language for parallel processing solutions. For more information on Occam see 'Occam2 Reference Manual' (Prentice Hall) and Pountain and May. The main features unique to Occam are :-

Indentation. The block structure of Occam is defined by the indentation of the source text. A number of lines of code are signified as relating to a previous construct by indenting the lines two spaces relative to that construct, eg.

```
IF
  a=1
    a:=0
  a=0
    a:=1
```

[2] Millions of Instructions Per Second

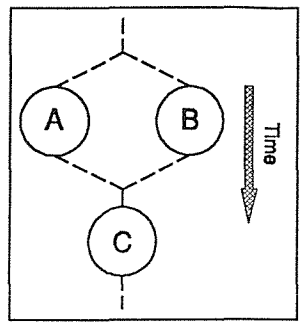
[3] Millions of FLoating point Operations Per Second

```
TRUE
  SKIP
```

has the effect of exchanging the values `0` and `1` in `a`.

Note that most languages require explicit block delimiters (eg. `{}` in C) with indentation used as an aid to readability. Occam enforces indentation in a consistent manner, removing the need for any other form of block delimiter.

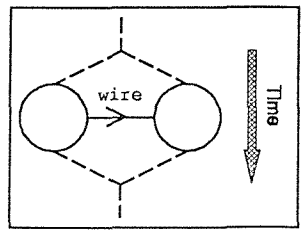
PAR. This construct states that the sub-processes it relates to are to be run concurrently. A PAR completes when each of its sub-processes has terminated. Unlike most languages, sequential execution must be explicitly stated using the **SEQ** construct eg.




```
SEQ
  PAR
    A()
    B()
  C()
```

runs the procedures `A()` and `B()` in parallel then, when both have terminated, runs `C()`.

Channels. Channels are a new data type which can be thought of as a wire between two concurrent processes. Each channel has an input end and an output end. Data may be sent from one process to another via the channel. eg.




```
CHAN OF INT wire:          -- declare wire
PAR
  wire ! 1                 -- sending process
  INT x:                   -- variable used by receiving process
  wire ? x                 -- receiving process
```

`wire` is defined as a channel which may transmit integers. One parallel process sends the value `1` down the channel `wire`. The other process receives the value and stores it in the variable `x`.

A communication involves two processes - one at either end of a channel. The first participant to reach an input or output construct will wait there until the second participant reaches the corresponding construct at the other end of the channel. At this point the communication takes place then both processes are free to continue executing code. Code must be designed so that all communications have a matching opposite end which will execute, otherwise 'deadlock' occurs. In parallel processing parlance, communication in

Occam is termed 'blocking' and 'synchronous'. Blocking means that the program flow stops at a communication construct until the communication has finished. Synchronous means that a communication finishes when the entire message has been transferred from its source to its destination.

ALT. This construct allows a process to wait upon a number of channels, responding to the one on which data is first received, eg.

```
INT x:
ALT
  a ? x
    c ! x
  b ? x
    c ! x
```

If data is sent on channel `a` first, it is stored in `x` then sent down channel `c`. However if data is sent on channel `b` first, it is stored in `x` then sent down channel `c`. This is a simple example of a 'multiplexer' process, where the data from several input channels is interleaved onto one output channel.

The Transputer hardware was designed to implement Occam efficiently and, at the time of writing, the Occam compiler still creates the fastest code. At the start of this work Occam was the only high level language compiler available; only recently have Fortran and C compilers become equipped with adequate debugging tools. If the project were to be extended significantly, translating the code to C would increase portability to parallel machines other than those based on the Transputer and would also make certain parts neater (where the natural implementation is to use data structures - Occam2 does not have these).

The specification of Occam is such that the compiler can perform extensive checking on the user's code. This greatly reduces the chance of programming errors slipping through the compilation/debugging phase of development. Logical errors are, of course, still a major problem.

One of the main drawbacks with using Occam for this type of project is the lack of existing code. Initial progress was slowed considerably by having to implement and validate 'standard' numerical routines. In addition, all the visualisation and hard output routines have been developed from scratch.

4.4 Programming Methodology

Initially, a test version of the code was written to evaluate the vortex method and to learn the Occam language. This version became somewhat unmanageable due to its 'organic' growth into large and complicated routines containing multiple copies of some sections of code.

The experience gained in the development of the test code guided the programming methodology used for the second version. The second version is functionally more complicated and the source code is over five times greater in size (the current version has over 10,000 lines of source in total) and yet is more manageable and proved easier to debug. This was achieved quite simply, by not allowing any piece of source code to grow to a size that made it difficult to comprehend as a whole. In addition any code which had to be used in more than one place was made into a procedure and called from the various places, rather than copying the section with an editor. This achieved a 'single path' for as much of the functionality as possible, making modifications to the code simpler as only one place needed to be altered.

4.5 Zonal Decomposition/Summation Algorithm

4.5.1 Cheaper Algorithms

A naive calculation of the velocity of the vortices in order to time step their positions has an operation count of $O(N_V^2)$ (because for each vortex a sum must be performed over all the others, see equation (14)). Since new vortices are generated on the boundary at each time step, even small runs become expensive quickly. One possibility for reducing the cost is to use the vortex-in-cell method, which introduces a grid. The velocity is calculated by interpolating the vorticity onto the grid, solving equation (2) for the streamfunction, differentiating to give the velocity on the grid and then interpolating it at the vortex positions. The operation count for this method is $O(N_V)$ in vortices, and $O(\boldsymbol{PlogP})$ where $\boldsymbol{P}$ is the number of grid points on the mesh. Good results have been obtained using this method by Smith and Stansby (1988), Graham (1988) and others. Two major disadvantages of the vortex-in-cell method are that grid generation is necessary and that computational effort must be expended in dealing with areas that contain no vorticity.

Another method is the zonal decomposition/summation method, described in detail below. The method has also been described as a 'fast multipole' technique and a 'vortex lumping' scheme. This scheme maintains the essence of the original Lagrangian scheme as no grid is required. The implementation of the method is significantly more complicated than the basic

scheme, but significantly reduces the operation count and yields results which are the same to within a specified numerical accuracy.

4.5.2 The Theory

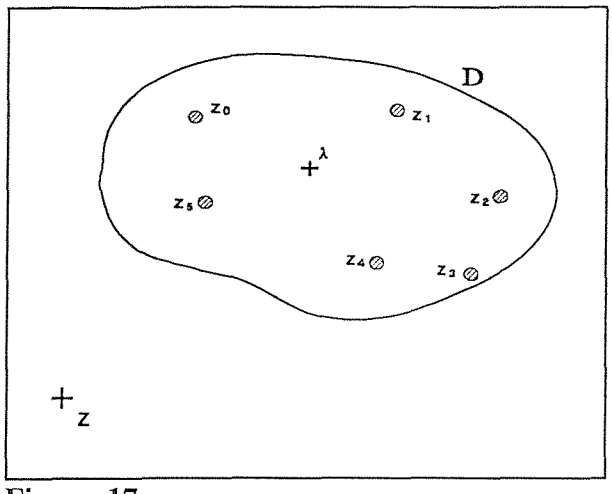


Figure 17

The zonal decomposition/summation method follows work by Van Dommelen and Rundensteiner (1989) but has been significantly restructured to make it more suitable for parallelisation. The method relies on the following theorem. Suppose there are N_p vortex blobs contained within a domain D (as illustrated in Figure 17), then sufficiently far from D the velocity induced by all N_p vortices may be calculated using a Laurent series in n_t terms (where n_t depends only on the accuracy required). 'Sufficiently far' in this case turns out to be relative to the maximum distance from any vortex in D to some arbitrary point in D (this maximum distance is termed the 'radius' of a zone). The coefficients of the Laurent series can be calculated in advance of the summation relative to this arbitrary 'centre' point, λ. Hence, beyond a certain distance from the centre, we can use the series and, if N_p is large, save work compared to a direct summation over each vortex. The formulae for the coefficients, velocity and convergence condition are

$$\text{let} \quad a_k = \sum_{j=1}^{N_p} \Gamma_j (z_j - \lambda)^{k-1} ,$$

$$\text{then} \quad u(z) - iv(z) = \frac{1}{2\pi i} \sum_{k=1}^{n_t} a_k (z-\lambda)^{-k} + O(\epsilon) \qquad (47)$$

$$\text{provided} \quad |z-\lambda| > h(n_t, \epsilon) . \sup\{ |z_j - \lambda| : j = 1, \ldots N_p \}$$

where a_k are the coefficients for the zone, Γ_j and z_j are respectively the strengths and positions of the vortices in the zone, λ is the (complex) centre point of the zone and h is a function depending on n_t, the number of terms used in the series and ε, the accuracy required. The derivation of these formulae are given in appendix A.

Equation (47) is, strictly speaking, valid only for point vortices. It has been observed, however, that no significant extra error is incurred by using the blob distributions given in

section 3.4. This is because these distributions decay rapidly and so are numerically indistinguishable from point vortices at only a few core radii from the vortex centre.

4.5.3 Algorithmic Details

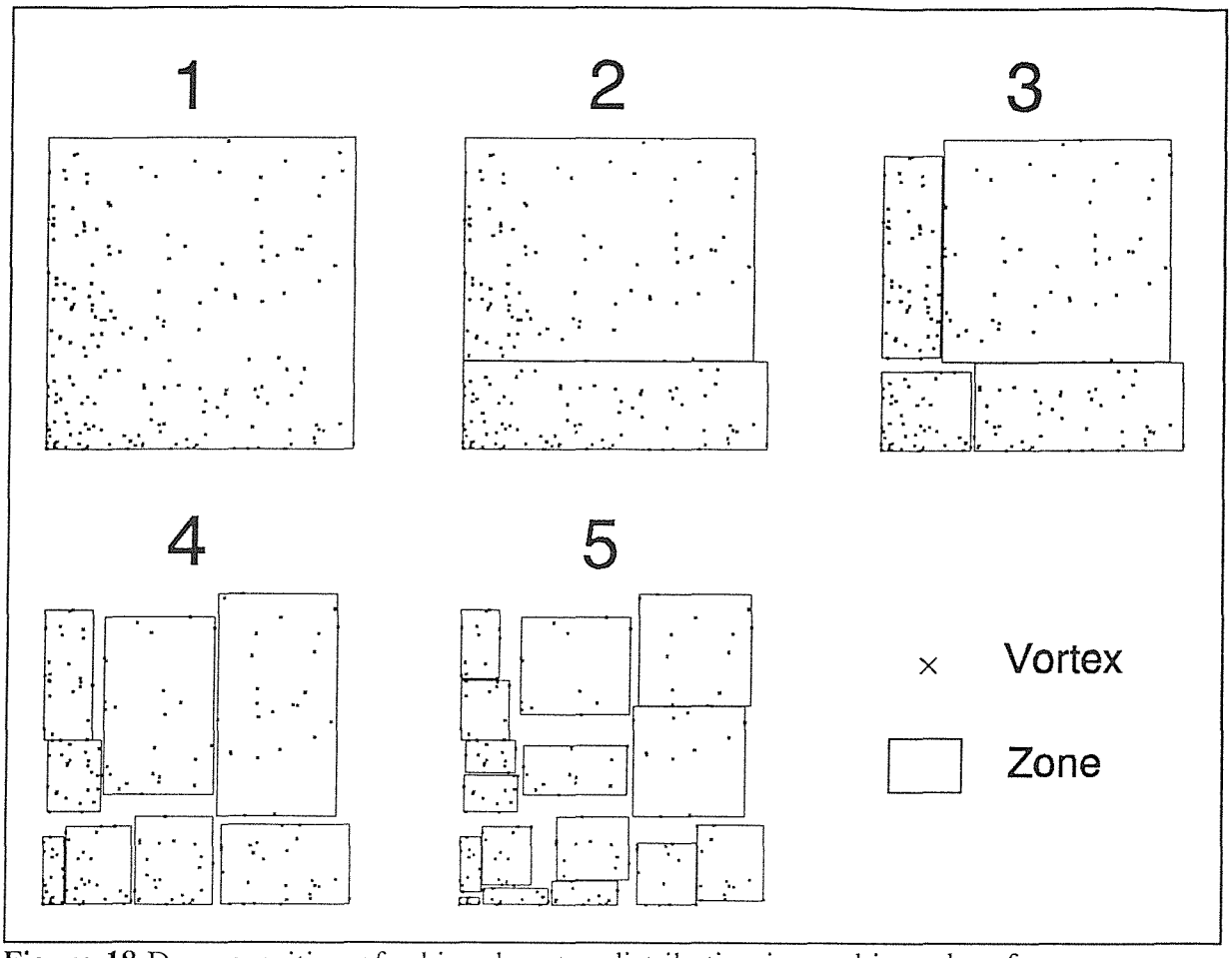


Figure 18 Decomposition of a biased vortex distribution into a hierarchy of zones

To use the theorem effectively, it is necessary to construct an algorithm to decompose the entire vortex field into a set of zones. The algorithm is made even more efficient by using a hierarchical zonal structure, so that the zone containing the largest number of vortices can always be used. For the parallel implementation it is also desirable to have zones at the same level in the hierarchy containing equal numbers of vortex blobs (for load balancing, see section 4.6).

Figure 18 shows the action of the algorithm developed for the parallel vortex code. The algorithm is of the 'divide and conquer' type. Initially the whole domain is split into two zones, each containing half of the blobs, the split being on the larger of its dimensions (See appendix B for pseudo code for the splitting process). Thus the resulting smaller zones are

as square as possible. Splitting is recursed until further division would leave the smallest zones containing too few vortices to save work in the forthcoming velocity summation.

In Figure 18, the dividing process is shown for a random distribution of vortices biased towards the bottom left corner of the region. The division is repeated in the example until each zone contains less than 20 vortices. Notice that, even on this relatively uneven distribution, most of the zones are nearly square. This minimises the distance of the furthest vortex from the centre (so the convergence condition is satisfied as close to the zone as possible).

The centre, λ, of *every* zone (ie. all the boxes in stages 1-5 in Figure 18) is then calculated and stored. The position of this point is arbitrary; we have found that the algebraic mean of the vortex positions is adequate. The distance of the vortex furthest from the centre (the radius) within each zone is also stored for checking the convergence condition of the Laurent series during the forthcoming velocity summation. The coefficients, a_k, for the zones are then calculated directly from the vortex blobs. All this involves an operation count, for each vortex, proportional to the 'depth' of the zonal structure, which can be seen[4] to be $O(log_2 N_V)$. Therefore the total operation count for forming the zonal structure is $O(N_V log_2 N_V)$ and thus will not dominate the calculation. In practice this process constitutes an almost insignificant amount of work compared to the velocity summation.

It has been discovered empirically that a minimum zone size of around 30 vortices (for n_t=25) is near optimal. The minimum zone size must be adjusted so that using the smallest level of zones is slightly cheaper than evaluating the velocity from the blobs themselves. Run times have proved remarkably insensitive to the minimum zone size used. In one extreme example the minimum zone size was reduced to 8, so that evaluating the velocity using the 25 term series would be more expensive than using the 8 blob sum. Since the zones are summed hierarchically, enough larger zones were used that the total run time was still far less than by direct summation.

The vortices must be sorted and indexed in a manner that allows the code to reference those within a particular zone. A velocity summation at a particular point is carried out in a piece-by-piece fashion from the zones. The zones are considered in the same order as for the splitting algorithm (largest first, then working towards smallest). A check is made, using the convergence condition from equation (47), on whether the Laurent series will converge for the zone under consideration and if so, the series is evaluated and the contribution from that zone is complete. If not, the smaller constituent zones are used. In effect this introduces another Laurent series. This process continues until either the convergence condition succeeds

[4] Doubling N_v would cause one extra zone splitting, the zone depth increasing by one.

or the smallest zones cannot be used, in which case summation is performed directly from the blobs. It can be seen that this summation process will take into account the effect of every vortex in the region.

The creation of the zonal structure and the velocity summation from the structure are most naturally implemented using abstract data types. Each zone requires a mixture of integers or pointers: to reference the correct vortices, and floating point types: to store the centre, zone size and coefficients. This can be achieved in Occam2 using the RETYPES construct which allows the code to access existing objects as a different type. However, the Occam2 implementation is clumsy and was difficult to debug. This part of the code would be much neater and more natural in C. It is interesting to note that the proposals for Occam3 include a record type which would be ideal for storage of the zonal structure.

4.5.4 Accuracy

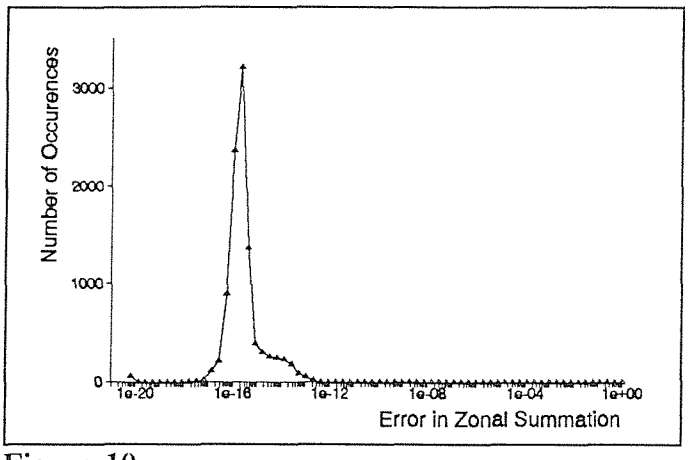


Figure 19

The accuracy of the zonal summation method has been tested by comparison with the velocity values as evaluated directly from the blobs, for a random vortex distribution. The results of a typical run of this test code are presented in the graph on the left. Figure 19 shows the number of errors falling in each error band $[2^i,2^{i+1}]$ (i=-67,...-1) for evaluating the velocity at 10,000 random locations (both inside and outside the region containing the blobs). In this case 200 Gaussian blobs were used, the number of terms in the Laurent series, n_t, was taken as 25 and the convergence function h was taken as 3. The error incurred for these parameter values is always smaller than 10^{-12}, and is most likely to be around 10^{-16}. The large peak in the graph at 10^{-16} corresponds to the round-off error in the machine representation of a 64-bit floating point number; this level of discrepancy would be expected from a simple reordering of the original computation. The actual error in the zonal algorithm is shown as a small plateau in the graph extending from 10^{-15} to 10^{-12}. The errors are much smaller than other numerical errors in the method, so the zonal summation gives effectively the same result as direct summation.

4.5.5 Computational Cost — Theoretical

The following argument indicates that the zonal velocity summation has operation count $O(N_V log N_V)$. This is reinforced by experiments where graphs of computation time versus number of vortices are nearly linear (see section 4.5.6). We start by considering the zones used in a typical velocity summation (this is illustrated in Figure 20). In the picture, the velocity is being calculated for the point z. A typical breakdown of sums from zones and from blobs in different regions is illustrated. Near the point z the direct sums must be used, then, moving further away, progressively larger zones can be used. The principle argument for evaluating the operation count is that (ignoring effects from the edges) a constant number of each zone size contributes to the sum at any given point. This statement is justified below for a uniform distribution of vortices. The amount of work in summing from a zone of any size, B, is fixed (it is $O(n_t)$) and the amount of work in summing from the blobs in a small zone, A, is fixed (it is $O(N_p)$). So the operation count for each velocity sum (not including the evaluation of the coefficients, which was shown to have cost $O(N_V log N_V)$ in section 4.5.2) is

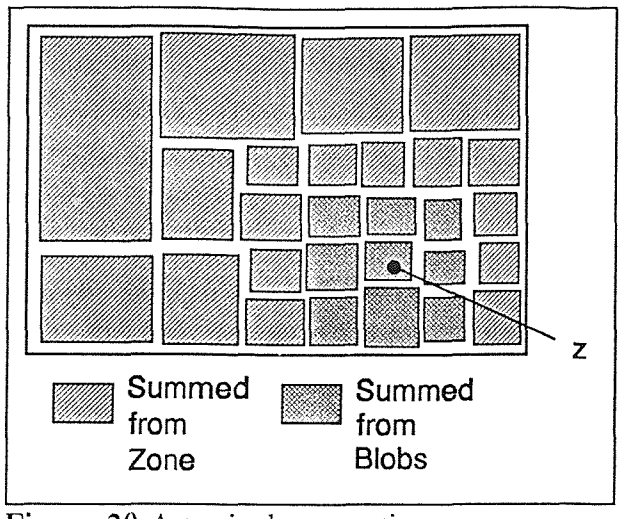


Figure 20 A typical summation

$$\text{OpCount} = Ak_b + B\sum_{j=1}^{depth} k_j \qquad (48)$$

where k_b is the number of zones for which a direct sum must be used and k_i is the number of zones at the i^{th} 'level' which contribute to the sum. Let the level be such that the zone at level 1 is the whole region, a zone at level 2 is half the region, etc. ***depth*** is the number of times the whole region is recursively split in the decomposition (see section 4.5.3).

The total number of level 1 zones is 1, level 2 zones is 2, ... level ***depth*** zones is 2^{depth}. Suppose (without loss of generality) the area of the whole region is 1. So the average area of a level 1 zone is 1, area of a level 2 zone is 1/2, ... area of a level ***depth*** zone is $1/2^{depth}$. This uses the assumption that the vortices are uniformly distributed and so the zones at a given level are of similar physical dimensions. The average 'radius' of each level of zones is the half the square root of the area.

The proof continues by estimating the regions where each level of zone contributes to the velocity summation. These regions are derived from the convergence condition of equation (47) and are circular annuli. Thus the area of the regions where each zone size contributes may be calculated and, since the average area of the zones is known, used to estimate how many zones of each size contribute to the sum. The argument uses the worst case assumption that the whole of the annulus where the condition holds is filled with contributory zones of a certain size. In reality, edge effects will cause parts of the annular regions to be completely empty of vortices, incurring less cost than that estimated. Throughout the following working, we have assumed a constant value for the quantity h, from equation (47).

For example, to estimate k_b, the region required is where no series can be used — the summation is from the blobs themselves. For such a condition to be satisfied, the point under consideration must be close enough to z that the convergence condition of equation (47) is not satisfied for the smallest (level ***depth***) zones. So the equation for possible values of λ for zones which would contribute to this part of the sum, is

$$|z-\lambda| < \frac{h/2}{2^{depth/2}} \tag{49}$$

which defines a circular region centred on z with an area $\pi h^2/(4.2^{depth})$. So, dividing this area by the area of a level ***depth*** zone gives an estimate of the number of zones which must be summed using blobs,

$$k_b \sim \frac{\pi h^2}{4} \tag{50}$$

To estimate k_{depth}, the condition that a level ***depth*** zone can be used, but that a level ***depth***-1 cannot, is required. As above, the convergence conditions for these two criteria give

$$\frac{h/2}{2^{depth/2}} < |z-\lambda| < \frac{h/2}{2^{(depth-1)/2}} \tag{51}$$

which defines an annulus centred on z with area $\pi h^2/(4.2^{depth})$. So, dividing by the area of a single level ***depth*** zone,

$$k_{depth} \sim \frac{\pi h^2}{4} \tag{52}$$

The estimation of the constants $k_{depth-1},\ldots k_1$ follows the same reasoning, they form concentric annuli. At some point edge effects will start to reduce the constants (the annulus cannot be filled with usable zones because the edge of the region has been reached), but we can still use the theory to put an upper bound on the number used. The upper bounds are

$$k_i \sim \frac{\pi h^2}{4} \quad \text{for } i=1,\ldots depth-1 \tag{53}$$

Putting the upper bounds for the k_i into equation (48) gives

$$\text{OpCount} \sim \frac{\pi h^2}{4}(A + depth.B) \tag{54}$$

In other words the operation count for each vortex is proportional to the zone depth, which is $O(log_2 N_V)$. So the total operation count for calculating the velocity for all the vortices is $O(N_V log_2 N_V)$. This whole argument can be explained by the simple idea that by increasing the total number of vortices in a sum, the effect is to create a few extra sums from large zones, with the number of direct sums and sums from smaller zones staying relatively constant.

Finally it is worth noting that, as explained above, this method gives effectively the same results as a direct $O(N_V^2)$ summation, whereas the vortex-in-cell method must introduce a grid-related numerical viscosity.

4.5.6 Computational Cost — Experimental

Possibly the most important performance measure of a discrete vortex method code is the algorithmic performance. To put this statement into explicit terms, the dependency of the execution time on the number of vortices used will ultimately govern the size of the problem that may be solved. It is therefore very important to verify experimentally that the application of the algorithmic theory given above has given the expected results.

The code outputs a file specifically for the purpose of analyzing performance. The file contains timings taken for the various phases of the overall algorithm, and for each complete time step. These timings are measured internally for each processor for each phase, then an average is taken to give a single result for the time taken by the code as a whole. The results of a typical large run are shown in Figure 21, which was executed on 17 T800 processors with a clock speed of 25MHz.

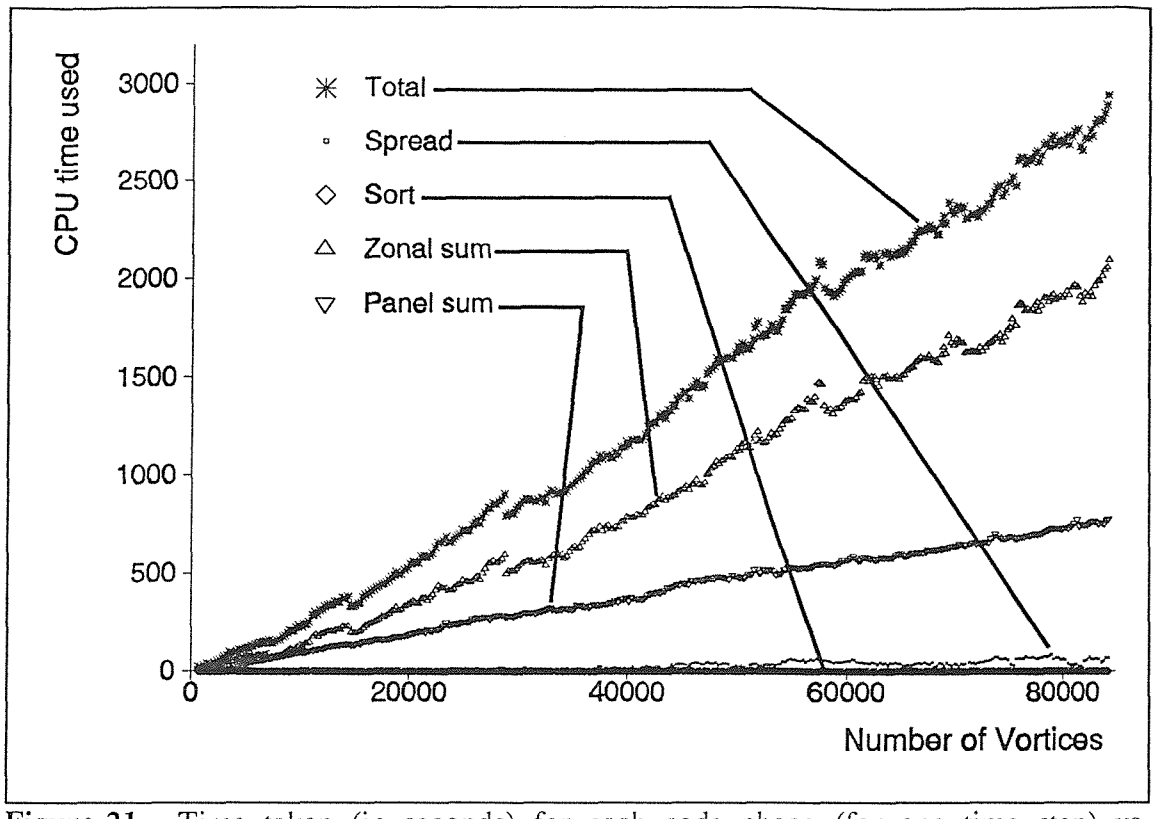


Figure 21 Time taken (in seconds) for each code phase (for one time step) vs. number of vortices, for a typical large run

The results show the total time taken for each time step, as well as the time taken for the four major sub-steps — the linear panel sum (see section 3.6), the zonal velocity summation, the global sorting of vortices (see section 4.7) and the spreading of vortices amongst processors (see section 4.8).

The two most time consuming stages are the panel summation and the velocity summation. The sorting proves to be an almost negligible amount of work and the spreading, although noticeable, is insignificant in comparison with the two work-intensive phases. The panel summation is trivially shown to be theoretically linear in the number of vortices, and the results verify this. The velocity summation is more complex, the graph appears to be divided into approximately linear segments, with discontinuities at their bordering points. The run time seems to grow linearly with number of vortices until a certain point, then drop slightly, then resume the linear behaviour at a slightly increased gradient. This behaviour is explained by revisiting the algorithmic construction and noticing that a change in behaviour would be expected with increasing numbers of vortices — at the point where there are a sufficient number for an extra zonal division to occur. At such a point, the summation would change in composition by adding a new set of 'smallest' zones, each containing just more than half the number of vortices of the previous smallest zones. Evidently this change initially causes the operation count to drop slightly, a somewhat unexpected result. The behaviour is

near-linear and, with slightly increased gradient between discontinuities, seems to fit the $O(N_V log_2 N_V)$ prediction given in section 4.5.5.

The total time taken for a time step is near linear in number of vortices, as was expected. This is still expensive for an algorithm and means that following a run for a long time period is dramatically more expensive than seeking an accurate short time solution. In fact, a reasonable estimate of the cost of calculating a flow to time t is given by $O(t^2)$, using the simple assumption that an equal number of vortices enter the flow on each time step. However, this should be contrasted with the basic algorithm, which was $O(N_V^2)$ in number of vortices leading, in a similar manner, to a total operation count $O(t^3)$ to calculate a flow to time t.

4.6 Parallel Implementation Details

The parallel implementation is a natural extension of the zonal decomposition. One processor deals with the boundary calculations; the vortex blobs are divided amongst the others in such a way that those close in physical space reside on the same processor.

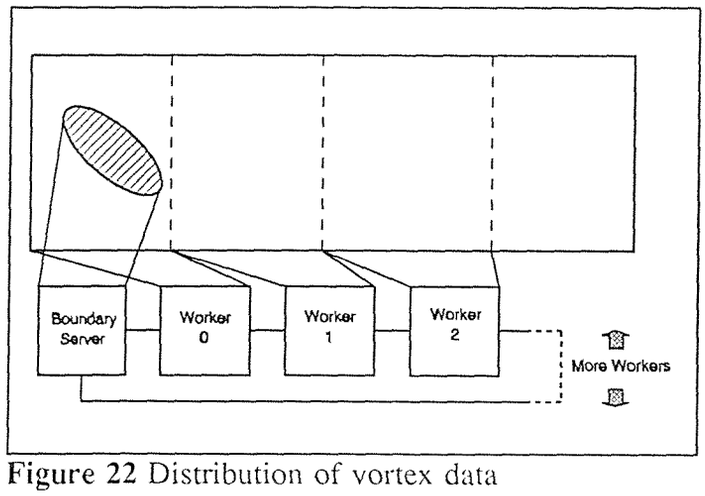


Figure 22 Distribution of vortex data

For the set of problems we are considering, the vortices form an elongated wake in the streamwise direction. Therefore, in order to simplify the algorithm, we decided to distribute the vortices in spatially disjoint regions on the streamwise coordinate only (as shown in Figure 22). The vortices are sorted globally at every time step, and partitioned so that equal numbers are held on each worker (for load balancing purposes). The boundary server holds the panel matrix and handles all the boundary condition calculations, including the introduction of new blobs into the flow. Each worker calculates the velocity of the vortices held locally and time steps their positions. New vortices are distributed from the boundary server to the workers and then the whole domain is re-sorted and the numbers of blobs are equalised (termed 'spreading'). In this way the task of updating the flow is shared equally amongst the available number of processors.

The hardware configuration is illustrated in Figure 23. The workers and boundary server are as explained above. The graphics server is optional and, if present, contains a graphics board and is used to display flow visualisations on a monitor as a run progresses (see section 4.10). The host server reads all the initialisation data for a run from files on the host CPU, then either disconnects completely from the host CPU or periodically saves the state of the flow to files (see section 4.11). The boundary server and host server need proportionately more memory than the workers (typically 4Mbytes each). The workers can have relatively small local memories, some of the results presented here were carried out using 256Kbyte workers. Obviously, more available memory allows more vortices to be used and hence flows can be followed further in time or resolved in finer detail.

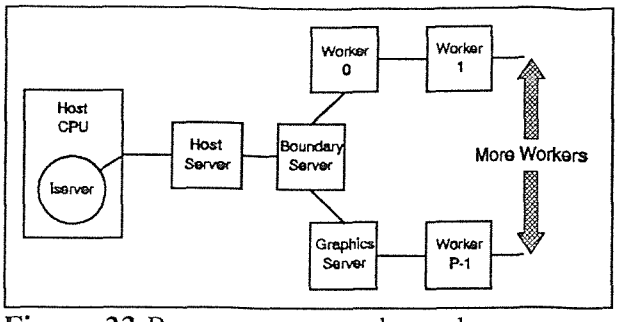


Figure 23 Processor network used

The overall process structure of the Occam code for any of the processors in the ring is illustrated in Figure 24. The Occam pseudo code for this arrangement is simply,

```
PAR
  Left()
  Right()
  Worker()
```

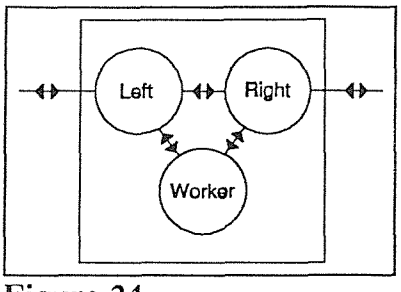


Figure 24

The interconnecting channels, omitted from the pseudo code for simplicity, are passed as parameters to the procedures. The left and right processes allow data to be received and sent concurrently with the worker process performing calculations on the current data. They also serve the purpose of a simple routing harness, ensuring that messages sent by the worker arrive at the correct destination process without causing deadlock.

The parallel algorithm for first order time stepping of the vortices is slightly simpler than the higher order version, so this will be explained first. The graphics server merely passes all data straight through in both directions, making a copy of any information needed for plotting (see section 4.10). The main part of the algorithm concerns the function of the ring of workers and of the boundary server over a fluid time step. The code is designed so that vortices can be deleted from the flow.

Worker processors:

1 Spread the vortices evenly amongst all the worker processors, blend the new vortices (from the end of the previous time step) into the main vortex data set and clear any deleted vortices.

2 Globally sort the vortices on the streamwise coordinate. Locally sort the vortices on the transverse coordinate.

3 Perform a zonal decomposition on the local vortices on each processor, form a data structure containing indexing information, zone sizes, zone centres and Laurent series coefficients.

4 On each processor, calculate the velocity induced by the local vortices at the panel control points.

5 On each processor, send the zonal data, the vortices, the partial boundary velocity from step 4, any visualisation information calculated on the previous time step and any run information for the host (eg. timings, forces) to the right process. This information is all to be circulated clockwise around the ring, with the left process delivering to each worker as it passes.

6 Calculate the velocity induced by the local vortices on each other, for each processor. Calculate the effect of the local vortices on any visualisation data.

7 Receive a remote worker's zonal structure and vortices from the left process. Calculate the velocity induced by the remote vortices on the local ones. Calculate the effect of the remote vortices on any visualisation data. After delivering the remote data to the worker, the left process sends another copy to the right process. The right process then sends the data to the next processor in the ring while with the left process receives the next set of remote data and the worker adds the effect of the current remote data. This step is repeated until all the remote data has visited this processor.

8 Send approximate spatial bounds of the local disjoint streamwise segment from each worker process to its local left process to help it acquire new vortices in the correct region. These values are merely guidelines to reduce the amount of spreading and sorting needed. The processors at either end of the chain adopt from regions extending well beyond the area of interest to ensure all new vortices are claimed.

9 Receive the panel strengths from the boundary server. The left and right processes pass them along the chain to each worker process. Add the velocity induced by the panels and the uniform flow to the velocity of the local vortices. Add the effects of the panels and the uniform flow to any visualisation data.

10 Time step the positions of the local vortices using the summed velocities and the random walk (if being used). At this stage some vortices may be deleted or reflected due to crossing a boundary.

11 For each processor, the left process passes (to the worker process) any new vortices (from the boundary server) falling in the region set up previously. The new vortices falling outside this region are passed along the chain.

Boundary Server:

1 (Corresponding to worker processor step 7) The left and right processes circulate data from the worker processors around the ring. The partial velocity at the panel control points from each worker processor is passed to the worker process on the boundary server. The total velocity at each control point is summed by the worker process.

2 The uniform flow is added to the velocity at the control points. The matrix calculation for the panel strengths is performed (an LU decomposition has been performed on the matrix in an initialisation phase).

3 The panel strengths are sent to the chain of worker processors.

4 New vortices are created around the boundary and their positions are time stepped appropriately (including deletion or reflection of any crossing the boundary, if required). Surviving new vortices are sorted on the streamwise coordinate.

5 The new vortices are sent to the workers in packets whose maximum size is the maximum number of vortices a worker processor can store (it is possible to generate more vortices in one time step than any individual worker could store).

The second order Runge-Kutta time stepping scheme requires a slightly different algorithm. The scheme is applied as follows,

$$\text{let} \quad \underline{h}_j = \frac{1}{2}\underline{u}(\underline{x}_j(t)) \,, \qquad \text{then} \quad \underline{x}_j(t+\delta t) = \underline{x}_j(t) + \underline{u}(\underline{x}_j(t)+\underline{h}_j)\,\delta t + O(\delta t^3) \tag{55}$$

gives the convective motion of a vortex at $\underline{x}_j(t)$. This can be achieved with no extra storage as follows. Suppose $\underline{x}_j$ and $\underline{u}_j$ are the array elements used to store the position and velocity of the jth vortex, then

1 Calculate $\underline{u}_j$ as the velocity of the jth vortex based on the current vortex positions.

2 Let $\underline{x}_j \rightarrow \underline{x}_j + \frac{1}{2}\underline{u}_j(t)\delta t$ be the intermediate position of the jth vortex. If the vortex enters the body at this stage, reflect it[5].

3 Let $\underline{u}_j \rightarrow -\frac{1}{2}\underline{u}_j$.

4 Add to $\underline{u}_j$ the velocity of the jth vortex based on the *intermediate* vortex positions.

5 Then $\underline{x}_j \rightarrow \underline{x}_j + \underline{u}_j(t)\delta t$ is the final position of the jth vortex as specified by equation (55). If the vortex enters the body, reflect or delete it.

To use this method the parallel algorithm must be adapted to perform the velocity summation twice successively. This entails repeating stages 3 to 7 for the worker processor, with some extra code to effect the second order calculation as described above. Stages 1 to 3 must be repeated for the boundary server. The vortices are not resorted between the substeps since reordering of the position arrays would occur, adding the complication of matching a particular velocity to a vortex after the reorder. We have found that this omission makes very little difference to the run times. This is because vortices which are initially sorted into zones will remain grouped together after a small time step. The zones may overlap slightly, but this appears not to affect the performance of the method significantly.

[5] Deletion within a partial step is considered inconsistent

4.7 Sorting

The global sorting of vortices across processors is by no means trivial. Naive sorting algorithms (eg. bubble sort) are typically $O(N^2)$ (N being the number of elements in the data set to be sorted), so care must be taken to not allow this to become the time consuming step. The vortex code uses an algorithm based on 'Pigeon sort', see Birch (1988). The algorithm sorts random data with uniform distribution (equal probability of each point lying anywhere in a given region) in an operation count of $O(N)$. It is similar to the commonly used quicksort algorithm, which recursively divides the data set to be sorted into two subsets : those lying below and above the point halfway between the minimum and maximum elements. In pigeon sort the same approach is adopted, except that the data set is recursively divided into N subsets, where N is the number of elements in the set being divided. The sets used are those elements lying in the 'pigeon holes' created by dividing the region between the minimum and maximum elements into N equal segments.

Since the data set to be sorted (ie. the vortices) must be ordered on two-dimensions (locally), the *actual* data cannot be sorted because sorting on transverse coordinate would spoil the streamwise order. Instead, indexing arrays must be used which, in Occam, are best implemented as arrays of integers. Using a double subscript gives the data in order. So, given the floating point data $x[i]$ and the index array $a[i]$, the object of the sort is to achieve

$$x[a[i]] \le x[a[i+1]] \quad \forall\ i \in [0, N-1). \tag{56}$$

The sequential pigeon sort algorithm consists of the following stages (for pseudo code for the algorithm, see appendix B),

1 Scan through the data set once to find the maximum and minimum elements, x_{min} and x_{max}. In the trivial cases where $N<2$, the sort is already completed.

2 Store in an array the pigeon hole which each element lies in. The boundary between the $(i-1)^{th}$ and the i^{th} pigeon hole is the point $x_{min}+(i/N)(x_{max}-x_{min})$. The correct pigeon hole is conveniently and quickly calculated by a scaling and rounding to an integer value.

3 Scan through the array created in step 2 to count the number of elements in each pigeon hole, store the results in another array.

4 Reorder the index array so that all the elements in the first pigeon hole appear first, then those in the second pigeon hole etc. This is carried out by summing the values in the array created in step 3 to work out the starting positions for each pigeon hole, then using the array created in step 2 to transfer the correct index values for each pigeon hole into the appropriate segments of a working index array.

5 Copy the working index array back to the main index array. Repeat step 1 using the segment of the index array containing only those elements from the first pigeon hole in place of the whole set.

It can be seen that this process will eventually achieve the condition (56). For uniformly distributed data, the expected proportion of pigeon holes containing m elements on the initial subdivision is given by the binomial distribution,

$$f(m) = \frac{N!}{(N-m)!m!}\left(\frac{1}{N}\right)^m\left(1-\frac{1}{N}\right)^{N-m} \qquad (57)$$

so as $N\to\infty$, $f(m)\to\frac{1}{e.m!}$.

So the percentage of pigeon holes containing m elements is asymptotically independent of N. Hence, for large N, each element must on average be examined and pigeon-holed a fixed number of times. Using equation (57) the fraction of pigeon holes containing one element is expected to be 37%. Since a pigeon hole containing one element needs no further sorting, on each pass 37% of the elements are completely dealt with, so if the operation count for sorting N elements is $W(N)$ and the operation count per element for a pass is A, then

$$\begin{aligned} W(N) &= AN + W(0.63N) = AN + 0.63AN + W((0.63)^2N) \\ &= AN(1+0.63+0.63^2+0.63^3+\ldots) \approx 2.7AN. \end{aligned} \qquad (58)$$

In other words, each element is on average examined approximately 2.7 times.

The code has been further optimised by explicit use of optimal sorts for pigeon holes containing two, three or four elements (eg. two elements can be sorted by one test then a swap if needed).

The parallel version of this algorithm is based on the observation that the sequential algorithm is symmetrical. The pigeon holes created by the first subdivision can be dealt with in any order. In particular, it is simple to work from the minimum upwards or the maximum downwards. Taking a pigeon hole alternately from the bottom then top we can sort the data set from the edges to the middle. The elements which have already been already sorted at the edges are sent to the left (minimum elements) and right (maximum elements) processes. The left and right processes exchange elements with the connected processes on neighbouring processors, comparing the values to check if the element should be swapped between processors. As soon as two elements do not need swapping, the sort can proceed purely locally.

As presented here, the parallel sorting algorithm does not correctly handle an element which ought to move by more than one processor to arrive at the correct destination in the sort. Such an element will remain at the correct end of the sort, on the processor neighbouring its original one. In an extreme case, it would migrate by one processor in the correct direction at each time step. The algorithm could be modified to deal with this situation correctly but, since the vortices always remain partially ordered by the time stepping, the added complication to the code was judged as unnecessary.

4.8 Spreading

It is extremely important to load balance any parallel code. This code contains a few synchronisation points throughout the time step. If one processor takes longer to execute the code between two synchronisation points, the others must all wait before starting the next phase of the calculation. This causes usable CPU cycles to be wasted and so makes the code inefficient.

The most obvious way to balance this application is to spread the number of vortices uniformly over the worker processors. This should give each worker an equal share of the calculation and reduce the idle processor time. The following algorithm is used to achieve the correct number on each worker,

1 Just after the acceptance of new the vortices from the boundary server, the local number of vortices currently on each worker is circulated around the ring (see section 4.6).

2 Calculate the total number of vortices, N_V, by summing the circulated local totals.

3 Calculate the desired number of vortices on the i^{th} worker (for P worker processors), which is $(N_V/P)+i>(N_V\backslash P)$, where the / and \ are the integer division and remainder operators respectively, which obey the rule $(N/M)M+N\backslash M=N$, and $a>b$ takes the Boolean value of 1 if true or 0 if false.

4 Calculate the desired total on either side of the i^{th} worker. Use these to calculate the deficit between the desired totals and the existing totals.

5 Send to or receive from each neighbouring processor exactly the number of vortices required to achieve the desired totals to either side. The overall effect of this is to achieve the correct number on every worker.

This keeps the number of vortices as near to equal as possible between workers (they can differ by at most 1).

The resulting code is reasonably well balanced, apart from a general tendency for the end processors to have slightly less work to do. This is believed to be due to the end processors being further from a larger proportion of the vortices; the zonal summation is more efficient at a greater distance from the zones being used.

Another improvement in efficiency would be gained by removing some existing synchronisation points. Overall, the code must run in lock-step so that the workers are all dealing with the same time step. However, given a global communications network (such as that provided by T9000s and C104s (see section 4.2)) some synchronisations which currently exist to keep the communications simple could be removed. This would involve restructuring the code for a fully connected network, where each processor is connected to all the others, as opposed to the current ring network. Performance would be improved, as there may be cases where a slight imbalance in workload exists which would average out over a longer time period.

4.9 Parallel Code Performance

The object of this section is to evaluate the performance of the discrete vortex code when executed on various numbers of processors. To compensate for the added effort of developing a parallel program, it is important to ensure that the performance benefits are being realised. A measure must be introduced which gives an idea of the increase in code performance with increasing numbers of processors, a property known as the 'scalability' of the code. An ideal parallel program will scale linearly with increasing numbers of processors. In practice, there is a limit to the scalability of any given problem (see the example in section 4.1.1), so the best that can be hoped for is near-linear scalability for a reasonably large number of processors.

The usual quantity introduced to measure scalability is called 'efficiency'. Various measures of parallel efficiency exist, all are designed to give the percentage of optimal performance attained by a code. The most rigorous and most commonly accepted definition of efficiency is,

$$\%\ \text{Efficiency} = \frac{\text{Run time on 1 processor}}{(\text{Run time on } N \text{ processors}).N} \times 100 \qquad (59)$$

where the single processor version should be a fully sequential version, which does not have the parallel overheads. Unfortunately, due the lack of any code maintenance provisions in Occam (such as the conditional compilation directives in C), the sequential version was not kept up to date. Thus it was necessary to use only the parallel code on varying numbers of processors to evaluate performance. Given this constraint, the whole idea of efficiency becomes fairly arbitrary as there is no absolute quantity to normalise against (this is usually provided by the sequential version). However, it is still worth considering the total CPU time elapsed for a given problem, defined by conducting a fixed amount of computation on N processors, then calculating

$$\text{Total CPU elapsed} = (\text{Run time using } N \text{ processors}) \times N. \tag{60}$$

This is a useful quantity as it gives the maximum available processor time, in other words the sum of the used and wasted CPU time. As the number of processors is varied for a particular calculation it is found that this figure becomes larger when more CPU time is wasted. The effect of not comparing to a sequential version is to ignore any extra calculation which must be performed in the parallel algorithm. It is typical to find that the fundamental inefficiency in a parallel algorithm can cause an extra 'factor of 2' to be introduced, so that the parallel code runs at approximately the same performance on two processors as the sequential code on a single processor. It is generally accepted that, provided a reasonably linear scalability is achieved for further increase in the number of processors, this parallelisation penalty is worth paying. In terms of the newly introduced quantity of total CPU time elapsed, linear scalability corresponds to a constant value with increasing numbers of processors.

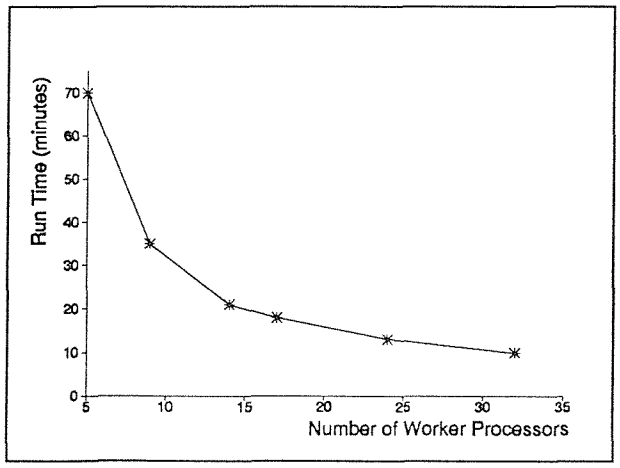


Figure 25 Run times vs. number of processors

Figure 25 shows a graph of run times for a fixed calculation which was performed using varying numbers of processors. The run in question was computing the flow past a circular cylinder at Reynolds number 500. 100 panel elements were used to represent the body and the time step was set to be 0.05. The code was set to generate streamline plots at every interval of 0.5 non-dimensional time units, so that four such plots were made. A fairly course plotting grid with a fixed number of points was used. The start time of each run was recorded, then the operating system time-stamp on the file containing the fourth streamline plot was used to work out the elapsed run time. By the end of the run approximately 1600

vortices were present in the flow. The processors used for these experiments were all 20MHz T800 transputers, with 4-cycle external memory. The results are difficult to analyze in this form, but do have the advantage of being in terms of a primitive quantity. The graph shows decreasing run time with an increasing number of processors. A closer examination reveals that the run times are approximately halved by doubling the number of processors — an encouraging sign. Another pertinent topic to raise at this point is: how far from real time simulation are these figures ? Taking a non-dimensional time unit from the model to be one second and using 32 processors, the run time is ten minutes. This is a factor of 300 slower than real time which means that, for this calculation, real time would be achieved using $O(10^2)$ processors $O(10^2)$ faster than a T800 Transputer. Realistically, this will still not be possible with the next generation of microprocessors, but is almost certain to have been surpassed by the turn of the century. Note that the term 'real time' is used here to indicate a frame rate suitable for animation purposes, rather than to refer to dimensional fluid time.

Figure 26 is a graph of the same data as that shown in Figure 25, this time it is represented in terms of the total CPU time elapsed as defined in equation (60). The total CPU time elapsed decreases up to a certain number of processors and then begins to gradually increase again. The expected behaviour for increases in numbers of processors beyond the values shown here would be to continue the gradual increase for a certain number, then for the gradient of the graph to start to increase. Eventually a point will be reached where there is no decrease in run time by adding an extra processor. The shallow gradient on the increase in total CPU between 17 and 32 processors is a good result — this shows that very little computing resource is wasted by increasing the number of processors used for the simulation.

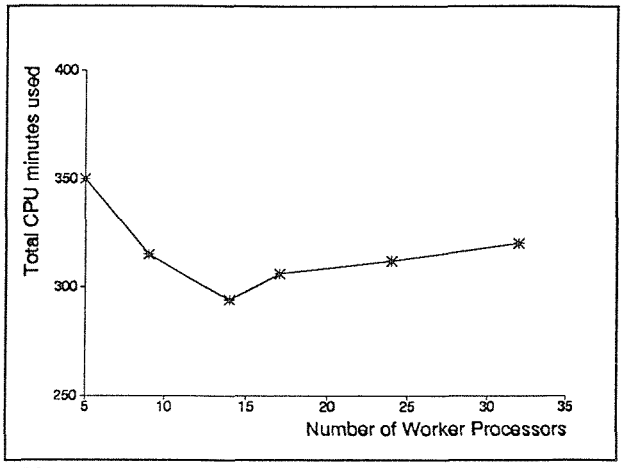


Figure 26 Total CPU time used vs. number of processors

It should be realised that the system being studied here is extremely complicated, involving several algorithms whose workload is based on run-time calculated quantities. Therefore a full analysis of why the CPU time varies like this is not possible, but some likely contributory factors can be mentioned. One significant factor is that the CPU time is being measured for all the processors, but the host server and boundary server have a role in the calculation which is independent of the number of processors used. This means that, in a typical calculation, the host server and boundary server will spend a significant amount of time idle. This has a larger effect when less processors are used as firstly, the two constitute a larger fraction of the total processing power and secondly, each worker processor has more

work to do, so the idle periods of the other two processors will be longer. The other major source of inefficiency worth noting is that slight inequalities will exist between workers in the amount of work to be done in the zonal summation stage. Due to the dynamic nature of the composite summation in equation (47), an accurate prediction of the amount of work involved cannot be made in advance. Assuming that this is essentially a random effect, more total CPU cycles would be wasted using a larger number of processors. However, one compensating effect is that using a larger number of vortices should help to average these random inequalities.

The code has been shown to give reasonable performance characteristics when the size of the processor network is increased. Most of the results presented here were derived from runs using either 14 or 17 processors, the region for which the total CPU time used is near minimum.

The example run was selected as an example of a small run to illustrate the performance of the code on a relatively fine-grain parallel problem. The code will have a better parallel performance on a larger problem. One of the hardware platforms we regularly use has processor performance lights, and we have noticed that these are illuminated for a high proportion of the time when conducting runs with many vortices. In some of the very long runs, which generated flows containing nearly 100,000 vortices, each time step was taking nearly one hour. During such a large run, the inequalities in work caused some processors to be inactive for typical periods of a minute. Thus we believe that the idle time drops significantly on larger runs.

Perhaps a more important issue than that of absolute performances, is the fact that the computing power for this project was completely provided by facilities available on the University of Southampton campus. If the same work were to have been constrained to the use of SISD or SIMD architecture machines, time on external resources would have had to be acquired, because no local machines of sufficient power have been available during the duration of the research. We estimate that in excess of 1000 hours of CRAY time would be needed to reproduce all the runs which have contributed to this work. Thus, we would probably not have been able to use such large runs if we had been constrained to SISD/SIMD machines.

4.10 Visualisation

The production of useful flow visualisations was one of the main aims of this project. An unexpected outcome is that visualisations have proved extremely useful in debugging code and finding suitable values for numerical parameters. All the visualisation modes can be displayed in colour on a high resolution graphics monitor or output to files in HPGL (Hewlett Packard Graphics Language) format for hardcopy devices.

Apart from simple vortex plotting (see section 4.10.1), the visualisation computations are extremely expensive compared with those for updating the flow. Although the operation counts for these procedures are all $O(N_V)$, the multiplicative constants can be large (eg. proportional to the number of grid points used). Consequently complex visualisations tend not to be calculated on every time step and vortices plotted on the intervening steps (this is specifiable by the user, see section 4.11).

In each of the display modes described below, a visualisation rectangle is specified by giving a minimum and maximum streamwise and transverse coordinate. The relevant modes are then displayed in this region, and clipped outside. Any part of the bluff body lying in the visualisation region is also plotted. The streamwise coordinate maps to the horizontal and the transverse to the vertical for plotting purposes.

4.10.1 Discrete Vortex Display Mode

This is the simplest mode and the most useful for debugging the code. On the graphics screen the vortices themselves are plotted as small blocks of coloured pixels. The vortices are coloured red for positive circulation and blue for negative circulation, the stronger vortices being coloured more intensely. In the hardcopy files, vortices are drawn as small triangles with the orientation depending on the sign, with no indication of strength. An example of the hardcopy produced by this display is shown in Figure 27, the roll-up of sheets of vortices can be clearly seen. Refer to section 5.2 for a full explanation of the terms α, Re and τ.

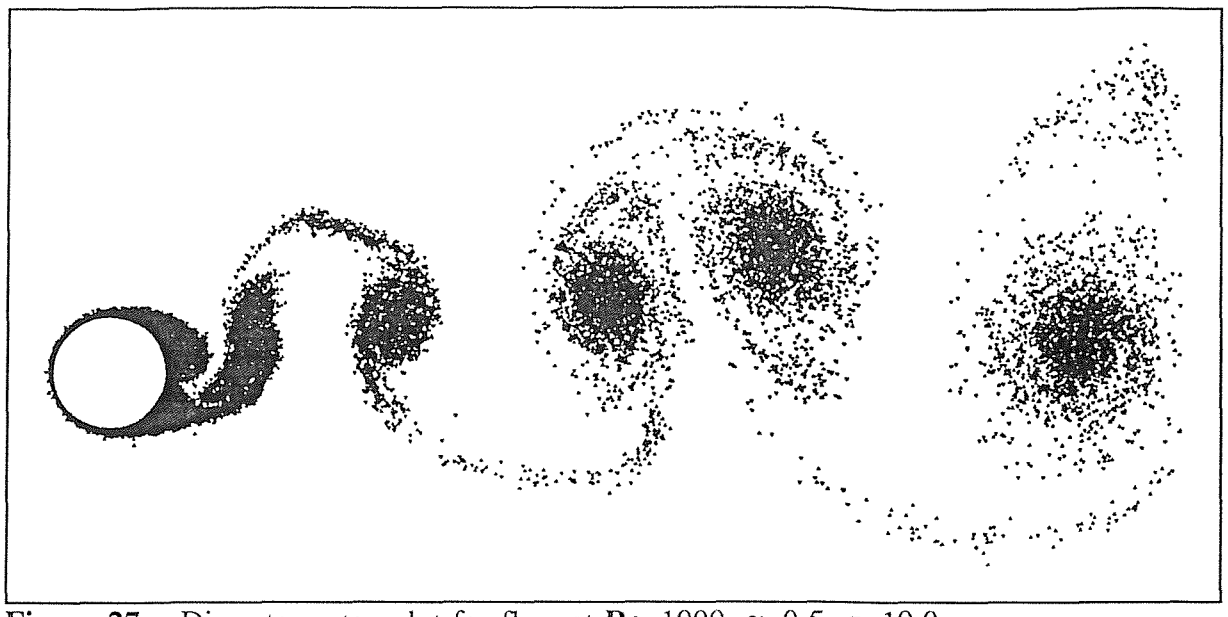


Figure 27 Discrete vortex plot for flow at **Re**=1000, α=0.5, τ=19.0

This mode is of limited usefulness for serious flow visualisation. The important flow regions tend to become crowded with vortices and, since the vortices are merely plotted on top of each other, the actual vorticity is difficult to deduce from the pictures. However, it does give the underlying information on which the method depends (ie. the positions and strengths of the blobs) and so can be useful.

4.10.2 Velocity Vector Display Mode

In this mode a grid is introduced for display purposes. The grid is distributed by dividing the visualisation region into a number of vertical strips and mapping each strip onto a worker processor. The grid is stored in single precision to save on memory and increase the speed of calculations; since the grid is only for visualisation, the accuracy does not need to be high. The use of vertical strips is an attempt to load balance the grid calculations. The wake spreads horizontally so will tend to spread across every processor's grid rather than being predominantly located on one processor's grid (as would often be the case using horizontal strips).

The Occam language presents a problem with the implementation of the visualisation grid. Occam is a static language, so array sizes must be determined at compile time. In all the implementations we have used during this project (product numbers IMS D700C, D700D, D705B and D7205A) the RETYPES command will not allow reshaping of a two-dimensional array. In other words, the code fragment

```
...
[10][10]REAL32 A:
INT x,y:
SEQ
  x:=20
  y:=5
  [y][x]REAL32 B RETYPES A:
  B[4][15]:=3.1415(REAL32)
...
```

would produce a compiler error at the line containing the RETYPES. This would be the obvious mechanism to use for different shaped grids. In earlier versions of the code, the maximum grid dimensions in each direction were fixed and a recompilation was necessary to change these. The code now uses one-dimensional arrays to store the grid and subscripting functions to access the correct element, allowing complete grid reshaping without the need to recompile. The fragment above would be implemented as

```
...
[100]REAL32 A:
INT x,y:
SEQ
  x:=20
  y:=5
  INT FUNCTION subscr(VAL INT sy,sx) IS (sy*x)+sx:
  A[subscr(4,14)]:=3.1415(REAL32)
...
```

which is nearly as concise, but is not as well checked by the compiler as the RETYPES would be.

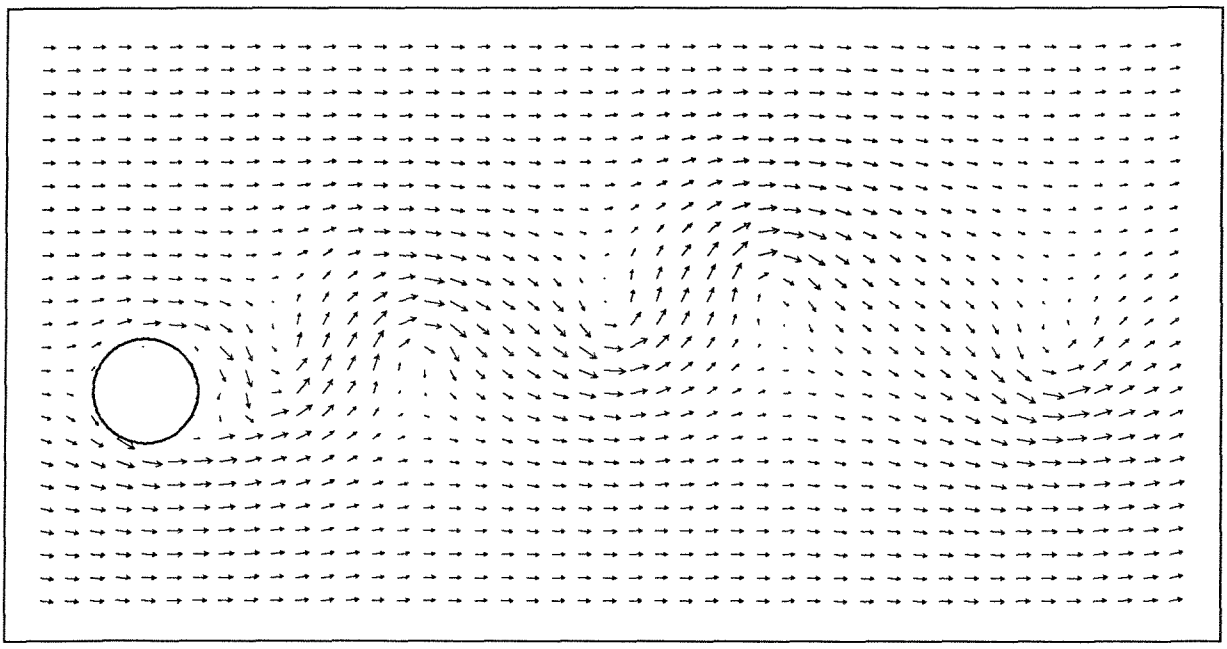


Figure 28 Velocity vector plot for flow at **Re**=1000, α=0.5, τ=19.0

The velocity is calculated at each grid point and then displayed as a scaled green line on the graphics screen. The vector is displayed in the hardcopy as a scaled arrow, as shown in Figure 28. This visualisation mode was relatively simple to implement and gives a good idea of the physics of a flow. The main drawback is that detailed resolution of the important flow regions is difficult — increasing the density of grid points tends to crowd the picture. Streamlines are generally better suited to judging the direction of the flow, but vectors have the advantage of giving a direct indication of the fluid speed.

4.10.3 Streamline Display Mode

In this mode a grid is created in a similar manner as in section 4.10.2. Neighbouring grids share a column of cells so that contours drawn on neighbouring grids join together. The streamfunction is calculated at the grid points (see appendix C for streamfunction formulae). Analytic integrals for the various vorticity elements are used to give values for the streamfunction. Initially, we tried to numerically integrate the velocity to produce streamfunction values. The number of points needed to sufficiently reduce the errors made this an extremely expensive process, because the velocity is relatively expensive to evaluate. The streamfunction formulae tend to be rather complicated and hence also expensive to calculate. However, as the results contain no grid-related integration error at the grid points, the grid can remain relatively coarse and induce no overall errors.

When the streamfunction has been evaluated, the streamlines are the contours of the values on the grid. A contour plotting routine was written to calculate the contours of a numerical quantity given on a uniform rectangular grid. The process is conducted in two stages. Firstly, a temporary array is used to store all gridcell sides crossed by the relevant contour value. Secondly, the temporary grid is 'walked' until a contour is found then this contour is followed and plotted until it completes a circuit or leaves the grid at the edge. An example of the output generated by the streamline visualisation mode is shown in Figure 29. The flow is the same as that used to illustrate the other two hardcopy display modes in order to provide a direct comparison.

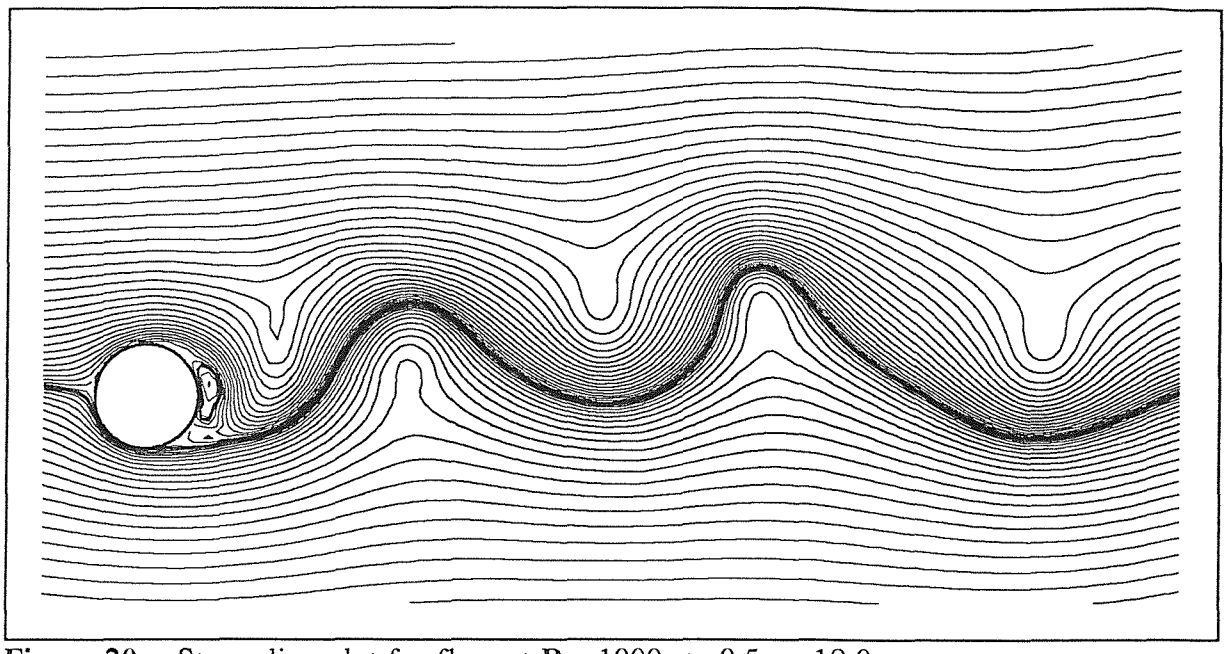


Figure 29 Streamline plot for flow at **Re**=1000, α=0.5, τ=19.0

This is the most useful of the visualisation modes as it provides a means for comparison with other research. The experimental flow visualisations we are using consist of long-exposure photographs of dust particles placed in the fluid and illuminated. These particles are travelling tangentially to the streamlines, so our visualisations should bear a close resemblance to these photographs. In addition, most authors who present results from numerical flow simulations give streamline plots as their visualisations.

Streamlines also enable detailed resolution of the flow in the important regions near the body. The distribution of contour values is clustered around zero and the streamfunction set to be zero slightly upstream of the body; the streamfunction is only defined to within an arbitrary additive constant.

4.10.4 Vorticity Plotting Display Mode

This is only available on the graphics screen. The grid is set up as explained in section 4.10.2. The vorticity within each grid cell is calculated using analytic integrals of the vorticity formulae. The vorticity is then displayed as intensities of red colour (positive) or blue colour (negative). This mode can be used simultaneously with the streamline plotting mode to give good visualisations of the velocity and vorticity fields. Vorticity contours were found to be less useful than colours due to the local noisiness of this field.

4.11 Host Services

This section briefly explains the function of the host server processor, then specifies how to actually run the code and the file formats used for specifying flow parameters and body shapes.

The host server processor has two main functions. It begins by reading, from the host disk, all the initialisation data, consisting of run parameters, a body shape and possibly a starting flow saved by a previous run. After sending all this information to the network, its function changes to that of becoming a run time file server. Information is sent to the host server at each time step and, depending on the options selected, some of this information is saved in an appropriate form at each step.

Possible hosts are those supported by the Inmos toolsets. Currently these include a range of Unix workstations, IBM-compatible PC's and IBM mainframes.

4.11.1 How to Run the Code

The code uses command line parameters to specify the input files and the display mode. The syntax for running the code is as follows (given in the Unix style of parameter passing),

```
vortex -p parameter-file -c curve-file -v flow-file
       -d display-mode -f
```

The file options specify filenames, the function and format of all files used by the code is specified in sections 4.11.2 and 4.11.3.

There are four possible display modes, specified by giving `display-mode` the following values,

- `n` No visualisation, the vortices are plotted to the graphics screen for debugging purposes, see section 4.10.1.
- `v` Velocity vector mode, see section 4.10.2.
- `s` Streamline mode (also displays vorticity on graphics screen), see section 4.10.3.
- `c` Contour mode, same as `s`, but no HPGL files are output (flow files *are* output).

The `f` option specifies that the current run is to file results. If this option is not specified, the host server terminates the host file server ('iserver') after the initialisation phase and continues to run 'disconnected'. Output is then purely via the graphics screen, leaving the host session free to edit files, for instance.

Each parameter may be specified or omitted. For any omitted parameters the code will assume sensible defaults. The default parameters are stored internally, the default curve file is called 'circle.crv', the default flow index file is 'default.idx', the default display mode is `n` and the default file mode is to run disconnected.

4.11.2 Parameter files

These files are a convenient way of passing the various run parameters to the code. It was found convenient to use files for this purpose as it allows the user to set up several different 'experiments' and then test them on different curves and using varying numbers of vortices. The generic format is given below

```
minimum.streamwise.plot.coord     maximum.streamwise.plot.coord
minimum.transverse.plot.coord     maximum.transverse.plot.coord
streamwise.gridpoints             transverse.gridpoints
number.of.streamlines
Reynolds.number          time.step                save.interval
uniform.flow             angular.velocity
random.number.seed
minimum.zone.size
```

The parameters must appear in the lines as shown, but space between them is ignored as is anything on the line after the parameters values (to allow comments). The first four parameters specify the size and position of the visualisation rectangle. The next two parameters specify the number of gridcells in each dimension, if they are not readable (eg. put some text on the line) then the maximum number of gridcells possible will be scaled into the visualisation rectangle. The next parameter is the number of streamlines to plot. The line below this contains first the Reynolds number, then the time step and then the interval (in non-dimensional time units) between saves to disk. The next line contains the value of the incident uniform flow, then the angular velocity of the body (which must be zero except for a circle). The following line contains the random number seed and the final line contains the minimum number of vortices for a zone in the decomposition (see section 4.5).

4.11.3 Curve files

These hold the coordinates of the curve defining the body boundary. The program reads this file at the start of the run and uses it to set up the panel method and vortex creation points. The curve is defined in the complex plane with 1 being the unit vector in the streamwise direction and i being the unit vector in the transverse direction. Then the curve positions and tangents are defined as follows,

let the function $C(t)$ trace a closed curve in $\mathbb{C}$, for $t \in [0,1)$
then $z_i = C(t_i)$

$$\text{and} \quad s_i = \frac{C'(t_i)}{|C'(t_i)|(t_{i+1}-t_i)} \tag{61}$$

are the discrete (complex) curve positions and tangents, assuming $0 \le t_i < t_{i+1} < 1$.

In practice, it has been found that the points should be reasonably uniformly spaced around the curve, possibly slightly closer together in regions of higher curvature. When specifying a new curve, the user should ensure that the maximum chord has unit magnitude. If this rule is not adhered to, the Reynolds number must be scaled from the value specified in the parameter file. The curve file has the format of one line for each point, in the form,

```
streamwise.position    transverse.position    streamwise.tangent    transverse.tangent
```

The number of lines of this form in the file is automatically used as the number of boundary points.

5. Results

5.1 Panel Methods

As mentioned in section 3.6, panel methods can only approximate the boundary conditions. As with any approximate method, it is important to assess the accuracy attained. Here we evaluate the panel method used in the discrete vortex code. Various inviscid visualisations will be used to demonstrate that the solutions are the expected inviscid ones. Graphs are given showing absolute errors and relative convergence using both the new singularity-free element and a traditional panel element. The new element is shown to give solutions which converge more rapidly to the correct solution as the number of elements is increased. The solutions are also shown to be smoother near the boundary.

Throughout the section, the panel methods are constructed as described in section 3.6. The new elements used are those whose velocity distribution is given by equation (31). As a reference, a second panel method is investigated using a vortex sheet of constant strength as an element, with velocity distribution given by equation (29). The constant vortex sheet element will in future be referred to as a 'textbook' element. The solutions produced using the two panel elements are compared, concentrating in particular on the boundary error. Initially, velocity vectors were plotted for this purpose but were found to convey limited amounts of information. Streamline plots were found to be much more useful as it can easily be seen if more than one streamline crosses the body surface, in which case fluid is entering or leaving the body.

5.1.1 Flow Visualisations

Figure 30 shows the solution using only ten new panels for uniform flow past a circle — the familiar inviscid solution is obtained. Figure 31 shows the equivalent solution using ten textbook panels. On a large scale, the two solutions look broadly similar, although a close examination reveals the new method to give a substantially smoother solution near the boundary. In all these diagrams the sides of the polygon shown follow the exact position of the textbook elements, but only indicate the position of the new curved elements by joining their end points. The curved elements used for the calculation shown in Figure 30 closely follow the surface of the actual circle. The introduction of curvature into the elements can be seen to help smooth the streamlines around the surface of the desired body shape, diminishing the errors due to the discrete nature of the method.

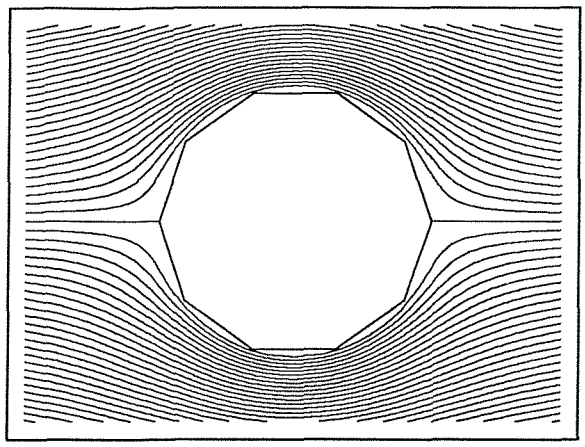


Figure 30 Flow past a circle using 10 new panels

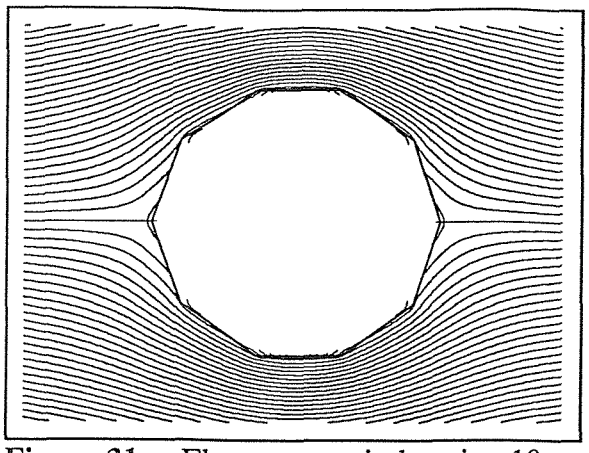


Figure 31 Flow past a circle using 10 textbook panels

Figure 32 and Figure 33 show the same solutions in more detail near a section of the circle boundary. The effect of the singularities in the textbook panels is clearly seen — many streamlines cross the boundary between the control points. The new panels maintain a smooth solution around the boundary, reducing the amount of fluid leakage. A quantitative analysis of boundary leakage is given in section 5.1.2.

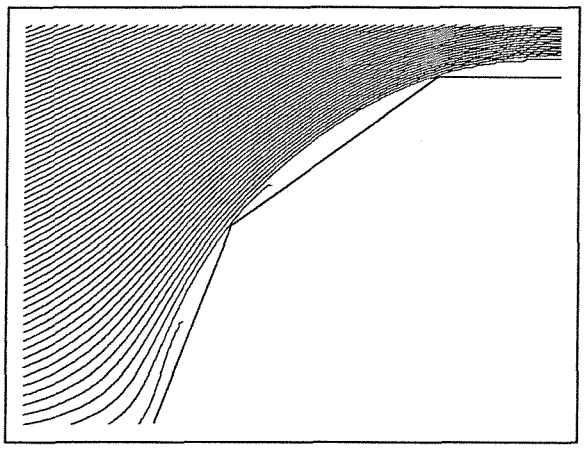


Figure 32 Flow past a circle using 10 new panels — boundary details

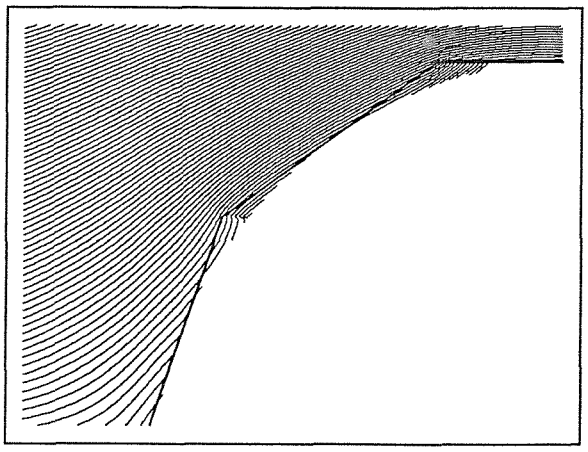


Figure 33 Flow past a circle using 10 textbook panels — boundary details

To help illustrate the ability of the method to cope with smooth, non-symmetrical bodies, Figure 34 shows flow past an ellipse with aspect ratio 2:1, placed at 45° to the incident flow. In this example fifty new panels were used to generate the solution. Again, the expected non-lifting inviscid solution is clearly obtained. Figure 35 is a solution using one hundred new panels to solve for flow past a 20:1 ellipse, placed at 90° to the incident flow. This is a difficult problem due to the high curvature of the body at the two ends — some leakage can be observed in these regions, indicated by streamlines entering the body. Such an ellipse could be used to smoothly approximate a thin plate. The leakage in regions of high curvature

is always a problem with panel methods; it can be reduced by placing a larger proportion of the panel elements in the highly curved regions.

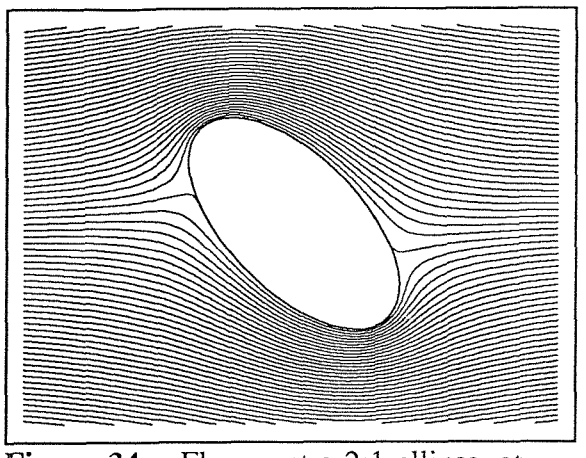


Figure 34 Flow past a 2:1 ellipse, at 45°, using 50 new panels

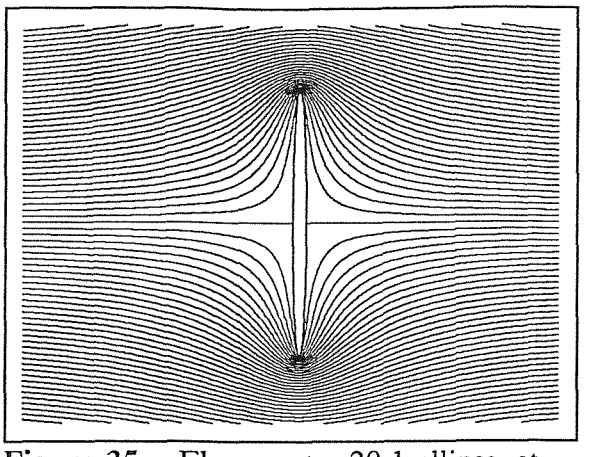


Figure 35 Flow past a 20:1 ellipse, at 90°, using 100 new panels

Finally, Figure 36 shows flow past a modified Joukowski aerofoil. The aerofoil family used are transformed from a circle as prescribed in equation (62) (taken from Paterson (1983)),

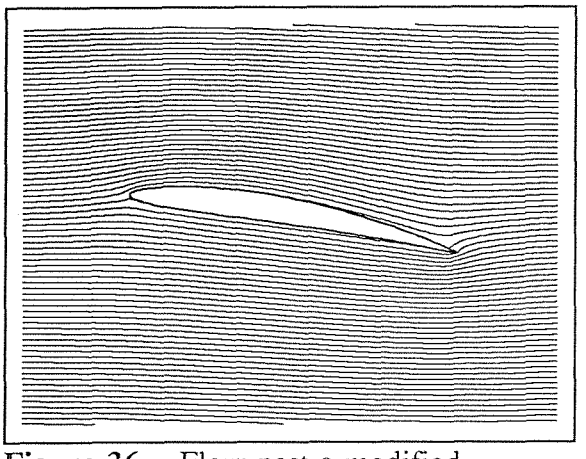


Figure 36 Flow past a modified Joukowski aerofoil using 150 new panels

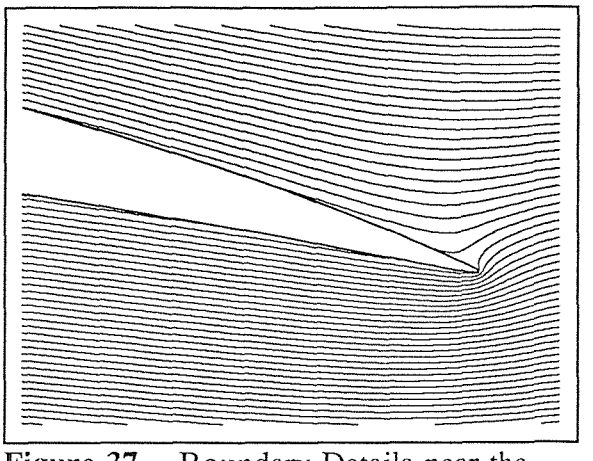


Figure 37 Boundary Details near the aerofoil tip

$$z = \delta e^{i\phi} + a e^{2\pi i t} \qquad ;0 \le t < 1$$

$$\text{where} \quad a = \sqrt{1 - 2\delta\cos\phi + \delta^2} \tag{62}$$

$$\eta = \frac{2-\epsilon}{2} \frac{(z+1)^{2-\epsilon} + (z-1)^{2-\epsilon}}{(z+1)^{2-\epsilon} - (z-1)^{2-\epsilon}}$$

This gives a whole family of aerofoils, where the thickness is proportional to δ, the angle at the rear tip is near $\varepsilon\pi$ and the shape/symmetry is affected by adjusting ϕ. The panel testing code can also calculate these aerofoils at any angle of attack to the incident flow. The aerofoil in Figure 36 uses an angle of attack of 10°, δ=0.08, ϕ=0.7 and ε=0.087 (giving a tip angle of about 5°). The solution looks accurate at a distance, although closer examination of the region around the tip, as shown in Figure 37, reveals an increased amount of boundary leakage. This problem was significantly more pronounced in earlier versions, but was improved to the current state by slightly rounding off the trailing point. This is achieved by transforming a slightly larger circle (equation (63)),

$$z = \delta e^{i\phi} + a_1 e^{2\pi i t} \qquad ;0 \le t < 1$$

$$\text{where} \quad a_1 = (1+\xi)\sqrt{1 - 2\delta\cos\phi + \delta^2} \tag{63}$$

for small ξ.

For the aerofoil shown here, the value ξ was set to 0.02. A similar rounding of the tip (using a different technique) will be used for NACA aerofoils in the vortex code (see section 5.6.1). The rounding of sharp corners is also necessary to desingularize the problem — with a genuine point, the problem tends to become numerically unstable as the number of panels is increased: the matrix becomes ill-conditioned. This is thought to be due to the two equations related to the panels either side of the corner tending towards the same panel position, but attempting to impose zero velocity in a different direction. The region of increased leakage can also be physically reduced in size by increasing the number of panels used.

The new panel method has been demonstrated to give the expected inviscid non-lifting solutions for the various types of bodies we wish to experiment upon. It is hoped that the improved boundary representation will yield more stable and accurate solutions, when carried through into the full discrete vortex calculations.

5.1.2 Examination of Boundary Leakage

In this sub-section, the details of the fluid velocity at the body surface will be quantified. The graphs given all show the magnitude of the velocity component normal to the surface contour, at a set of points on the surface obtained by traversing the body in an anti-clockwise direction. It is of key importance to include several samples between the control points, as the solution is specifically calculated to have minimum error at the control points themselves. The behaviour between the control points matters for discrete vortex methods, as vortices can travel to any position outside the body and thus their motion is affected by these errors.

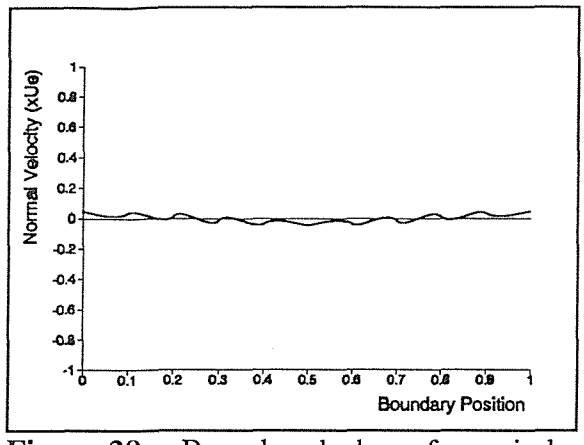


Figure 38 Boundary leakage for a circle, 10 new panels

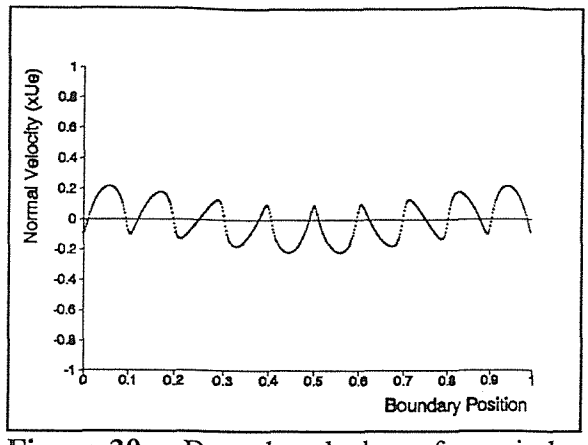


Figure 39 Boundary leakage for a circle, 10 textbook panels

Figure 38 and Figure 39 show boundary leakage for a circle using 10 new and textbook panels respectively. The horizontal axis shows boundary position in terms of a curve parameter in the interval [0,1); the vertical axis shows the normal component of velocity at the boundary, normalised with U_e, the incident uniform flow speed.

The new panels show a smooth solution which is minimised near control points, this is the best we can hope for from the limited information available; to further improve the solution between control points it would be necessary to make assumptions about the nature of the solution being sought. The equivalent scenario using textbook panels shows the effect of the singularities — large peaks in the normal velocity between the control points. In a discrete vortex method these would cause vortices to either be forced into the body or pushed violently away from it, both are non-physical and would undoubtedly have an adverse effect upon solutions.

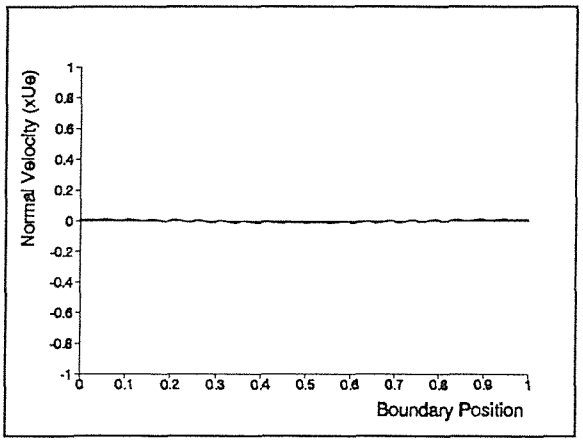


Figure 40 Boundary leakage for a circle, 20 new panels

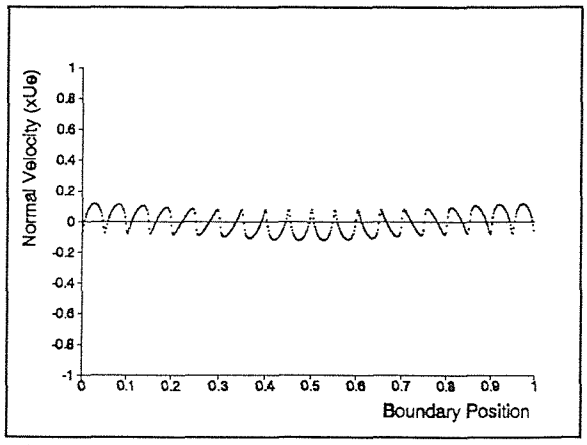


Figure 41 Boundary leakage for a circle, 20 textbook panels

Figure 40 and Figure 41 show the boundary leakage for a circle using twenty new and textbook panels respectively. Convergence of the boundary condition is illustrated in both cases. The new panels maintain a smooth solution, with the small errors between the control points reduced in magnitude and physical size. The textbook panels also show less error between control points, but the peaks of the errors remain undesirably large.

5.1.3 Convergence of the Methods

A more rigorous analysis of the convergence of the methods is now presented. There seems to be no single preferred measure of error which is predominant in the literature. The three most popular are the norms L_1, L_2 and L_∞, so results are presented using each of these. The norms are defined in terms of boundary leakage as follows :

$$L_1 = \oint_C |\underline{u}.\underline{n}|\,ds \approx \sum |\underline{u}.\underline{n}|\,\Delta s \tag{64}$$

$$L_2 = \sqrt{\oint_C |\underline{u}.\underline{n}|^2 ds} \approx \sqrt{\sum |\underline{u}.\underline{n}|^2 \Delta s} \tag{65}$$

$$L_\infty = \sup\{\,|\underline{u}.\underline{n}| \text{ on } C\} \approx \max\{\,|\underline{u}.\underline{n}| \text{ at sample points}\} \tag{66}$$

So L_1 is the average modulus of error, L_2 is the 'root mean square' of error and L_∞ is the magnitude of the maximum error.

Figure 42, Figure 43 and Figure 44 show convergence in the three norms for the problem of uniform flow past a circular cylinder, as shown in Figure 30. The graphs are plots of the values of the respective norms using various numbers of panels. The axes are labelled in real values, but are scaled using a log_{10} transformation.

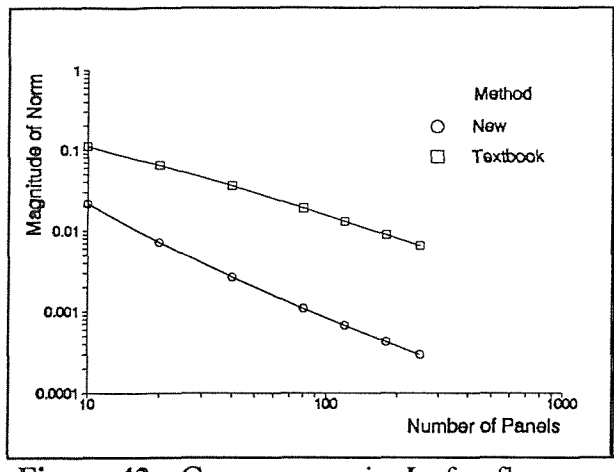


Figure 42 Convergence in L_1 for flow past a circle

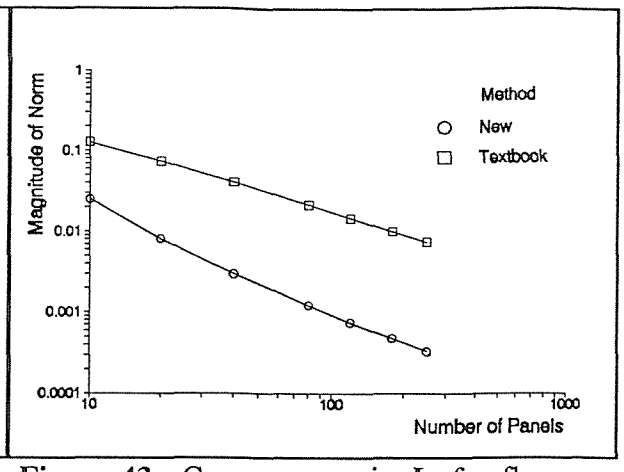


Figure 43 Convergence in L_2 for flow past a circle

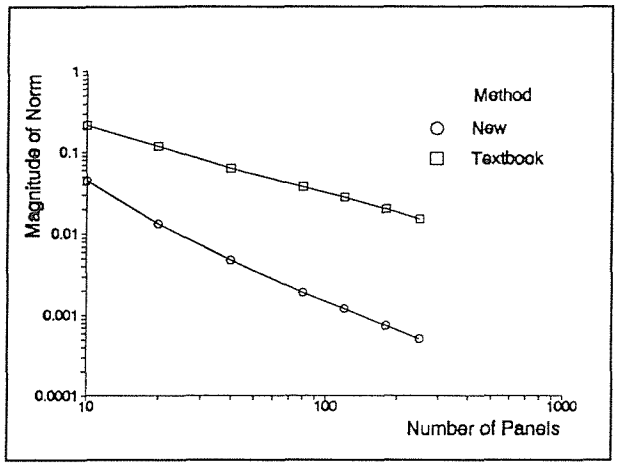


Figure 44 Convergence in L_∞ for flow past a circle

Both methods clearly converge in all three norms, since all show the error decreasing in value with increasing numbers of panels. Without rigorous analysis, it would seem that both convergence curves are asymptotic to a straight line. The axes are logarithmic, therefore this implies a power-law decay of error. Since the gradient is more negative on the asymptotic line for new panels, it is reasonable to conclude that, not only is the magnitude of the error much smaller, but it is decaying at a faster rate. To summarise this conclusion, the total error is smaller and the method converges more rapidly to the correct solution using the new panel elements.

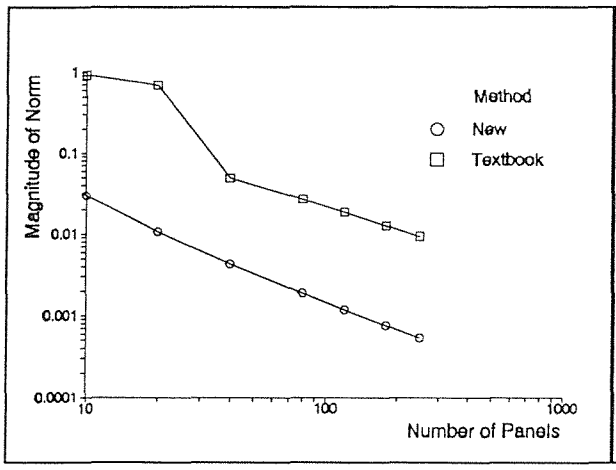


Figure 45 Convergence in L_1 for flow past a 2:1 ellipse at 45°

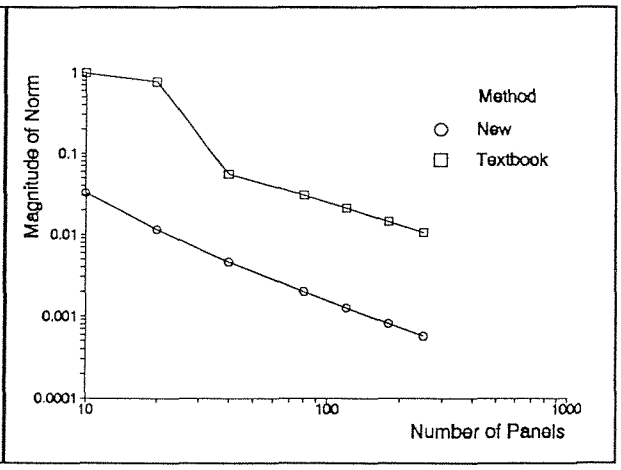


Figure 46 Convergence in L_2 for flow past a 2:1 ellipse at 45°

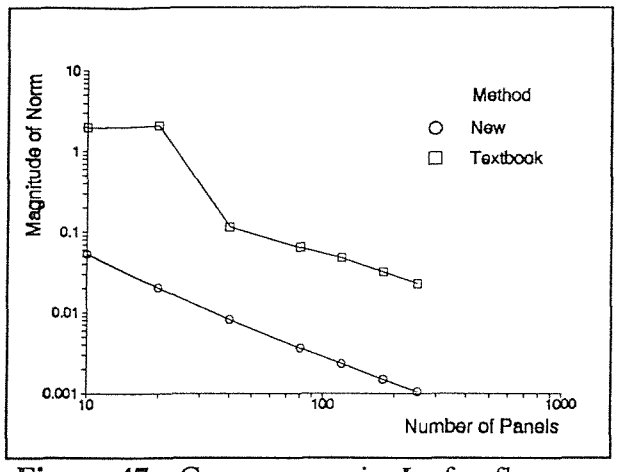


Figure 47 Convergence in L_∞ for flow past a 2:1 ellipse at 45°

Figure 45, Figure 46 and Figure 47 show convergence in the three norms for the problem of uniform flow past a 2:1 ellipse at 45° to the flow, as shown in Figure 34. Again, both methods converge in all three norms, because the error clearly decreases with increasing numbers of panels. the textbook elements show little improvement in moving from 10 to 20 panels. This is typical of the numerical inaccuracy problems which can occur using these elements. Again, the new panels show a smaller error and more rapid convergence in all three norms.

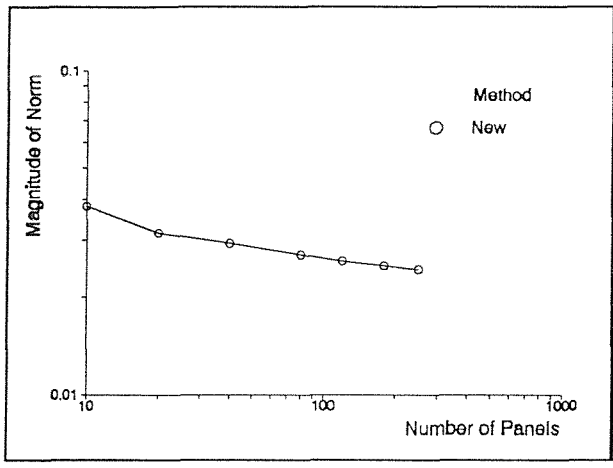


Figure 48 Convergence in L_1 for flow past a modified Joukowski aerofoil

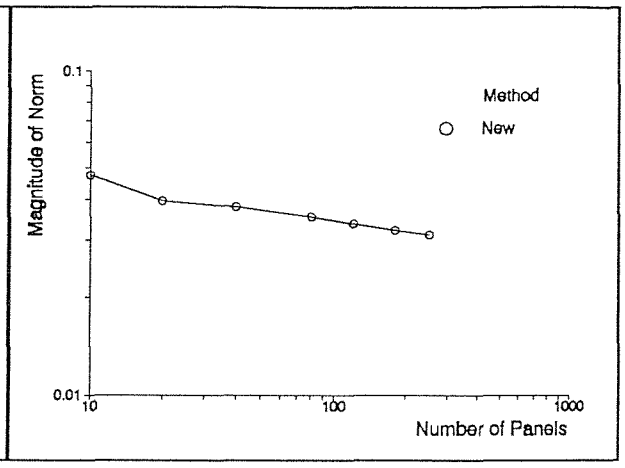


Figure 49 Convergence in L_2 for flow past a modified Joukowski aerofoil

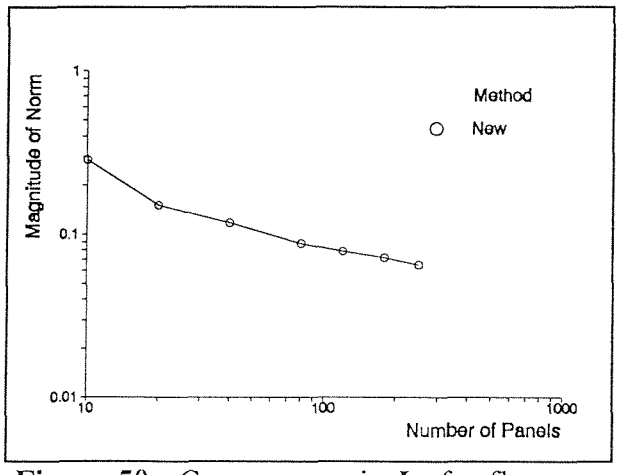


Figure 50 Convergence in L_∞ for flow past a modified Joukowski aerofoil

Figure 48, Figure 49 and Figure 50 show convergence in the three norms for uniform flow past a modified Joukowski aerofoil, as illustrated in Figure 36. The solution using textbook elements became unstable for more than 20 panel elements and so no convergence results are given. We presume that this instability is due to our positioning of panels, but did not expend much effort investigating the problem as it was not central to the work. The new panels show convergence in all three norms with increasing numbers of panels. The rate of

convergence, although slower than for the two previous smooth problems, still appears to become asymptotic to a straight line. This implies a power-law decay of error in the solution of the boundary condition.

Convergence to the correct boundary condition has been demonstrated to be more rapid for the new panels. The panels have also been shown to correctly solve the various types of aerodynamics problems which we are interested in. Considerable effort was invested in making this part of the vortex model as accurate as possible. It is hoped that these panel elements may find wider uses in the field of aerodynamics. Panel methods are an extremely useful tool and, since the matrix inversion is the most expensive part of these calculations, using fewer elements to achieve a certain accuracy has a large payback.

5.2 Testing Numerical Solutions of the Discrete Vortex Code

To test the solutions produced by the code, we chose to use the well-studied problem of impulsively started flow past a circular cylinder. The non-rotating and rotating cases are both considered, for a variety of Reynolds numbers corresponding to those for which reliable experimental or numerical data are available. Our primary method of evaluation is by qualitative and quantitative comparison of flow visualisation diagrams with comparison of force coefficients exerted by the fluid on the cylinder providing a check on results.

Comparison of flow visualisations is non-trivial, since the amount of information contained in, for example, a streamline plot is extremely large. The usual method for comparison is to check that each key flow feature is present and then to compare the dimensions of these features. For each case here, the visualisations will be presented and then a discussion of the successes and failings of the numerical schemes, as compared to the experiments, will be given.

Force coefficients are extremely valuable aerodynamic results, but are of limited use for verification of results. The problem is that, although a poor result for force coefficients may highlight a source of error, a good result for these quantities alone is not sufficient to verify the method. Calculating a force coefficient in a numerical simulation always involves a numerical integration of the more basic variables available, hence there is scope for so called 'fortuitous' results. A force figure that appears to be accurate can be derived from a subtle cancellation of errors in the integration process, a dangerous effect to endorse. Conversely, it is possible for an accurately calculated flow to give a poor force result, due to errors in the calculation of the force. Thus we use force coefficients only as a check on results.

The diameter of the cylinder, D, is used to non-dimensionalise distances and time, τ, and to define the Reynolds number, Re,

$$Re = \frac{U_e D}{\nu}, \qquad \tau = \frac{t U_e}{D} \quad . \tag{67}$$

This is the same non-dimensional form as is used in the majority of the experimental papers used here. Some papers use the cylinder radius as the characteristic length, where possible these results have been adjusted so that all units are given in the same form.

For the rotating cylinder, the ratio of the rotational surface speed to the incident flow, α, is used as the extra non-dimensional parameter to categorise the flows. If ω is the angular velocity of the cylinder,

$$\alpha = \frac{\omega D}{2 U_e} \quad . \tag{68}$$

In the following discussions of visualisations, the word 'eddy' is used to describe a region of closed streamlines in a frame of reference fixed with respect to the body. For a circular cylinder, the 'main' eddies are the two largest eddies which initially form in the wake of the body. The 'secondary' eddies are the two counter-rotating eddies which form within the main eddies near their outer edges and at the body surface. The 'tertiary' eddies also follow this pattern, and are formed within the secondary eddies. The word vortex is used to describe a region of highly rotational fluid, but not necessarily with closed streamlines, such as occurs when an eddy is shed into the main flow.

5.3 Convergence of the Method — an Experimental Investigation

The formal convergence of the method to solutions of the Navier-Stokes equations has been proved, as discussed in section 3.5. However, the convergence proofs in the literature are for unbounded domains, and thus do not allow for the inclusion of boundary models. We must therefore perform experimental testing on solutions. We have conducted an extensive programme of testing using the example of impulsively started flow past a circular cylinder at Reynolds number 5000. Flow visualisations were compared to experimental visualisations using a variety of time steps and numbers of boundary points to represent the body. The code is set up so that increasing the number of boundary points also introduces more discrete vortices, reduces the core size and causes more vortex overlap — all of the convergence conditions needed for infinite fluids (see section 3.5.1). A summary of the results of the

investigation is presented here and a discussion is given on the effects of varying the time step and the number of boundary panels used.

Figure 51 - Figure 55 show the computed streamline plots for τ=1.0, using a time step of 0.1. Figure 56 shows an experimental visualisation of the same situation, from Bouard and Coutanceau (1980). This sequence is presented to show the effect of increasing the number of boundary points used whilst keeping the time step fixed.

The trend is generally towards the correct qualitative flow features with increasing number of boundary points. Figure 56 shows that the experimental flow is approximately symmetrical above and below the cylinder. On the upper side, a main eddy sits at the rear of the cylinder, with an enlarged central rotating section. Towards the top of the cylinder, two small counter-rotating secondary eddies can be seen. Between the core of the main eddy and the line of symmetry is a narrow section of wake, termed the 'forewake' by Bouard and Coutanceau.

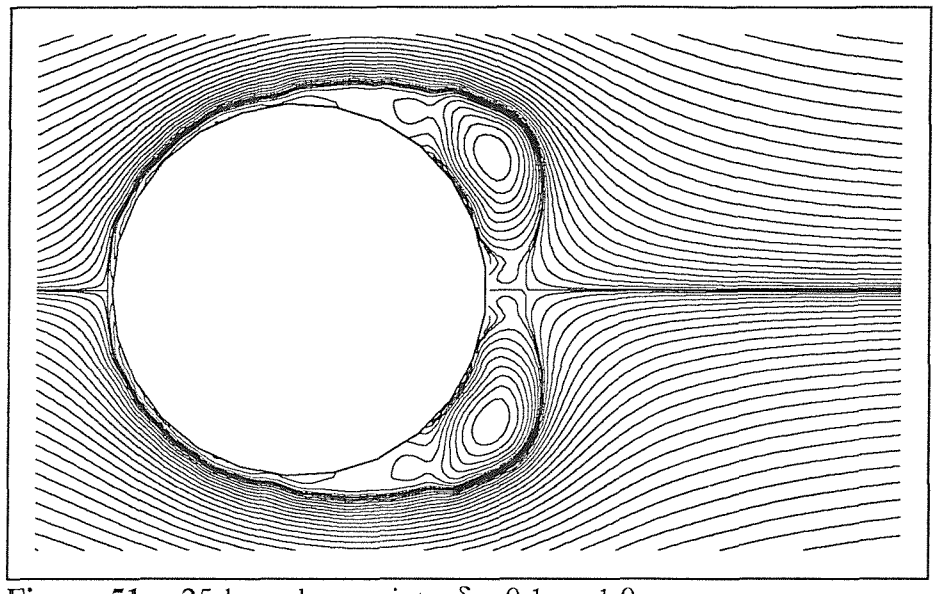


Figure 51 25 boundary points, $\delta\tau$=0.1, τ=1.0

Figure 51 uses 25 boundary points and, as would be expected, leads to a very crude result. However, it is pleasing that all the key flow features described above are present in this result, but the dimensions are incorrect. Notice that there is a fairly large amount of fluid leaking through the boundary in this flow, indicating that more panels are needed to satisfy the boundary condition. Notice also that the layer of fluid attached to the front of the cylinder, the 'boundary layer', is far too thick. This is a result of the vortex cores being too large to be capable of representing the boundary at the correct thickness.

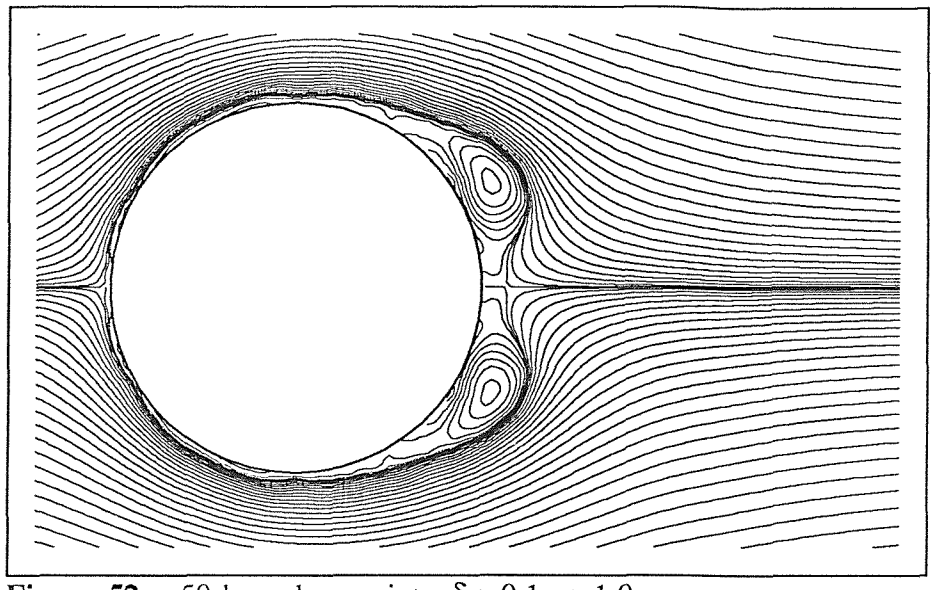


Figure 52 50 boundary points, $\delta\tau$=0.1, τ=1.0

In Figure 52, which uses 50 boundary points, the scales of the flow features become close to those seen in the experiment, but the main eddies are too close to each other and the secondary eddies have not developed properly. Some boundary leakage is still visible using 50 panels and the boundary layer is still too thick at the front of the cylinder.

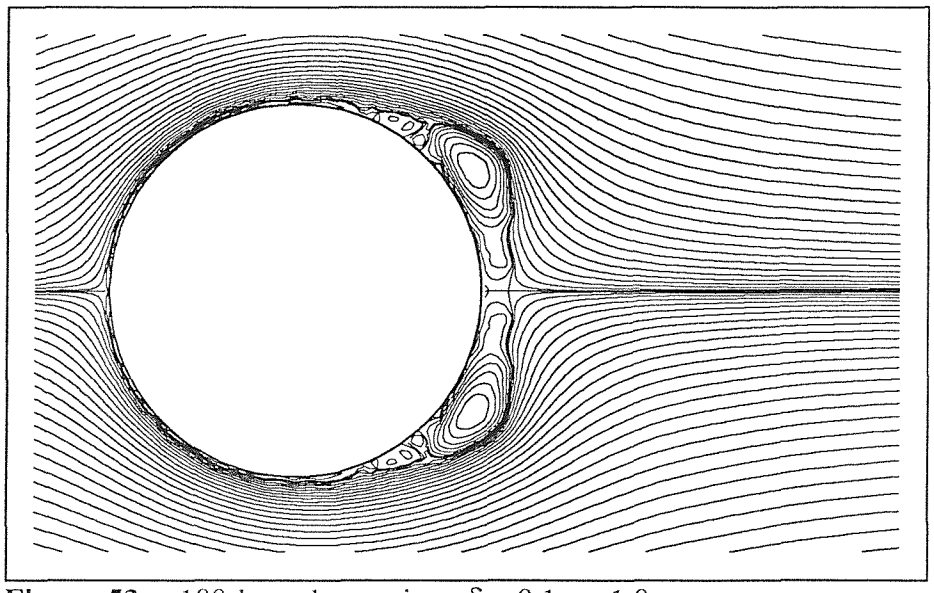


Figure 53 100 boundary points, $\delta\tau$=0.1, τ=1.0

Figure 53 shows the solution obtained using 100 boundary points. The dimensions and positions of the key flow features are very close to those seen in the experiment. An interesting new phenomenon has occurred in this visualisation — the secondary eddies that rotate in the opposite direction to the main eddy have both joined up with the external flow. This phenomenon is seen in several of our numerical solutions at high Reynolds numbers and does occur in real flows (see Figure 93). However, the experiment shows this secondary eddy

as closed with the body boundary in this case, so here the phenomenon is predicted wrongly. However, the clear presence of the secondary eddies and the forewake is of merit. Notice that the leakage of fluid across the boundary has been almost entirely eliminated by using 100 panels. The boundary layer is still just visible, shown by the layer of streamlines close to the front surface of the cylinder.

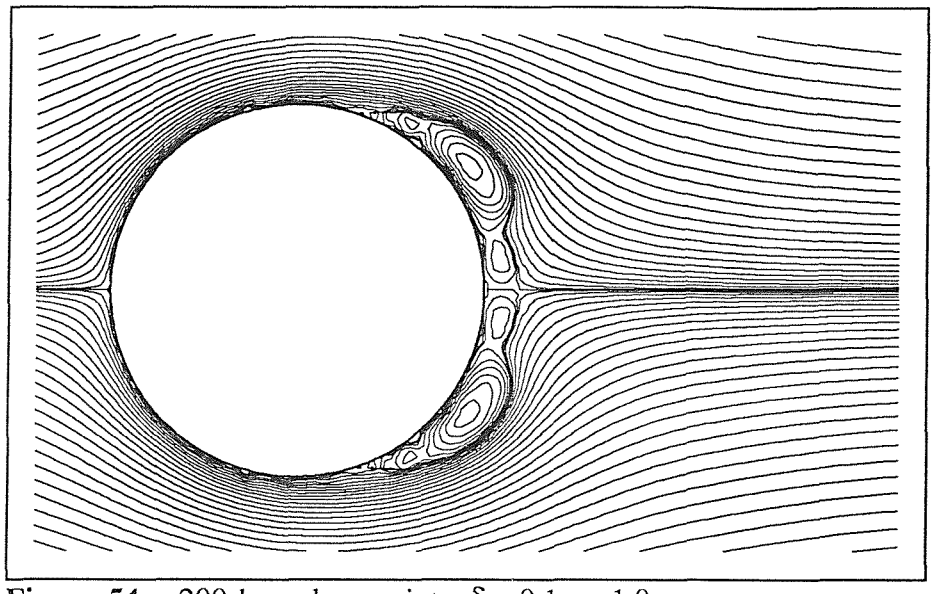


Figure 54 200 boundary points, $\delta\tau$=0.1, τ=1.0

Moving to Figure 54, which uses 200 boundary panels, the numerical solution is very close to the experiment. The secondary eddies are clearly visible, as is the forewake. The main eddy looks slightly too flat towards the boundary, but almost no boundary leakage is visible and the boundary layer is now thin enough not to be visible on the front cylinder surface.

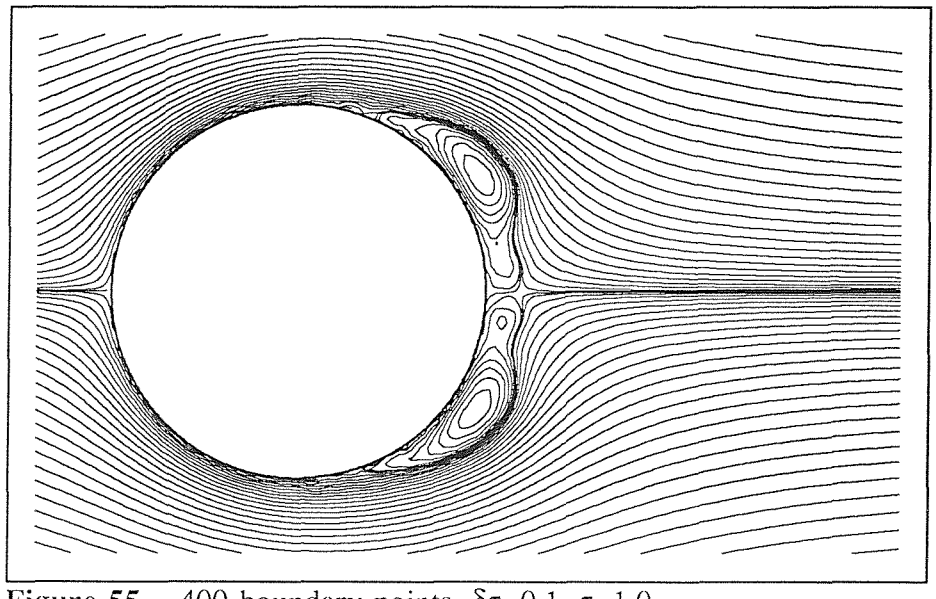


Figure 55 400 boundary points, $\delta\tau$=0.1, τ=1.0

Figure 55 shows the results of the same simulation using 400 boundary points, the maximum figure possible given the current constraints on memory and CPU time. The result is similar to that obtained with 200 boundary points, the dimensions of the flow features are nearly the same. The core of the main eddy has become slightly larger, and the details of the forewake appear to be slightly better resolved. However, details of the secondary eddy system appear to have been lost. The overall impression is that this solution represents no improvement, or is possibly a worse prediction than the previous case.

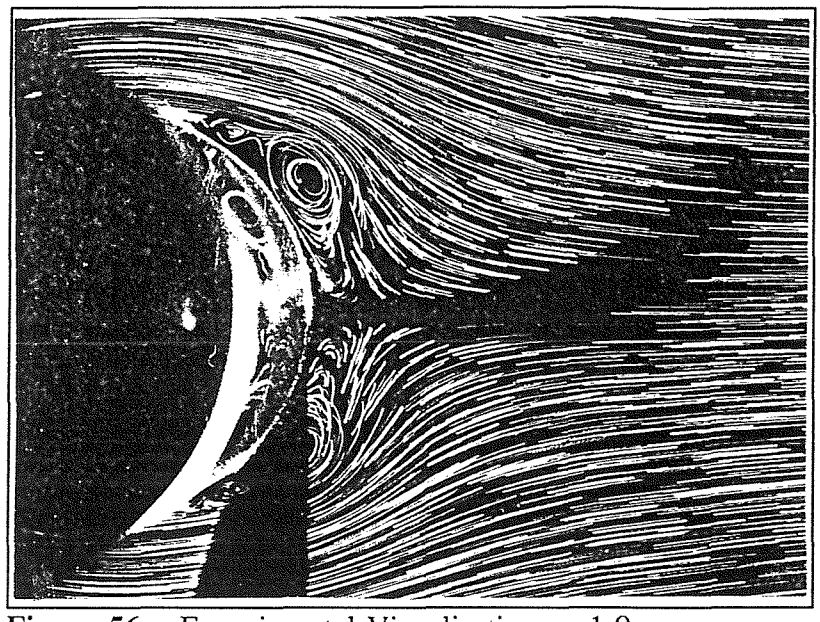


Figure 56 Experimental Visualisation, τ=1.0

Figure 56 shows the experimental visualisation of the same flow at the same non-dimensional time as the numerical visualisations above. The photograph is produced by taking a long exposure of illuminated dust particles suspended in the fluid in order to highlight the particle paths, which approximate the streamlines of the flow. The photograph must be studied carefully in regions where the paths cross in order to ascertain the behaviour of the fluid there. In this case, several paths cross just downstream from the forewake; this may indicate a rapid expansion of the main eddies into this region.

The trend of the solutions is towards the experimental flow, except that the solution using 400 boundary points and $\delta\tau$=0.1 fails to resolve the secondary eddy system with the same accuracy as that using 200 boundary points. This unexpected effect led us to conduct a broad investigation of the results of the discrete vortex model for this flow using a range of time steps and numbers of boundary points. The results of this survey are summarised below.

It is essential to use a sufficient number of boundary points that the vortex core size decreases sufficiently to accurately represent the boundary layer. If the vortex cores are larger than the desired boundary layer thickness, an accurate solution cannot be achieved for any

value of time step. The observed effect of large cores is that an artificially wide boundary layer develops, preventing important flow features near the body from being fully resolved.

There is a startup phenomenon which is dependent on the number of boundary points and the time step used. This phenomenon affects the rate at which the early flow develops, causing the formation of major flow features to be brought forward in time or delayed. Thus the non-dimensional times at which the discrete vortex plots closely match experiments are 'shifted'. This phenomenon is illustrated by Figure 57, which shows the flow generated using 400 boundary points and $\delta\tau$=0.0125, but at the later time of τ=1.4. Thus the comparison time has been shifted by 0.4 non-dimensional time units. This visualisation is in extremely close agreement with the experiment. The visualisations at later shifted-times are also in agreement with experiments[6], which indicates that the times are *shifted* and not scaled. Thus a shift-time must be added to obtain equivalent experimental times, and *not* a multiplicative constant introduced.

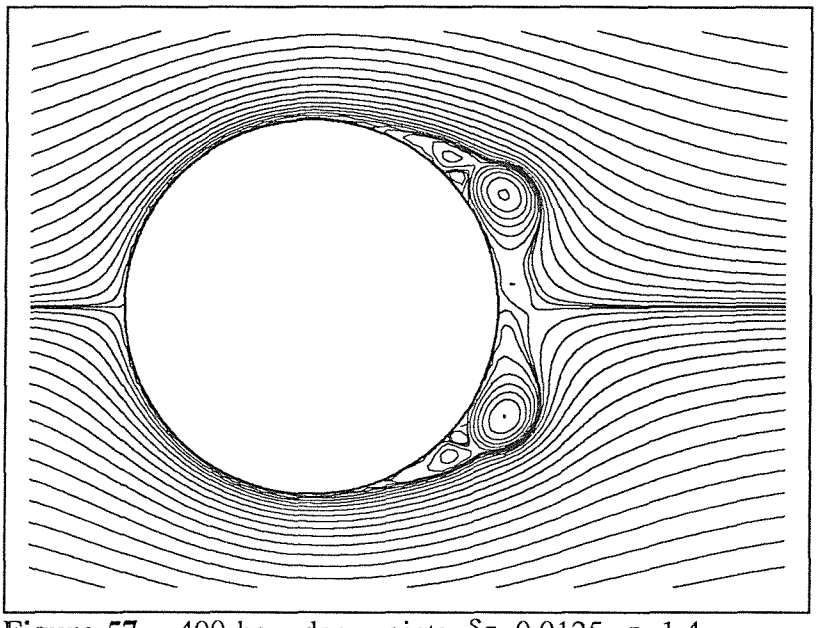


Figure 57 400 boundary points, $\delta\tau$=0.0125, τ=1.4

A further example of this time-shifting is given in Figure 58, where the equivalent run using a time step of 0.025 is shown. The visualisation at τ=1.3 is in close agreement with the experiment, as are the visualisations thereafter whose non-dimensional times of computation are 0.3 units later than the experimental photographs.

[6] apart from occurrences of the instability described below

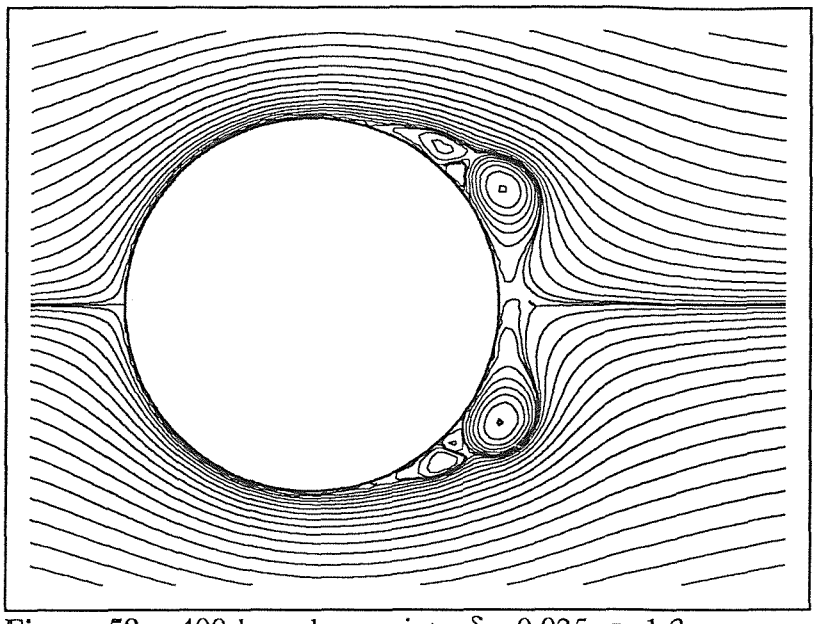


Figure 58 400 boundary points, $\delta\tau$=0.025, τ=1.3

A pattern emerges which shows that using more boundary points, or a smaller time step, delays the initial formation of the major flow features. An analogous situation exists in the experiments, which are not genuinely impulsively started and thus are initiated by a (usually undocumented) acceleration phase.

Allowing for these comparison shifts, results are generally closer to experiments using a shorter time step. However, short time steps can also cause instability after longer times (these instabilities have been seen from τ=2.0 onwards). A disturbance appears to develop in the shear layer, which eventually causes the solution to enter a completely different phase from the experiments. This process is illustrated in Figure 59 and Figure 60, which are two frames at later times from the same run which produced Figure 57.

Strong secondary eddies develop from τ=2.0. At τ=2.8, these eddies are in the process of joining the main wake, destabilising it and causing symmetry to break. These secondary eddies do not become fully entrained by the main wake as seen in most of the other computations at this Reynolds number. Instead, they travel around the core of the main wake, visible as sharply angular deviations in the streamlines (see Figure 60). The wake becomes destabilised and eddy shedding commences earlier than in computations using a larger time step.

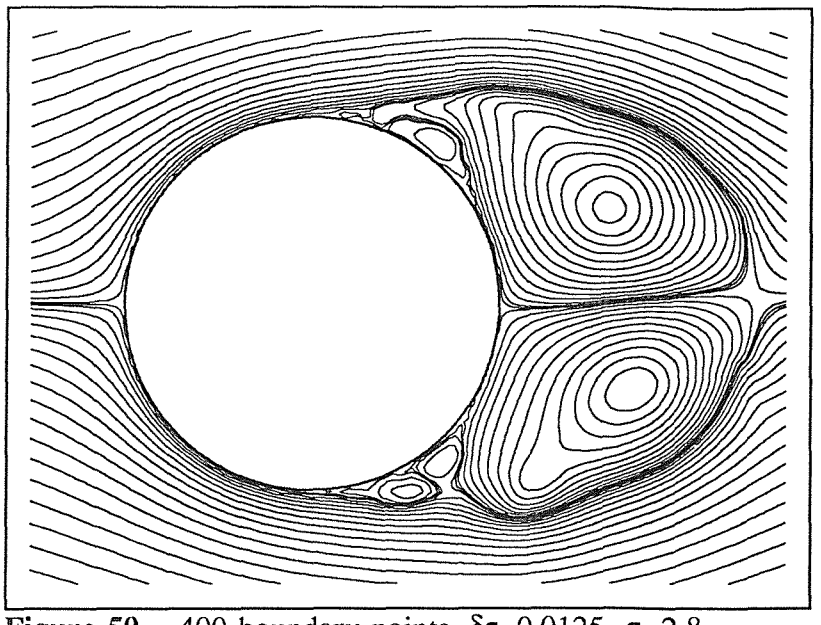


Figure 59 400 boundary points, $\delta\tau$=0.0125, τ=2.8

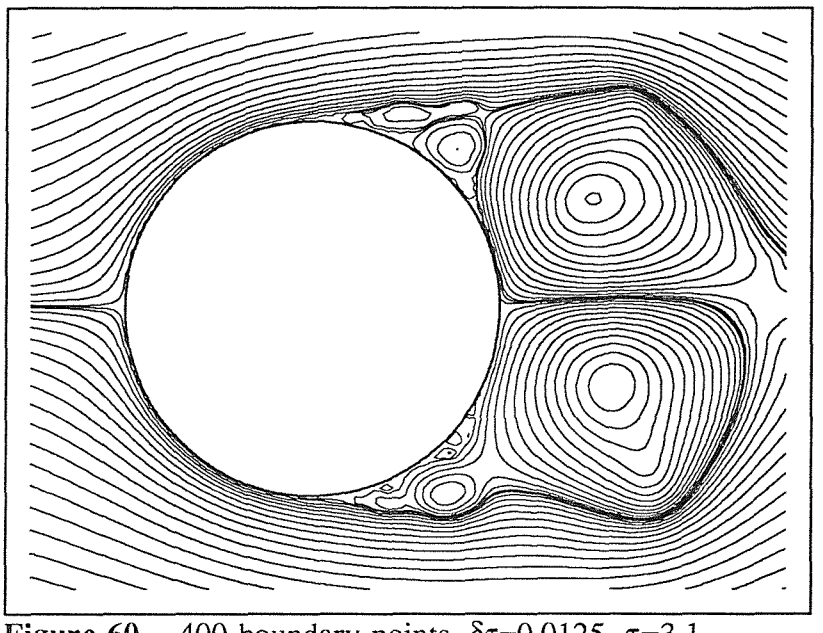


Figure 60 400 boundary points, $\delta\tau$=0.0125, τ=3.1

One possible cause of this instability is the error due to the transition of the vortex sheets to vortex blobs in the creation of new vortices. This error is discussed in section 3.8, and will become large when using a very small time step. The velocity distribution of a blob is similar to that of a sheet at distance, but close-up they differ significantly. Using a very small time step will leave the sheet and blob close together, it is then possible that the error in transforming sheets into blobs will have a more significant input to the panel method on the next time step than the actual motion of the blobs. Thus a feedback mechanism could be set up leading to instability. Small time steps have been used to give accurate long term solutions for the rotating cylinder, where the surface speeds are higher than in the non-rotating case.

It may be that the high surface speeds prevent the instability developing by transporting the blobs sufficiently far from the sheets, even with a small time step.

Through all this complicated behaviour, there is a significant trend towards solutions matching experiments more closely when more boundary panels are used. The complex startup phase discussed above can cause early time solutions to appear less accurate, but the extra spatial resolution significantly improves long time solutions.

For the accurate short time solutions presented in section 5.4.1, we have used 400 boundary points and a time step of 0.1. This produced little time-shift, which allows us to compare experiments with computations at the same non-dimensional times. We also chose the time steps for the rotating cylinders in section 5.5 so as to produce small time-shifts, however the values of time step required seemed to become smaller with an increasing rotation rate.

For the long term solutions the choice of parameters was made pragmatically. Runs were conducted with the least number of boundary points and the largest time step which produced valid solutions, to allow the solutions to be taken as far in time as possible. These long runs were found to be qualitatively insensitive to variation of parameters. The same types of behaviour occur, but not always at the same non-dimensional times. In practice this meant using either 100 or 200 boundary points, with a time step of 0.05 or 0.1.

The differences generated by changing the time step may be related to the boundary model we have used. Future studies could significantly benefit from an examination of the details of the boundary model. However, we have shown that excellent experimental comparisons can be produced with careful choice of the input parameters. There are also cruder parameter values which always produce results with reasonable accuracy, and we have tended to favour these.

5.4 Test Case 1 — Translating Circular Cylinder

5.4.1 Short Time, High Accuracy Solutions — Setup

Bouard and Coutanceau (1980) give experimental flow visualisations for impulsively started flow past a circular cylinder, for a range of Reynolds numbers and short times. In addition, Ta Phuoc Loc and Bouard (1985) give details of the evolution, in both experimental and numerical visualisations, of the flows at Reynolds number 3000 and 9500. In this section, the output of the discrete vortex code will be compared with data from these papers. The method will be shown to give solutions of comparable accuracy to other numerical methods

for the range of Reynolds numbers 300 to 9500. Tests indicate that solutions may have some validity for Reynolds numbers as high as 31,700 — for a discussion see section 5.4.11. As mentioned above, the following runs were conducted (unless otherwise stated) using 400 boundary points and a time step of 0.1.

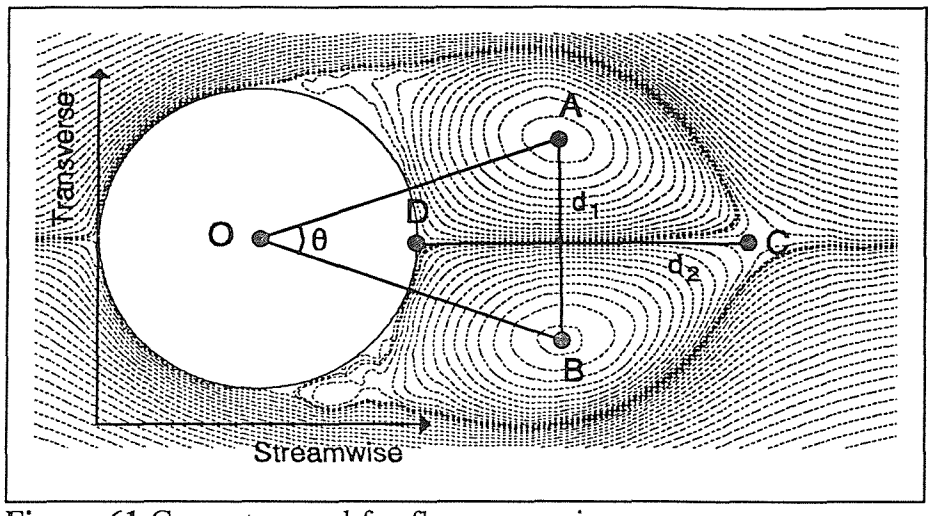


Figure 61 Geometry used for flow comparison

Figure 61 illustrates the geometry used for comparison of flow visualisations. The flow streamlines are shown as dotted lines; the positions of the points used to measure characteristic distances and angles are shown as small shaded circles. The three quantities compared are d_1=**AB**, the separation distance of the centres of the two main eddies, d_2=**DC**, the distance between the two stagnation points at the rear of the cylinder on the line of symmetry and θ=∠**AOB**, the angle subtended by the centres of the main eddies from the centre of the cylinder.

These three quantities were chosen as important physical flow measures of the solution, as they measure the length, width and position of the main eddies. This effectively measures the size of the main wake behind the cylinder. The quantities are also relatively easy to measure, although it can be difficult to estimate the position of the surface and the centre of the cylinder in some of the experimental photographs. The figures for the numerical solutions should be reasonably accurate as the locations of the points are usually clear. The three quantities have the additional advantage that they do not assume the flow to be symmetrical.

5.4.2 Short Time Solution - Re=300

Figure 62 shows the computed streamlines for the flow at τ=2.5, Reynolds number 300. Figure 63 shows the experimental visualisation of exactly the same flow. In the calculated flow, some 'noise' can be seen in the region of the separation points. This noise is observed in all the solutions at the lower range of valid Reynolds numbers, indeed it is this effect

which prevents the method solving accurately for even lower values of Reynolds number. The noise is caused by the large diffusion gradients which occur in certain regions of the flow. The solution shown here uses the combined diffusion velocity/random walk technique discussed in section 3.5. Hence, a large diffusion gradient in a region near the boundary is modelled by assigning a large diffusion velocity to the vortices in that region. This implies a rapid motion of these vortices, causing large time-stepping errors and causing the vortex field to become relatively fragmented. These combined effects increase the magnitude of errors in such regions of high diffusion, visible as noise in the streamline plot and force coefficient figures. A similar, but larger, effect is observed in pure random walk calculations where the standard deviation of the random walk becomes increased to the point where the noise is visible in solutions.

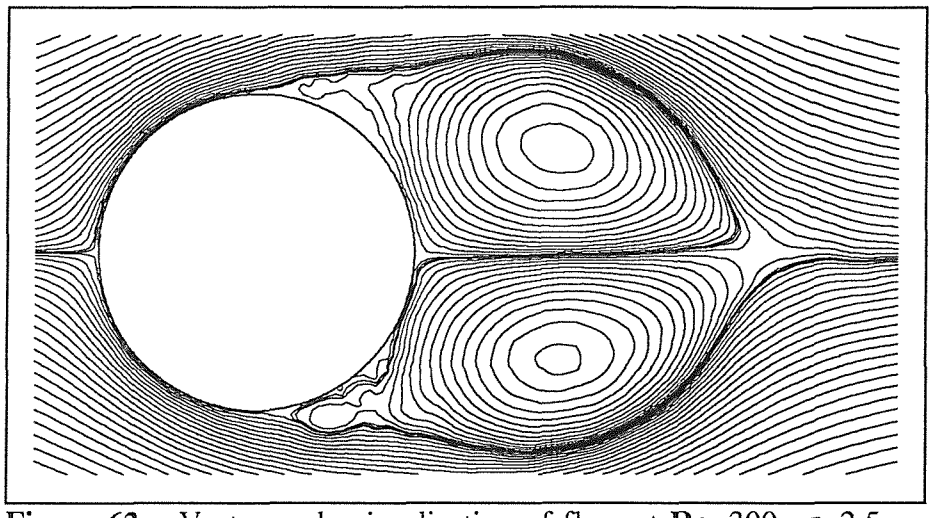


Figure 62 Vortex code visualisation of flow at **Re**=300, τ=2.5

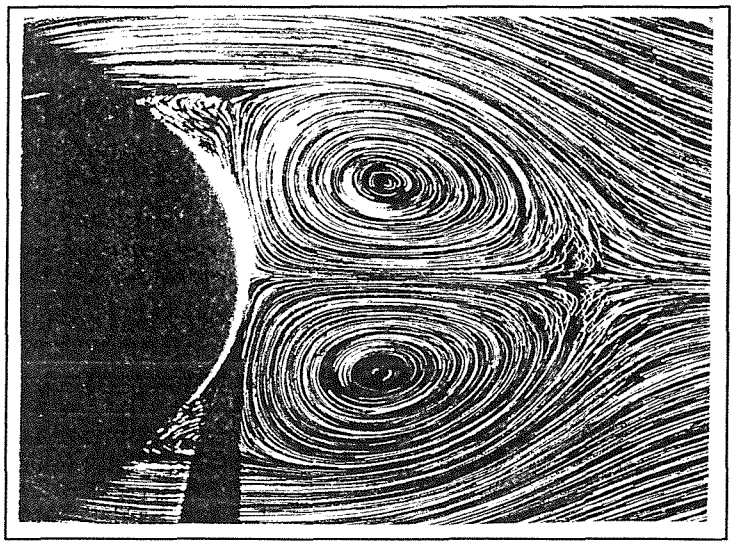


Figure 63 Experimental visualisation of flow at **Re**=300, τ=2.5

Comparing overall flow characteristics, the gross features are fairly well predicted by the computation. The secondary eddies are not fully developed in both visualisations — showing only as slight deviations in the streamlines near the body. It is also worth noting that neither flow is perfectly symmetrical.

The distance comparison table is given below, each of the measured quantities agrees with the experiment to within 10%.

	Figure 62	Figure 63
d_1	0.53	0.54
d_2	1.04	0.94
θ	31.1°	33.9°

5.4.3 Short Time Solution - Re=550

Figure 64 shows the computed flow for Reynolds number 550, at τ=2.5, with the equivalent experimental visualisation shown in Figure 65. The noise discussed above can be seen in the region of the separation points, although at a reduced level. The major flow features are well predicted. The secondary eddies are fully developed into closed regions against the cylinder surface and are clearly not joined to the external flow. In the calculation, details of the upper secondary eddy system are much closer to the experimental observations. The shape of the main eddies is accurately predicted.

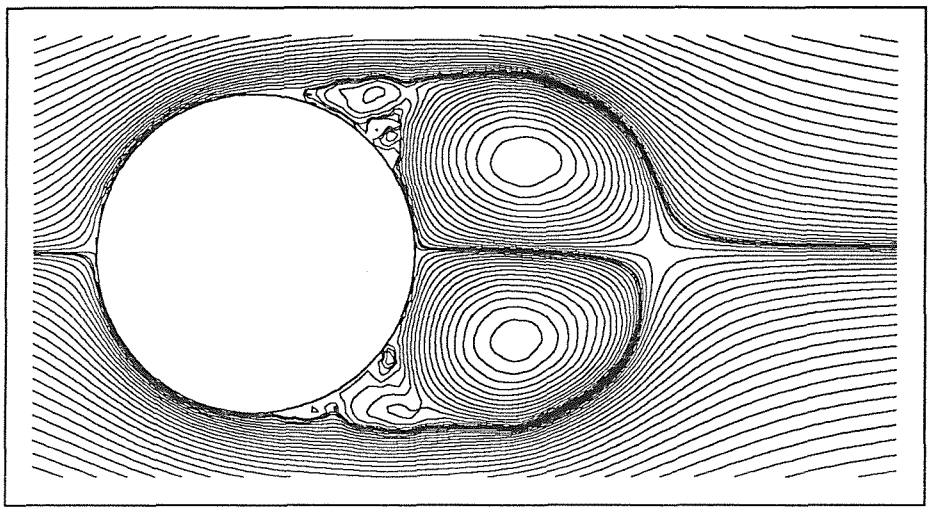


Figure 64 Vortex code visualisation of flow at **Re**=550, τ=2.5

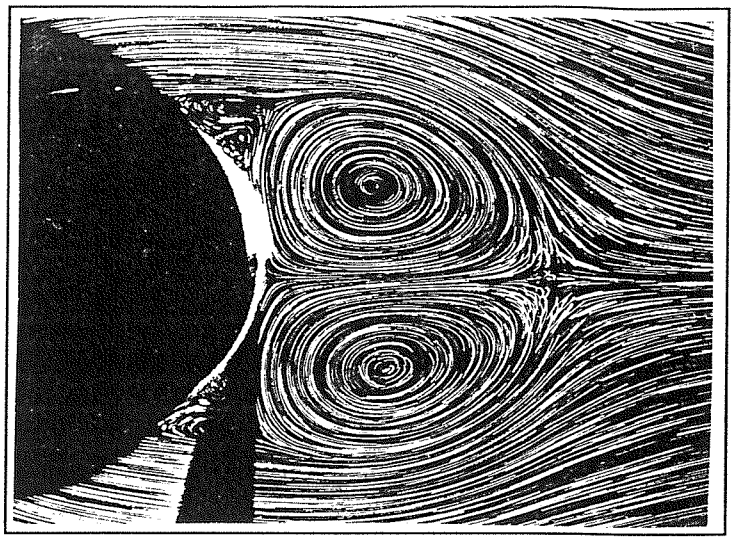


Figure 65 Experimental visualisation for flow at **Re**=550, τ=2.5

The geometry comparison table for the above flows is given below. Agreement is good, although the angle θ is noticeably too large, indicating that the centres of the main eddies are too close to the cylinder surface (if θ is too large but d_1 is near the experimental value, then the eddies must be too close to the cylinder).

	Figure 64	Figure 65
d_1	0.56	0.52
d_2	0.74	0.80
θ	38.0°	33.7°

5.4.4 Short Time Solution - Re=3000

Figure 66 - Figure 77 show the evolution of the flow at Reynolds number 3000, from τ=1.0 to τ=2.5. For each time, a comparison is made between visualisations produced by the vortex code, an experiment and a finite difference computation, the latter two are both taken from Ta Phuoc Loc and Bouard (1985). A geometry comparison table is also given for each time. Note that the centres of the eddies are not clearly indicated in the finite difference plots, so the entries in the table concerning the position of these centres must be left blank.

Figure 66 shows a streamline plot produced by the vortex code at τ=1.0. Figure 67 and Figure 68 show visualisations from an experiment and a finite difference calculation respectively. The scale of the main eddies is predicted accurately by both numerical solutions.

Details of the secondary eddies are not apparent in the vortex code solution, this was also observed in section 5.3 using 400 boundary panels and may be due to a small time-shift.

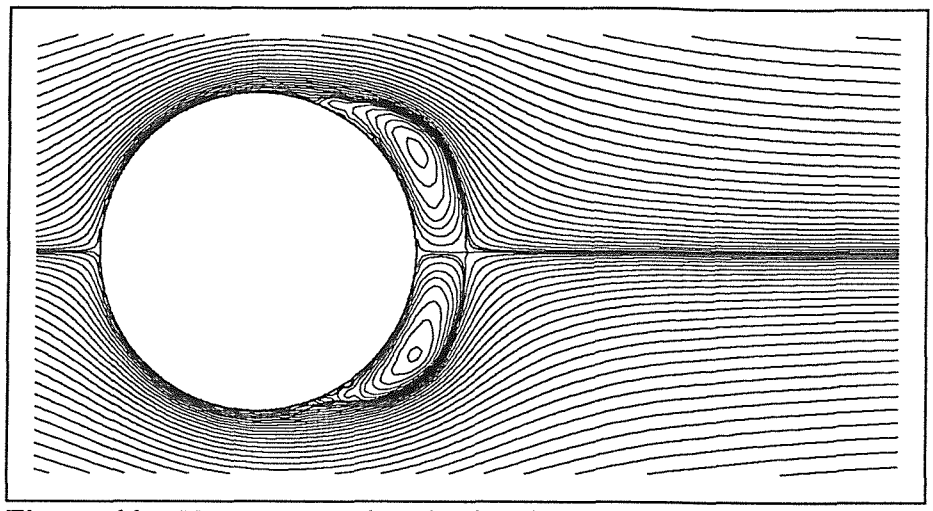


Figure 66 Vortex code visualisation for flow at **Re**=3000, τ=1.0

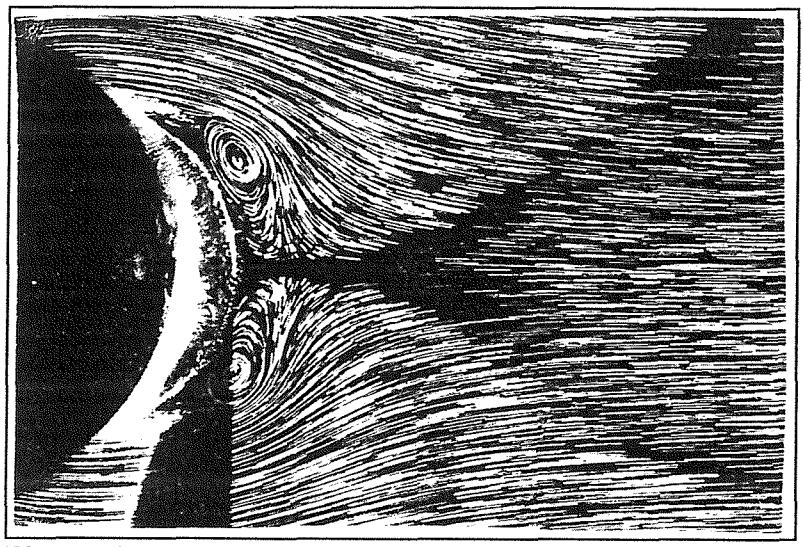


Figure 67 Experimental visualisation for flow at **Re**=3000, τ=1.0

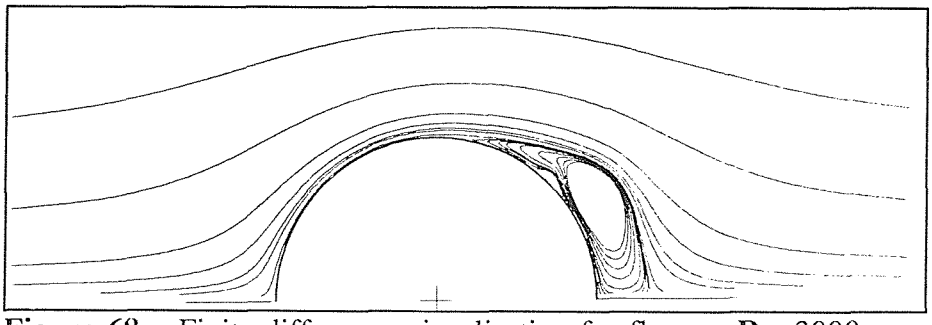


Figure 68 Finite difference visualisation for flow at **Re**=3000, τ=1.0

The table below gives a quantitative comparison of the flow geometries. The figures show a close correspondence between both of the numerical solutions and the experiment, for the quantities considered.

	Figure 66	Figure 67	Figure 68
d_1	0.64	0.65	-
d_2	0.15	0.15	0.17
θ	66.8°	65.4°	-

Figure 69 shows the vortex code visualisation for τ=1.5, the equivalent experiment is shown in Figure 70 and the finite difference solution is shown in Figure 71. Both numerical results closely match the experimental visualisation. The secondary eddy system has now formed in the vortex code solution and has a size and shape close to that seen in the experiment.

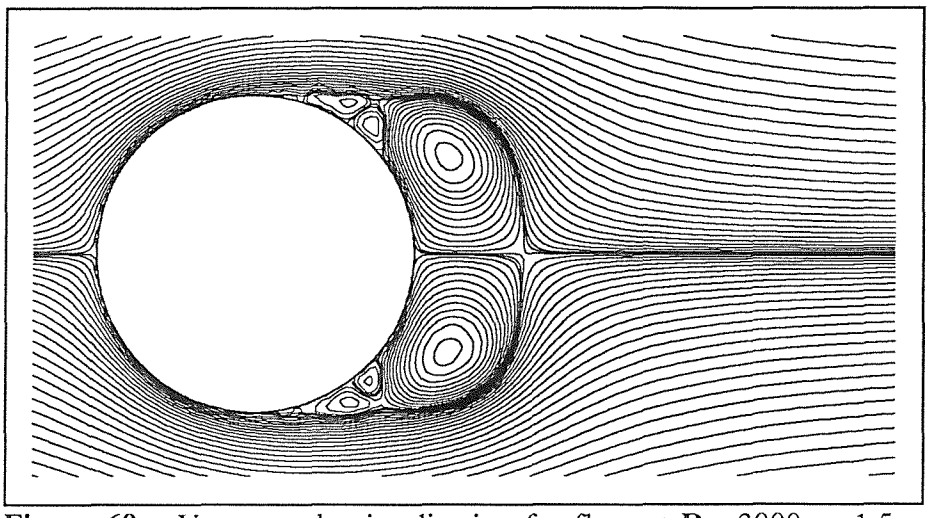


Figure 69 Vortex code visualisation for flow at **Re**=3000, τ=1.5

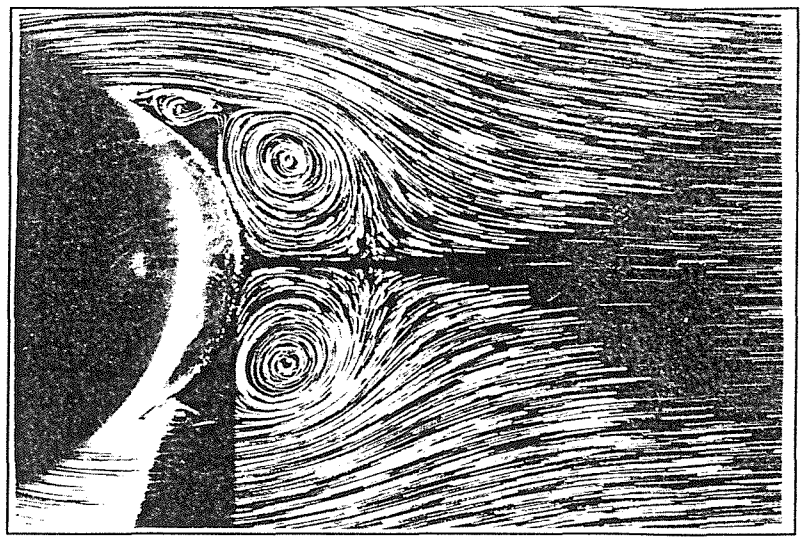


Figure 70 Experimental visualisation for flow at **Re**=3000, τ=1.5

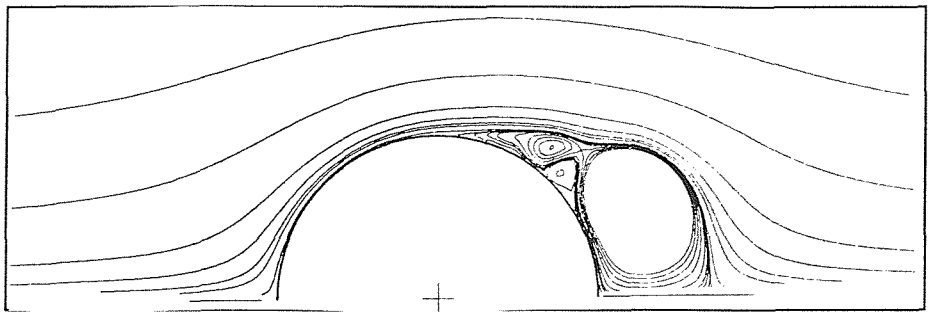


Figure 71 Finite difference visualisation for flow at **Re**=3000, τ=1.5

The geometry comparison table for the above flows is given below. All measurements are close to the experimental results. Both numerical solutions predict the wake to be too short, the vortex code also predicts the centres of the main eddies to be too close to the cylinder surface.

	Figure 69	Figure 70	Figure 71
d_1	0.62	0.61	-
d_2	0.33	0.38	0.34
θ	55.3°	50.5°	-

Figure 72, Figure 73 and Figure 74 show the same triplet of visualisations for τ=2.0. Both numerical solutions are close to the experiment. The wake has lengthened and the secondary eddy system remains well developed.

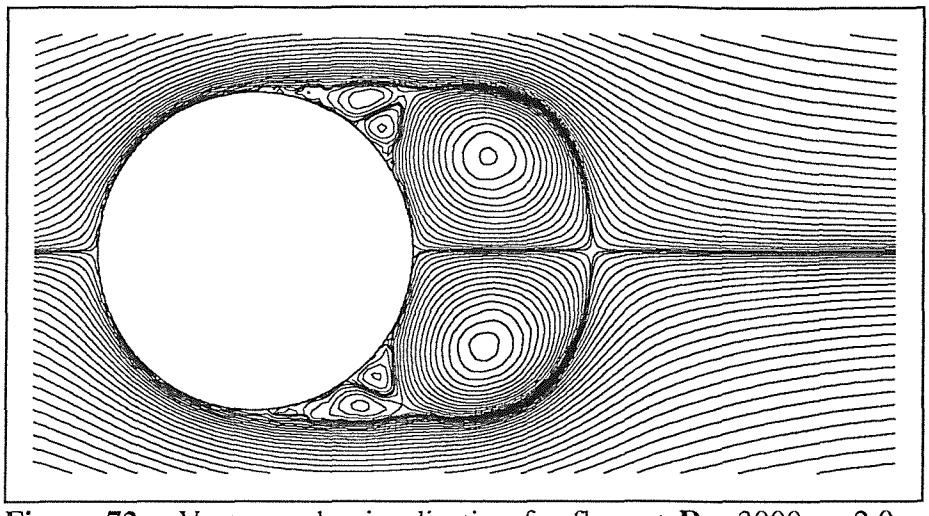


Figure 72 Vortex code visualisation for flow at **Re**=3000, τ=2.0

Figure 73 Experimental visualisation for flow at **Re**=3000, τ=2.0

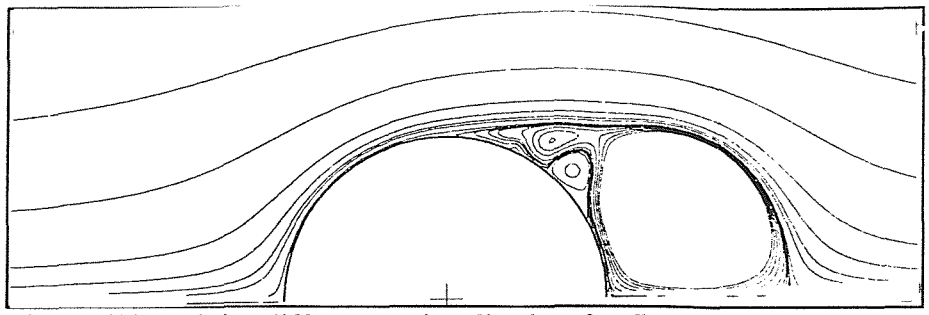


Figure 74 Finite difference visualisation for flow at **Re**=3000, τ=2.0

The geometry comparison table given below shows each of the quantities measured to be fairly well predicted. The vortex code still predicts the main eddies as being slightly too close to the cylinder surface.

	Figure 72	Figure 73	Figure 74
d_1	0.60	0.55	-
d_2	0.55	0.56	0.54
θ	44.5°	39.9°	-

Figure 75, Figure 76 and Figure 77 show the next time frame of τ=2.5, for vortex code, experiment and finite difference respectively. The vortex code solution has become asymmetrical in the region of the secondary eddies, the asymmetry eventually leads to the shedding of one of the main eddies.

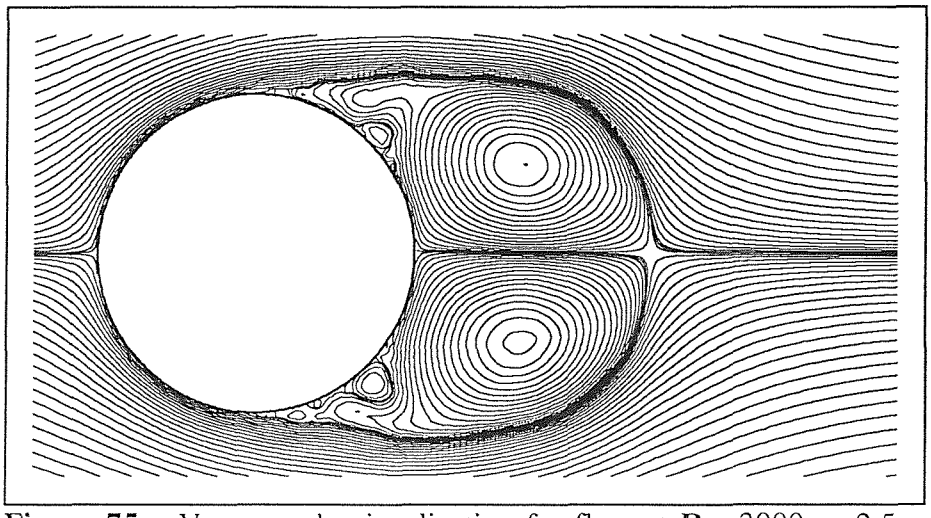


Figure 75 Vortex code visualisation for flow at **Re**=3000, τ=2.5

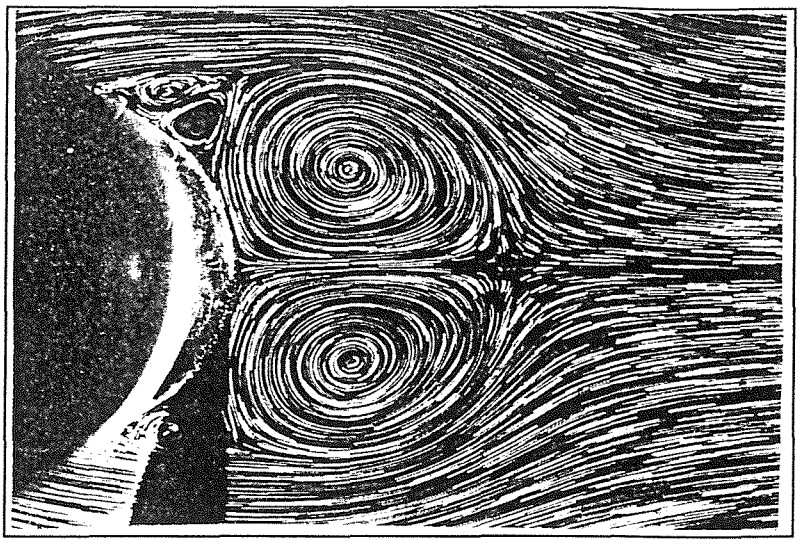


Figure 76 Experimental visualisation for flow at **Re**=3000, τ=2.5

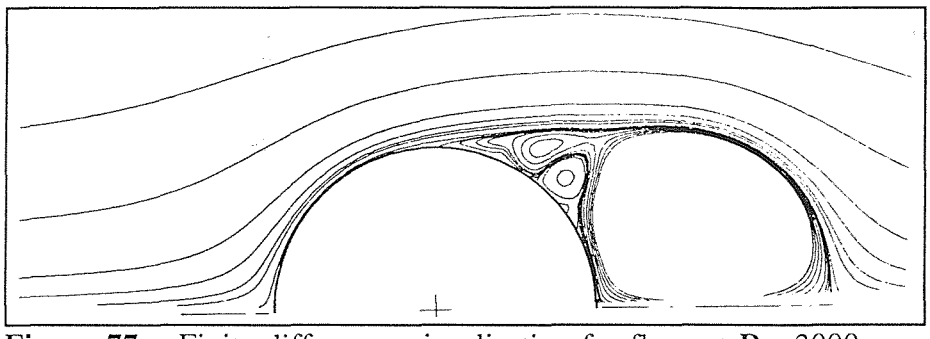


Figure 77 Finite difference visualisation for flow at **Re**=3000, τ=2.5

The geometry comparison table for τ=2.5 is shown below. The same pattern as for previous time steps emerges — results are fairly good, but both numerical solutions predict the wake to be too short and the vortex code predicts the centres of the main eddies to be too close to the cylinder surface.

	Figure 75	Figure 76	Figure 77
d_1	0.56	0.58	-
d_2	0.72	0.79	0.71
θ	46.8°	38.6°	-

In summary, results for the early flow at Reynolds number 3000 are in agreement with experimental data and results produced by a finite difference method. The finite difference

code appears to produce slightly results better than the discrete vortex method for this set of parameters. The discrete vortex solution has the centres of the two main eddies too close to the cylinder surface, a phenomenon associated with excessive roll-up of the vortex sheet which makes up the main eddies. Such excessive roll-up is known (see Smith and Stansby (1988)) to be an effect associated with time-stepping errors from the second-order scheme used here, which may explain this result. Better results could be obtained by using a smaller time step and estimating the time-shift, but our motivation here is partly to justify results with a large time step in order to give credibility to our long term solutions in section 5.4.8.

5.4.5 Short Time Solution - Re=5000

Now looking at the flow at Reynolds number 5000, Figure 78 - Figure 85 show the early time evolution. For each of four times, the output of the vortex code is shown alongside an experimental visualisation of the same flow, taken from Bouard and Coutanceau (1980).

Figure 78 shows the flow computed by the vortex code for τ=1.0, with Figure 79 showing the equivalent experimental visualisation. The existence of an enlarged central region to the main eddy and an elongated 'forewake' extending from this central region to the line of symmetry are correctly predicted. Details of the secondary eddy system are not correctly resolved, although see section 5.3, page 91. The geometry comparison table is also given below, showing good agreement for the quantities measured.

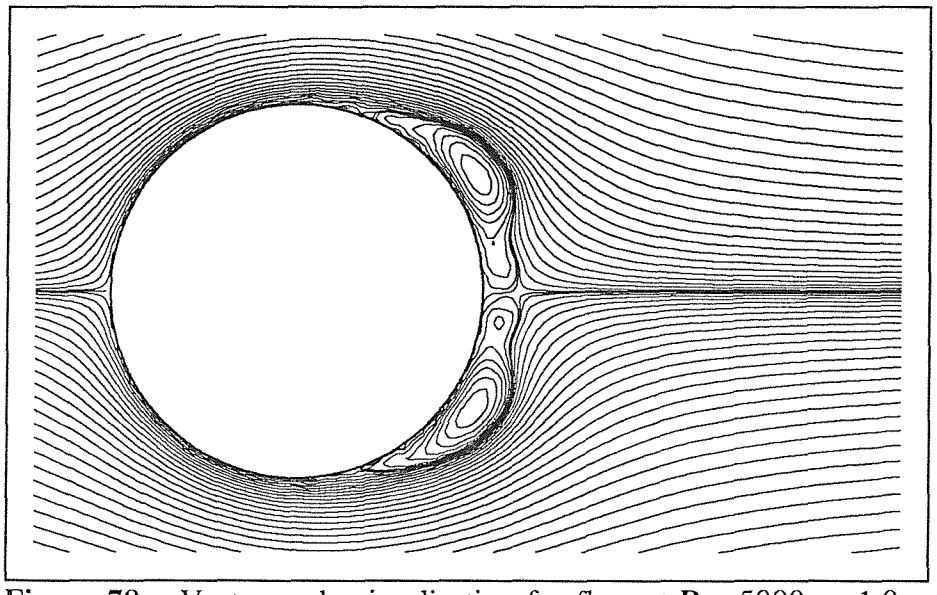


Figure 78 Vortex code visualisation for flow at **Re**=5000, τ=1.0

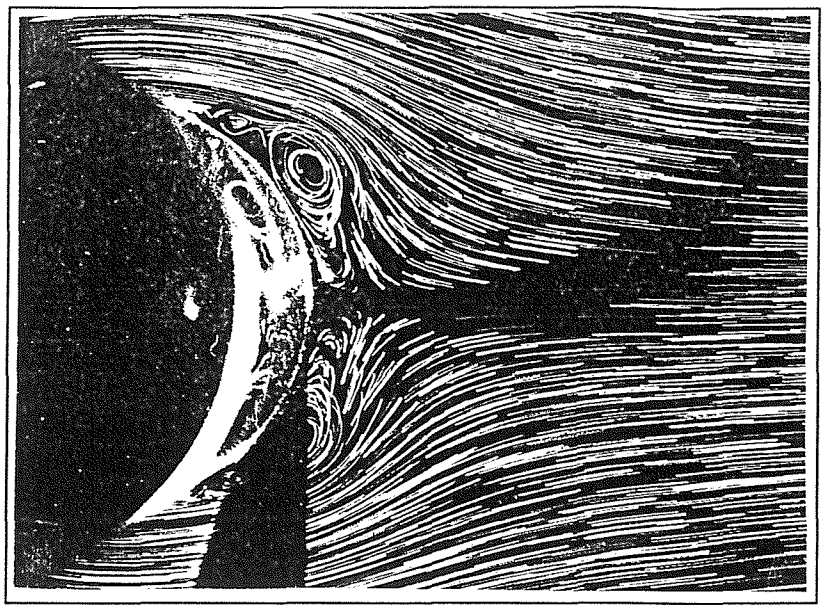


Figure 79 Experimental visualisation for flow at **Re**=5000, τ=1.0

	Figure 78	Figure 79
d_1	0.63	0.62
d_2	0.10	0.13
θ	67.2°	64.9°

Figure 80 and Figure 81 show the vortex code and experimental visualisations, respectively, for τ=1.5. The secondary eddy system has become joined with the main flow in the vortex code solution. It is worth noting that there is evidence of the flow becoming three-dimensional in the region of the secondary eddies, with dust particle paths crossing there, which indicates that these flow regions are unstable.

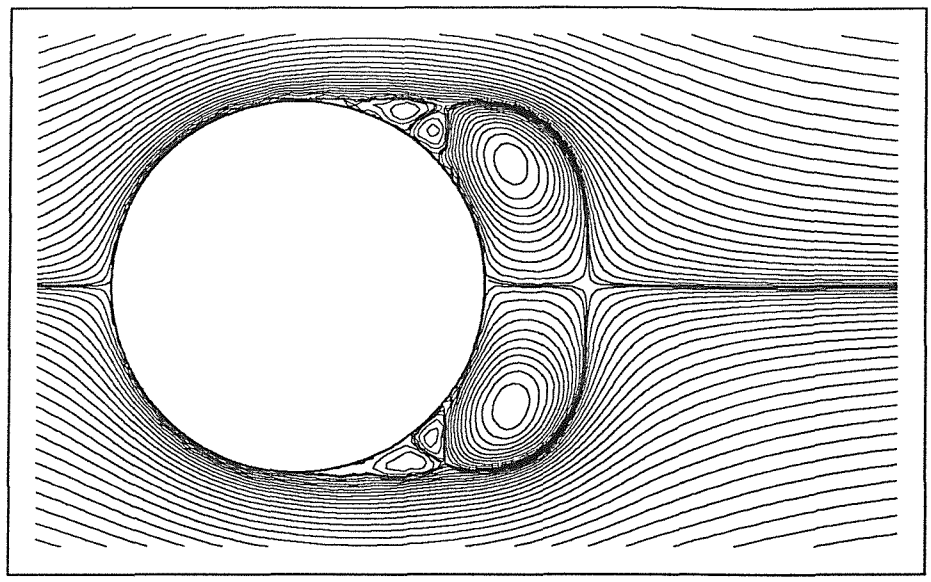


Figure 80 Vortex code visualisation for flow at **Re**=5000, τ=1.5

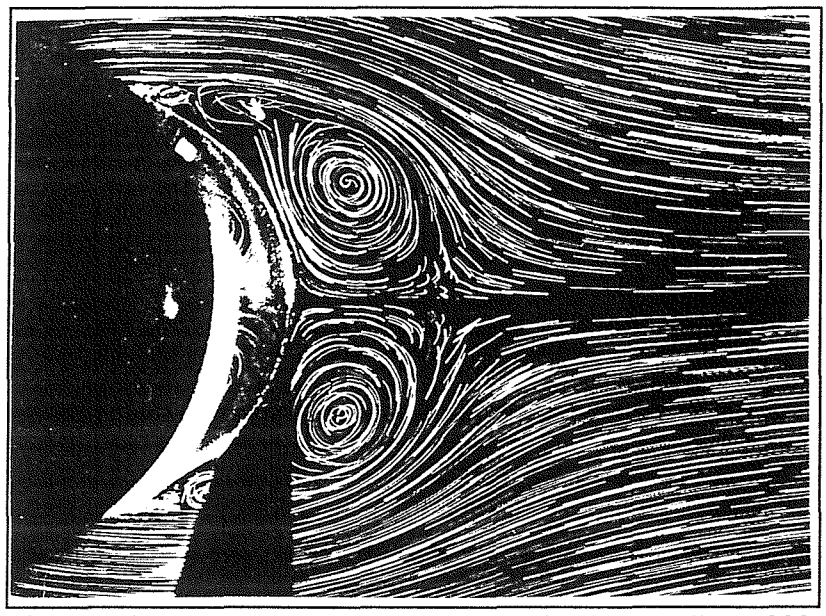


Figure 81 Experimental visualisation for flow at **Re**=5000, τ=1.5

The geometry comparison table, shown below, confirms the observation that the main wake is predicted as too short. The angle θ is also predicted as too large, showing that the centres of the two main eddies are too close to the cylinder and too far apart from each other.

	Figure 80	Figure 81
d_1	0.65	0.53
d_2	0.28	0.36
θ	60.0°	47.7°

Figure 82 shows the vortex code streamline plot for τ=2.0, with Figure 83 showing the experimental visualisation at the same non-dimensional time. The prediction at this time is very close to the experiment (particularly if the vortex plot is viewed as a vertical reflection of the experiment — remember that the direction of initial symmetry breaking is arbitrary). Thus the code appears to have maintained agreement with the experiment, despite forming a transient solution which differs at τ=1.5. It could be that the correct amount of vorticity continues to separate from the boundary layer during the τ=1.5 phase, eventually becoming entrained into the main eddies and restoring them to their correct dimensions. The change in the flow is very rapid near τ=1.5 so if there is a small time-shift, it would be more noticeable. A more accurate solution could be obtained by using a smaller time step and taking into account the appropriate time-shift. The geometry comparison table below confirms the close agreement with the experiment in quantitative terms.

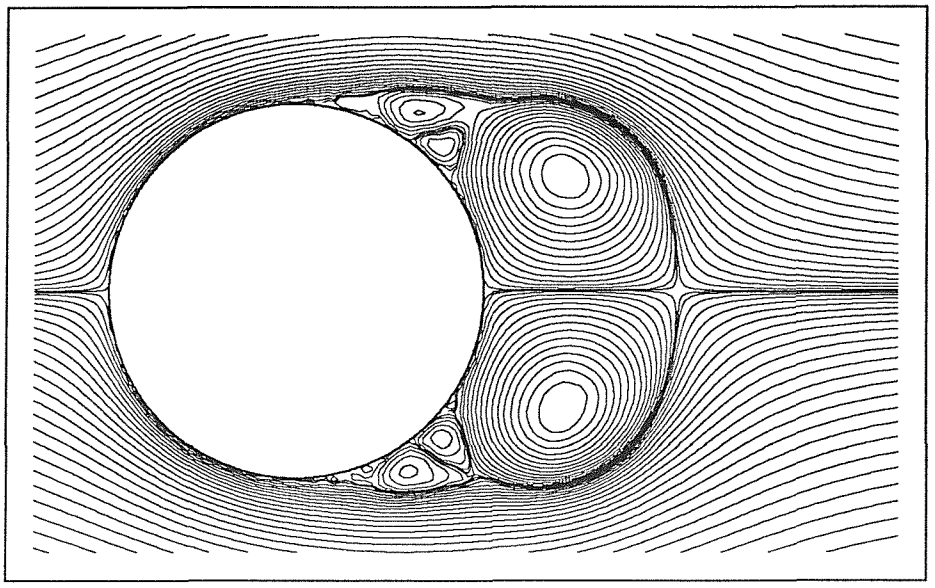


Figure 82 Vortex code visualisation of flow at **Re**=5000, τ=2.0

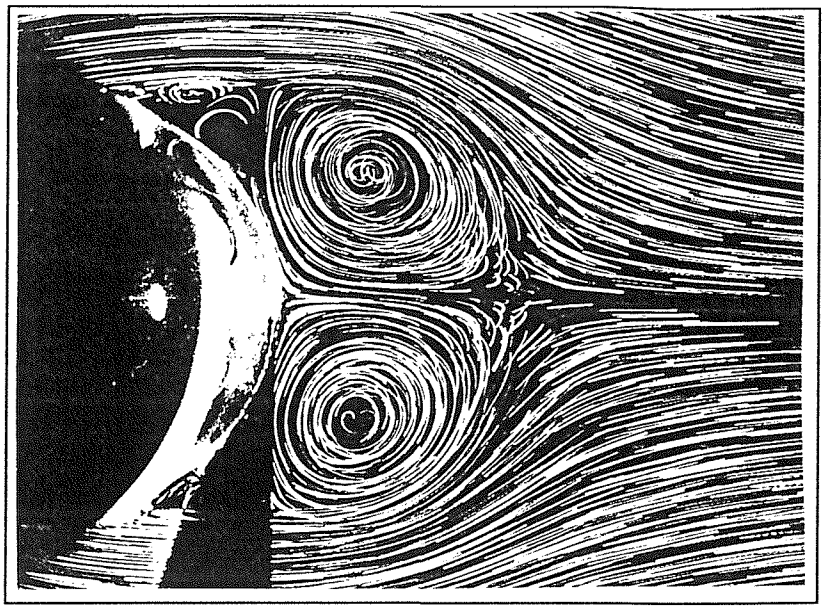


Figure 83 Experimental visualisation for flow at **Re**=5000, τ=2.0

	Figure 82	Figure 83
d_1	0.61	0.59
d_2	0.51	0.51
θ	46.5°	46.9°

Figure 84 shows the vortex code output at τ=2.5, Figure 85 the experimental visualisation for the same time. Again, agreement is close confirming that the code and experiment have entered a mode of closer agreement after a transient phase. Notice, in both experimental and numerical visualisations, the formulation of a very small tertiary eddy system, visible as a slight deviation on the streamlines close to the cylinder in the outer secondary eddies. The fact that the code appears to predict the flows more accurately after longer times indicates that we can use the same time step, $\delta\tau$=0.1, for long time runs. The geometry comparison table given below confirms the similarity between numerical and experimental visualisations.

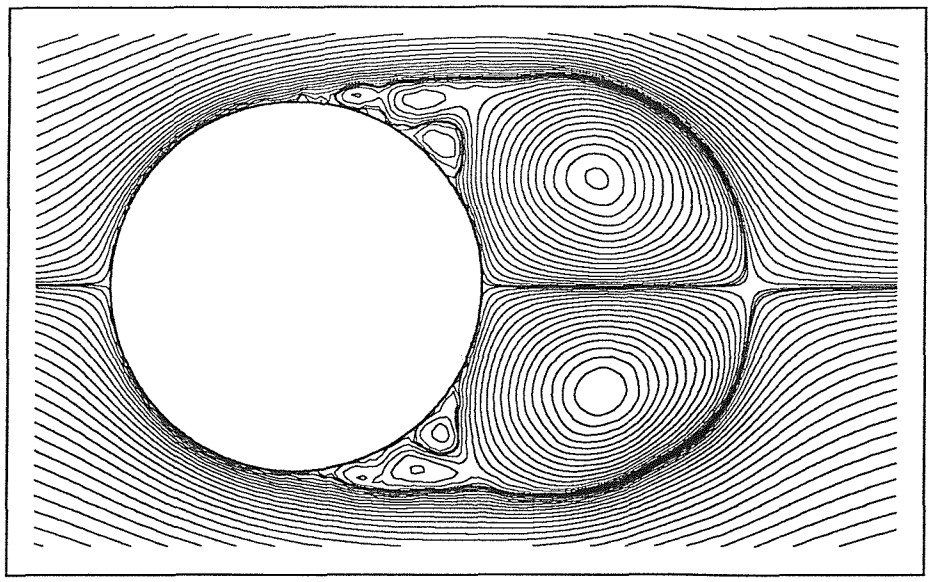


Figure 84 Vortex code visualisation for flow at **Re**=5000, τ=2.5

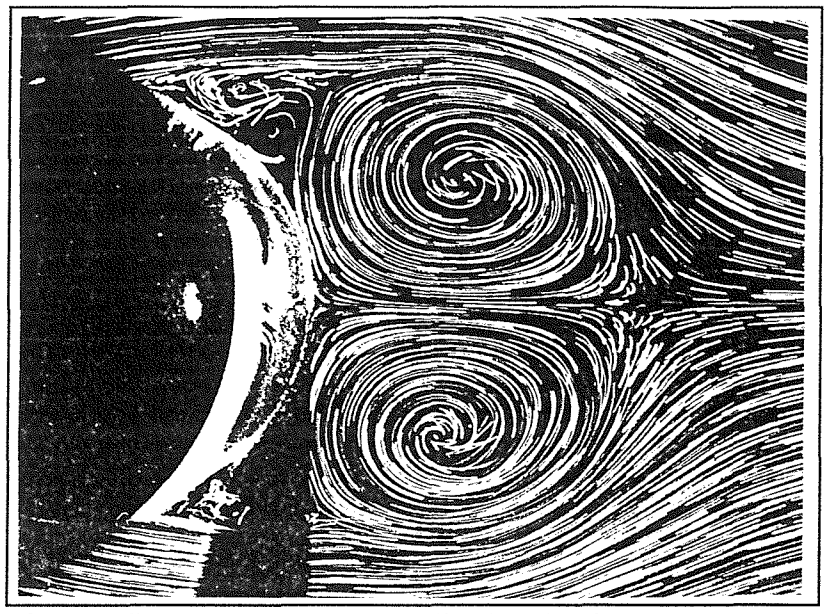


Figure 85 Experimental visualisation for flow at **Re**=5000, τ=2.5

	Figure 84	Figure 85
d_1	0.57	0.60
d_2	0.71	0.74
θ	38.7°	40.7°

5.4.6 Short Time Solution - Re=9500

Figure 86 - Figure 107 show the evolution of the impulsively started flow past a non-rotating circular cylinder at Reynolds number 9500. A flow visualisation generated by the vortex code is shown for each time frame, 400 boundary points and a time step of 0.1 were used. Experimental and finite difference visualisations, taken from Ta Phuoc Loc and Bouard (1985), are shown where available (the paper does not include visualisations at every value for the regular time intervals used). For each time frame a discussion is given of the successes and failings of both numerical methods and a quantitative geometry comparison table is presented. We have conducted runs at this Reynolds number using a smaller time step and obtain better results which are time-shifted. The results shown here are not intended to be the best possible using the discrete vortex code, but rather demonstrate that reasonable results can be achieved using a fairly large time step.

Figure 86, Figure 87 and Figure 88 show the discrete vortex, experimental and finite difference streamline plots for τ=0.6. The experimental flow is very difficult to read, but appears to be in closer agreement with the finite difference solution. In the discrete vortex solution an elongated eddy has formed on the cylinder surface away from the line of symmetry.

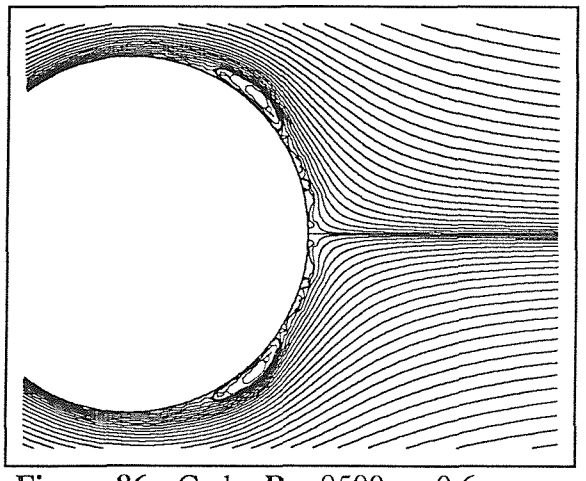


Figure 86 Code, **Re**=9500, τ=0.6

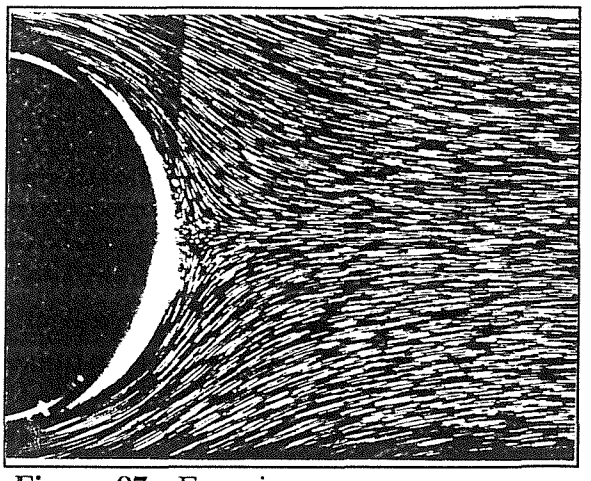


Figure 87 Experiment

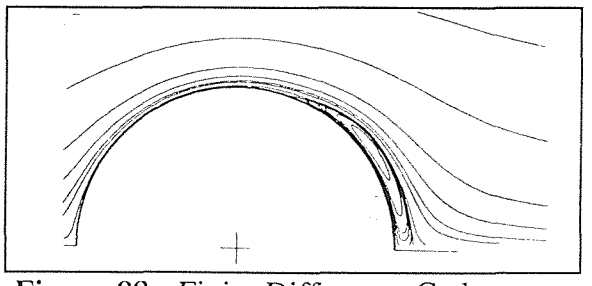


Figure 88 Finite Difference Code

The geometry comparison table is given below. It was possible to estimate the centre of the eddy in the finite difference diagram, so the table is full. Agreement is good between the experiment and finite difference solutions, with a slightly less accurate result given by the vortex code.

	Figure 86	Figure 87	Figure 88
d_1	0.77	0.69	0.66
d_2	0.02	0.05	0.06
θ	97.2°	80.6°	75.7°

Figure 89 shows the vortex code output for non-dimensional time τ=0.8, with Figure 90 showing the equivalent experimental visualisation and Figure 91 the finite difference equivalent. Both of the numerical solutions predict an enlarged central section to the main eddies, although this phenomenon is less pronounced in the experiment.

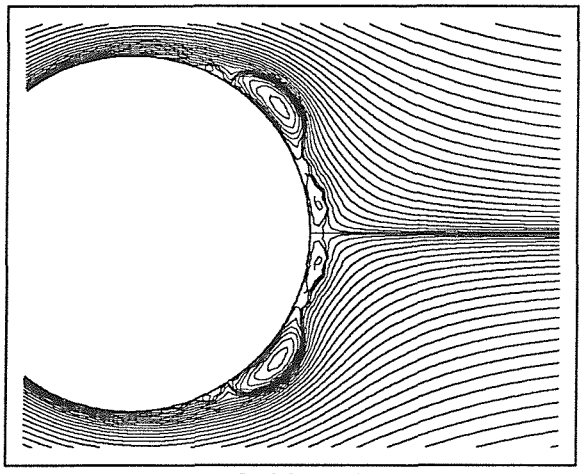


Figure 89 **Re**=9500, τ=0.8

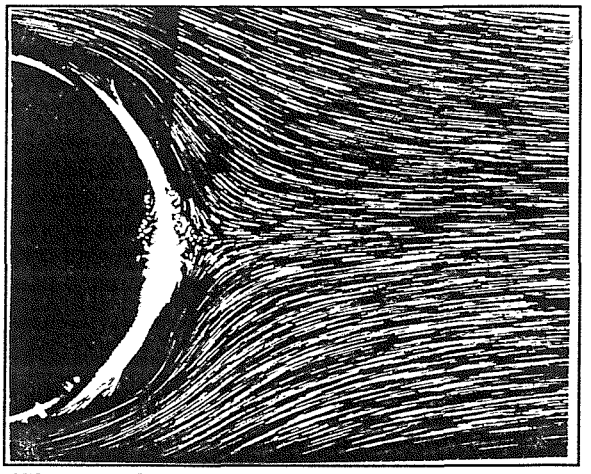


Figure 90 Experiment

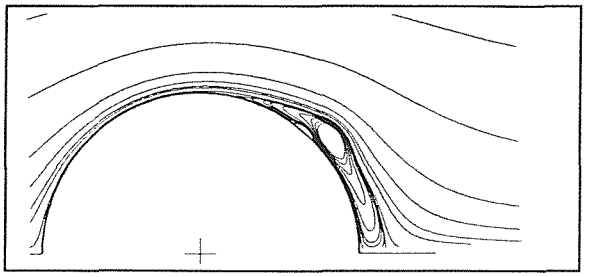


Figure 91 Finite Difference Code

A quantitative comparison of the dimensions of the flow features is given below. Both numerical schemes predict the angle θ as being too large, although the centres of the eddies are difficult to estimate in the experimental photograph.

	Figure 89	Figure 90	Figure 91
d_1	0.71	0.75	0.73
d_2	0.04	0.07	0.09
θ	82.6°	73.7°	82.5°

Figure 92, Figure 93 and Figure 94 show the same three visualisations at τ=1.0. There is clear evidence, in all three visualisations, of the counter-rotating secondary eddies interacting with the external flow. In the vortex solution, only the secondary eddy on the lower body surface is joined to the external flow; in the experiment the opposite has occurred, with the lower secondary eddy just contained within the wake and the upper secondary eddy interacting directly with the external flow. At this time, all three visualisations agree upon the existence of an enlarged central section to the main eddies and a forewake extending from this enlarged section to the line of symmetry.

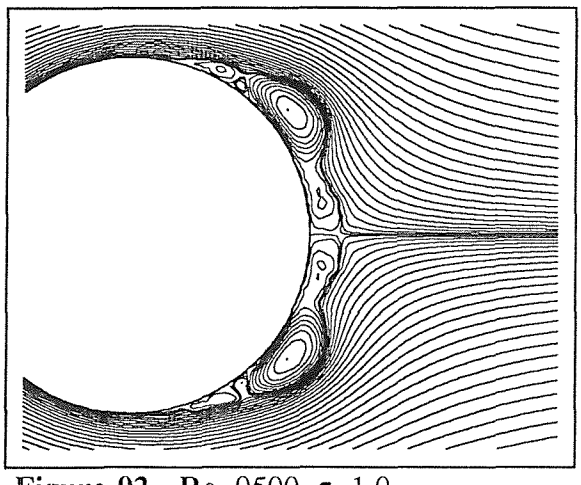


Figure 92 Re=9500, τ=1.0

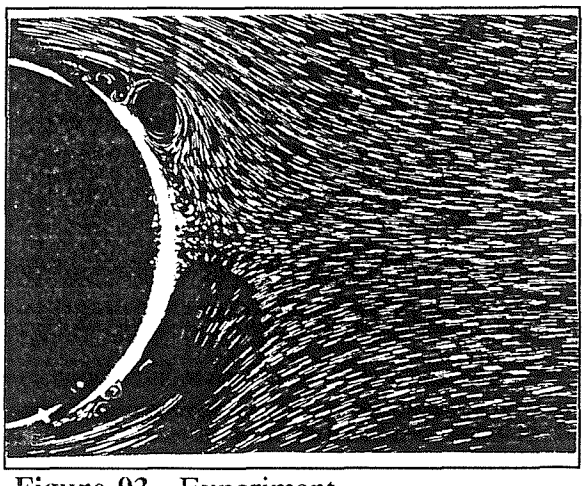


Figure 93 Experiment

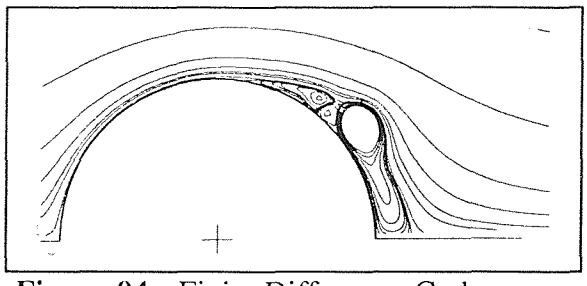


Figure 94 Finite Difference Code

	Figure 92	Figure 93	Figure 94
d_1	0.69	0.67	0.69
d_2	0.09	0.09	0.11
θ	74.3°	80.6°	74.3°

Figure 95 shows the streamline plot produced by the vortex code for τ=1.2, with Figure 96 showing the equivalent finite difference output. The experimental visualisation for this time frame is omitted from our source, Ta Phuoc Loc and Bouard (1985). The two numerical schemes show the same flow features, with the centres of the main eddies further apart in the discrete vortex solution, see the table below.

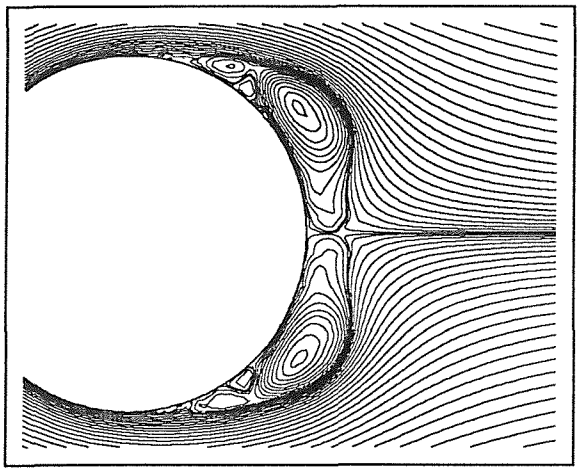


Figure 95 **Re**=9500, τ=1.2

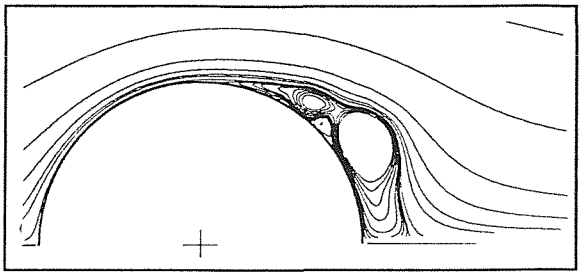


Figure 96 Finite Difference Code

	Figure 95	Figure 96
d_1	0.71	0.57
d_2	0.13	0.14
θ	73.0°	65.4°

Figure 97 - Figure 99 show the three comparative visualisations at non-dimensional time 1.4. At this time, the experiment shows the main eddies to have two distinct rotational centres. This phenomenon is correctly predicted in the finite difference calculation, but absent from the vortex code output. Consequently the centres of the main eddies, which appear to be situated at some position lying between the two centres seen in the other visualisations,

are too far apart. At this time the forewake can still just be seen — the streamlines start to turn upstream where they meet on the line of symmetry between the two main eddies. Figure 100 is a visualisation produced by the discrete vortex code at τ=1.6, using a time step of 0.0125, and illustrates that fact that it is possible to obtain a more accurate solution. As mentioned in section 5.3, a comparison time-shift is introduced — in this case all the vortex code plots for this run compare well with the experiments 0.2 non-dimensional time units earlier. The overall shape of the main eddies is in much closer agreement in this more accurate solution.

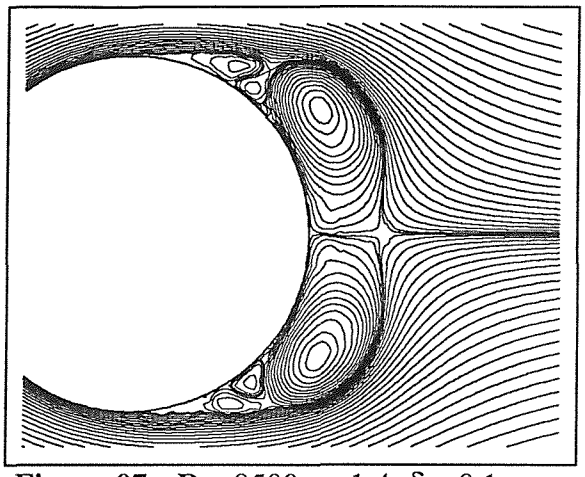


Figure 97 **Re**=9500, τ=1.4, $\delta\tau$=0.1

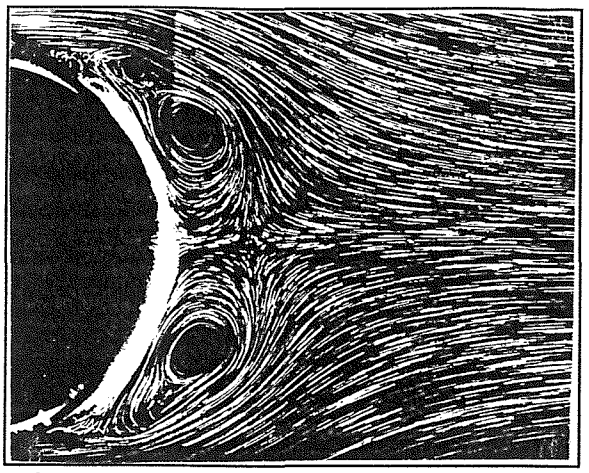


Figure 98 Experiment

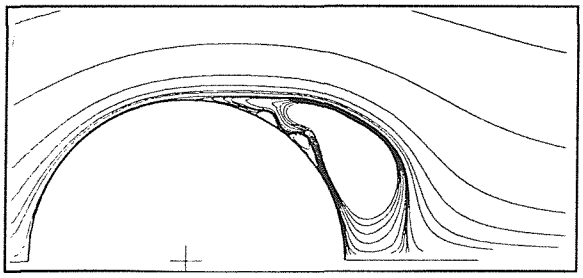


Figure 99 Finite Difference Code

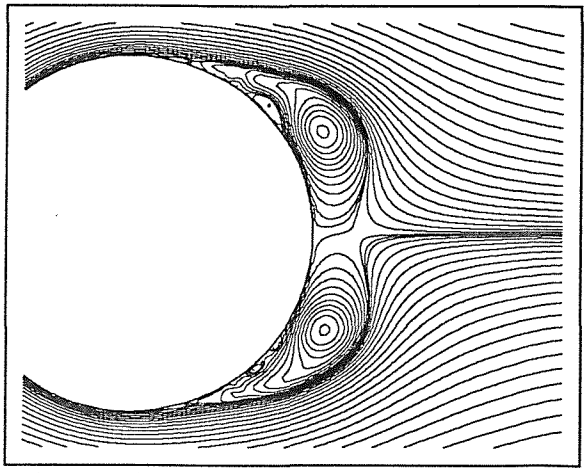


Figure 100 **Re**=9500, τ=1.6, $\delta\tau$=0.0125

At this stage, the centres of the main eddies in the finite difference plot could not be estimated with accuracy, so the geometry comparison table given below is incomplete. The figures confirm the observation that the centres of the main eddies are predicted as too far apart by the vortex code.

	Figure 97	Figure 98	Figure 99	Figure 100
d_1	0.69	0.61	-	0.59
d_2	0.22	0.20	0.21	0.15
θ	66.3°	60.1°	-	57.3°

Figure 101 shows the output of the vortex code at τ=1.6, Figure 102 shows the experimental visualisation at the same time and Figure 103 shows the equivalent finite difference plot. All three plots appear to be very similar, with the vortex code solution closely matching the experiment. In the experiment and vortex code, the lower secondary eddy is just joined to the outer flow. The forewake region has now been completely absorbed into the main eddies. The shape of the bounding contour of the wake is well predicted by both numerical schemes.

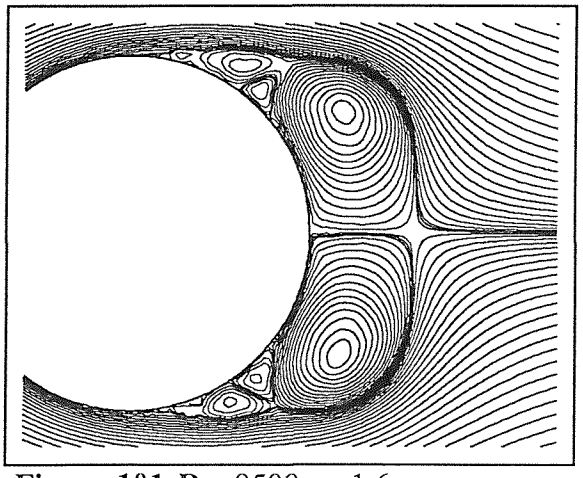


Figure 101 Re=9500, τ=1.6

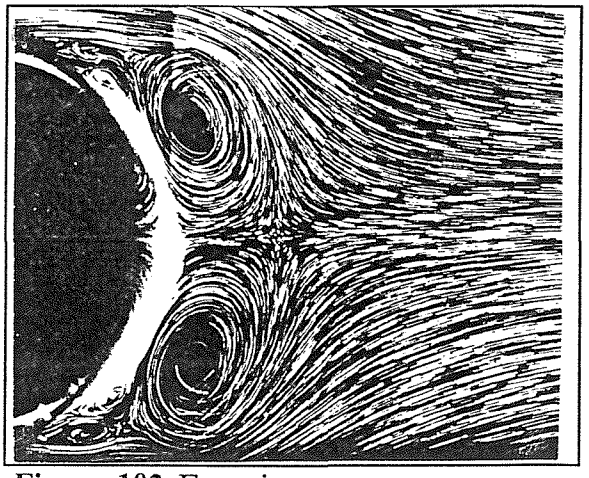


Figure 102 Experiment

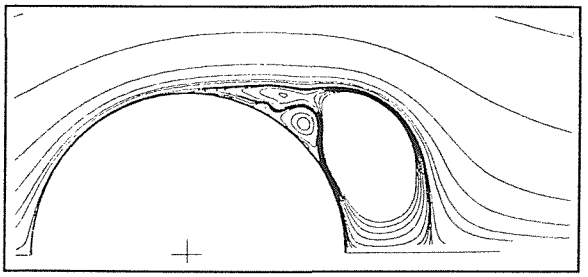


Figure 103 Finite Difference Code

The good agreement between vortex code and experiment is quantitatively confirmed by the geometry comparison table shown below.

	Figure 101	Figure 102	Figure 103
d_1	0.69	0.66	-
d_2	0.30	0.27	0.28
θ	60.5°	61.0°	-

Figure 104 and Figure 105 show the streamlines produced by the two numerical schemes at τ=1.8, the relevant experimental plot being absent from our source. The two agree on the scale and shape of the main eddies. The finite difference scheme shows the counter-rotating secondary eddy completely joined to the outer flow, it remains closed to the body in the discrete vortex solution.

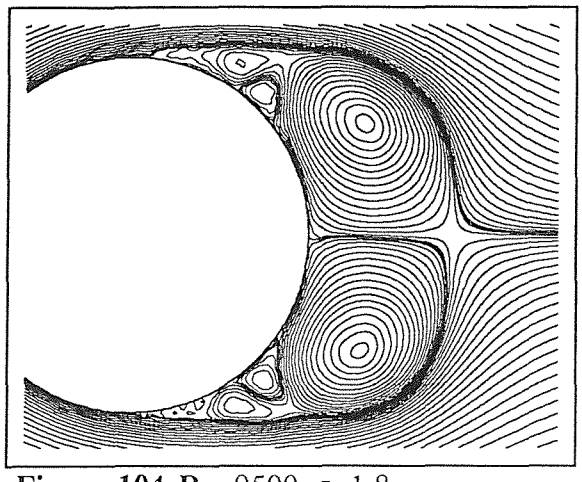


Figure 104 **Re**=9500, τ=1.8

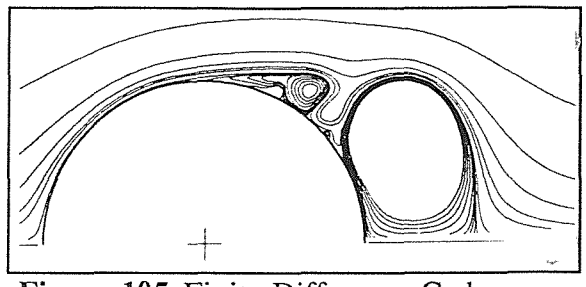


Figure 105 Finite Difference Code

The geometry comparison table is provided for completeness, the vortex code predicts the wake to be slightly longer than the finite difference code.

	Figure 104	Figure 105
d_1	0.65	-
d_2	0.41	0.36
θ	52.8°	-

Figure 106 shows the streamlines computed by the vortex code at τ=2.0, Figure 107 shows the equivalent experimental visualisation and Figure 108 shows the finite difference

solution at the same time. At this time, the vortex code appears to have predicted the shape of the wake more closely then the finite difference code, in which the main eddies are too 'square'. The secondary eddy system is predicted correctly by both codes although the experiment has asymmetry in this region. There is also evidence of some three-dimensional behaviour in the experiment, in the region of the centres of the main eddies, where dust particle paths can be seen to cross.

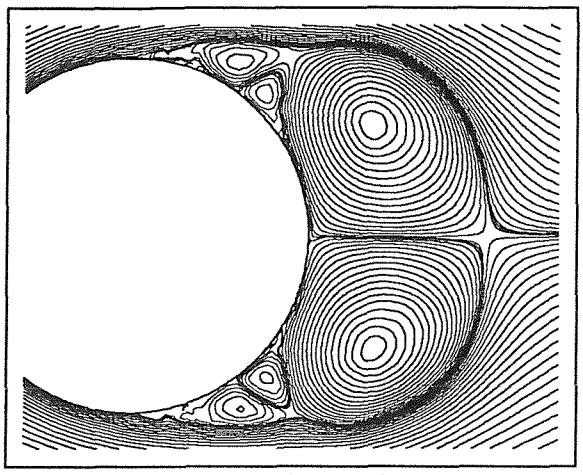


Figure 106 Re=9500, τ=2.0

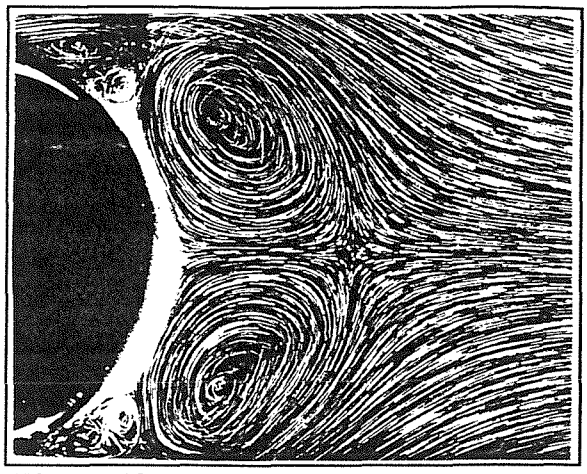


Figure 107 Experiment

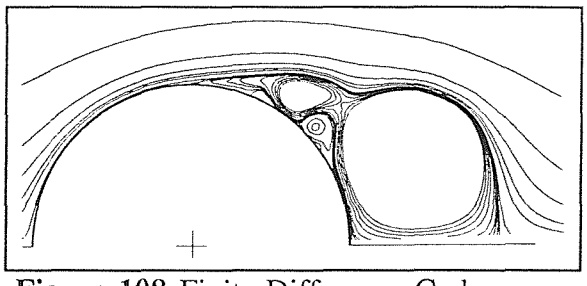


Figure 108 Finite Difference Code

The geometry comparison table given below shows good agreement between experiment and vortex code. The centres of the main eddies are predicted as too close but, as already mentioned, there are three-dimensional effects in this region in the experiments so their precise position is unclear.

	Figure 106	Figure 107	Figure 108
d_1	0.63	0.76	-
d_2	0.49	0.47	0.49
θ	49.8°	55.6°	-

In summary of the comparison at Reynolds number 9500, the two computational solutions are in overall agreement with the experimental visualisations. The vortex code visualisations appear to match the experiments more closely at some time frames, the finite difference plots at others. As would be expected, the flow is more unstable at this higher Reynolds number.

5.4.7 Short Time Solutions - Discussion

We have presented short time solutions at a range of Reynolds numbers and shown the results to be in reasonable agreement with experiments and finite difference calculations. In particular, the results for the later times seem to be in closer agreement, possibly due to starting effects occurring in the simulations and/or the experiments. There is some transient inaccuracy at earlier times.

More accurate solutions could be obtained by using a smaller time step and taking a comparison time shift into account, for an example see Figure 100. However, one of our main intentions in this section has been to show that results using a large time step have reasonable validity, so that we can use such a large time step to take long runs as far as possible in time in the next section, in order to generate novel fluid dynamics results.

5.4.8 Long Time Solutions - Objectives and Setup

In the following sub-sections, long time solutions at various Reynolds numbers are presented. The objective of these sub-sections is to compare the solutions with the few available results and to show the interesting phenomena which occur in some of the solutions. The results are presented in terms of streamline plots and graphs of force coefficients. The flow visualisation techniques have already been discussed, but some comment must be made to explain the force graphs.

The mathematical techniques used to calculate the force coefficients are given in section 3.11. Equation (39), from Quartapelle and Napolitano (1983), has been reported to give good results for an impulsively started circular cylinder, see for instance Smith and Stansby (1988). Our results agree and hence we have used the formula as a reference in order to evaluate other techniques. This equation should give accurate force figures because it integrates over the whole flow so no 'sampling' error is introduced. For this reason we refer to forces calculated using this technique as 'Lagrangian' forces, to indicate that the formula is in keeping with the Lagrangian flow model. Unfortunately, generalisation of the method to non-circular body shapes is a non-trivial task. Two extra numerical solutions of the Laplace equation would be needed, equivalent to two extra boundary solves at each time step and thus significantly increasing the storage and CPU requirements of the calculation.

The much cheaper, although less accurate, method based on equation (45) was evaluated as suitable for use with non-circular bodies. The method relies on evaluation of flow quantities at fixed 'grid' points placed around the body boundary. This makes the calculation of force quantities highly sensitive to the placement of the vortices relative to the grid, resulting in a large noise component being introduced into the force results. Fortunately, this noise has proved simple to eliminate using a simple moving average of the data over the time variable. Figure 109 and Figure 110 show force results at Re=550, before and after the smoothing has been applied. The integrity of the data has been largely retained, with the noise component effectively removed to show the underlying trend. All smoothed results presented here are averaged over three data points, which was found to be enough to remove the noise without losing essential details. We have termed results from equation (45) 'Eulerian' forces, to indicate the introduction of the grid.

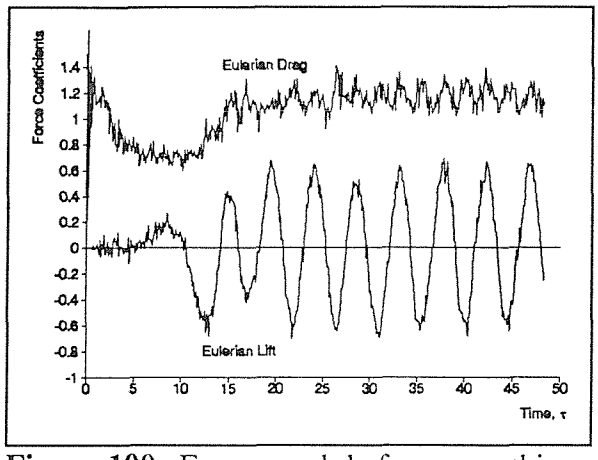


Figure 109 Force graph before smoothing

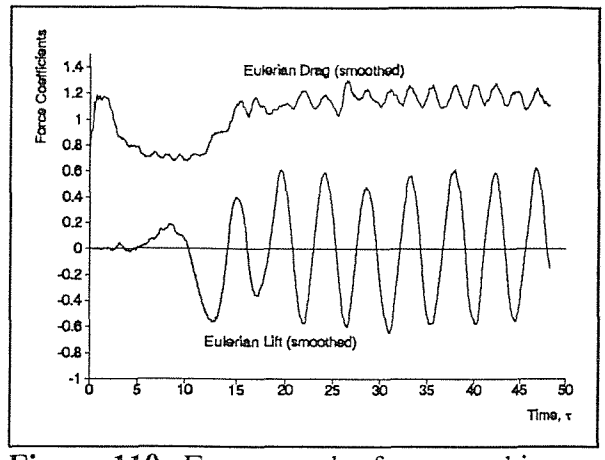


Figure 110 Force graph after smoothing

Most of the force graphs presented in this section show the total lift or drag exerted on the cylinder. This is composed of the sum of a pressure force and a skin friction force. The pressure component, whose evaluation is discussed above, dominates the total force at the Reynolds numbers considered here. However, it is of interest to examine the frictional component in isolation. The magnitude of the skin friction diminishes rapidly with increase in Reynolds number. Thus the friction component only contributes a significant proportion of the total force at the lower Reynolds numbers. Figure 111 shows the skin friction force coefficients at Reynolds number 550. These were calculated using equation (46) and are in reasonable agreement with figures given in Franke et al. (1990). Our calculations give a mean frictional drag of 0.21, Franke et al. quote 0.21 at Re=300 and 0.13 at Re=1000. The behaviour of the frictional lift is similar to that exhibited by the pressure lift — zero at early times and then oscillating after symmetry has broken showing the lifting effect of the alternate eddy shedding. The friction lift lags slightly behind the pressure lift. In the graph, the frictional drag is initially large then drops to a near constant value; the theoretical result is an infinite value at the impulsive start. A separate discussion has not been included for the

pressure force because it dominates the higher Reynolds number calculations and so its behaviour can be seen clearly in the other graphs.

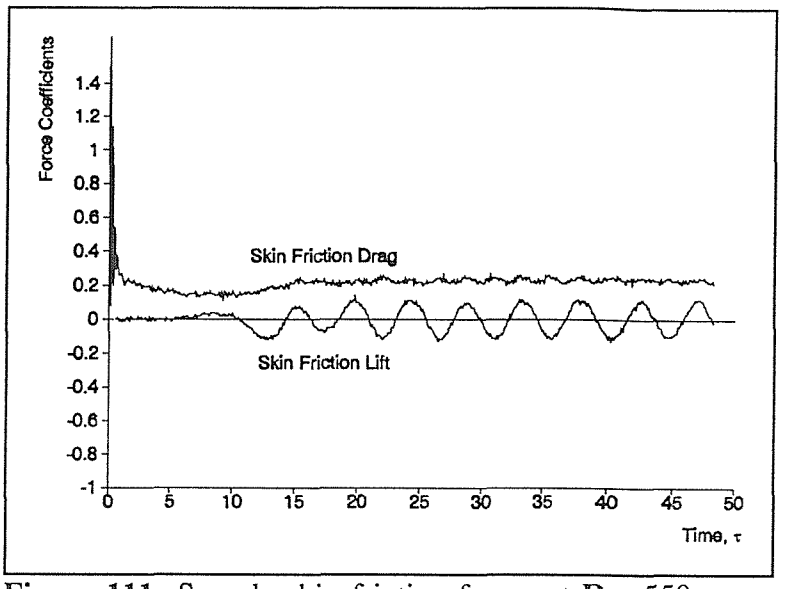


Figure 111 Sample skin friction forces at **Re**=550

5.4.9 Long Time Solution : Re=550

In this sub-section, the results of the discrete vortex model are presented for a long run at Reynolds number 550. The flow is shown near the time of the initial symmetry breaking after which it becomes periodic and sheds eddies alternately from either side of the cylinder. These eddies travel downstream to form the well known Karman vortex street. The results were generated using 200 boundary points and δt=0.1.

Figure 112 shows the streamlines at τ=5.0, where the flow is still near-symmetric. Slight oscillations at the rear of the eddy pair occur up to this point, then the pair become very unstable and symmetry completely breaks. The direction of this extreme symmetry breaking is arbitrary and depends upon essentially random effects such as noise in the model. One eddy becomes dominant near the cylinder and impinges on the other, which becomes elongated in the streamwise direction.

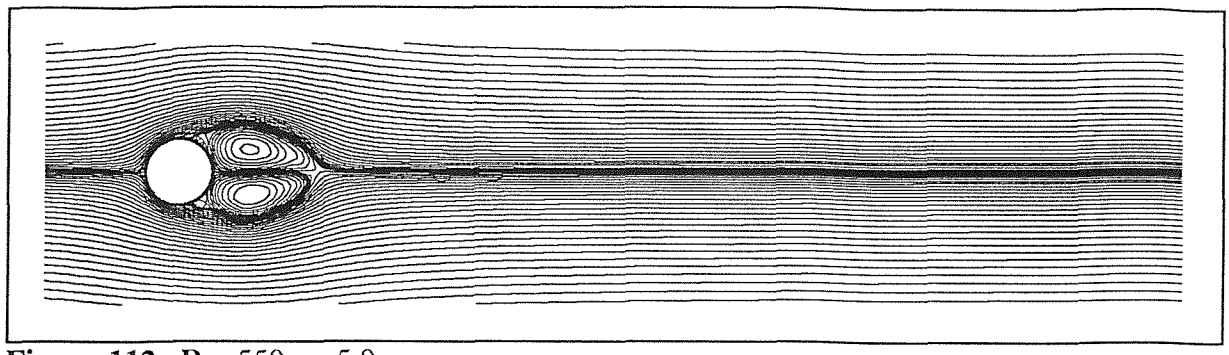


Figure 112 Re=550, τ=5.0

Figure 113 shows the flow at τ=10.0, the lower eddy is starting to shed. The secondary eddies near the upper separation point appear to be rectangular in this plot, but this is purely an effect of the visualisation grid being too coarse to fully resolve such fine features.

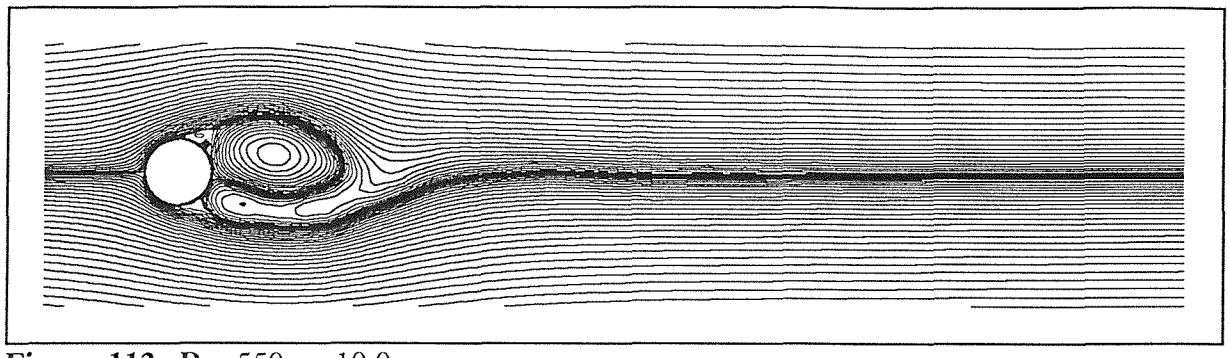


Figure 113 Re=550, τ=10.0

In Figure 114, where the flow has progressed to τ=11.0, the upper eddy increased further in size. The vorticity which was previously causing the lower eddy has split into two regions; the part further from the cylinder has shed and is travelling downstream while the other has formed into another eddy attached to the cylinder surface. Thus the first vortex to shed is smaller than subsequent ones.

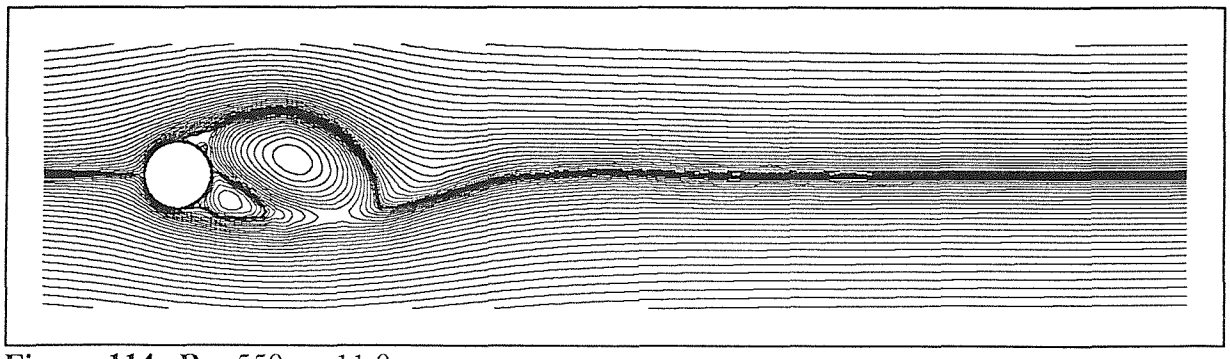


Figure 114 Re=550, τ=11.0

As the flow progresses to τ=12.0, as shown in Figure 115, the upper eddy/vortex continues to move downwards and starts to open out and join with the outer flow. This

eddy/vortex has also become detached from the cylinder. The small vortex which has completely shed continues to travel downstream, causing a kink in the streamlines. The lower eddy attached to the cylinder continues to grow larger.

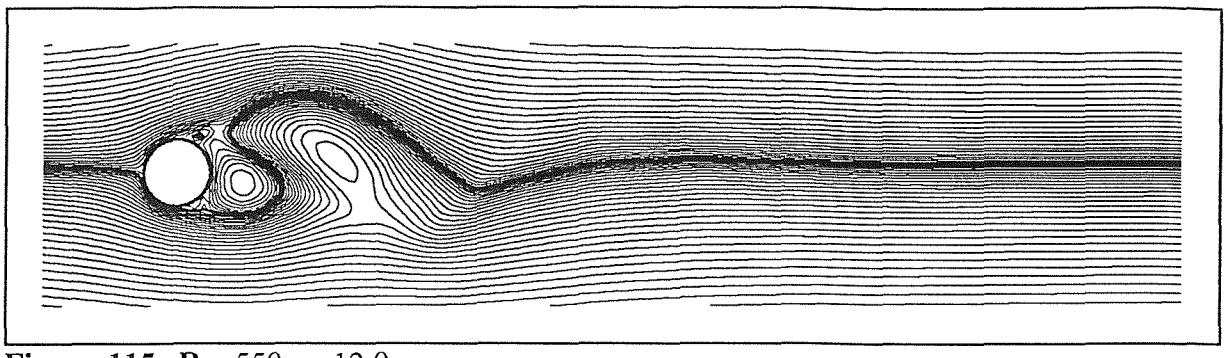


Figure 115 **Re**=550, τ=12.0

From τ=12.0, as confirmed by the force graphs given in Figure 118 and Figure 119, there is an initial build-up phase where vortices shed which are smaller than those in the subsequent periodic phase. After the first three eddies have shed, the system settles down to regular periodic eddy shedding at a time interval of approximately 2.0 non-dimensional time units. Figure 116 shows a streamline plot for τ=47.5, which is as far as we took this particular computation. A regular Karman vortex street has developed, demonstrating the periodic shedding pattern. The discrete vortex locations for the same flow are shown in Figure 117, a region containing a dense patch of discrete vortices corresponds to each of the shed vortices. The effect of the diffusion of vorticity can clearly be seen in the vortex clusters further downstream; the process of rolling up of 'sheets' of discrete vortices can be seen in the vortices nearer the cylinder.

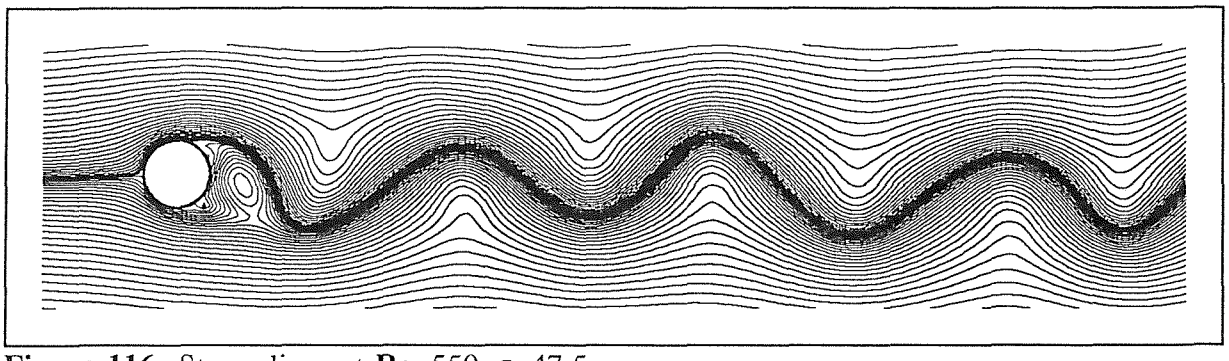


Figure 116 Streamlines at **Re**=550, τ=47.5

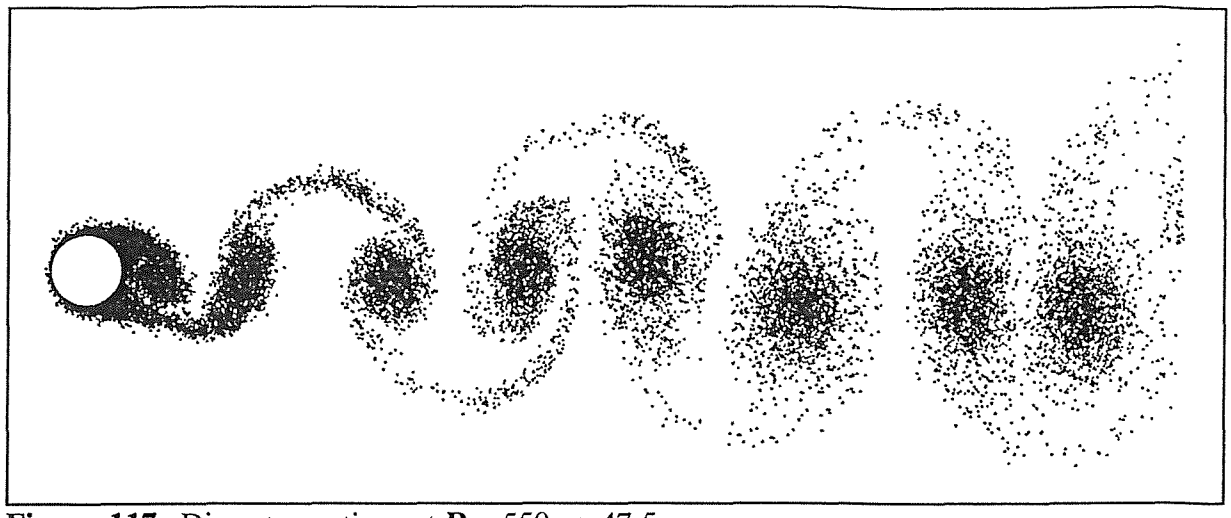


Figure 117 Discrete vortices at **Re**=550, τ=47.5

Figure 118 shows the force coefficients as calculated by the Eulerian technique; Figure 119 shows the results using the Lagrangian technique. The lift graphs are in close agreement. The early near-symmetrical flow generates almost no lift then the oscillations in lift grow to become periodic as eddies shed from alternate sides of the cylinder. The drag graphs agree in their mean value and their gross behaviour, however the variations in the Lagrangian graph are more exaggerated. Note that the pressure drag should be near zero at τ=0.0 (consider the similarity with the inviscid problem). Other published results also confirm that the pressure drag should start at zero and gradually increase. The Lagrangian drag tends towards this behaviour; the Eulerian drag appears to be inaccurate for early times (τ<2.0), which is the case in all our test problems. The severe oscillations seen in some of the force graphs at very early times are believed to be a direct consequence of the impulsive start.

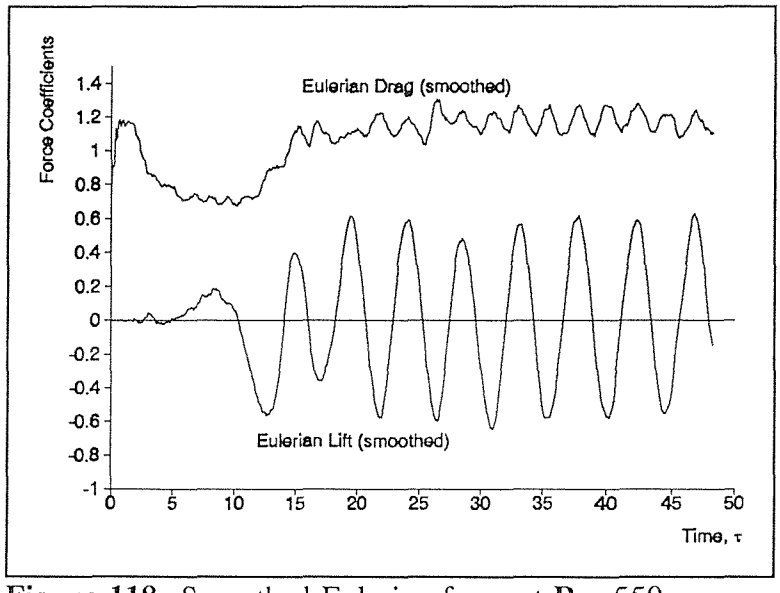


Figure 118 Smoothed Eulerian force at **Re**=550

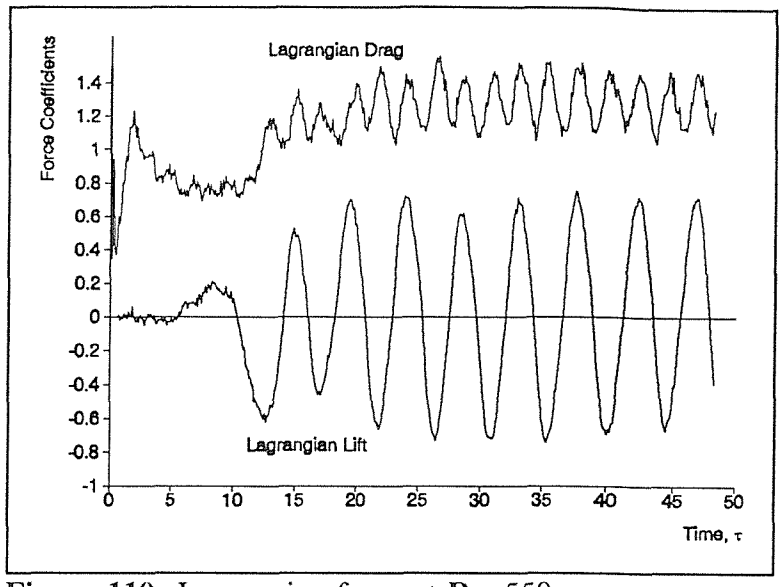


Figure 119 Lagrangian force at **Re**=550

These results give an average Strouhal number (for τ>15.0) of 0.22 as compared to the experimental value of 0.21 in Schlichting and the finite-difference results of Franke et al. who quote 0.215 at ***Re***=300 and 0.236 at ***Re***=1000. The average long time drag (taken for τ>15.0) is 1.16 (Eulerian) and 1.27 (Lagrangian); the experimental value given in Schlichting is 1.2. Franke et al. give mean drag figures of 1.32 at ***Re***=300 and 1.47 at ***Re***=1000. Kuwahara (1990) has performed two-dimensional and three-dimensional finite difference calculations and reports that the two-dimensional computations of drag are consistently greater than those reported in experiments.

5.4.10 Long Time Solution : Re=5000

At Reynolds numbers of 5000 and above, the discrete vortex solution exhibits interesting irregular behaviour. We have not found any other flow visualisations for long times at such high Reynolds numbers in the literature. As regards the method, we should keep some reservations in mind. There is no turbulence modelling in the method; turbulence cannot be modelled merely by accurate solution of the *two-dimensional* Navier-Stokes equations. The fundamental three-dimensional fine-scale nature of turbulence demands that either a two-dimensional turbulence model is used or an accurate three-dimensional solution is sought. Turbulence starts to occur in the wake at some value of increasing Reynolds number; Schlichting states that the wake is turbulent at 5000 but van der Vegt and de Boom claim that the wake is laminar at 31,700. In the discrete vortex method boundary layer details are mainly influenced by the flow near the boundary and we hope that, provided the gross features in the wake are reasonably well represented, the flow at the boundary can still be relatively accurately modelled. In particular, the effect on force calculations of a vortex decays rapidly as it moves away from the boundary, so the forces should not be directly influenced by minor

inaccuracy in the wake. Turbulence occurs in the boundary layer as the Reynolds number increases past 250,000 (from Schlichting), and we expect the model to give poor results in this flow regime.

The diffusion caused by the time stepping errors in the convective motion of the vortices will effectively put a ceiling on the achievable Reynolds number. In practice, we have found that results using Reynolds numbers greater than 10,000 are all broadly similar due to the non-deterministic nature of flows at these Reynolds numbers. However, in the next sub-section we achieve a reasonable degree of comparison with an experiment at Reynolds number 31,700.

On the positive side, we note that the short time results were in good agreement with experiments up to Reynolds number 9,500 and, apart from cumulative time integration errors, there is no fundamental reason why the quality of the solution for a laminar flow should degrade with time.

Figure 120 - Figure 127 show streamline plots for the flow at $\boldsymbol{Re}$=5000, up to τ=75.0. Figure 128 and Figure 129 show the force coefficients for the same run. This run was conducted using 100 boundary points and δt=0.1.

Figure 120 shows the flow at τ=7.5; symmetry has broken and the first eddy is starting to shed. The initial shedding is different from that at $\boldsymbol{Re}$=550 (it has also broken in the opposite direction) — the lower eddy has not stretched and split into two parts, but remains close to the cylinder surface in its entirety.

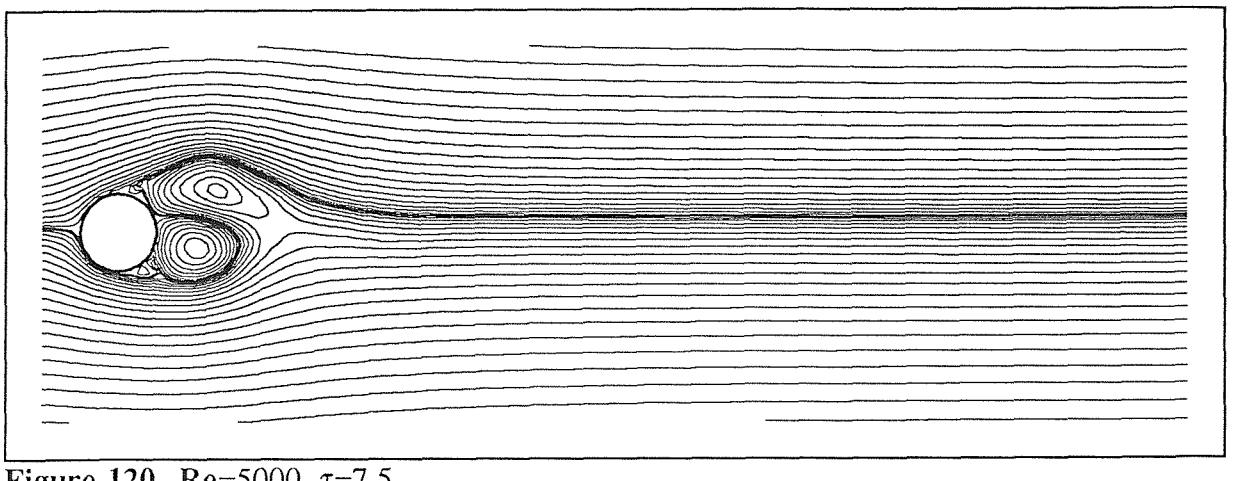


Figure 120 **Re**=5000, τ=7.5

In Figure 121, the flow has progressed to τ=10.0, where the first vortex has completely shed and is travelling downstream. The second vortex has nearly shed from the body surface and is occupying a position directly downstream of the cylinder. A complex secondary eddy

system is attached to the cylinder surface, consisting of a small eddy/vortex on the upper surface and a counter-rotating eddy pair on the lower surface.

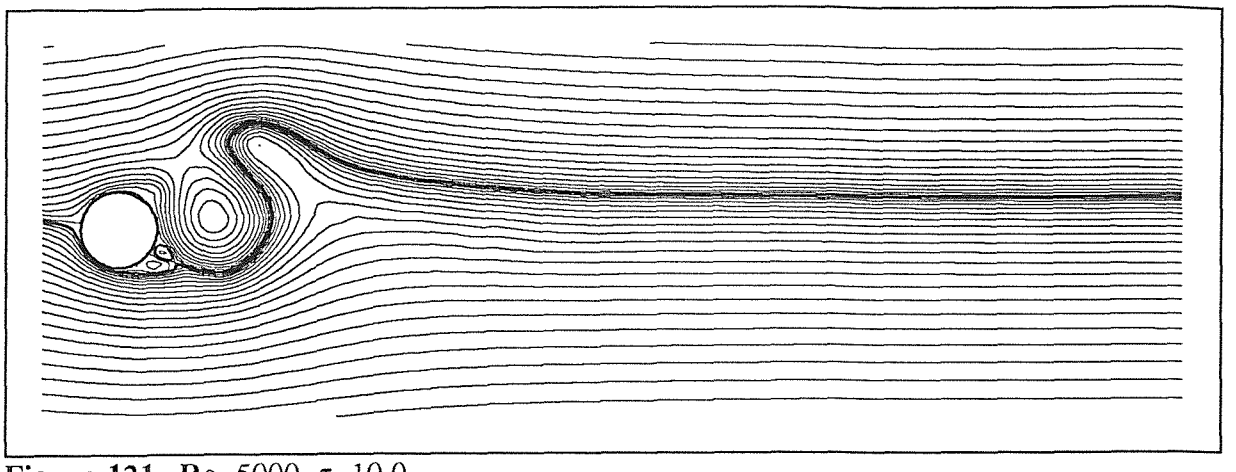


Figure 121 **Re**=5000, τ=10.0

By τ=12.5, as shown in Figure 122, the first pair of vortices have become completely detached from the cylinder and are travelling as a pair in a downstream and transversely upwards direction. A second large eddy-pair is forming at the cylinder surface. This style of shedding, where two vortices travel away from the cylinder as a closely-coupled pair, is seen in many of the high Reynolds number computations. Such a pair are often seen to travel slightly in the transverse direction (in addition to the normal downstream motion) due to their mutual dipole-like induction. In Figure 122 the pair induce a transversely upwards component of motion upon each other, but downwards motion has been found in other runs.

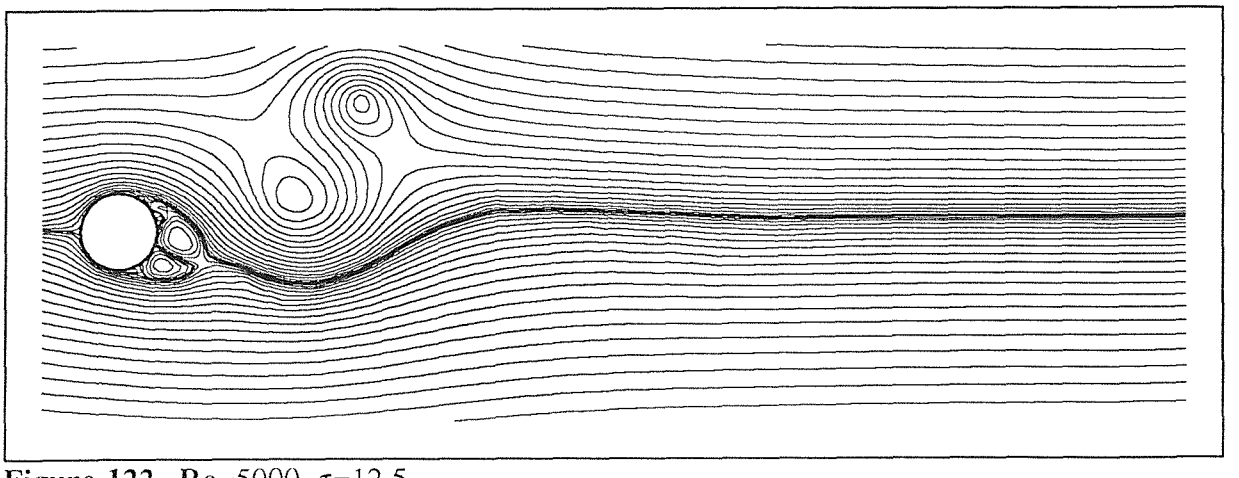


Figure 122 **Re**=5000, τ=12.5

Figure 123 shows the flow at τ=15.0. The initial vortex pair continues to travel downstream and transversely upwards. A third vortex has shed from the cylinder and a fourth eddy is forming at the cylinder surface.

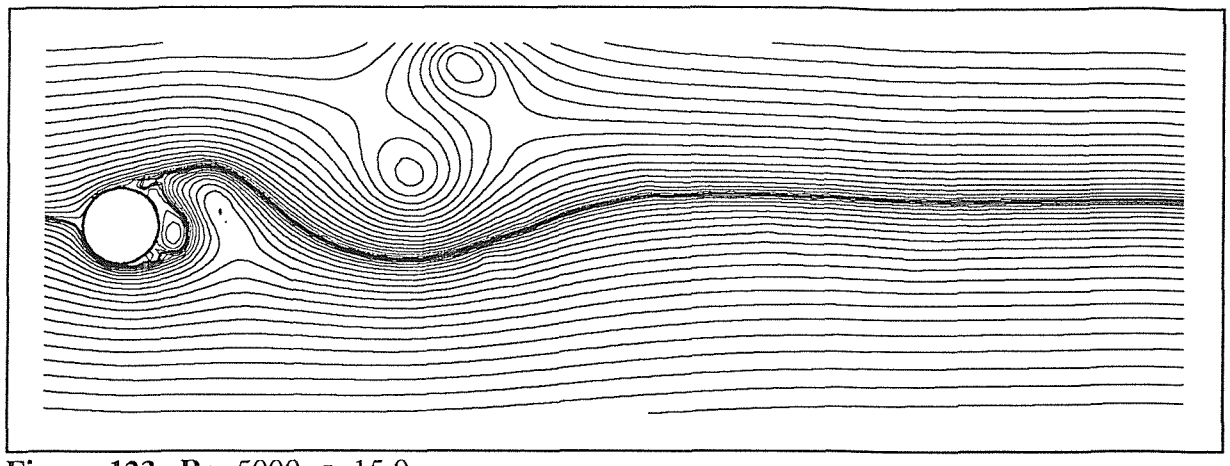


Figure 123 **Re**=5000, τ=15.0

The shedding continues to τ=17.5, as shown in Figure 124, where the eddy forming at the cylinder in the previous time frame has shed as a vortex and a two-eddy system is attached to the cylinder surface.

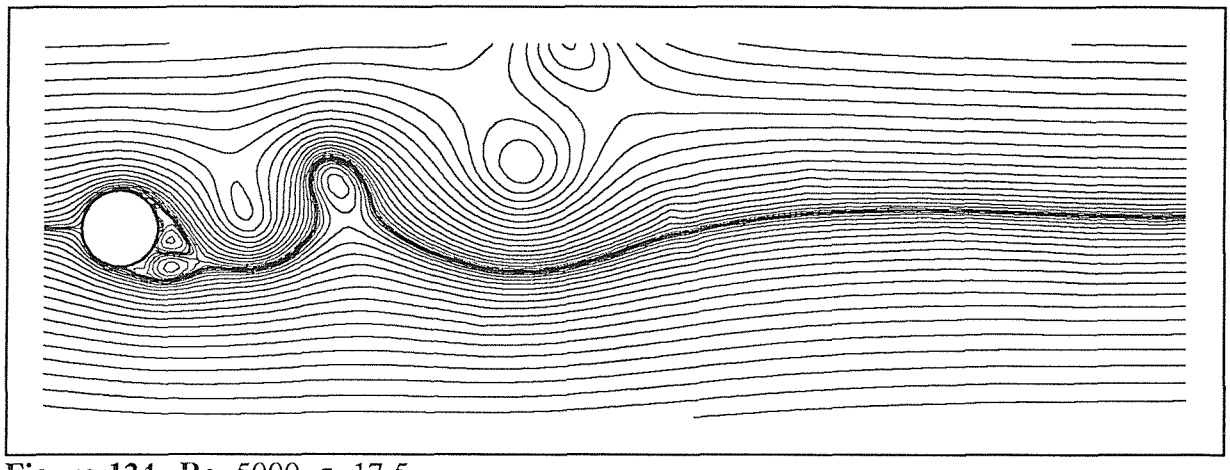


Figure 124 **Re**=5000, τ=17.5

As time progresses, vortices continue to shed in pairs in a similar fashion to those which have shed in the flow diagrams up to τ=17.5. These early eddies travel downstream and transversely upwards, causing a net negative lift on the cylinder. At τ=20.0, the shedding mode changes — eddies continue to shed in pairs but now travel downstream and transversely downwards. The effect of this new shedding mode is to cause a net positive lift on the body.

At τ=40.0, as shown in Figure 125, three shed eddy pairs can be seen travelling downstream and transversely downwards. At this point the shedding mode changes again, returning to a configuration similar to that at early times. An eddy pair grow together at the rear of the cylinder, which is accompanied by a drop in the drag and the corresponding change in the lift takes place over a longer period of time than those seen previously (see Figure 128).

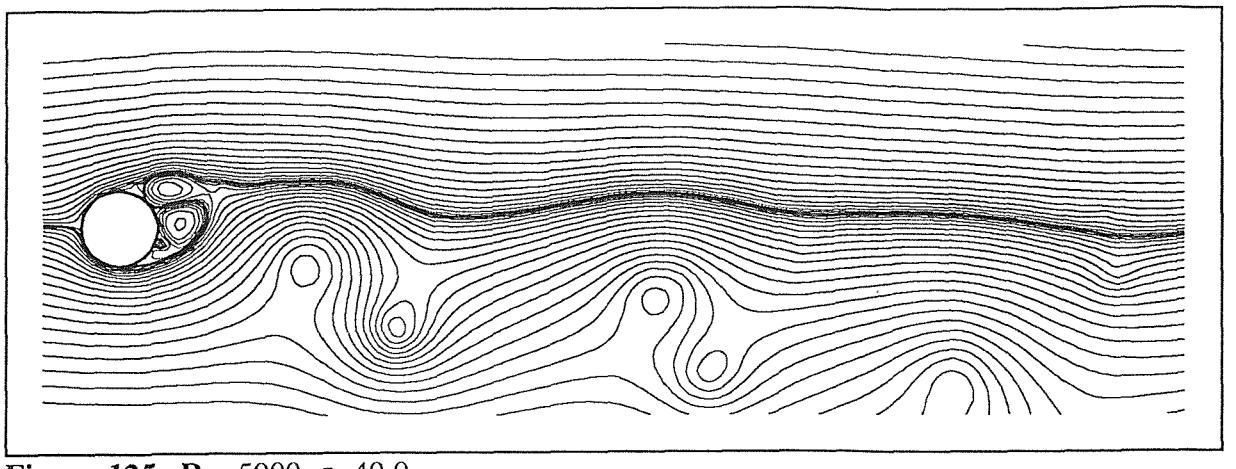


Figure 125 **Re**=5000, τ=40.0

Figure 126 shows the flow at τ=42.5, where the lower eddy/vortex at the cylinder has started to shed. Notice the similarity at the cylinder between this plot and Figure 120, which shows the flow at τ=7.5. This adds weight to the statement that the flow has returned to a mode similar to that occurring after the impulsive start. As time progresses from this point, eddies continue to shed from the cylinder in pairs.

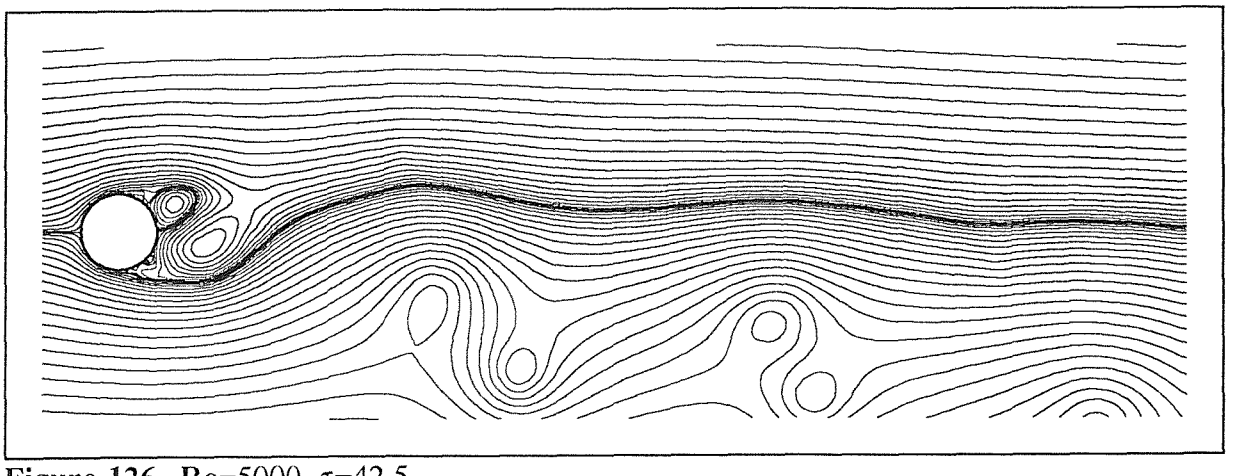


Figure 126 **Re**=5000, τ=42.5

Another shedding pattern starts after τ=55.0, when smaller eddies shed at a slightly higher frequency and singly rather than in pairs. At τ=75.0, as shown in Figure 127, the smaller eddies have formed into a configuration similar to the conventional Karman vortex street similar to that formed at ***Re***=550. The lift graph, which may be used as a measure of the shedding frequency, indicates that the mode is about to change again at this point. This prediction is based on the observation of a wide extremum in the lift graph and a large drop in the drag graph leading up to τ=75.0, phenomena which have accompanied earlier changes in shedding mode. These changes in the force on the body seem to be caused by the formation of an eddy over a longer than typical time interval.

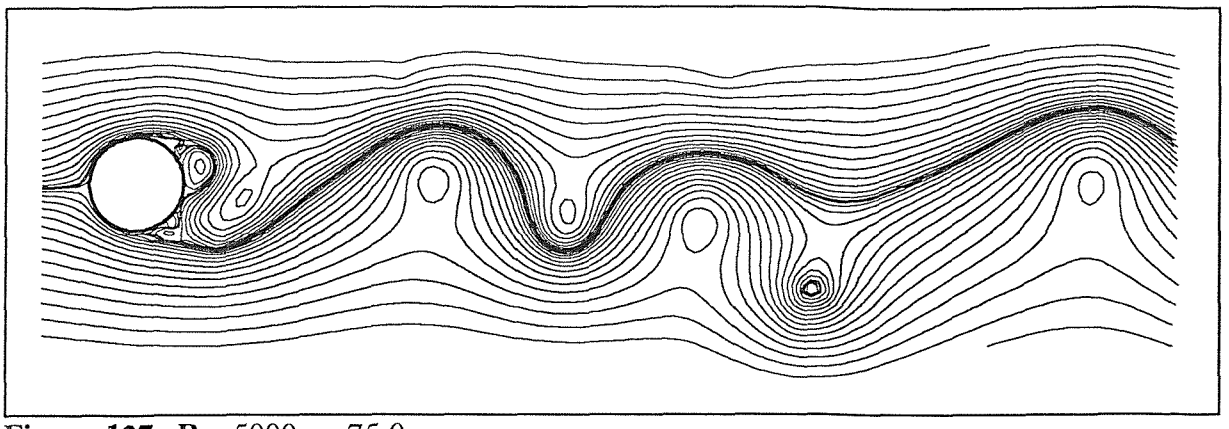


Figure 127 **Re**=5000, τ=75.0

Figure 128 shows the Eulerian forces as calculated by the discrete vortex code for the above run; Figure 129 shows the Lagrangian forces. The lift figures are in good agreement between the two methods; the pattern of behaviour of the drag figures agree, but the average value is lower in the Eulerian case and the magnitude of the variations is also smaller. The Eulerian drag gives the same average result as the Lagrangian drag when using a smaller time step (see p. 134 for results using δt=0.05).

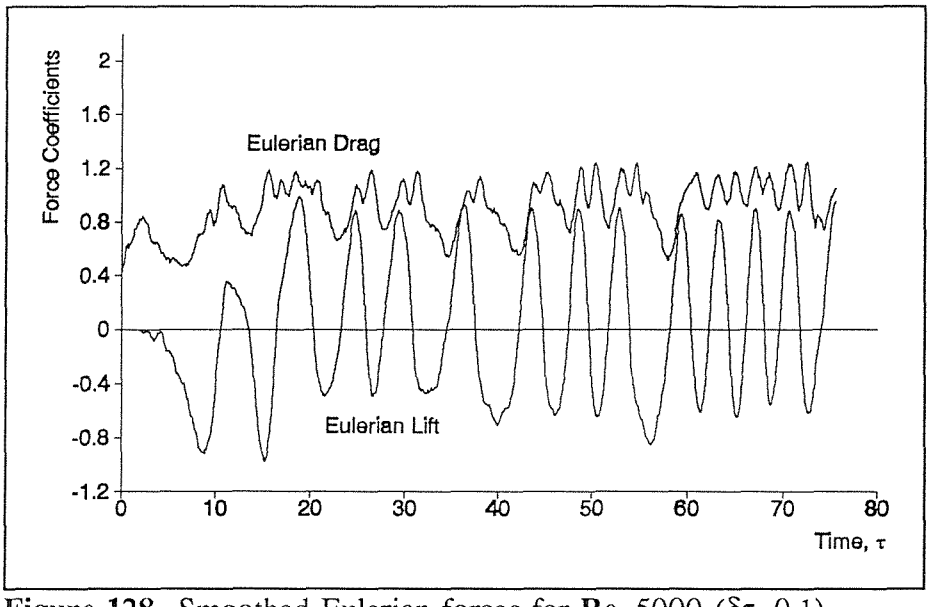


Figure 128 Smoothed Eulerian forces for **Re**=5000 ($\delta\tau$=0.1)

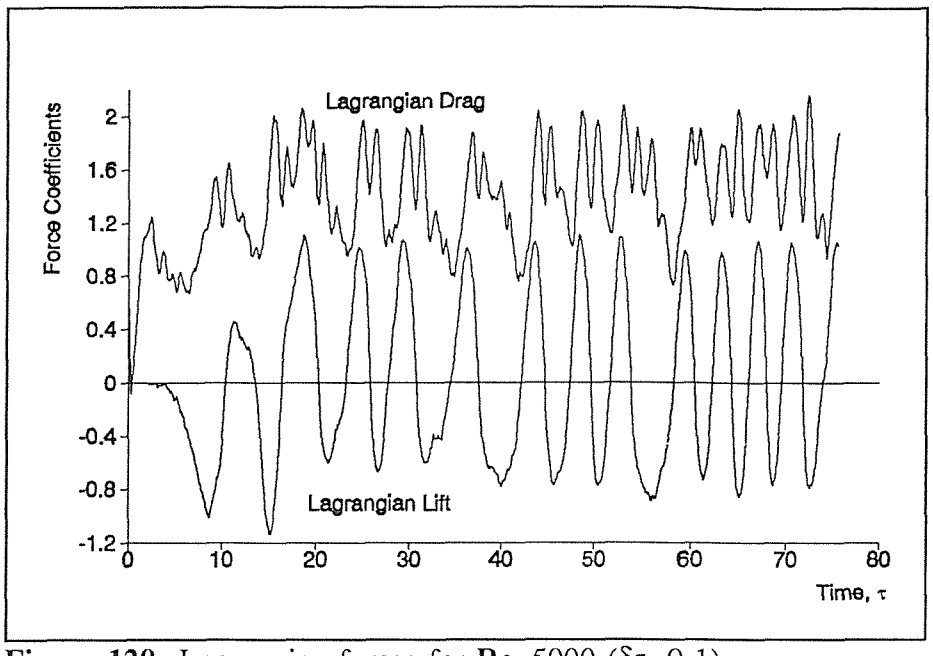


Figure 129 Lagrangian forces for **Re**=5000 ($\delta\tau$=0.1)

As mentioned earlier, the force results (the lift in particular) can be used to classify the eddy shedding pattern. An eddy which sheds from the upper surface of the cylinder tends to accompany a negative lift, and vice versa. Also, when no shedding occurs for a period of time the lift tends to stay nearly constant (for example $\tau \approx 40$ and $\tau \approx 55$ above). The long term average Lagrangian drag is 1.45.

Figure 130-Figure 132 also show results for the flow at Reynolds number 5000, this time using δt=0.05. These show that a slight alteration in the model can cause large changes in the flow, although similar patterns of behaviour occur. The long term average Lagrangian drag is nearly the same for both runs. However, the reduction in time step brings the average value of Eulerian drag close to the value given by the Lagrangian method.

Figure 130 shows a sample streamline plot at τ=37.5, from the alternative run at ***Re***=5000. The flow is not dissimilar to a vertical reflection of Figure 126. Three shed vortex-pairs are travelling downstream and transversely upwards, producing the expected net negative lift.

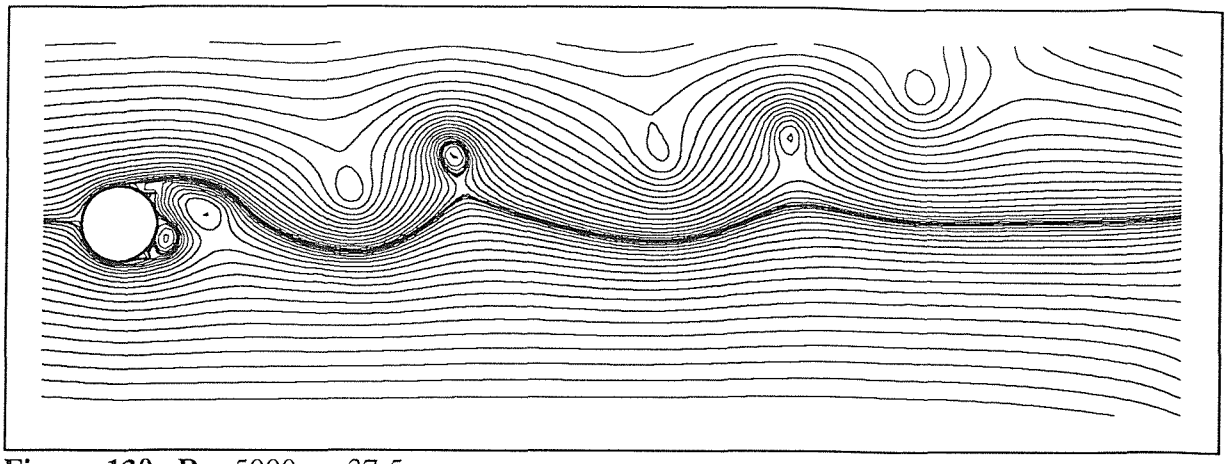


Figure 130 **Re**=5000, τ=37.5

Figure 131 shows the Eulerian force coefficients for the run; Figure 132 shows the Lagrangian force coefficients. The lift calculated by the two methods agrees well as we have seen before. The agreement between the two sets of drag figures has been much improved by reducing the time step from 0.1 to 0.05. The average drag values are now much closer, and the magnitude of the oscillations in the Eulerian graph have increased to nearly the same level as seen in the Lagrangian graph.

The Lagrangian drag graph gives a long term average (from τ=10.0 onwards) of 1.36; the Eulerian drag (δt=0.05) gives 1.21. Franke et al. give a value of 1.68 from a finite difference calculation. The experimental value of average drag, given in Schlichting, is approximately 1.1.

Our results give a Strouhal number of 0.21, Franke et al. report a computed value of 0.245, Schlichting gives an experimental value of 0.21. So our prediction is in better gross agreement then the finite difference calculation, although Reynolds number 5000 is the upper value Franke et al. test, so it may be on the edge of the valid range for their method.

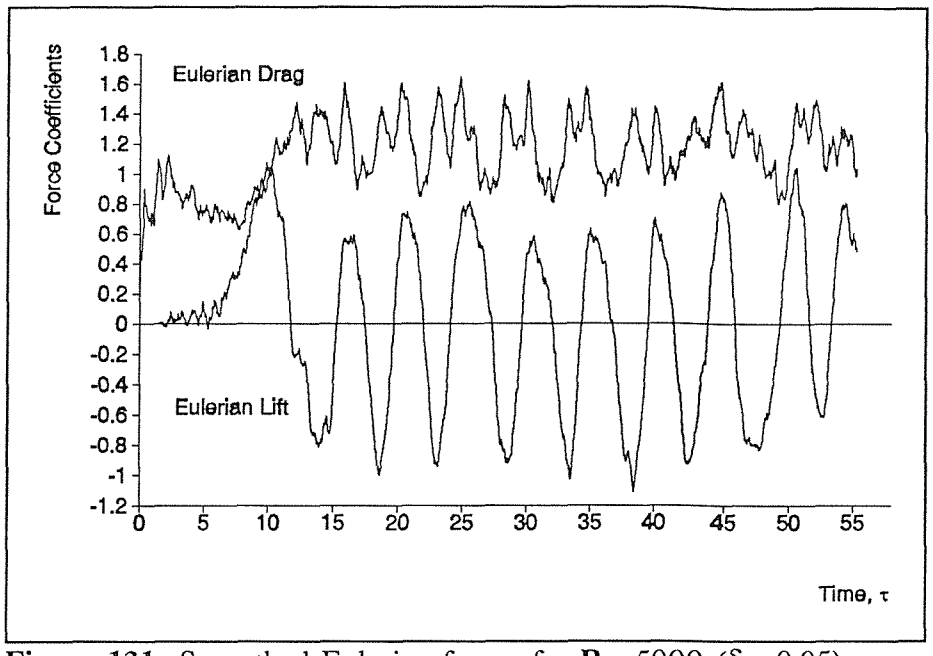


Figure 131 Smoothed Eulerian forces for **Re**=5000 ($\delta\tau$=0.05)

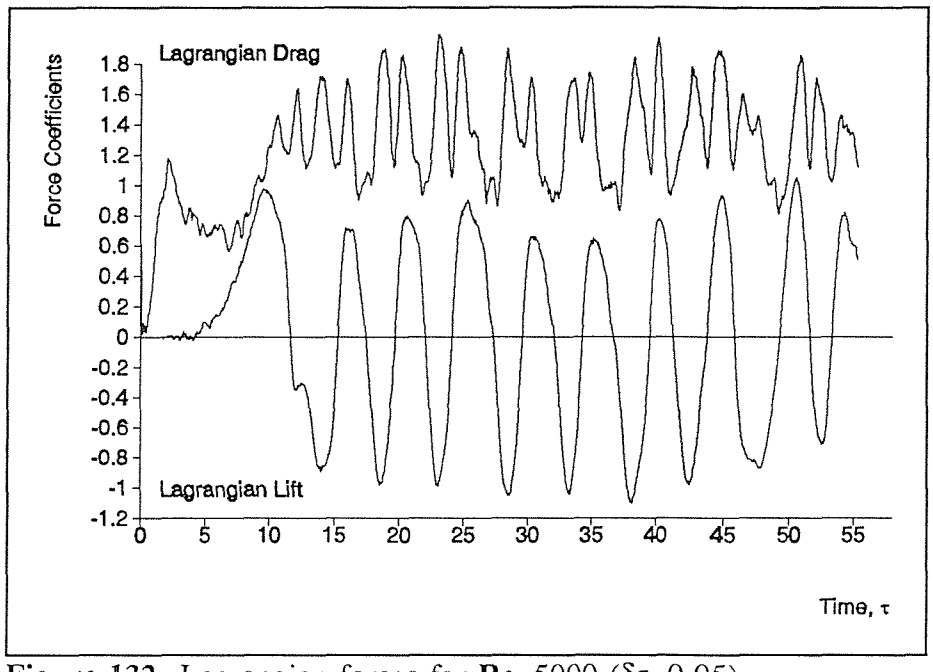


Figure 132 Lagrangian forces for **Re**=5000 ($\delta\tau$=0.05)

The behaviour of the force graphs appears to be broadly similar when comparing the run using δt=0.05 with the run using δt=0.1. Indeed, examining flows at ***Re***=5000 over a range of parameter values we have found that similar shedding modes are set up in all of these flows, although not necessarily at the same non-dimensional time or even in the same order. This leads us to conclude that there are several different shedding modes which can occur and are robust to numerical details, and that the mechanisms for changing from one to another allow transition from any particular mode to several others.

The existence of an eddy shedding mode for Reynolds numbers greater than approximately 40 is a bifurcation of the Navier-Stokes equations (see Bakker pp. 195-203). At Reynolds numbers less than 40 a symmetrical solution is always obtained, giving only one possible state for the system. As Reynolds number rises above 40 symmetry breaking is introduced into solutions. This can occur in either direction, leading to two possible states which the system can enter. The symmetric solution remains a possible, although unstable, solution, thus a pitchfork bifurcation occurs. We conjecture that further increases in Reynolds number, which we know cause stronger nonlinearity, will increase the tendency toward flow instability. Also, the existence of important smaller scale flow features will lead to further bifurcations and hence a multiplicity of possible system states. If this is true, it provides a consistent mathematical explanation for a wide variety of shedding modes being possible from the same model. Minor differences, such as the time step used, may also cause such a system to swap between the different states.

In a precise mathematical solution, each possible state could only be entered from an 'ancestor' state in the bifurcation tree. However, in numerical simulations or experiments, both of which contain noise, if the possible states are 'close' enough (in some parametric space) then the flow may be able to 'jump' between modes. This may explain why the pattern of transitions between eddy shedding modes is extremely complex.

If this bifurcation mechanism is correct, a possible outcome is that chaos will ensue at values greater than some critical Reynolds number. In effect, the system bifurcates infinitely to result in a continuum of possible states. At this point, even the exact mathematical solution can switch repeatedly between different modes of behaviour.

For the problem of flow past a circular cylinder, we have found little mention in the literature of eddies travelling in the transverse direction. Van der Vegt and de Boom (1985) mention that the initial eddy "undergoes large side motions of about two cylinder diameters". Also, we know that a simple consequence of vortex dynamics is that two concentrated regions of opposite-sign vorticity may convect each other, causing a flow pattern similar to a dipole. We note that placing side walls on an experiment will tend to suppress side motions. Finite difference boundary conditions will probably have a similar effect.

5.4.11 Long Time Solution : Re=31,700

Figure 133-Figure 137 show a sample of the flow visualisations from a run of the discrete vortex code at Reynolds number 31,700. The corresponding drag and lift graphs are shown in Figure 138 and Figure 139 respectively. A time step of 0.1 and 100 boundary points were used. The results are similar to those found at Re=5000, although the eddies in the wake are visibly less diffuse. This run was conducted principally for comparison with the discrete

vortex and experimental results of van der Vegt and de Boom (1985). The flow visualisations show a variety of different shedding modes occurring, as with Re=5000.

Figure 133 shows the flow at τ=32.5, where the flow has been shedding pairs of vortices for some time. The shed pairs have been travelling in the transverse downwards direction. Also, a weak vortex has become detached from the main wake and sits slightly above the vortex street.

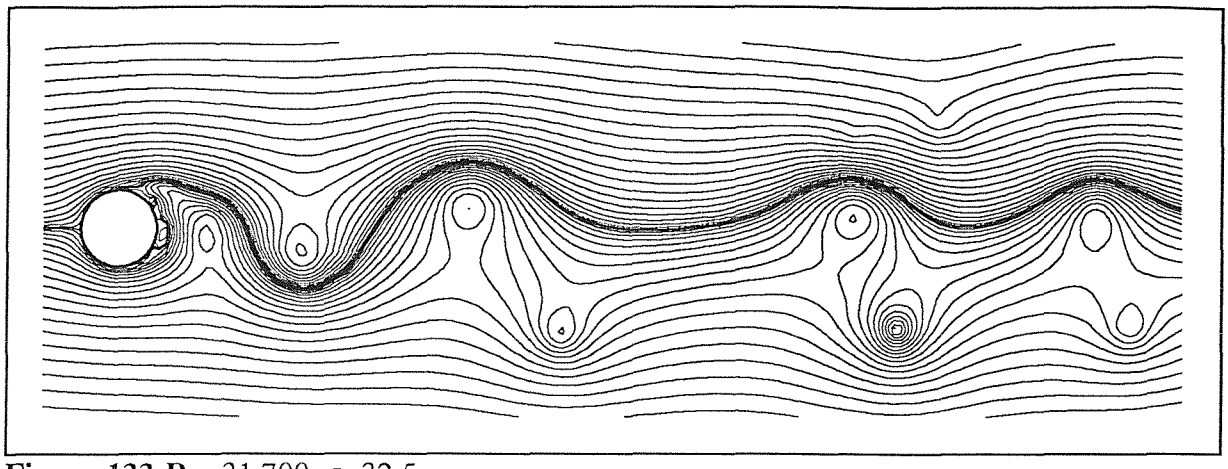


Figure 133 **Re**=31,700, τ=32.5

By τ=60.0 the shedding mode has completely changed (Figure 134). The vortices in the wake have been shed singly and have formed into a near-regular Karman vortex street.

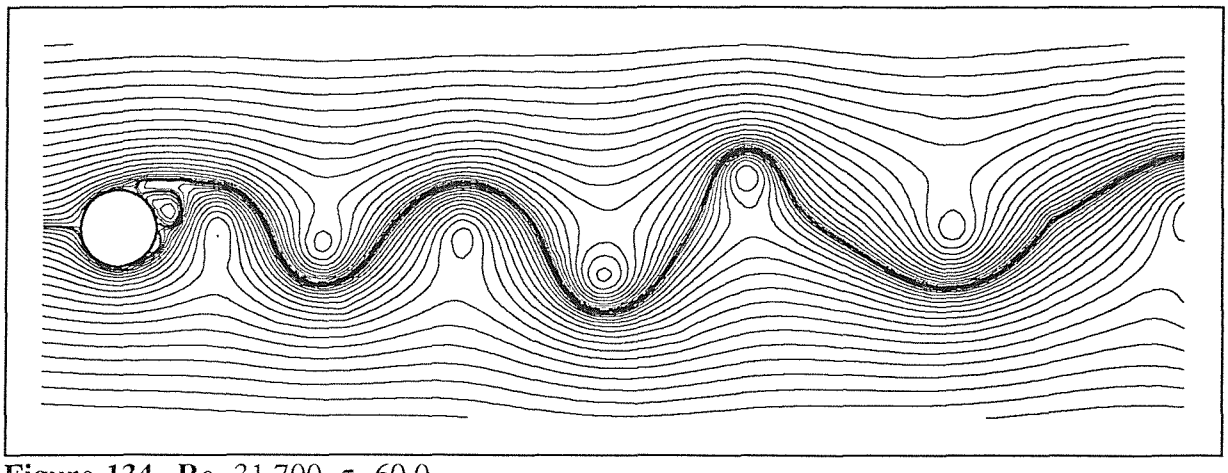


Figure 134 **Re**=31,700, τ=60.0

Figure 135-Figure 137 show the development of the flow from τ=135.0 to τ=140.0, where a state has developed which is similar, near the cylinder, to the initial symmetrical flow. Eddy shedding has been suspended for a period of time while a pair of eddies grow behind the cylinder. Eventually symmetry breaks, in a similar manner to that seen near the start. The upper eddy then breaks away from the body and vortex shedding is resumed. The effect of

this mode change can be seen in the force graphs (Figure 138 and Figure 139) — the lift stays near-constant for almost 10 non-dimensional time units and the drag drops to its lowest value since the initial symmetry breaking.

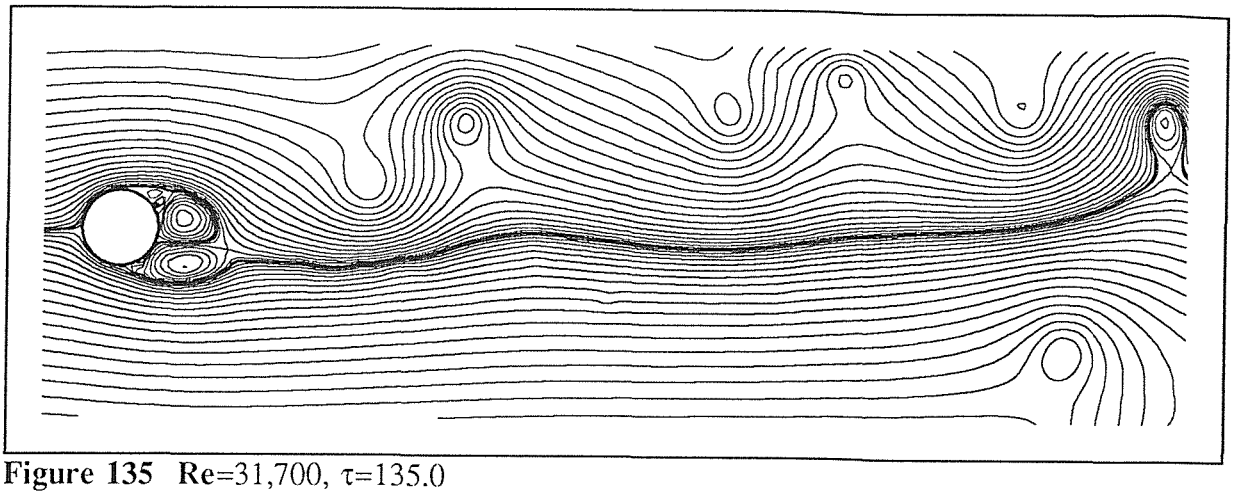


Figure 135 Re=31,700, τ=135.0

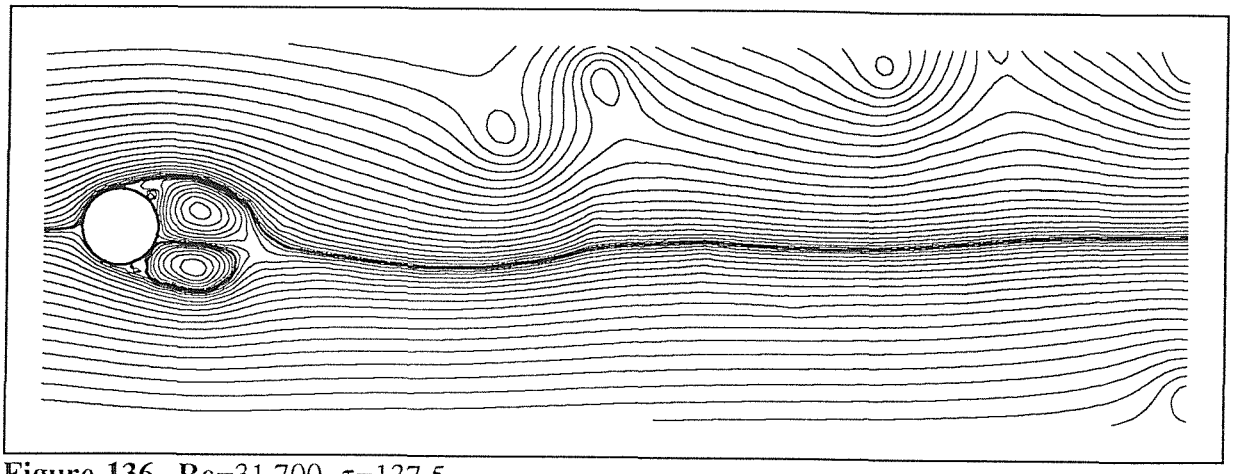


Figure 136 Re=31,700, τ=137.5

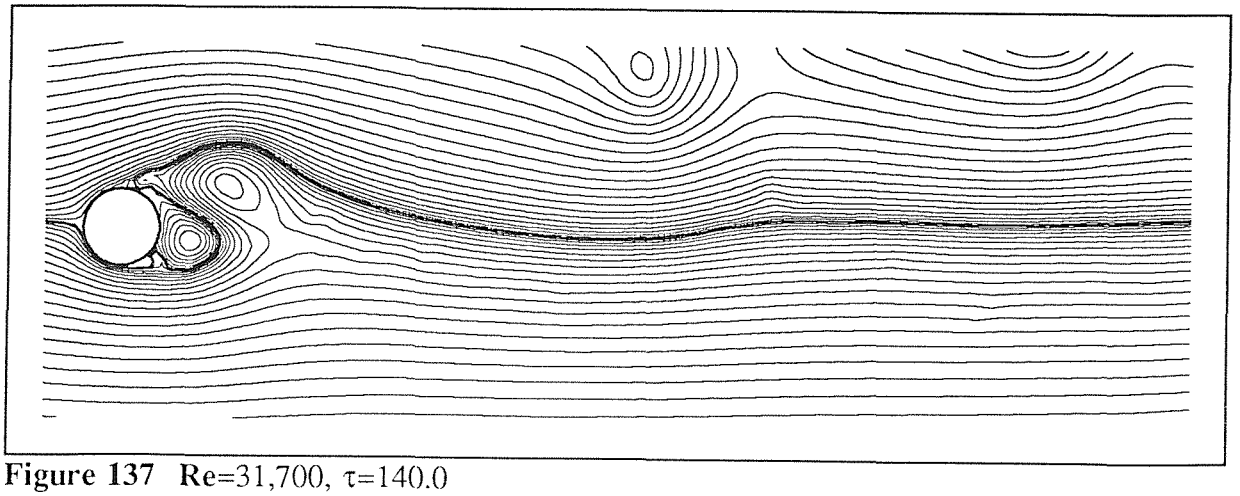


Figure 137 Re=31,700, τ=140.0

Eulerian drag and lift figures for the same run are shown in Figure 138 and Figure 139 (The Lagrangian force calculation had not been added to the code at the stage when this computation was made — a separate run is shown below with both types of force calculation given). The graphs reflect the complex nature of the flow. Examining the lift graph, the shedding frequency clearly changes several times. In several places a drop in drag which is accompanied by a long shedding cycle (eg. $\tau \simeq 40$, $\tau \simeq 80$, $\tau \simeq 135$) indicates a change in shedding mode.

Van der Vegt and de Boom (1985) give force graphs for two experiments, both conducted under the same conditions. Our lift behaviour is in good qualitative agreement with their experiments in the frequency, magnitude and variations of the oscillations. Our Eulerian drag figures are too small, with a long term average of 0.83 using a time step of 0.1 (first run). Van der Vegt and de Boom report an average drag ranging from 1.18 to 1.4 over four different experiments. However their experimental drag figures behave in the same erratic way as our computed values and the experimental lift graph shows times where the lift is almost constant for an extended period. The lift graph shown in Figure 139 gives an average Strouhal number of 0.18, the same value as that given in van der Vegt and de Boom. Perhaps the most significant result from their experiments is that widely varying eddy shedding behaviour is seen, over four experiments conducted in identical conditions. Thus it seems that non-deterministic effects occur in the experiments. Van der Vegt and de Boom also examine the three-dimensional behaviour in the flow, the conclusion is that the large scale structures are almost perfectly two-dimensional.

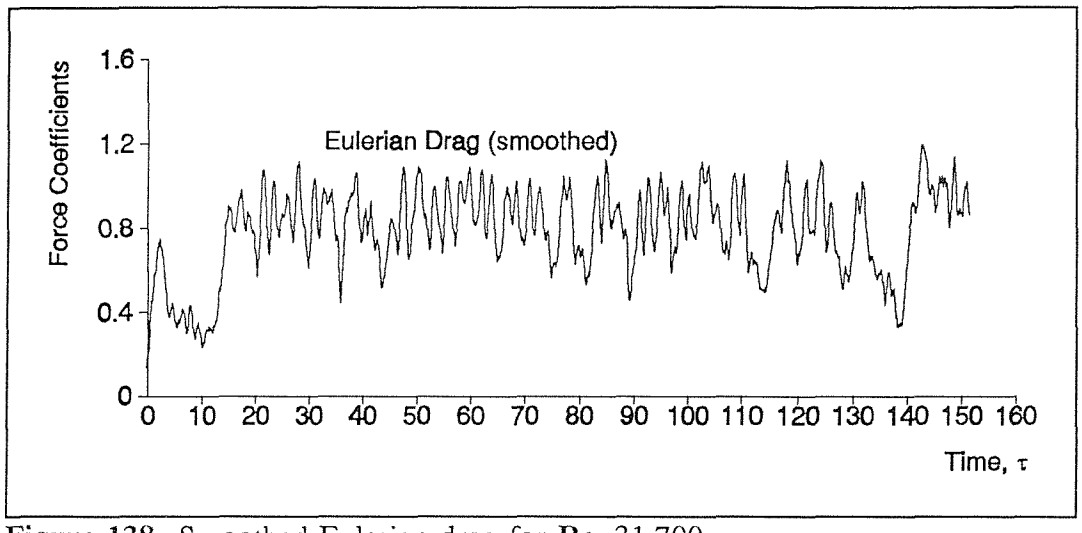


Figure 138 Smoothed Eulerian drag for **Re**=31,700

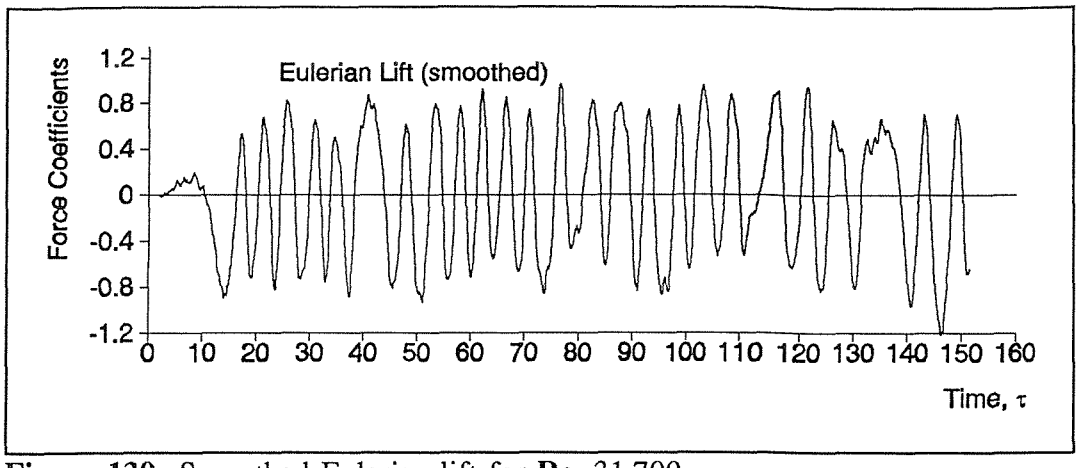


Figure 139 Smoothed Eulerian lift for **Re**=31,700

Figure 140-Figure 143 show the Eulerian and Lagrangian force coefficients for an alternative run, using δt=0.1, 100 boundary points and a different random number seed. The Lagrangian drag is in good agreement with the drag figures of van der Vegt and de Boom. This confirms our finding for ***Re***=5000, that a smaller time step would be needed to bring the Eulerian drag figures into close correspondence with experiments. Storage and CPU limitations have prevented us from taking a run with δt=0.05 as far in non-dimensional time.

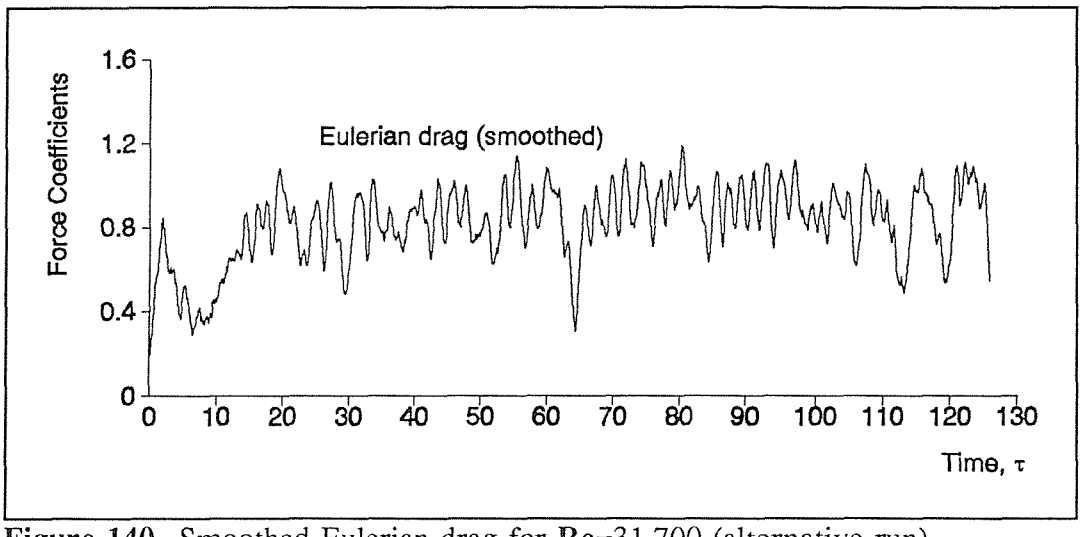


Figure 140 Smoothed Eulerian drag for **Re**=31,700 (alternative run)

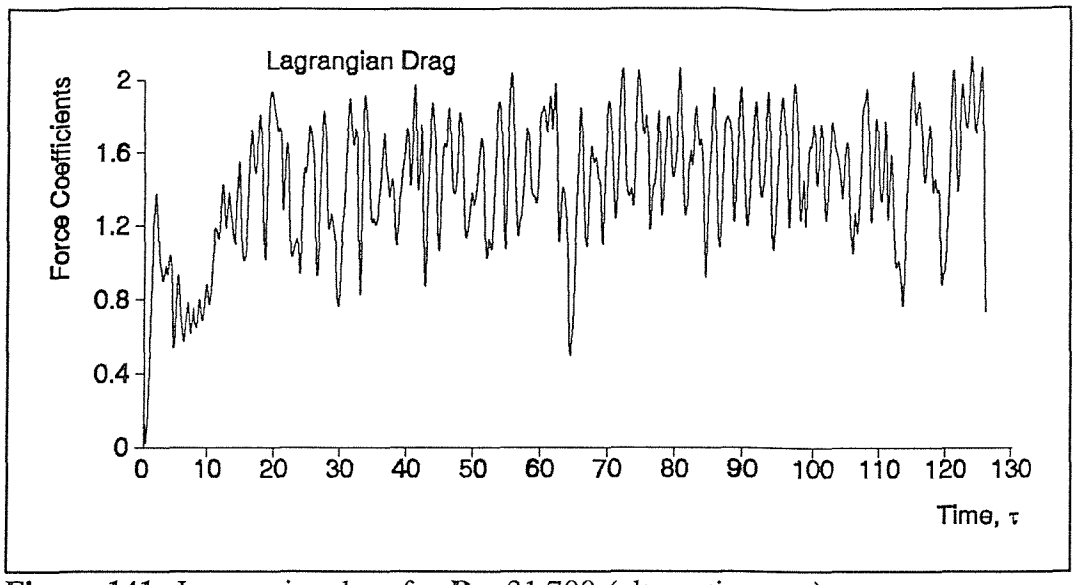


Figure 141 Lagrangian drag for **Re**=31,700 (alternative run)

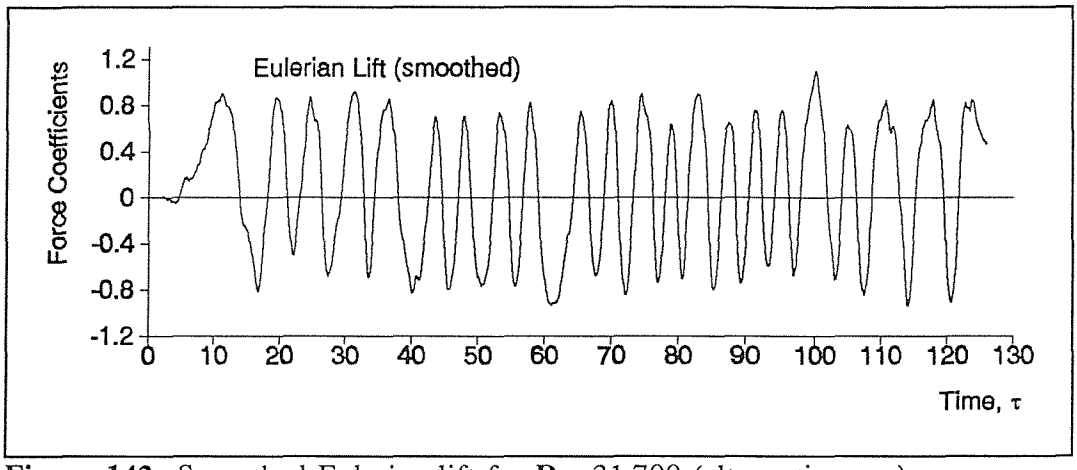


Figure 142 Smoothed Eulerian lift for **Re**=31,700 (alternative run)

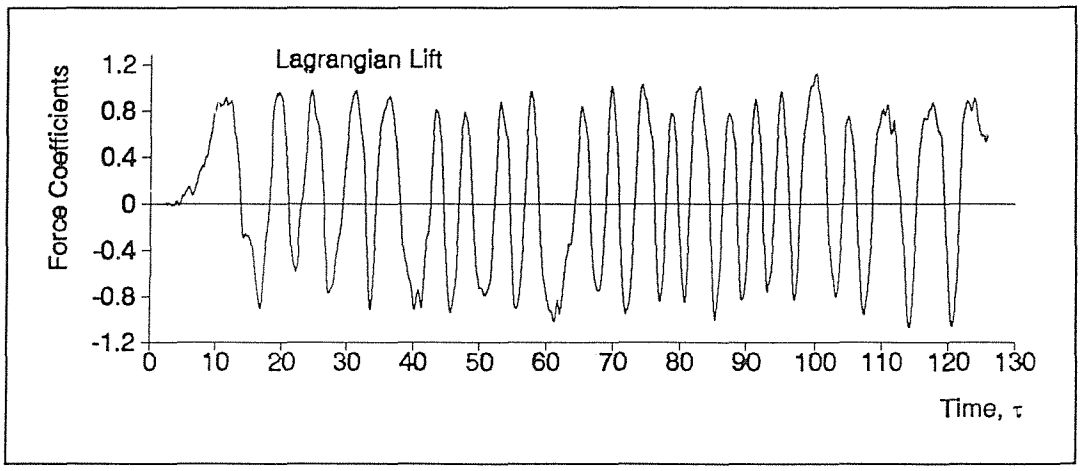


Figure 143 Lagrangian lift for **Re**=31,700 (alternative run)

The alternative run gives an average Strouhal number of 0.19. The average drag values are 0.86 (Eulerian) and 1.50 (Lagrangian). The experimental value for drag given in Schlichting is 1.2.

5.4.12 Long Time Solutions - Conclusion

We have presented the output of the discrete vortex code for long time flow past a translating circular cylinder. The results are in good agreement with the few experimental data available. Agreement is not so good with the finite difference calculations of Franke et al., although the largest discrepancy is at Reynolds number 5000 which is the largest value they have used and thus probably the least accurate. It is also of merit that the force graphs presented by Franke et al. (for Reynolds numbers less than 500) show a very similar pattern of behaviour as our Lagrangian forces. Their drag starts at zero then rises rapidly to a peak value, drops slightly to a plateau, then settles to a regular periodic behaviour.

The irregular behaviour which occurs at the higher Reynolds numbers has been interpreted in terms of the strong nonlinearity in the Navier-Stokes equations. There is evidence, from the experiments of van der Vegt and de Boom, that similar behaviour occurs in the real flows. It would be of great interest to conduct flow visualisation experiments at high Reynolds numbers to fully document this behaviour in real flows.

Experiments have been performed on soap films, by Couder et al. (1989) , in order to examine the behaviour of a genuinely two-dimensional fluid. The experimental photographs display a very similar erratic behaviour to our high Reynolds number computations. Vortices are clearly seen to be transversely displaced, typically in pairs.

5.5 Test Case 2 — Rotating Circular Cylinder

5.5.1 Setting up the Problems

The second test case used for evaluation of the method is prediction of the impulsively started flow past a rotating circular cylinder. Results from the discrete vortex method will be compared to numerical and experimental results from Badr et al. (1990).

The flows considered are all at Reynolds number 1000; Badr et al. state that results are similar at Reynolds numbers 200 and 500. The authors postulate that the similarity at the lower Reynolds numbers is on the 'macro' scale and that flows differ on a 'micro' scale, principally because secondary vortices form less readily at lower Reynolds numbers. We could find no experiments for rotating cylinders at higher Reynolds numbers. Badr et al.

present a very short time numerical calculation for Reynolds number 10,000. An 'exploded' view of the flow is given in order to make a comparison with an earlier boundary layer calculation, see Ece et al. (1984). This meant that our results for Reynolds number 10,000 would not have been directly comparable without substantial recoding for a specific case. Therefore we chose not to attempt a comparison at this Reynolds number.

We have used the same ratios of surface speeds as in Badr et al., α=0.5, 1.0, 2.0 and 3.0. The results at α=0.5 and 1.0 are broadly similar to those at α=0, as given in previous sections. A simple periodic behaviour is adopted, with large eddies shedding from alternate sides of the cylinder. In these flows, the major effect of the rotation is to create proportionately more vorticity in the opposite rotational sense to the motion of the cylinder. At the higher rotational speeds of α=2.0 and 3.0 a different mode of behaviour is approached. A closed layer of fluid is observed, which rotates with the cylinder; dynamic behaviour such as eddy shedding occurs outside this layer.

The moving boundary is introduced by placing a point vortex of circulation $\pi\alpha$ at the centre of the cylinder. The velocity induced by this vortex is added to all velocity calculations and the circulation of $\pi\alpha$ is added to the zero circulation condition of equation (34). Thus the total circulation around a contour far from the body will remain zero, a condition needed to keep the pressure single valued (see Badr et al.).

The additional point vortex is taken into account in the force calculations by considering it as a constant strength vortex sheet spread over the surface of the cylinder (these two vorticity distributions give the same velocity distribution outside the cylinder). Then, inserting the sheet into equation (39) leads to the following extra term in the drag and lift,

$$C_L' = \frac{\pi\alpha}{N_b}\sum_{1}^{N_b} u_s n_x$$
$$C_D' = \frac{\pi\alpha}{N_b}\sum_{1}^{N_b} u_s n_y \qquad (69)$$

where N_b is the number of boundary points, u_s is the tangential surface speed at the centre of a panel element and (n_x,n_y) is the unit normal to the cylinder. These terms are added to the usual force formulae, but in practice make an almost insignificant contribution.

In Badr et al. the time is non-dimensionalised on the cylinder radius, thus their results at time 2τ are comparable to our results at time τ. Where their results are shown in this thesis, the times have already been modified. The scheme used by Badr et al. is a composite finite

difference and spectral scheme, but in the interests of brevity will be referred to as a finite difference scheme.

5.5.2 Re=1000, α=0.5

Figure 144 - Figure 164 show a sequence of flow visualisations at *Re*=1000 and α=0.5. The output from the vortex code is shown alongside the output from the finite difference code and the experimental photographs of Badr et al. A discussion is given at each time frame of the differences between both of the numerical methods and the experiment. This run was made using 200 boundary points and a time step of 0.05, which was found to give a very small time-shift for this problem.

Figure 144 shows a streamline plot generated by the discrete vortex code at τ=0.5, with Figure 145 giving the equivalent plot as calculated by the finite difference code. In both cases a thin wake has formed on the downstream side of the cylinder. The wake has been distorted, as compared to the α=0 solution, in the direction of the rotation of the cylinder (anticlockwise).

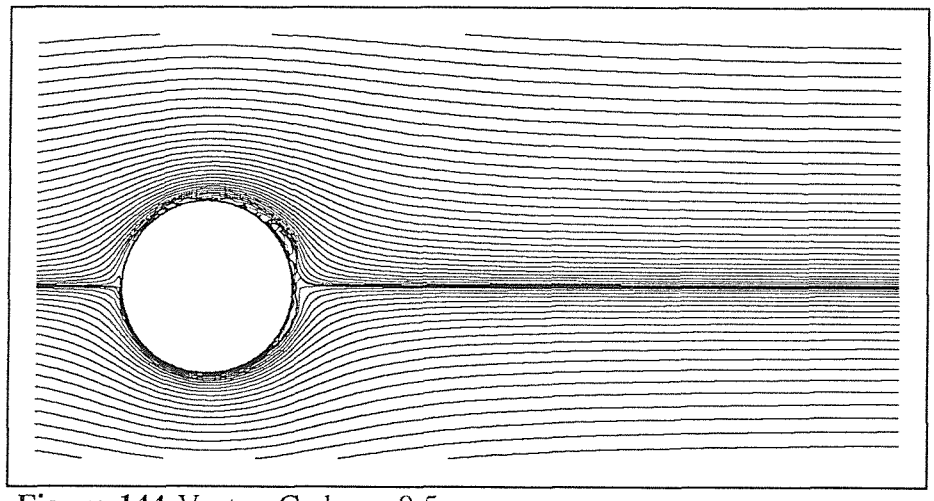


Figure 144 Vortex Code, τ=0.5

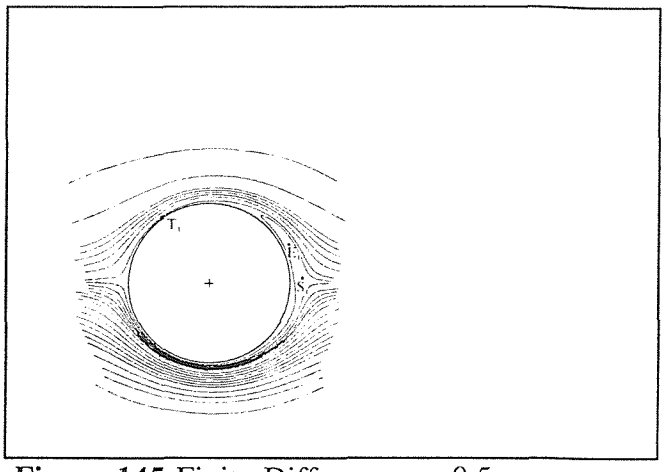


Figure 145 Finite Difference, τ=0.5

In Figure 146 and Figure 147, which show the plots from the two numerical schemes at τ=1.0, two eddies are clearly visible in the wake. The eddies have similar shape and dimension in both calculations — the upper eddy is physically larger and will contain more total vorticity. The moving surface of the rotating cylinder generates a net clockwise (negative) vorticity, of which the bulk eventually accumulates in the upper wake eddy. This makes it physically larger than the lower eddy.

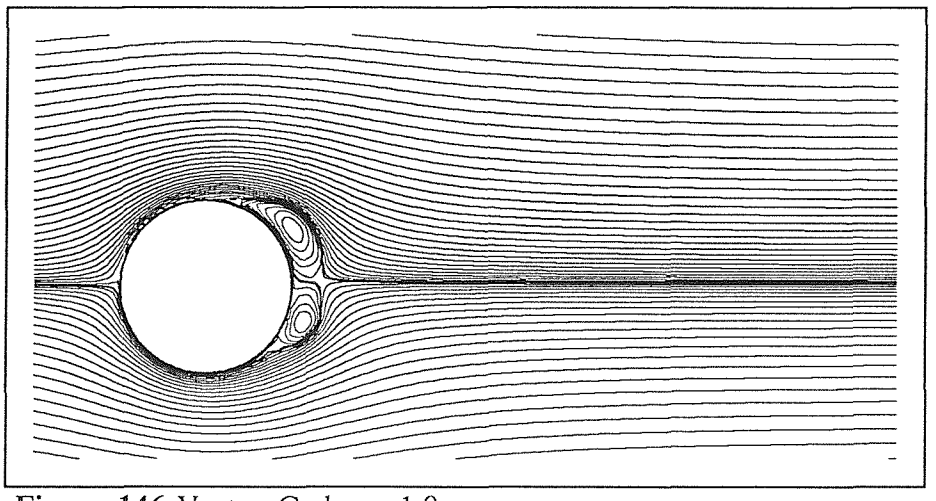


Figure 146 Vortex Code, τ=1.0

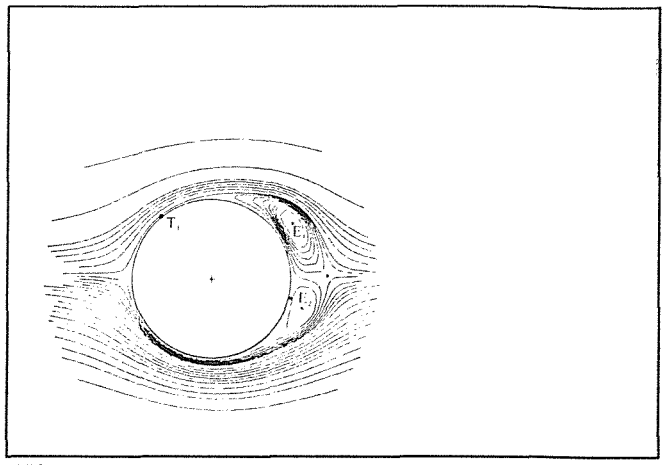


Figure 147 Finite Difference, τ=1.0

Figure 148 and Figure 150 show the continuation of the sequence of plots generated by the two numerical schemes; Figure 149 shows the experimental visualisation at the same non-dimensional time of 1.5. Both schemes have predicted the experimental flow accurately. The two eddies have become enlarged since the previous time frame and the upper eddy is still more rotationally intense. Secondary eddies have clearly formed in both numerical schemes. In the experiment, a secondary vortex similar to those predicted can be seen on the upper cylinder surface, the lower surface is obscured by a shadow.

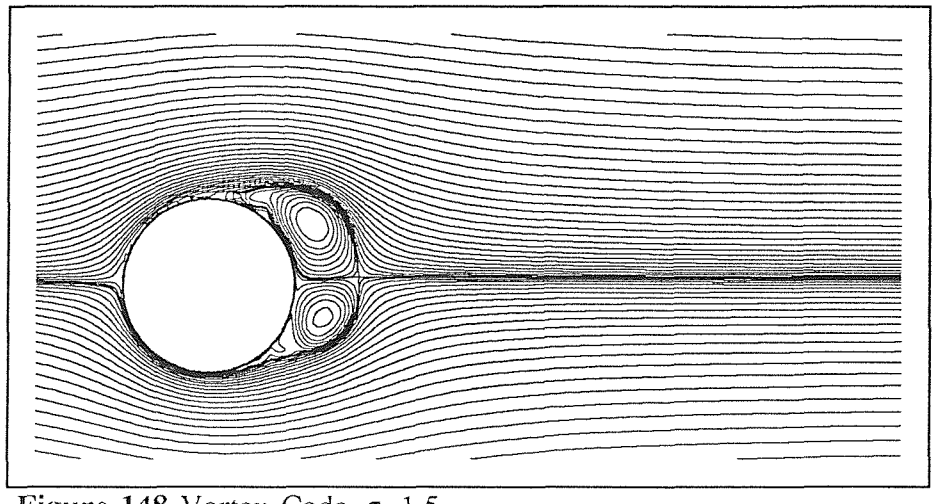


Figure 148 Vortex Code, τ=1.5

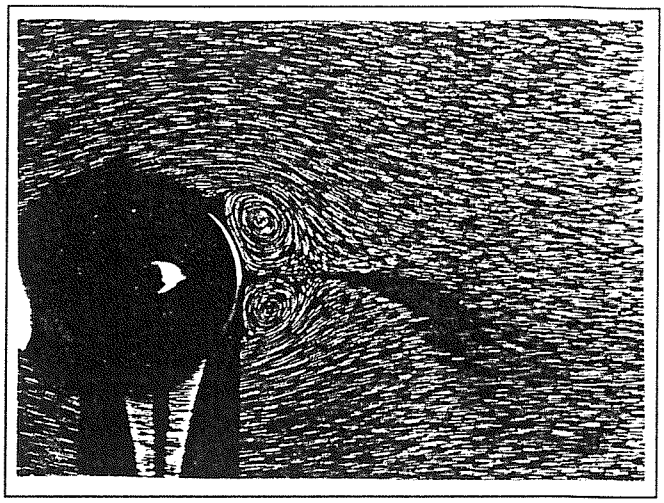


Figure 149 Experiment, τ=1.5

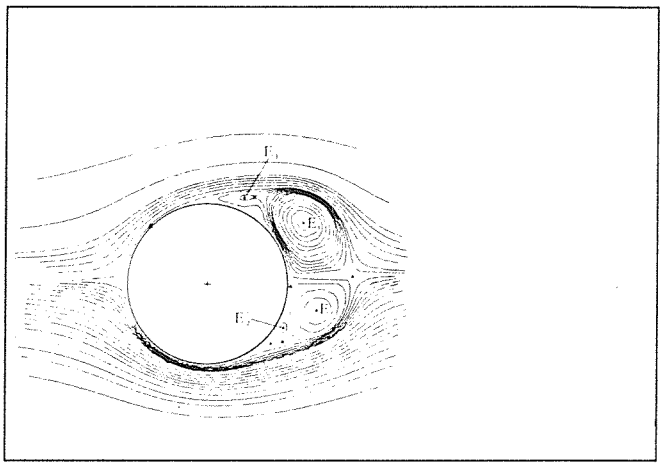


Figure 150 Finite Difference, τ=1.5

Figure 151, Figure 152 and Figure 153 show visualisations at τ=2.0. The vortex code is in close agreement with the finite difference calculation, the shape and size of the main eddies being very similar in both. The experiment, although less clear, appears to have been well predicted by both schemes. The upper secondary vortex can be seen clearly in the experiment and has the same form as in the computed solutions. The lower secondary vortex is partly obscured but there is evidence that it has become joined with the outer flow, as predicted by both calculations.

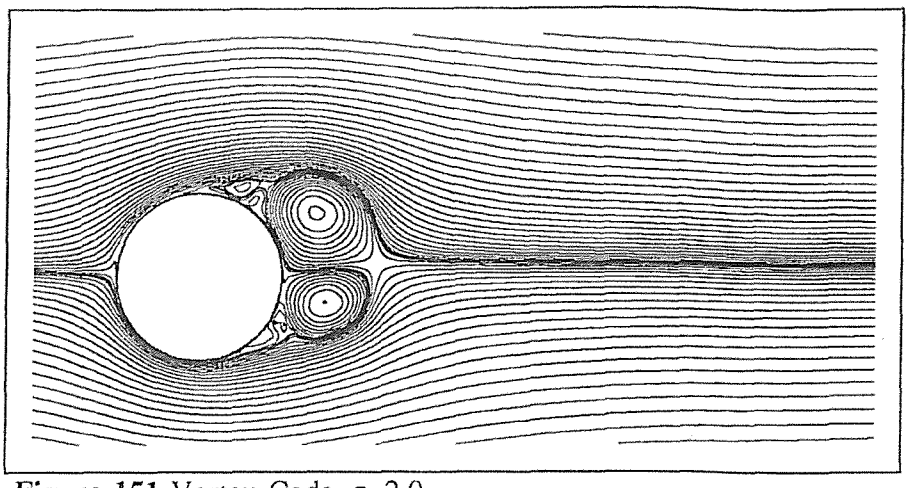


Figure 151 Vortex Code, τ=2.0

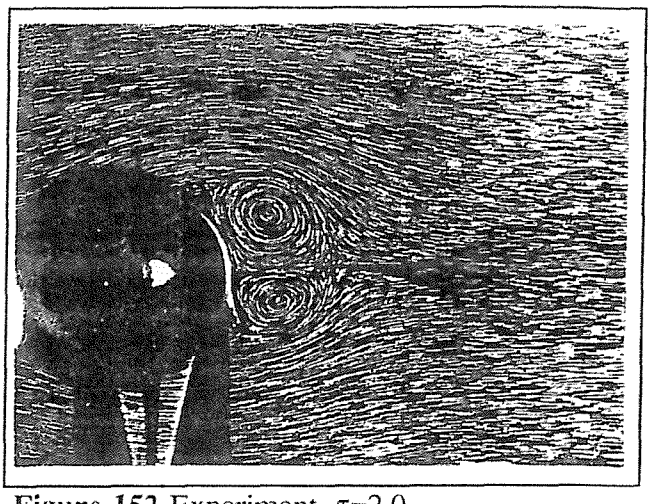


Figure 152 Experiment, τ=2.0

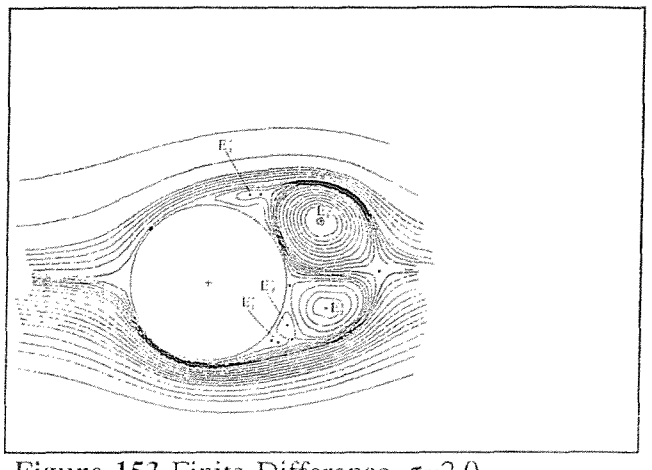


Figure 153 Finite Difference, τ=2.0

No experiment is given in Badr et al. for τ=3.0, but Figure 154 and Figure 155 show the streamlines computed by the two methods. The two calculations remain in close agreement,

although the upper secondary vortex is more tightly rolled-up in the discrete vortex solution. At this time, the upper main eddy has started to shed from the body and move downstream from the cylinder.

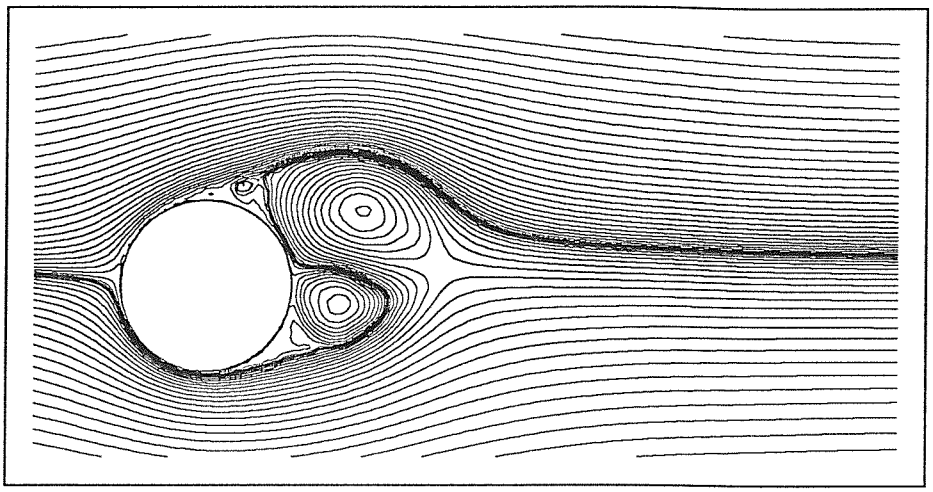


Figure 154 Vortex Code, τ=3.0

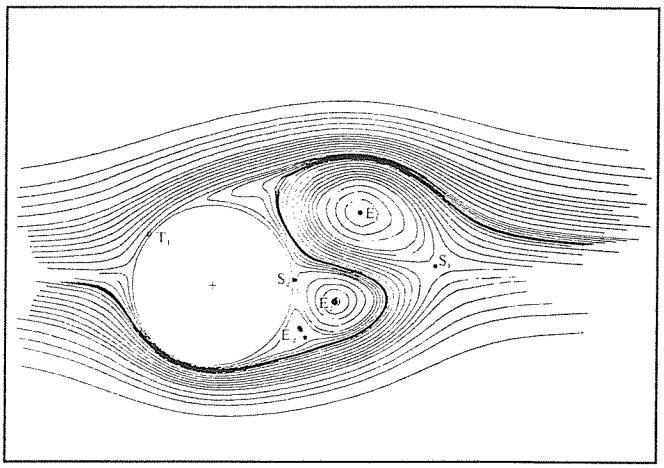


Figure 155 Finite Difference, τ=3.0

Figure 156 and Figure 157 show the progression of the two computations to τ=4.0. The two methods are still in close agreement, although slight differences can be seen. Details near the boundary appear to be slightly different, although a slight time-shift may be responsible for this, because the flow is changing very rapidly at this time.

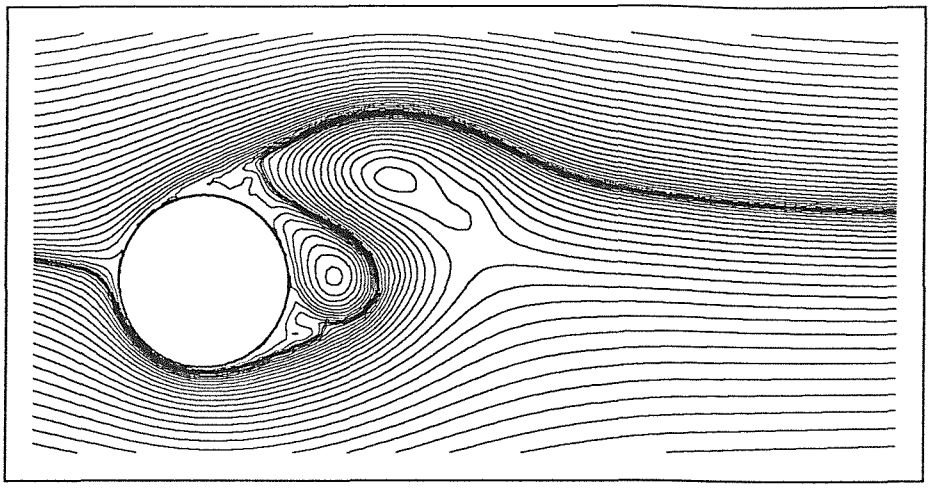


Figure 156 Vortex Code, τ=4.0

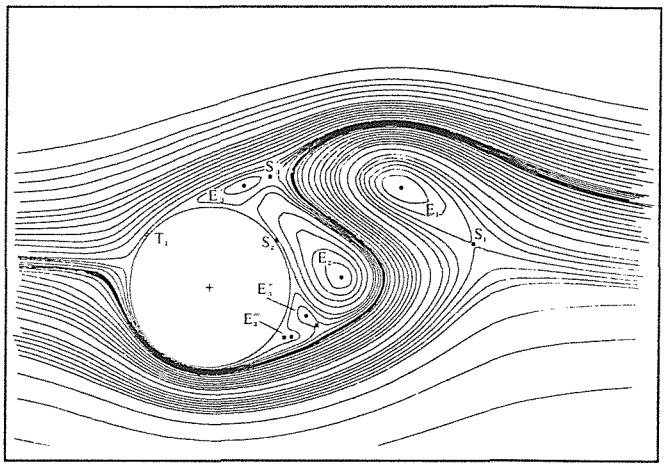


Figure 157 Finite Difference, τ=4.0

At τ=5.5, an experimental visualisation is available and is shown in Figure 159. The two numerical visualisations at this time are shown in Figure 158 and Figure 160, and are still in broad agreement. In the computed flows, both initial main eddies have now fully shed and the original lower eddy has 'opened out' on its upper side. The experiment shows the original lower main eddy closer to the cylinder then predicted by the calculations. A complex secondary vortex system has developed in the experiment and both computations, with one vortex on the upper surface and two counter-rotating vortices on the lower surface.

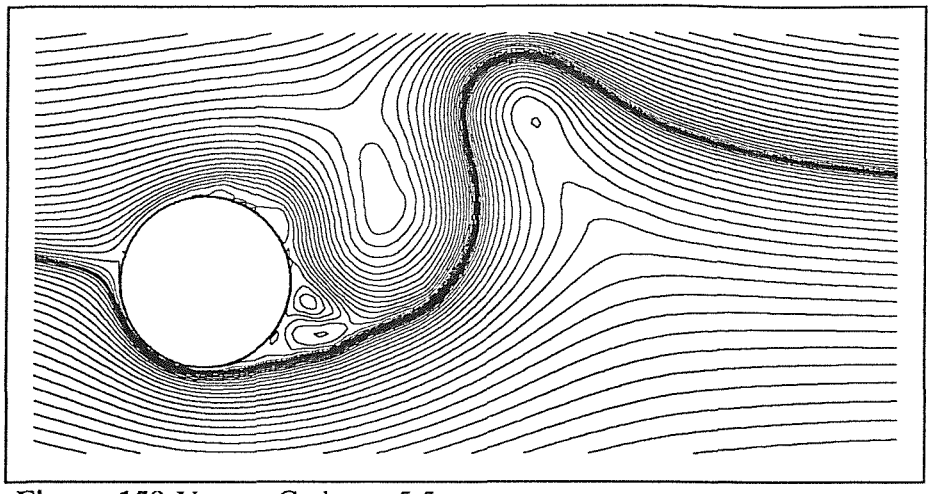


Figure 158 Vortex Code, τ=5.5

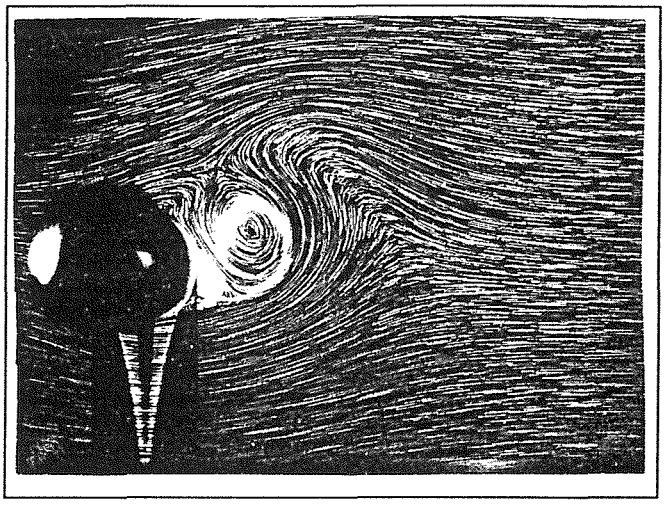


Figure 159 Experiment, τ=5.5

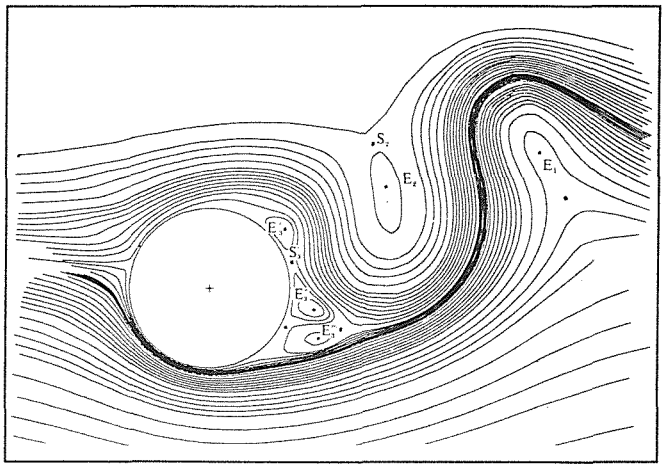


Figure 160 Finite Difference, τ=5.5

Figure 161 and Figure 162 show the two computed solutions at τ=6.5; no experimental photograph is supplied in our source paper for this time frame. The computations still agree

well, both show the original main eddies further downstream than at τ=5.5 and a second eddy pair forming. The new eddy on the upper surface has become dominant.

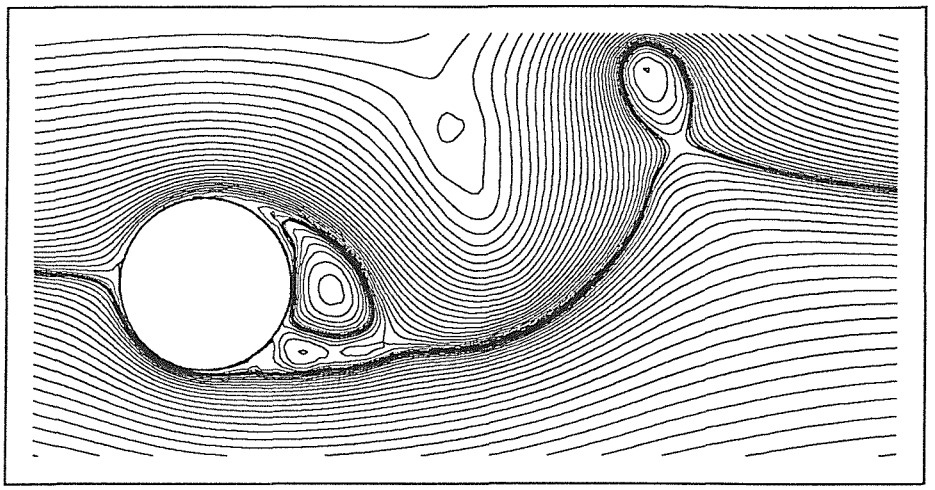


Figure 161 Vortex Code, τ=6.5

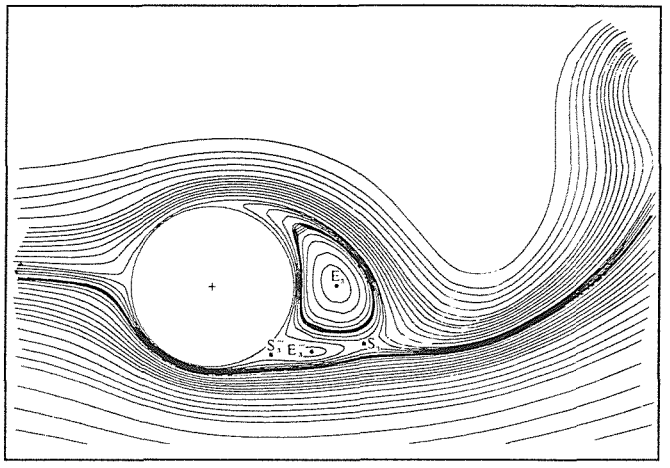


Figure 162 Finite Difference, τ=6.5

In Figure 163 and Figure 164, the two calculated solutions have reached τ=8.0; agreement is still good. At this time, the eddy which was forming at τ=6.5 has started to shed and a strong lower eddy is starting to form.

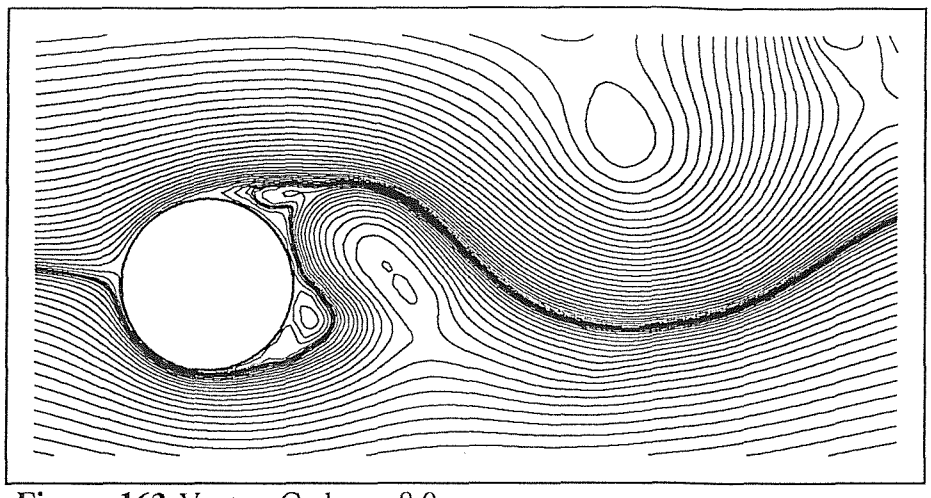


Figure 163 Vortex Code, τ=8.0

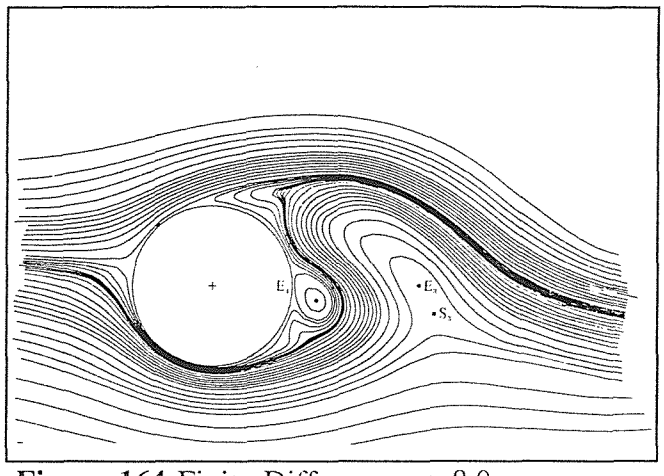


Figure 164 Finite Difference, τ=8.0

Figure 165 is a graph of the Eulerian force coefficients produced by the vortex code for the same run which produced the above visualisations; Figure 166 is a graph of the Lagrangian force coefficients for the same run. The Eulerian calculation has been smoothed, in order to show the underlying trends, by plotting a moving average of three consecutive data points. The two results are in broad agreement. After the initial peak caused by the impulsive start, the drag has a small magnitude in the Lagrangian calculation and gradually builds up. The Eulerian drag appears to be inaccurate for early times, in agreement with our previous findings. Both drag graphs have a dual-periodic behaviour (ie. there are two dominant frequencies) corresponding to the shedding of two different sizes of eddies from the upper and lower sides of the cylinder. The lift shows a periodic behaviour, remaining negative for almost the entire run, on the Lagrangian graph.

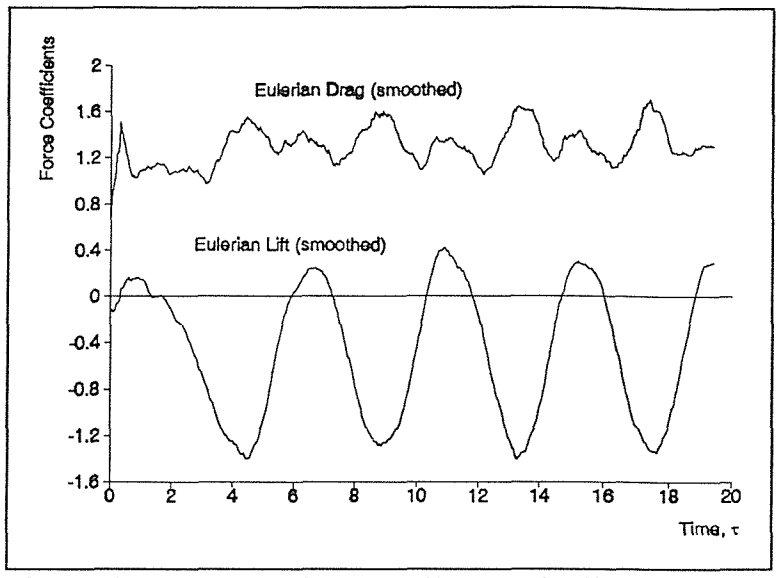


Figure 165 Eulerian force coefficients for **Re**=1000, α=0.5

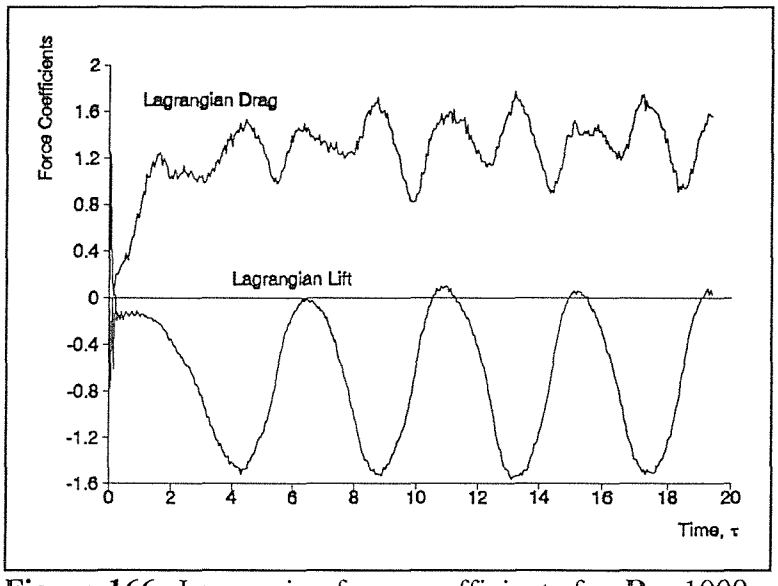


Figure 166 Lagrangian force coefficients for **Re**=1000, α=0.5

The force figures given by Badr et al. show the same behaviour as those given here. The drag has a dual periodicity, of the same characteristic shape as that produced by the discrete vortex method, and peak values around 1.9. The lift has a simple periodicity with the maximum magnitude values close to -2.0. We could not find experimental values for force coefficients. The inviscid value of lift for this set of problems is $-2\pi\alpha$, much larger in magnitude than even the peaks of either scheme.

We have demonstrated close agreement, at α=0.5, between the discrete vortex calculation and the finite difference calculation of Badr et el. Agreement with the experiment is close for

early times, but the final experimental photograph available shows both computations exhibiting a slightly different behaviour from the experiment. It would be of interest to obtain experimental photographs for the later stages of the flow to determine whether the apparent error is transient or whether the numerical and experimental solutions have started to diverge away from each other.

5.5.3 Re=1000, α=1.0

Figure 167 - Figure 173 follow Badr et al. in demonstrating the periodicity of the flow at α=1.0. A few time frames are shown which demonstrate that the flow repeatedly returns to a certain configuration of eddies. The drag and lift graphs, given on p. 157, also confirm the periodic behaviour of the flow. The discrete vortex code was run with 200 boundary points to represent the cylinder and a time step of 0.033.

The first time frame is at τ=3.5, with Figure 167, Figure 168 and Figure 169 showing the streamlines from the discrete vortex code, experimental photograph and finite difference code respectively. All three visualisations are in broad agreement, the upper main eddy has shed in a manner similar to the flow at α=0.5 (see Figure 156 and Figure 157, p. 149). A complex three-eddy system is attached to the surface of the cylinder.

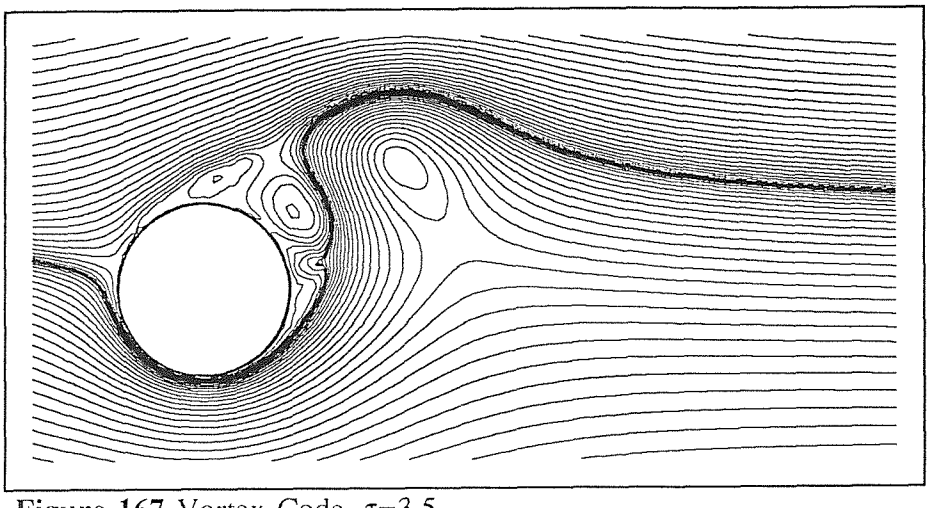


Figure 167 Vortex Code, τ=3.5

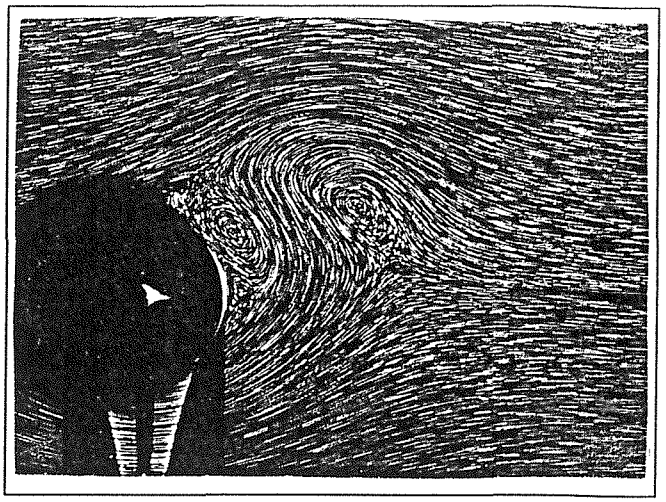


Figure 168 Experiment, τ=3.5

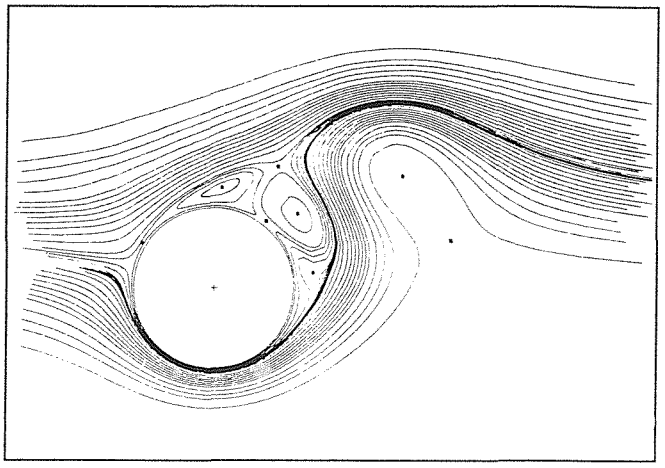


Figure 169 Finite Difference, τ=3.5

Figure 171 and Figure 172 show the state of the computed flow at τ=8.0. A similar configuration of eddies has developed to that shown above, with a vortex street forming behind the cylinder and a three-eddy system attached to the cylinder surface. Some differences are apparent between the vortex code solution and the finite difference solution. The flow generated by the discrete vortex method at τ=7.5 is shown in Figure 170. Studying the two adjacent time frames, it is reasonable to conclude that, at a time between τ=7.5 and 8.0, the vortex code would be close to the finite difference solution at τ=8.0. In other words, a slight time-shift has caused the solutions, which are changing rapidly at this time, to be in agreement at slightly different non-dimensional times.

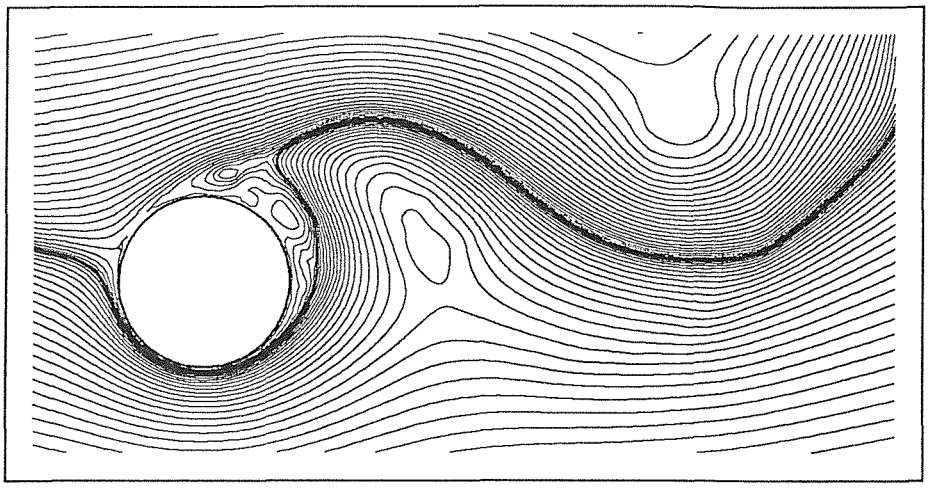


Figure 170 Vortex Code, τ=7.5

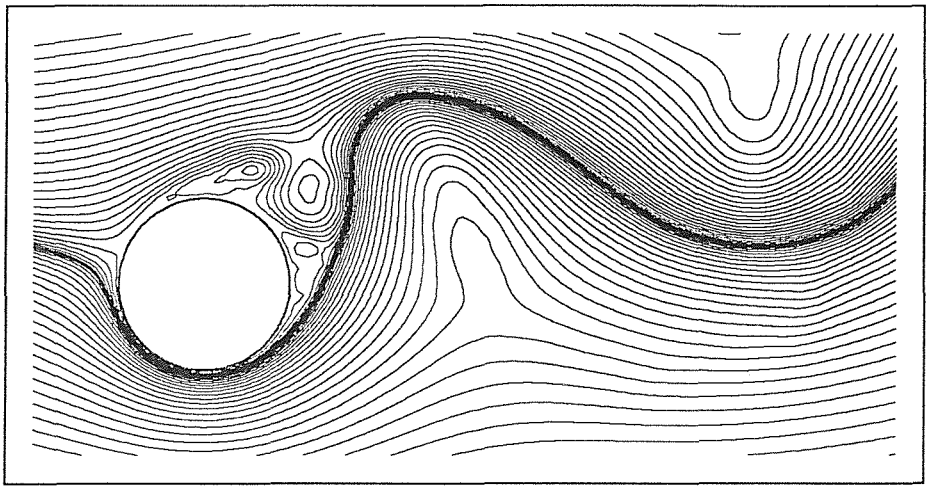


Figure 171 Vortex Code, τ=8.0

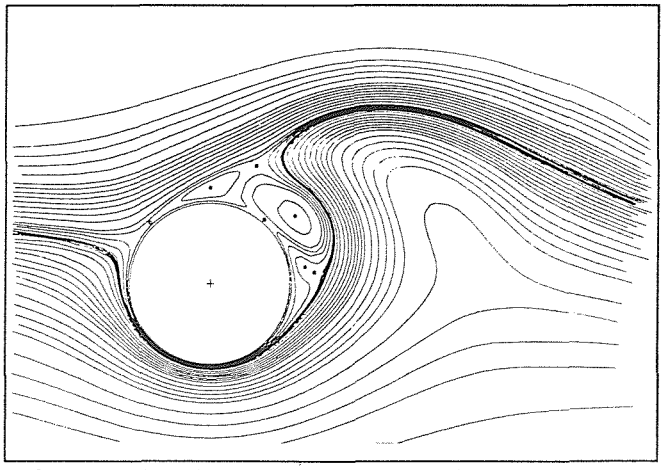


Figure 172 Finite Difference, τ=8.0

To further confirm the periodicity of the flow, we have continued the calculation and found that the same configuration of eddies occurs again at τ=12.0, see Figure 173.

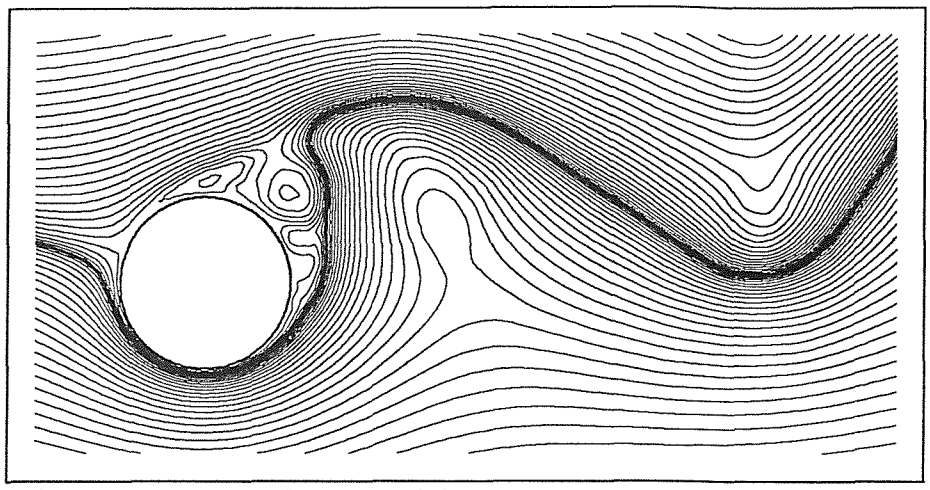


Figure 173 Vortex Code, τ=12.0

Figure 174 and Figure 175 show the force coefficients generated by the vortex code using the Eulerian and Lagrangian methods respectively (see p. 120). The two techniques are in broad agreement, with differences similar to those seen in the results for α=0.5. Again, the Eulerian graph has been smoothed to highlight the underlying trend. The two calculations appear to be in broad agreement, with the exception of the early time values which show a significant discrepancy. The drag graph has a dual periodic trend showing alternate vortex shedding from either side of the cylinder; the lift graph oscillates with period slightly greater than 4.0.

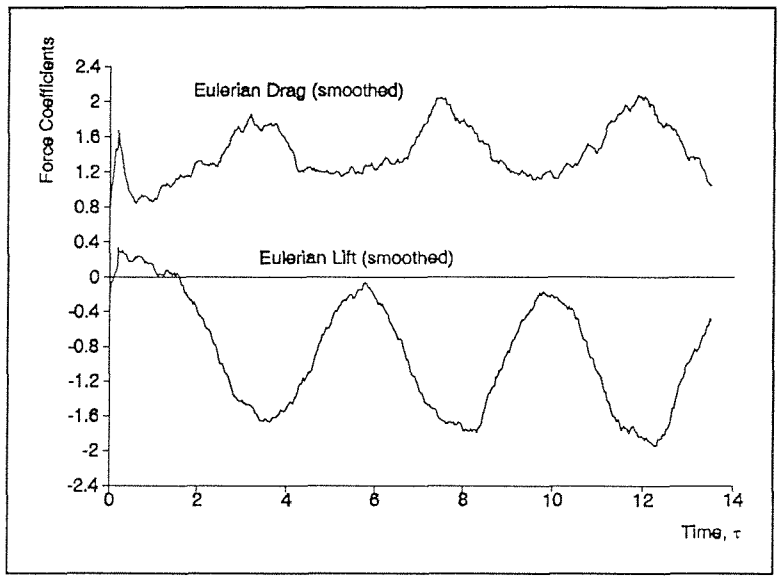


Figure 174 Eulerian force coefficients for **Re**=1000, α=1.0

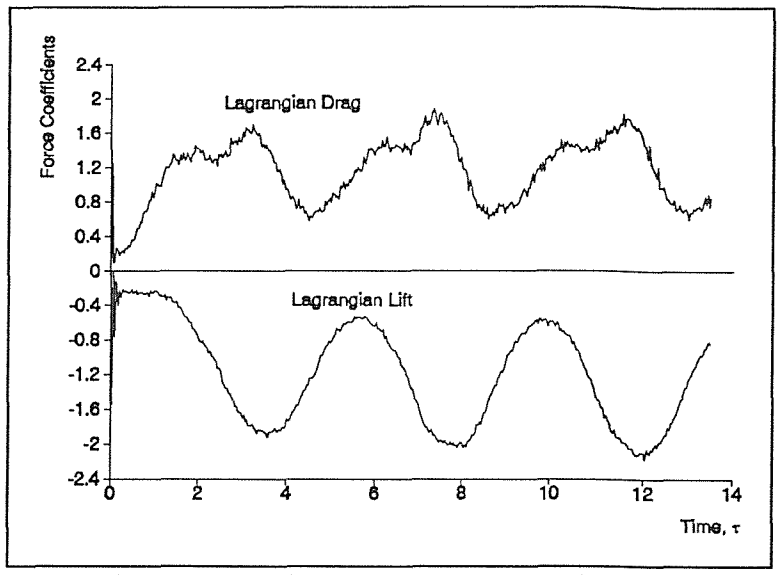


Figure 175 Lagrangian force coefficients for **Re**=1000, α=1.0

The values for force coefficients at α=1.0 given in Badr et al. show the same behaviour as those shown above, but with different values. The drag graph shows the same dual periodicity, with an almost flat plateau followed by a maximum of approximately 2.0; the lift graph shows a period of just over 4.0, oscillating between approximately 1.0 and 3.0. Notice that the lift graph as a whole is drifting towards more negative mean results, indicating that the state is not completely periodic.

5.5.4 Re=1000, α=2.0

Figure 176 - Figure 200 show the flow at ***Re***=1000 and α=2.0. A significantly different behaviour is exhibited at this rotational speed — the flow has no simple periodicity within the observed time interval. The discrete vortex solutions use 200 boundary points and a time step of 0.025. Figure 176 and Figure 177 show the streamlines for τ=0.5 as computed by the vortex code and the finite difference code respectively. Both show a thin wake developing behind the cylinder, predominantly attached to its upper surface.

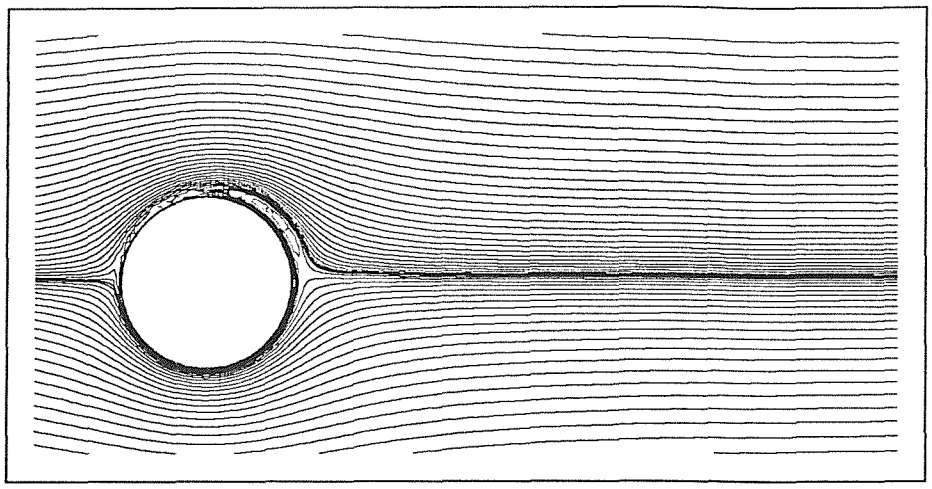


Figure 176 Vortex Code, τ=0.5

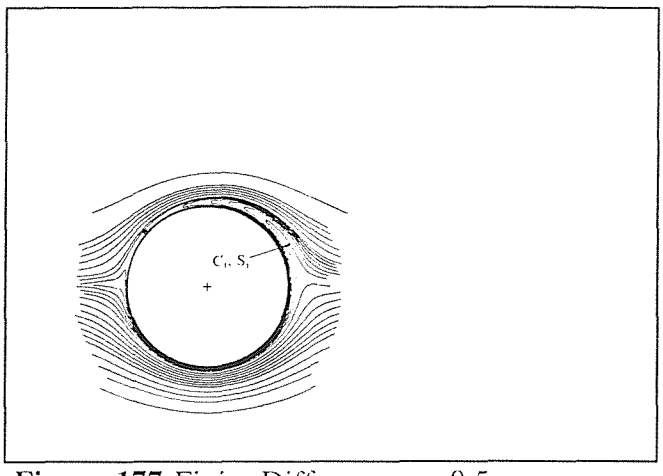


Figure 177 Finite Difference, τ=0.5

In Figure 178 and Figure 179, the two solutions have progressed to τ=1.0. The two solutions are in good agreement showing a single thin, but rotationally intense, eddy sitting on the upper surface of the cylinder. The solutions appear slightly different, but this is mainly due to a difference in streamline clustering. If the two are compared carefully, all the key features can be seen to be in similar positions.

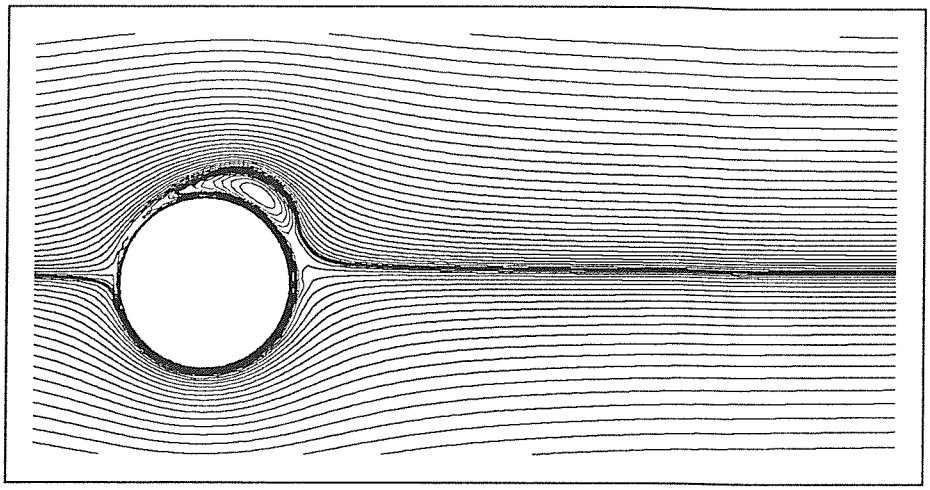


Figure 178 Vortex Code, τ=1.0

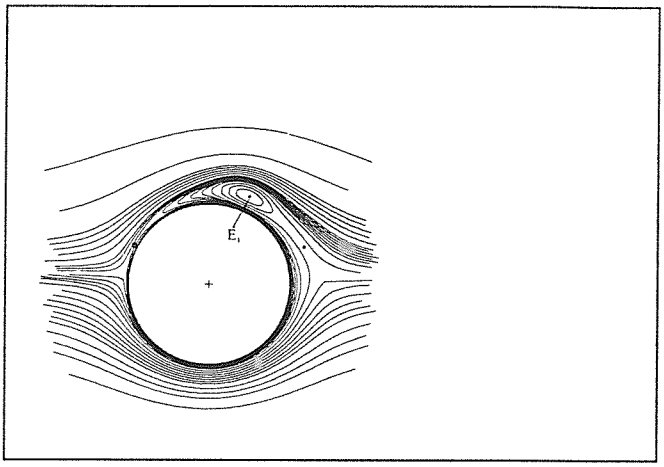


Figure 179 Finite Difference, τ=1.0

Figure 181 shows the first experimental photograph for α=2.0, taken from Badr et al. Alongside are presented the vortex code (Figure 180) and finite difference code (Figure 182) visualisations at the same non-dimensional time of 2.0. All three visualisations show the same situation, the single eddy from the previous time frame has intensified and has developed an enlarged central region. The eddy remains attached to the cylinder surface. Both numerical calculations predict the experimental solution accurately.

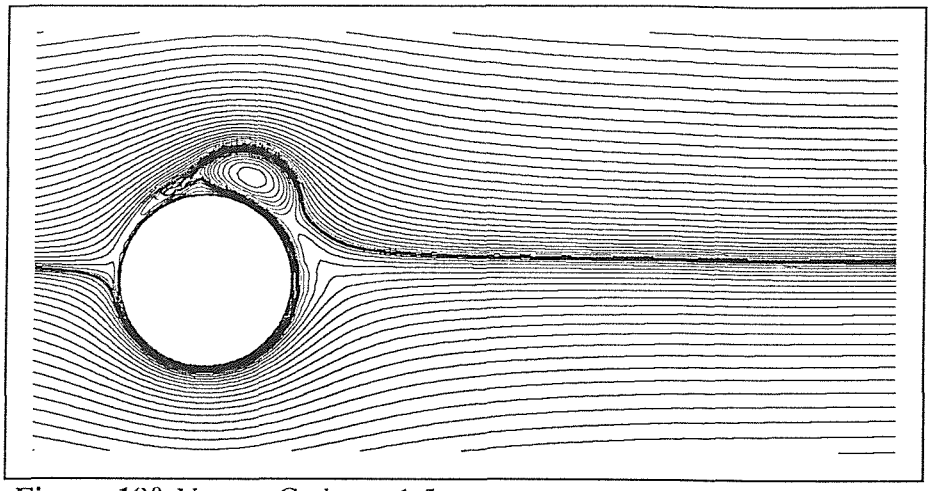


Figure 180 Vortex Code, τ=1.5

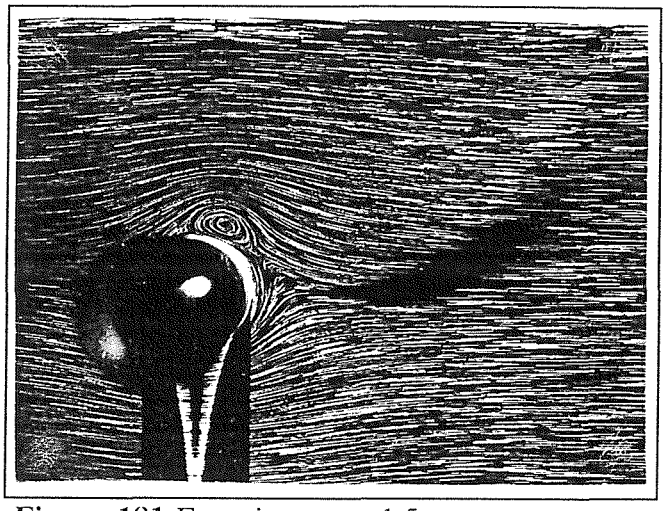


Figure 181 Experiment, τ=1.5

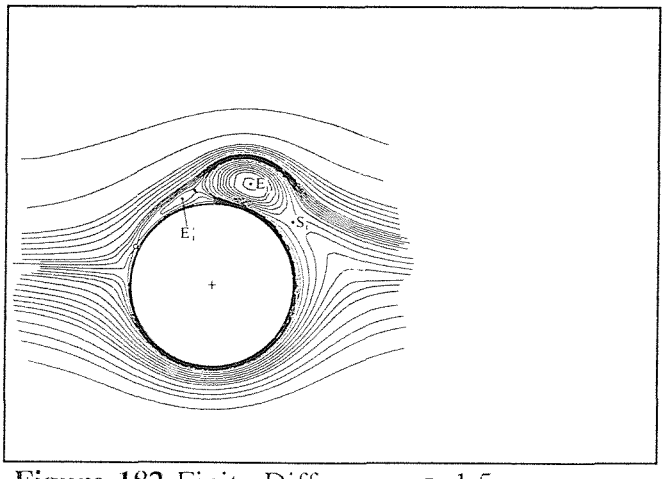


Figure 182 Finite Difference, τ=1.5

Figure 183 - Figure 185 show all three streamline diagrams at τ=2.5. The experiment shows the single eddy to have enlarged since the last time frame, it has also opened out to

partially join with the main flow at the rear of the cylinder. The joining with the outer flow signifies the early stages of shedding from the cylinder. Both computations predict the experimental situation accurately.

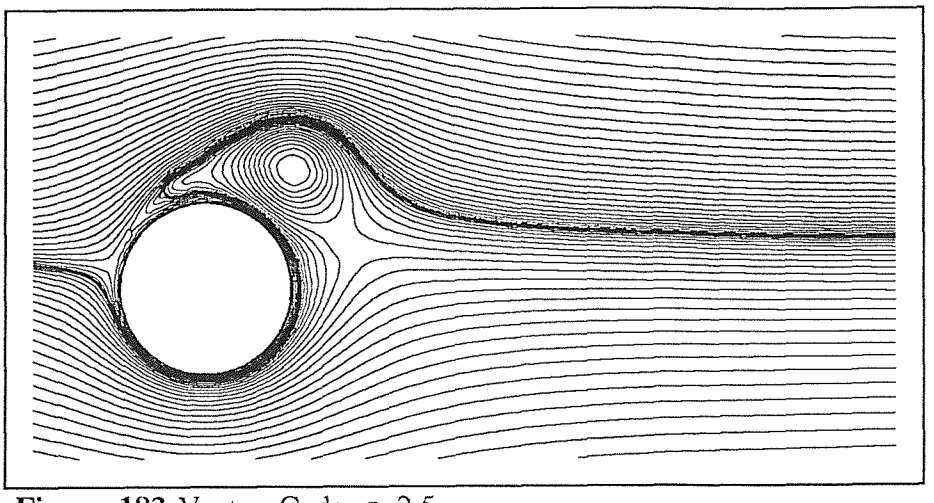


Figure 183 Vortex Code, τ=2.5

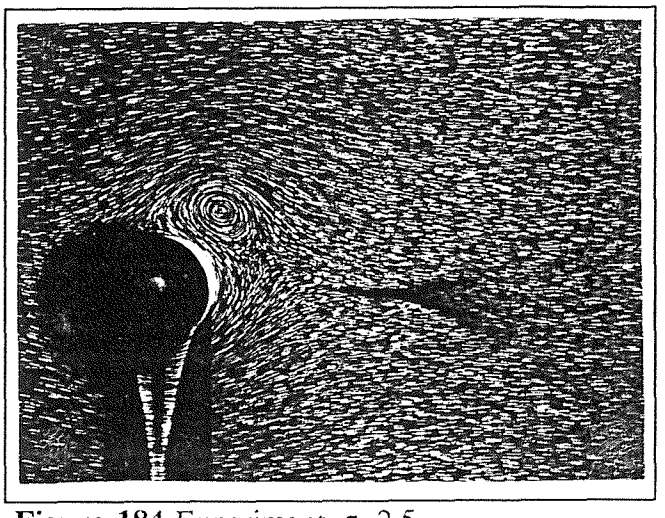


Figure 184 Experiment, τ=2.5

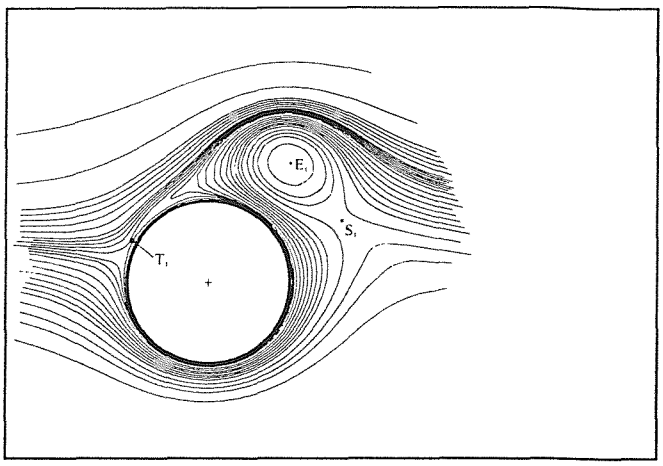


Figure 185 Finite Difference, τ=2.5

In Figure 186, the vortex code has moved on in time considerably, to τ=4.5. Figure 187 and Figure 188 show the equivalent experimental photograph and finite difference solution. At this time, the initial eddy has completely shed and is starting to travel in a downstream direction. Both calculations predict the experiment accurately. In the finite difference solution too few streamlines are used to show the centre of the shed eddy but the centre and stagnation point are marked as two dots and appear to be positioned approximately correctly. Some waves are apparent in the streamlines above the cylinder in the finite difference solution, these waves are not present in the experiment or in the discrete vortex solution.

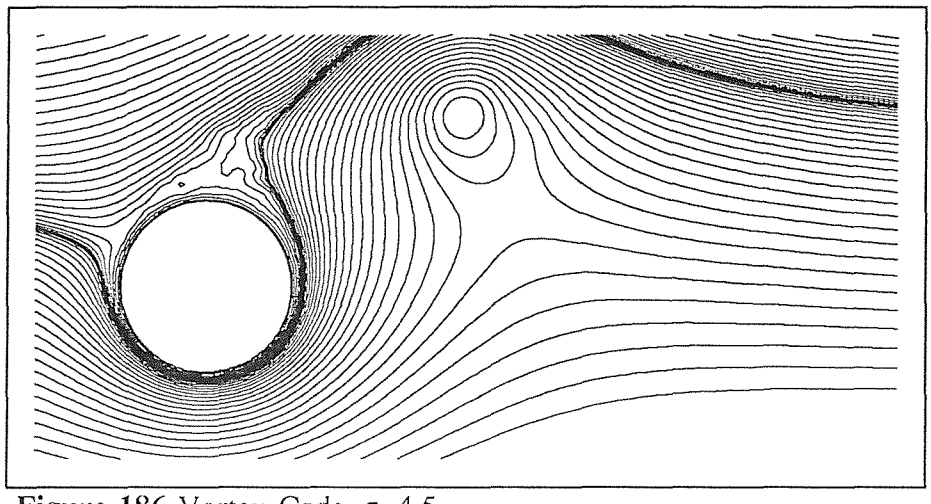


Figure 186 Vortex Code, τ=4.5

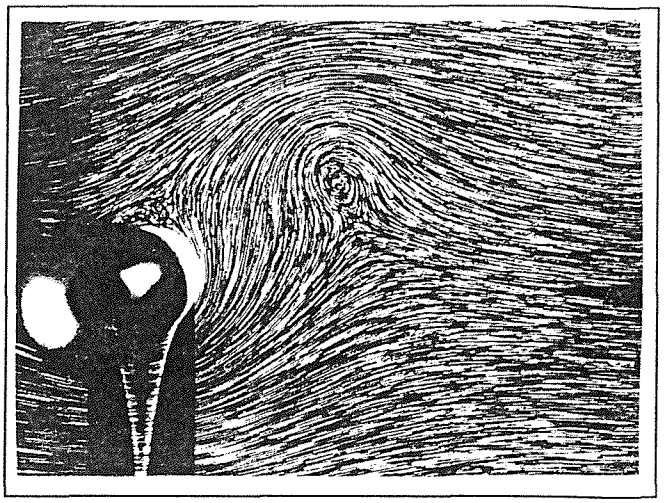


Figure 187 Experiment, τ=4.5

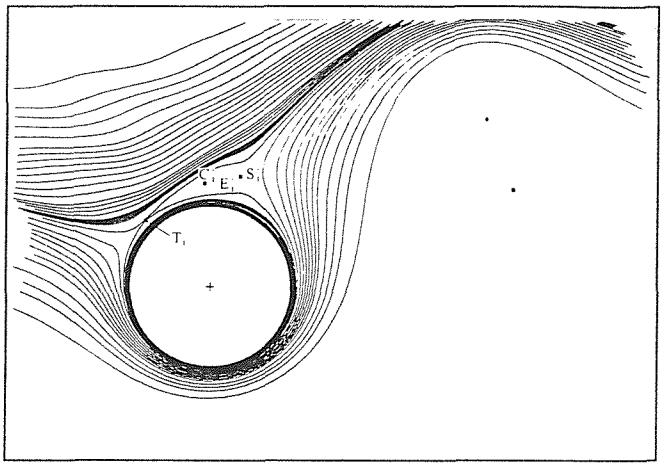


Figure 188 Finite Difference, τ=4.5

Figure 189 - Figure 191 show the same triplet of visualisations at τ=5.0. The shed vortex from the previous time frame has moved slightly further downstream. Two smaller eddies have formed on the upper surface of the cylinder, one of these has joined with the outer flow. Both calculated flows are in good agreement with the experimental photograph. The waves in the streamlines above the cylinder are still present in the finite difference calculation; neither the experiment nor the discrete vortex calculation show any signs of these waves.

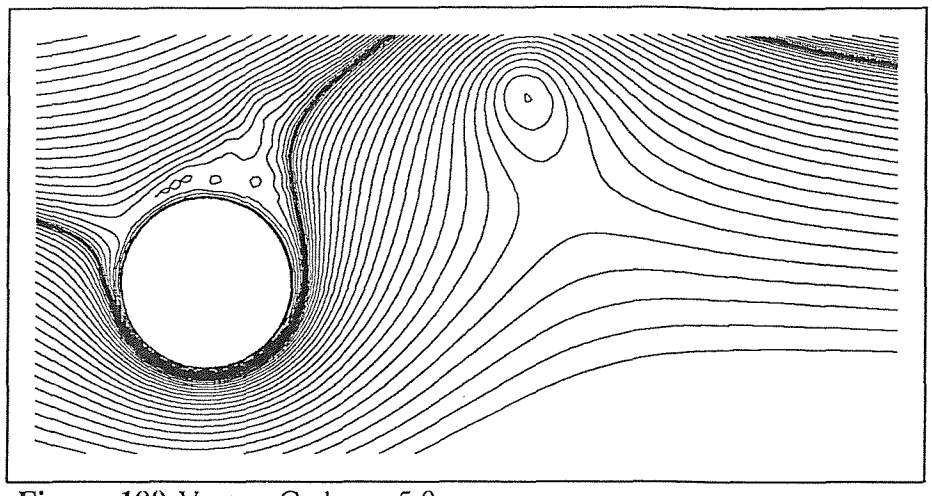


Figure 189 Vortex Code, τ=5.0

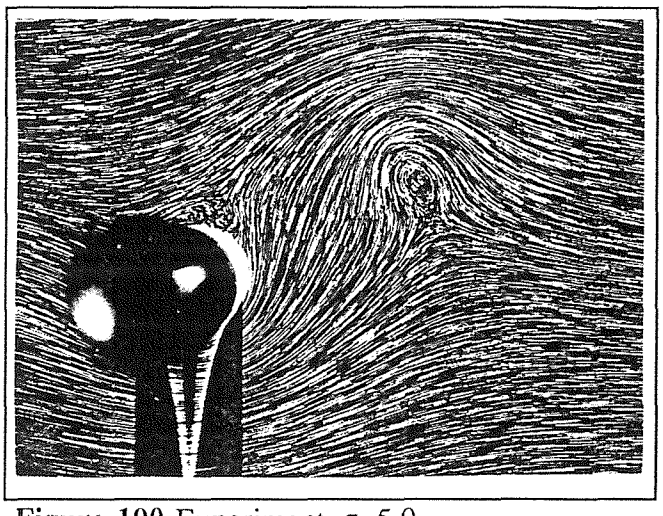


Figure 190 Experiment, τ=5.0

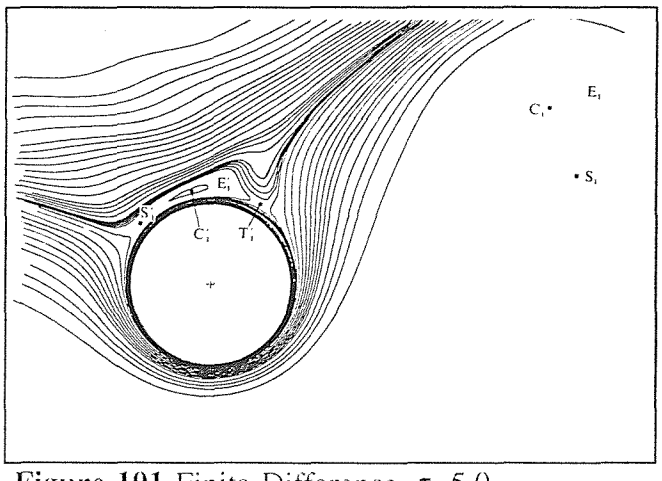


Figure 191 Finite Difference, τ=5.0

Figure 192 - Figure 194 show the three visualisations of the flow at τ=6.5. The shed eddy has travelled still further downstream. The smaller anticlockwise-rotating eddy has started to

shed from the cylinder. Both of the computations are in good agreement with the experiment. The waves in the streamlines above the cylinder are still present in the finite difference solution.

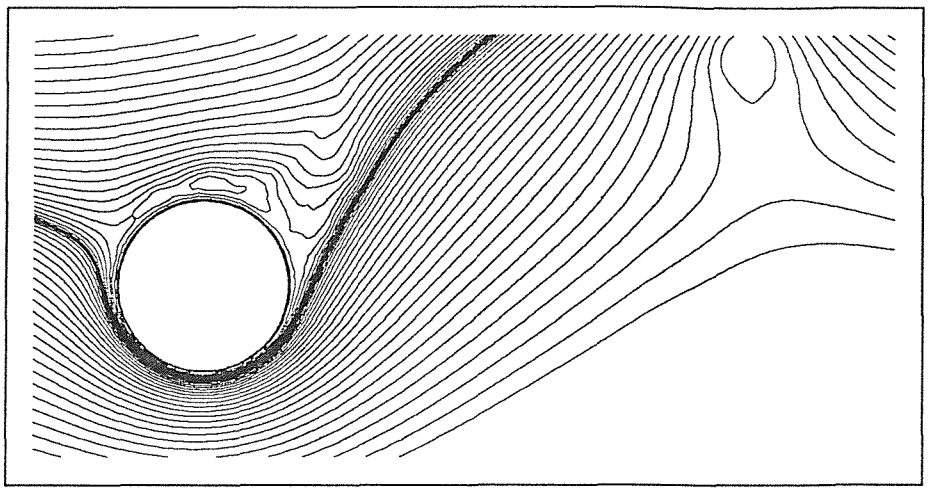


Figure 192 Vortex Code, τ=6.5

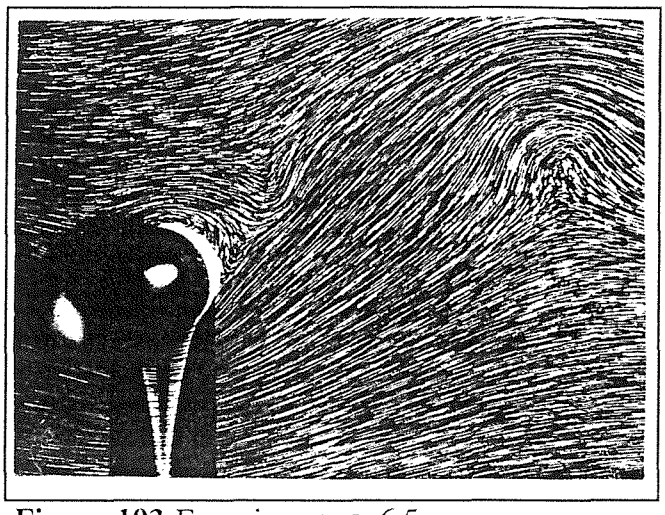


Figure 193 Experiment, τ=6.5

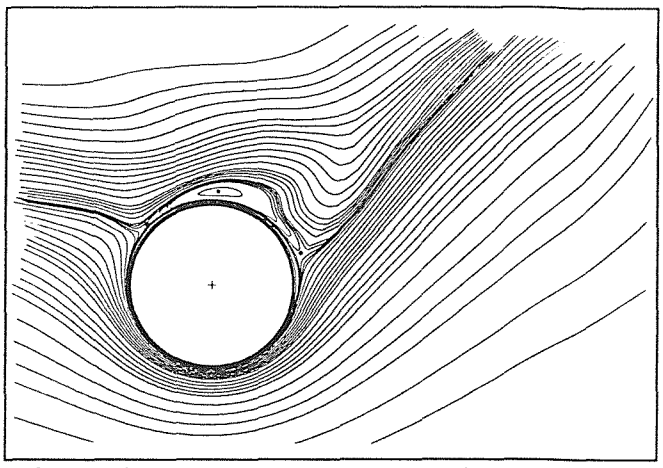


Figure 194 Finite Difference, τ=6.5

No further experimental photographs are given for α=2.0 in Badr et al., but we now present further comparisons between the two computations. Figure 195 and Figure 196 show the two computed solutions at τ=7.5. The two are still in broad agreement, with a small anticlockwise vortex continuing downstream and a second clockwise eddy starting to form on the upper surface of the cylinder.

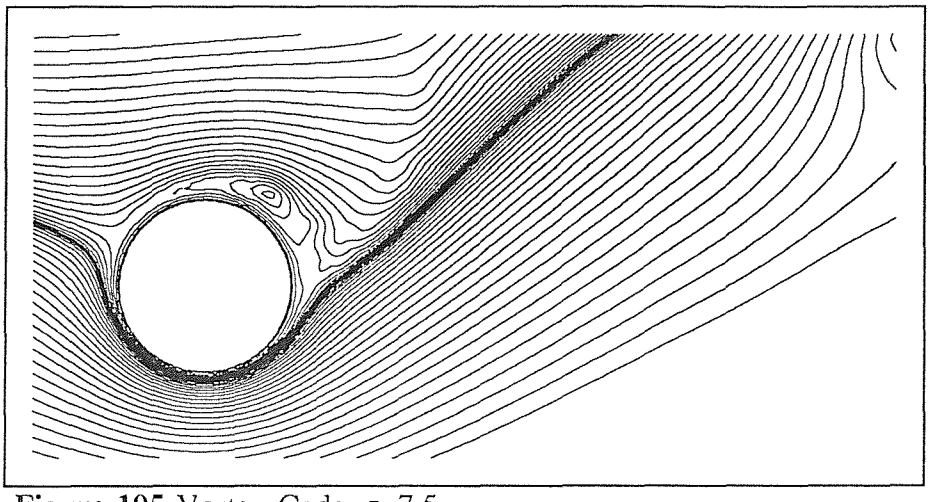


Figure 195 Vortex Code, τ=7.5

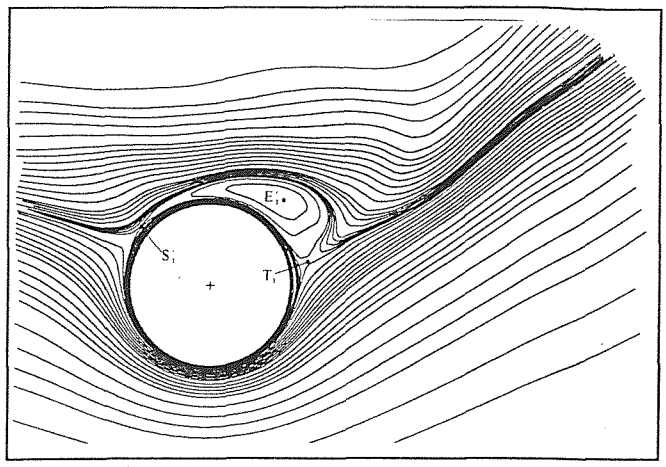


Figure 196 Finite Difference, τ=7.5

At τ=8.0, as shown in Figure 197 and Figure 198, the two calculated solutions start to exhibit different behaviour. In the finite difference solution, the eddy forming in the previous time frame has started to join with the outer flow, signifying the start of shedding. In the vortex code, the attached eddy is growing in size but remains attached to the cylinder surface.

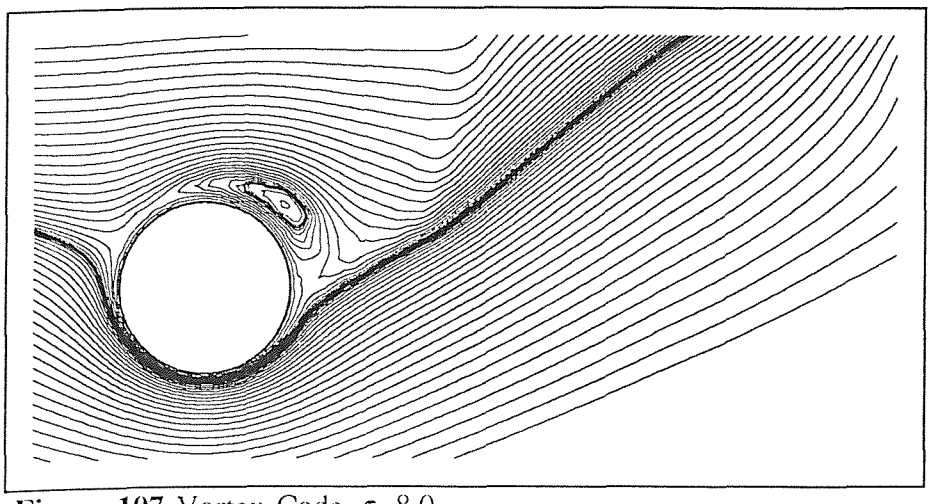


Figure 197 Vortex Code, τ=8.0

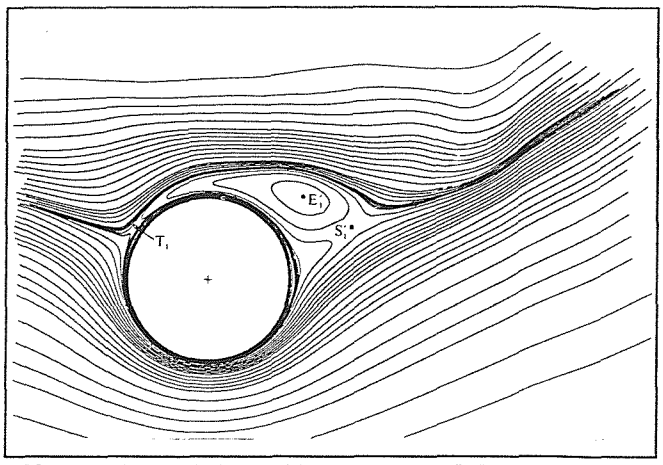


Figure 198 Finite Difference, τ=8.0

Figure 199 and Figure 200 show the two computed flows at τ=10.0, now the two are showing completely different situations. In the finite difference solution, the eddy which was partially attached in the previous time frame has almost completely shed; in the discrete vortex solution, the previously attached eddy remains attached and has grown to a substantial size. By continuing this calculation we have found that this eddy sheds over the next non-dimensional time unit.

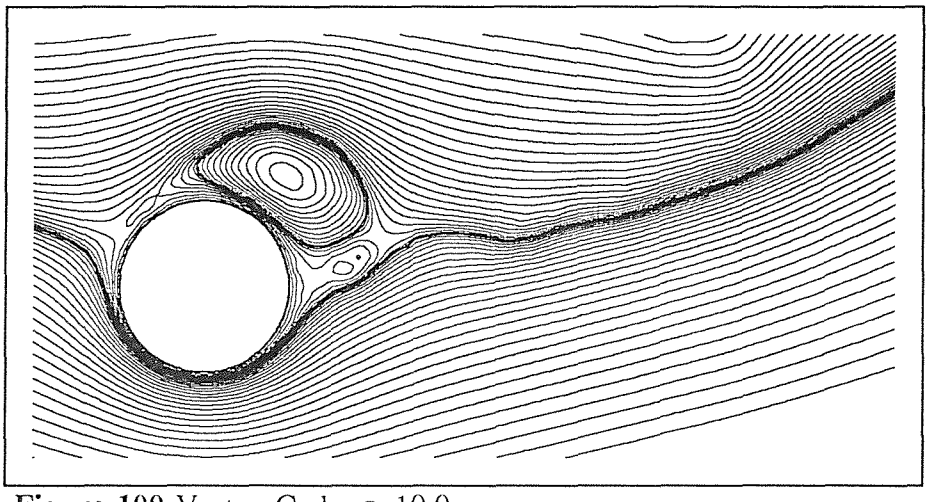


Figure 199 Vortex Code, τ=10.0

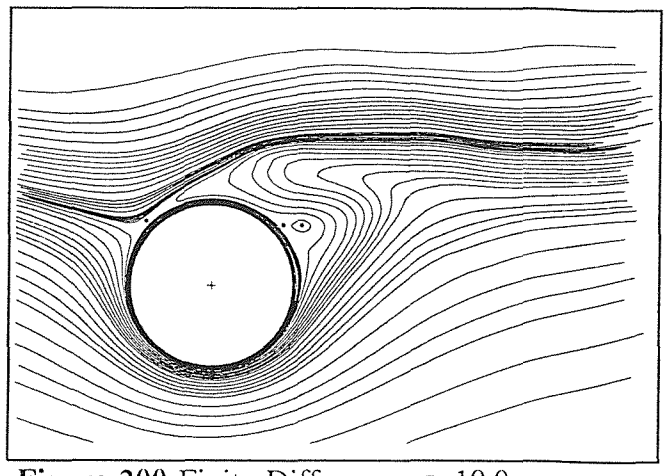


Figure 200 Finite Difference, τ=10.0

Figure 201 and Figure 202 show the force coefficients calculated by the vortex code using the Eulerian and Lagrangian techniques respectively. Comparing the two results, the familiar pattern emerges with the two calculations in broad agreement but differing in certain details. The initial values in the Eulerian graph are in error — in particular, we expect the lift to be negative. The other main difference is that the drag returns to zero near τ=7 in the Lagrangian graph but has a non-zero minimum there in the Eulerian case. The force graphs confirm that the flow has become non-periodic or at least periodic on a larger time scale. No force values are given at α=2.0 in Badr et al.

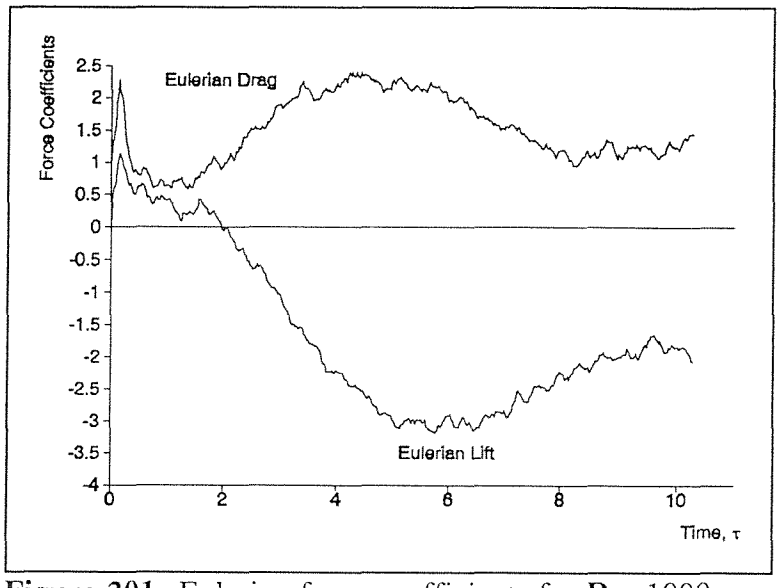


Figure 201 Eulerian force coefficients for **Re**=1000, α=2.0

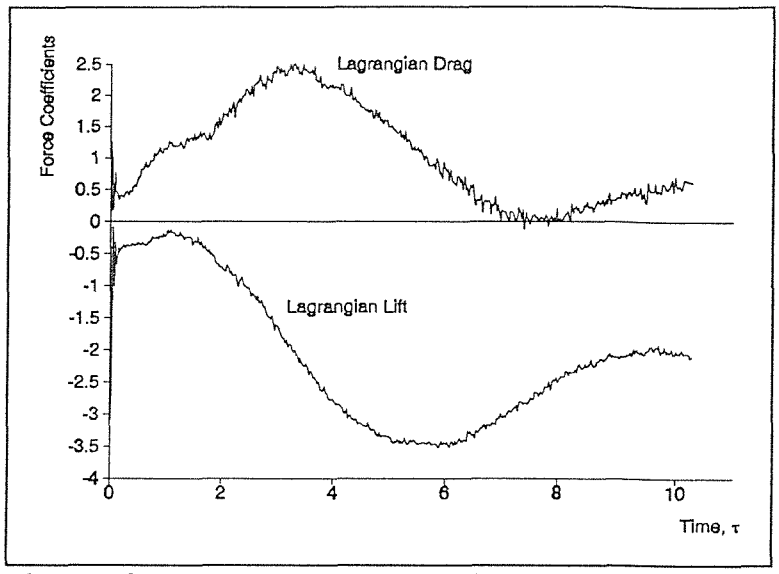


Figure 202 Lagrangian force coefficients for **Re**=1000, α=2.0

5.5.5 Re=1000, α=3.0

Figure 203 - Figure 220 show the evolution of the flow at ***Re***=1000 and α=3.0. The discrete vortex code was run using 200 boundary points and a time step of 0.016. Comparisons similar to those given before are presented for each of the non-dimensional time frames chosen by Badr et al. The behaviour at this rotational speed continues the trend towards less periodic flow (or possibly towards longer periods of repetition). The rotational speed is now sufficient to eventually cause a layer of fluid to rotate with the cylinder and to cause an eddy to travel around to the front of the cylinder.

The first time frame is at τ=0.5, with the discrete vortex solution shown in Figure 203 and the finite difference solution shown in Figure 204. The flow has developed into a similar configuration to the lower rotational rates at this time, with a thin wake wrapped around the cylinder in the direction of rotation. The wake is wrapped further around the cylinder as would be expected with the higher rotational speed. The vortex code and finite difference code visualisations are in agreement, the wake appears to be slightly wider in the finite difference solution.

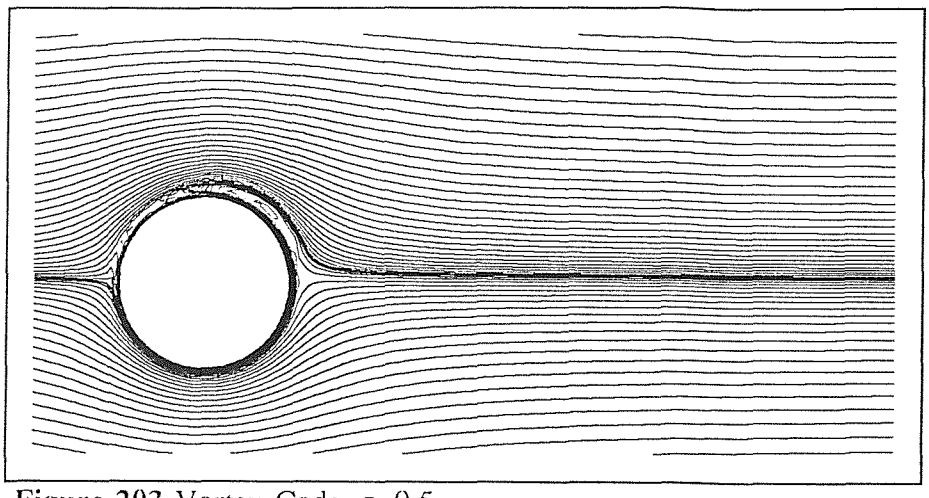


Figure 203 Vortex Code, τ=0.5

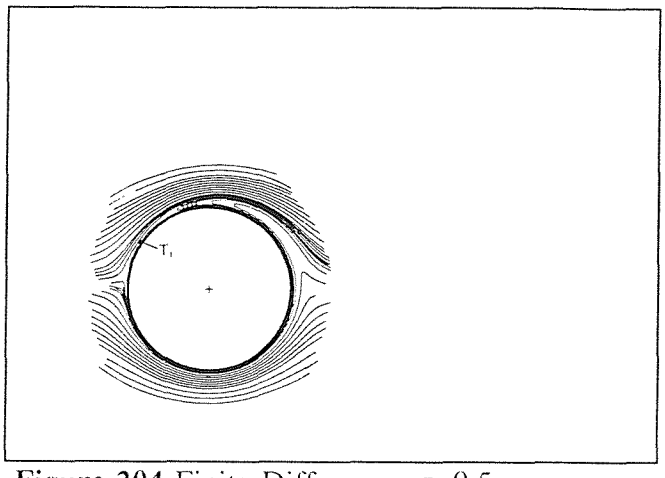


Figure 204 Finite Difference, τ=0.5

In Figure 205 and Figure 206 the computed solutions have progressed to non-dimensional time 1.0. Both plots show a rotationally intense eddy, with an elongated central region, forming on the upper surface of the cylinder. A careful study shows the solutions to be in closer agreement then they at first appear. The illusion of discrepancy is created a different clustering of streamlines in the two calculations. The main *actual* difference is that, in the finite difference solution, the eddy is slightly further around the cylinder in the direction of rotation.

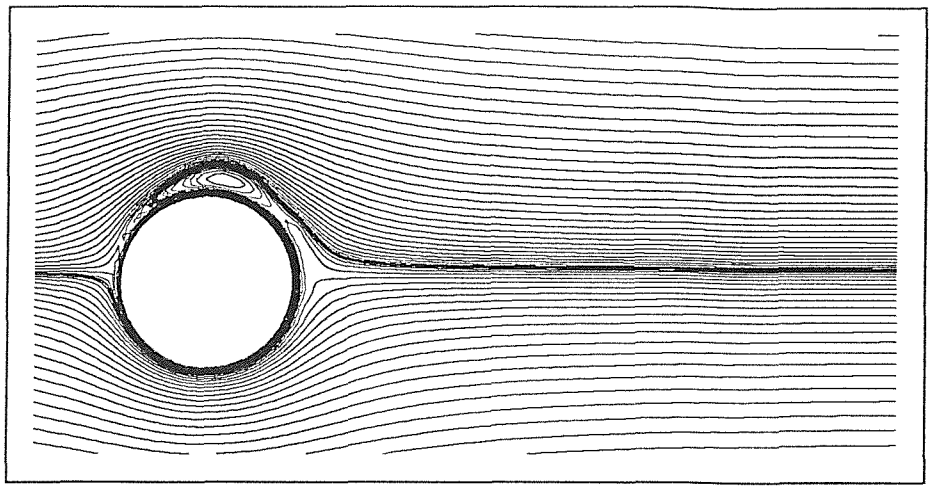


Figure 205 Vortex Code, τ=1.0

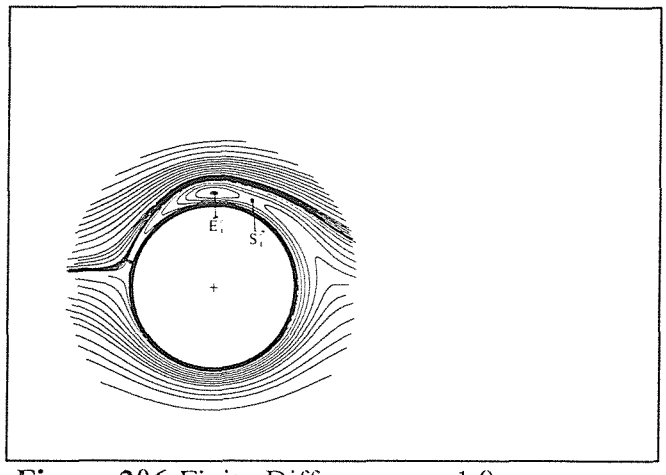


Figure 206 Finite Difference, τ=1.0

At τ=1.5, the first experimental photograph is available (Figure 208). The discrete vortex and finite difference predictions of the same non-dimensional time are shown in Figure 207 and Figure 209 respectively. By now, the vortex at the rear of the cylinder, which was just starting to form in the previous time frame, has grown into a second eddy. Both numerical schemes are in agreement with the experiment. The vortex code has the upper eddy positioned slightly further in the clockwise direction around the cylinder surface. In the discrete vortex solution, some streamlines completely encircle the cylinder, indicating that a layer of fluid has started to rotate with the cylinder.

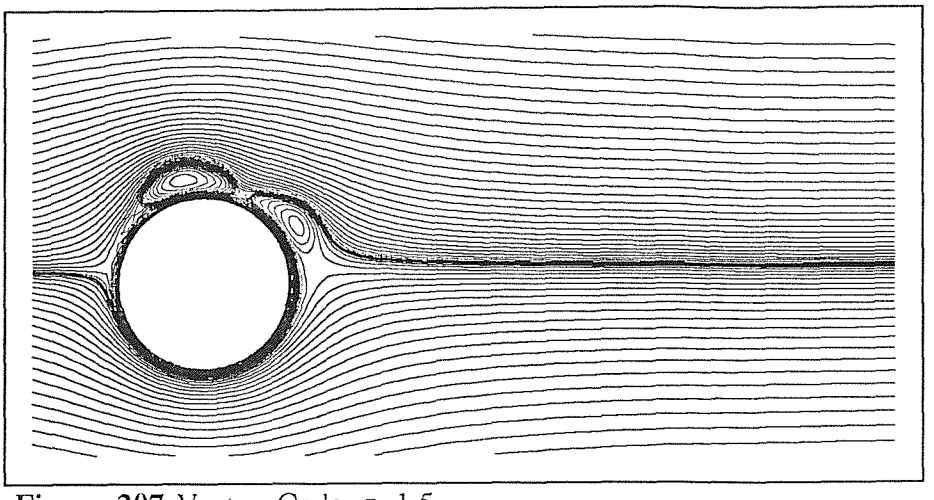


Figure 207 Vortex Code, τ=1.5

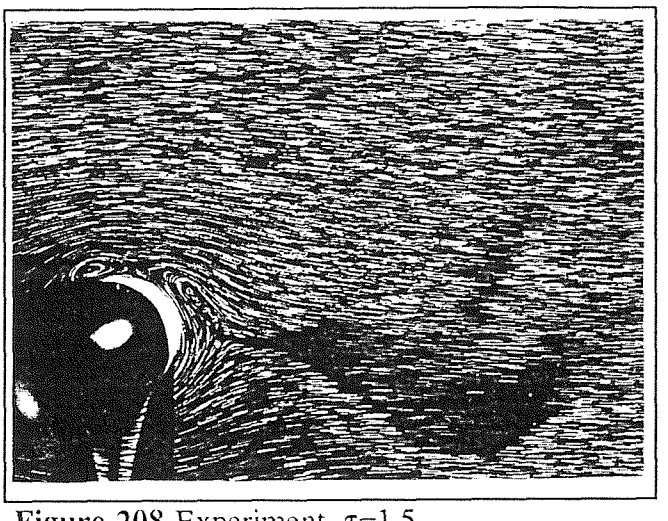


Figure 208 Experiment, τ=1.5

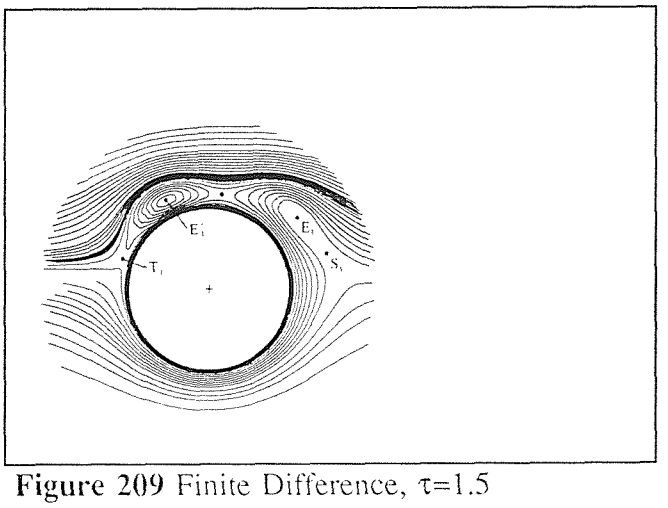


Figure 209 Finite Difference, τ=1.5

In Figure 210 - Figure 212, the three visualisations at τ=2.0 are shown. The eddy configuration is similar to the previous time frame, with two distinct eddies attached to the

upper cylinder surface. The eddy which is further clockwise has grown and is showing characteristic signs of starting to shed. The other eddy has moved further around the cylinder in the direction of rotation but has not changed significantly in size. At this time frame, the vortex code solution predicts the experiment more closely than the finite difference solution. The location of the more anticlockwise of the two eddies is near the uppermost point of the cylinder in the vortex code; this region is difficult to read in the experiment but there is evidence of an eddy in a similar position. The finite difference code predicts this eddy to have travelled round to the front of the cylinder.

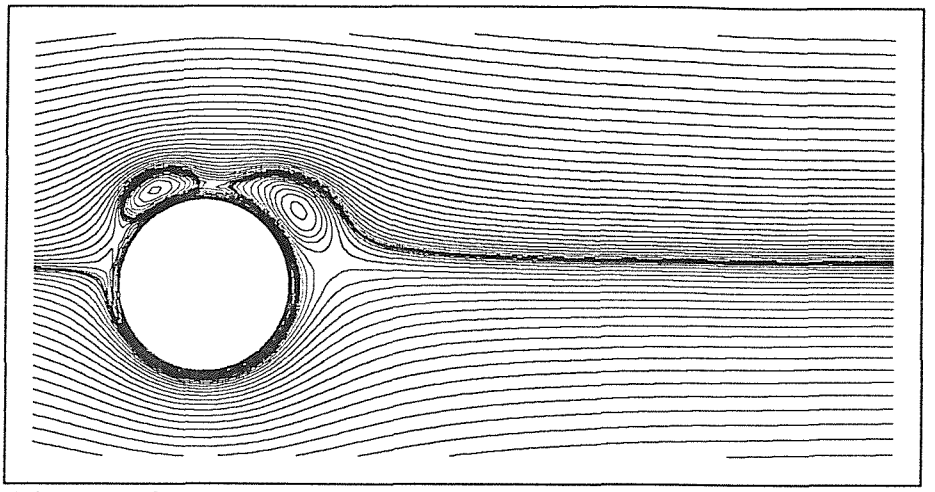


Figure 210 Vortex Code, τ=2.0

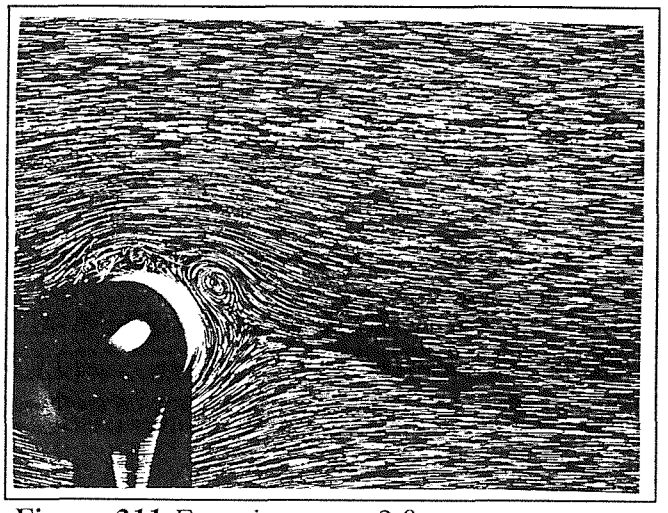


Figure 211 Experiment, τ=2.0

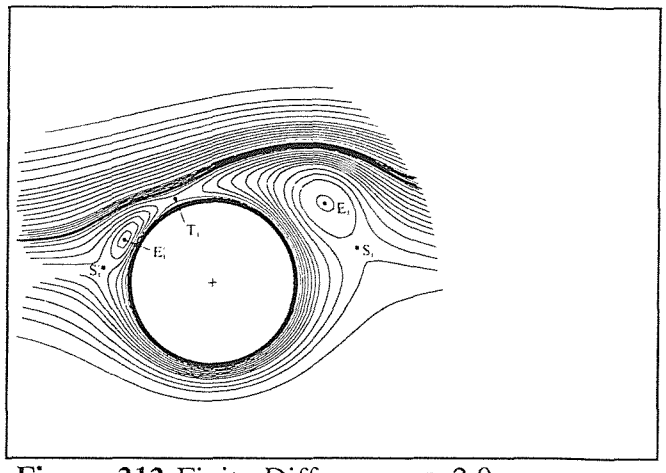


Figure 212 Finite Difference, τ=2.0

In subsequent time steps, the forward eddy on the front continues to travel around the cylinder. As it moves it becomes stretched flatter to the cylinder surface until it is eventually absorbed into the intense velocity gradient of the rotating fluid layer. One consequence of this behaviour is that much of the vorticity from this eddy will eventually find its way into the more clockwise eddy.

Figure 213 shows the discrete vortex solution at τ=3.0. Figure 214 shows the experimental photograph and Figure 215 shows the finite difference solution at the same non-dimensional time. All three visualisations are in broad agreement, with a large eddy starting to shed from the cylinder and vorticity of the same sign as this eddy entrained to the front of the cylinder. In the finite difference solution, the eddy appears to have moved further away from the cylinder than in the discrete vortex solution or the experiment.

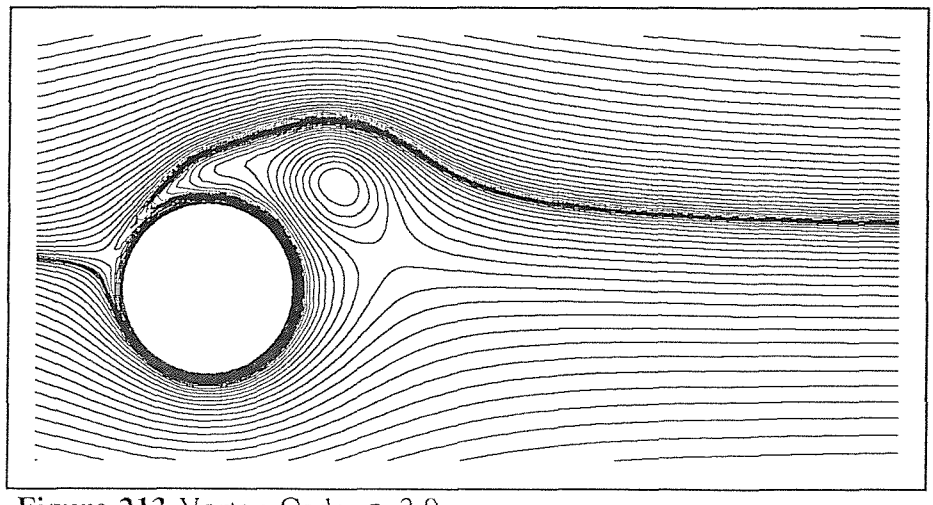


Figure 213 Vortex Code, τ=3.0

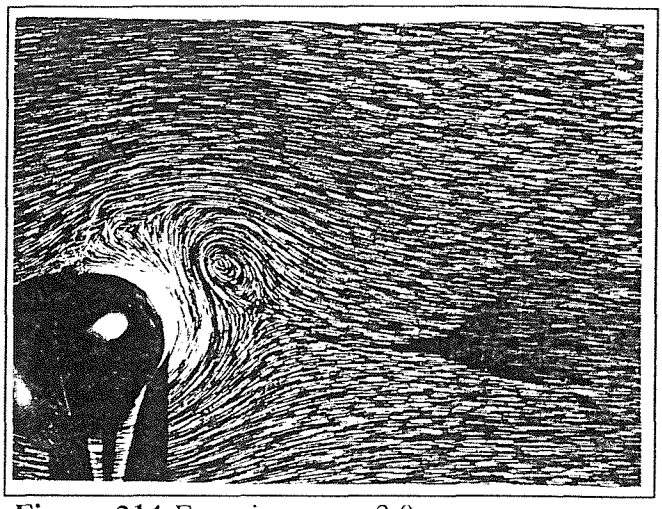


Figure 214 Experiment, τ=3.0

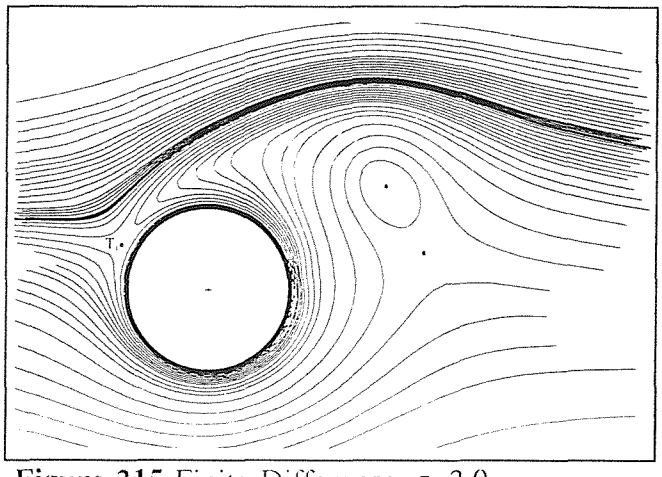


Figure 215 Finite Difference, τ=3.0

Figure 216 - Figure 218 show the three flows at τ=5.0. The shed eddy/vortex continues to move downstream from the cylinder. The finite difference solution shows the upstream end of the eddy smoothly stretched around the cylinder surface. The discrete vortex solution shows a separate eddy forming in this region. The experiment is difficult to read in this region, showing evidence of three-dimensional instability, but there is some evidence of the presence of a separate eddy and there is generally better agreement with the discrete vortex solution.

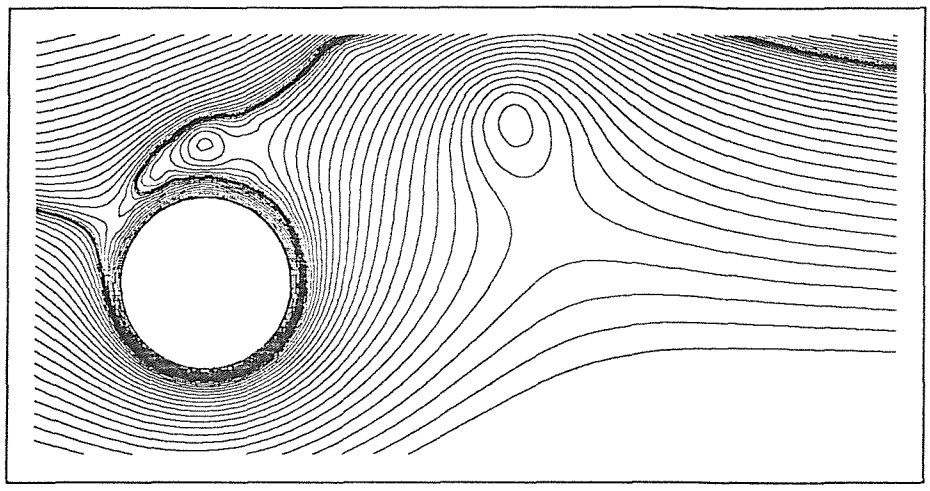


Figure 216 Vortex Code, τ=5.0

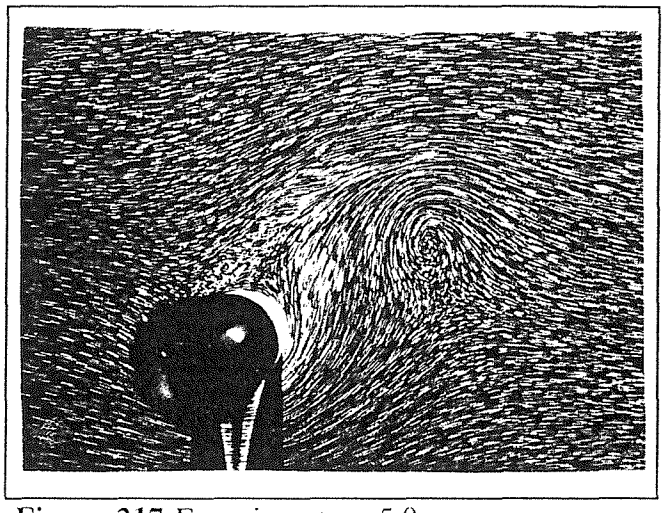


Figure 217 Experiment, τ=5.0

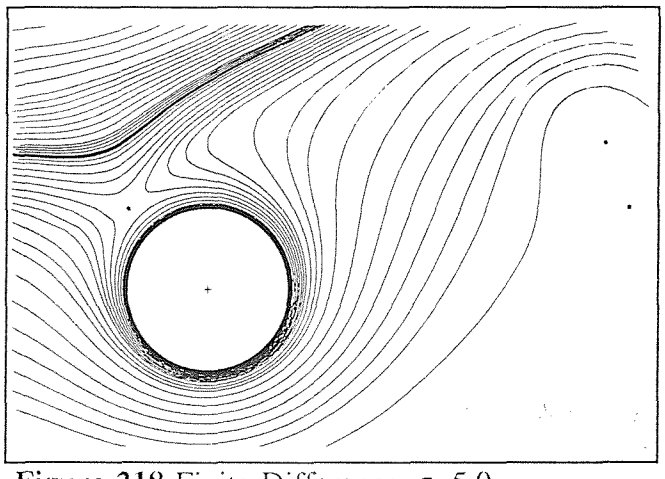


Figure 218 Finite Difference, τ=5.0

Figure 219 and Figure 220 show the two numerical solutions at non-dimensional time 10.0. The two are in agreement, showing a wider layer of fluid now rotating with the cylinder

and much of the local fluid passing around the cylinder in a similar fashion to the inviscid solution. Badr et al. claim that the finite difference code has reached a steady state, although their calculation still shows a significantly increasing lift at this time (see below).

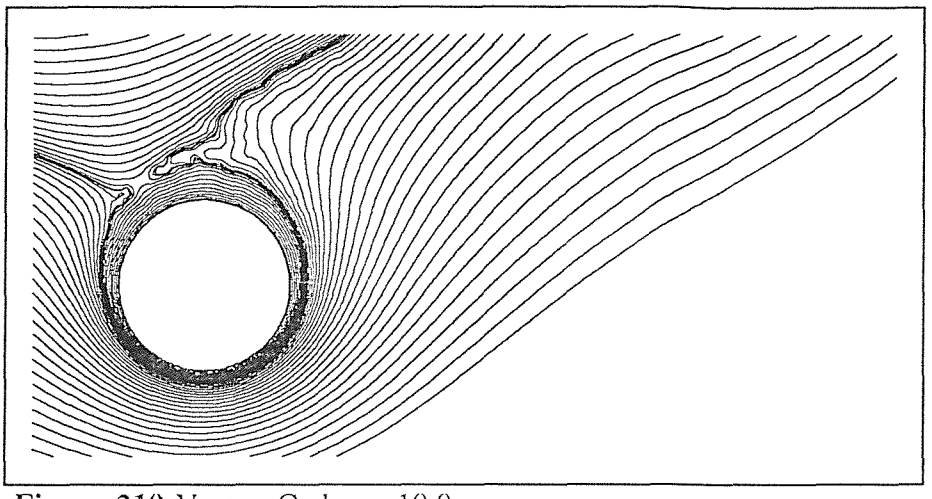


Figure 219 Vortex Code, τ=10.0

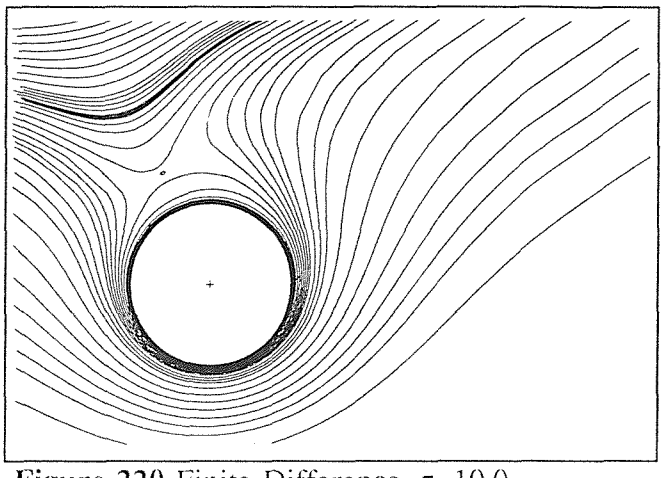


Figure 220 Finite Difference, τ=10.0

Figure 221 and Figure 222 show the Eulerian and Lagrangian calculations for the force coefficients. The two methods are in agreement on the value of the lift coefficient apart from at early times, when the Eulerian scheme shows it to be positive. The two drag graphs show the usual discrepancy at early times (the Eulerian graph does not start near zero and gradually build up) but also the Lagrangian graph shows the drag dropping towards the end of the calculation; the Eulerian calculation shows the drag approaching a constant value after long times.

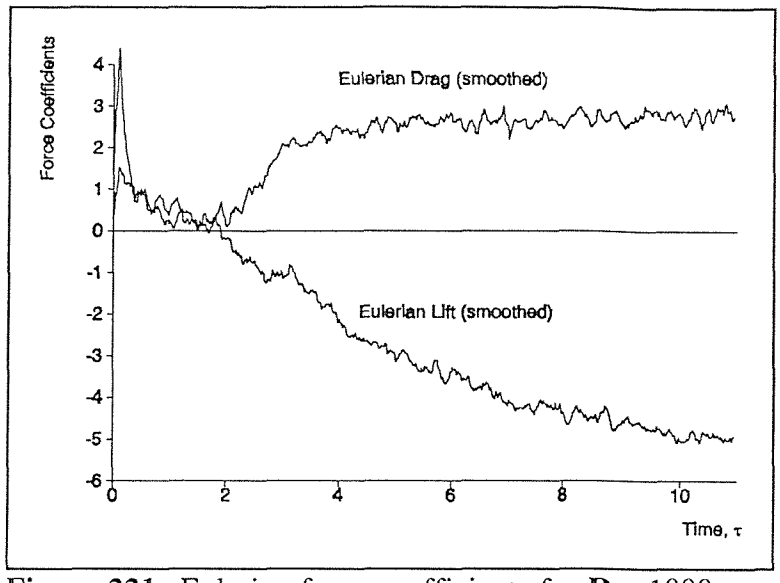


Figure 221 Eulerian force coefficients for **Re**=1000, α=3.0

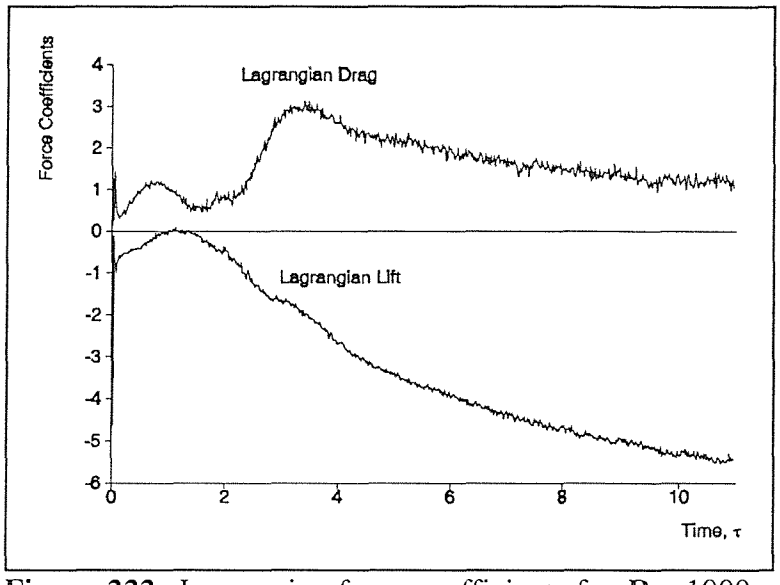


Figure 222 Lagrangian force coefficients for **Re**=1000, α=3.0

In the calculations of Badr et al., the qualitative behaviour of the lift is in reasonable agreement with our calculations. They obtain larger values for lift, reaching -8.0 at τ=10.0, but the overall behaviour is very similar to that given by the vortex code. The drag values presented in the paper show the early behaviour to be in agreement with our Lagrangian graph and the later behaviour similar to our Eulerian graph. The asymptotic value of the long time drag is higher than ours, approximately 5.0.

We have performed one long run, up to τ=65, for a rotating cylinder with α=3.0 and ***Re***=1000. This was performed with a time step of 0.05 and 100 boundary points. We do not

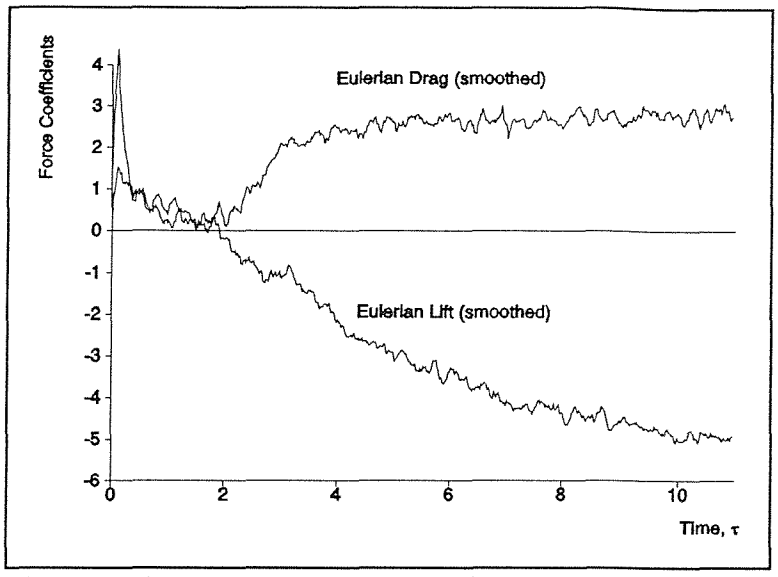


Figure 221 Eulerian force coefficients for **Re**=1000, α=3.0

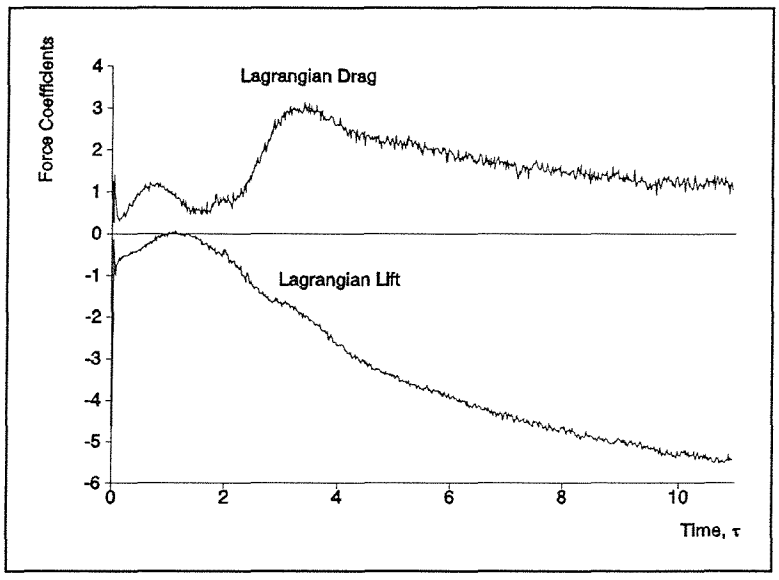


Figure 222 Lagrangian force coefficients for **Re**=1000, α=3.0

In the calculations of Badr et al., the qualitative behaviour of the lift is in reasonable agreement with our calculations. They obtain larger values for lift, reaching -8.0 at τ=10.0, but the overall behaviour is very similar to that given by the vortex code. The drag values presented in the paper show the early behaviour to be in agreement with our Lagrangian graph and the later behaviour similar to our Eulerian graph. The asymptotic value of the long time drag is higher than ours, approximately 5.0.

We have performed one long run, up to τ=65, for a rotating cylinder with α=3.0 and Re=1000. This was performed with a time step of 0.05 and 100 boundary points. We do not

expect the short time results from this run to be as accurate as the other results presented in this section, but the (Eulerian) forces were consistent with those shown here up to τ=10. However, after τ=30, the flow moved away from the pseudo-steady state and a small amplitude shedding mode was initiated with a corresponding fluctuation in the forces.

5.5.6 Summary

The discrete vortex scheme developed here has been shown to give solutions for flow past a rotating circular cylinder which are comparable to, and in some cases better than, the finite difference/spectral calculations presented in Badr et al.

The discrete vortex method gives solutions which are in better agreement with experiments for this set of problems than for the non-rotating cylinder. We have also found no evidence in these problems of the non-determinism found in the translating cylinder problems.

The force coefficients have been shown to display the same qualitative trends as those given in Badr et al. The Lagrangian method gives smoother, more accurate figures; the Eulerian calculation gives noisier values which are consistent, apart from at very early times. The absolute values of the forces differ in some cases from the calculations of Badr et al. We have noticed that none of the calculations ever come close to the inviscid value for lift, which is $-2\pi\alpha$. The discrete vortex long term force values are consistently of smaller magnitude than the finite difference calculations for these problems, this discrepancy increases for higher rotational rates.

5.6 Flow Past NACA Aerofoils

5.6.1 Setting up the Problems

Having thoroughly tested the code against a range of standard circular cylinder problems, we move on to examine results for aerodynamics problems. The NACA four-digit series of aerofoil profiles were used, as they are the standard set of wing sections for computational fluid dynamics tests.

The Reynolds number and non-dimensional time are now based on the chord length,

$$R = \frac{U_e C}{\nu}, \qquad \tau = \frac{t U_e}{C} \quad . \tag{70}$$

Taking air at one atmosphere pressure and 20° C as the fluid medium gives $\nu=1.5\times10^{-5}$ m^2s^{-1}, from Batchelor (1967). From the cylinder problems we expect that meaningful solutions could be obtained up to Reynolds numbers of at least 35,000. Thus, we expect to be able to solve for aerofoil problems in air such that $U_eC<0.5$ m^2s^{-1}. Unfortunately, this is not high enough even for the wing of a large scale model, eg. $U_e=30$ ms^{-1} and $C=0.5$ m, giving $U_eC=15$ m^2s^{-1}. However, the code will allow modelling of the low-speed stall process, and we will present detailed visualisations of this using NACA wing sections.

An accompanying code was written to generate curve files (see section 4.11.3) for the aerofoils. This code generates a set of discrete points lying on the aerofoil surface and a set of normalised tangents to the curve at these points. The code also allows any angle of attack for the aerofoil to be specified.

As mentioned in section 5.1, the panel method we have used can become ill-conditioned for non-smooth curve shapes, with an increasing number of panels. To avoid this problem, we have slightly rounded the trailing edge of the aerofoils. This was achieved by using a fitted hyperbola for the thickness distribution at the rear of the aerofoil, as shown in Figure 223. The hyperbola is fitted to be tangential to the usual thickness distribution at the points A and B in the figure, and is perpendicular to the chord line at the point C, the trailing edge. This creates an aerofoil with continuous first derivative around its entire perimeter, which is close in shape to the usual NACA four-digit section.

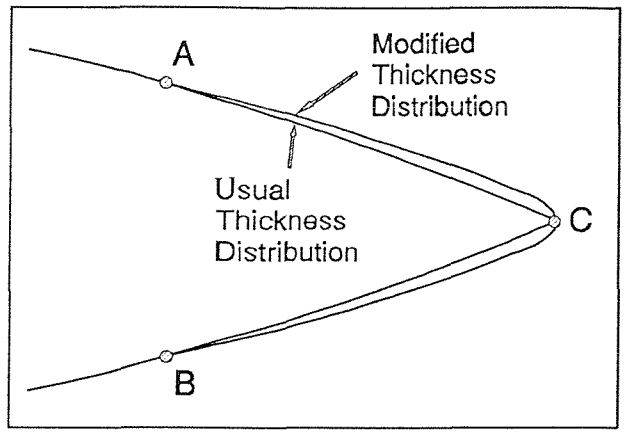


Figure 223 Illustration of the rounding used at the trailing edge

The equations for the unmodified NACA sections were taken from Abbot and Von Doenhoff (1949), and are,

$$y_c = \begin{cases} \dfrac{m}{p^2}(2px - x^2) & ; x \le m \\ \dfrac{m}{1-p^2}(1 - 2p + 2px - x^2) & ; x > m \end{cases}$$

$$y_t = 5t\,(a_1\sqrt{x} + a_2 x + a_3 x^2 + a_4 x^3 + a_5 x^4)$$

$$\theta = \tan^{-1}\left[\frac{2m}{p^2}(p - x)\right] \qquad (71)$$

$$x_u = x - y_t \sin(\theta)$$
$$y_u = y_c + y_t \cos(\theta)$$

$$x_l = x + y_t \sin(\theta)$$
$$y_l = y_c - y_t \cos(\theta)$$

Where the parameters are,

m = maximum y-coordinate of mean line expressed as a fraction of chord length
p = fraction along chord of maximum y-coordinate
t = maximum thickness as fraction of chord length
x = curve traversing parameter (corresponds to x-coordinate of point on mean line).

The output quantities (x_l, y_l) and (x_u, y_u) are the positions of the points on the lower and upper surfaces of the aerofoil, respectively.

The quantities a_1-a_5 are predefined constants,

a_1 = 0.2969
a_2 = -0.1260
a_3 = -0.3516
a_4 = 0.2843
a_5 = -0.1015.

The intermediate quantities are,

y_c = y-coordinate of point on mean line
y_t = thickness distribution
θ = angle between tangent to the mean line at (x, y_c) and the chord line.

We round the trailing edge by using the same formulation with modified thickness distribution,

$$Y_t(x) = \begin{cases} y_t(x) & ; \; x<m_1 \\ H_a\sqrt{1-\dfrac{(x-H_c)^2}{H_b^2}} & ; \; x\geq m_1 \; . \end{cases} \tag{72}$$

We then apply the following matching conditions,

$$\begin{aligned} Y_t(m_1) &= y_t(m_1) \\ Y_t'(m_1) &= y_t'(m_1) \\ Y_t(1) &= 0 \; , \end{aligned} \tag{73}$$

which give the matching constants as,

$$\begin{aligned} H_a &= (m_1-1)\frac{y_t(m_1)+1-m_1}{y_t(m_1)+2(1-m_1)y_t'(1-m_1)} \\ H_b &= H_a\sqrt{y_t'(m_1)} \\ H_c &= 1+H_a \; . \end{aligned} \tag{74}$$

Then we simply use $Y_t(x)$ in place of $y_t(x)$ in equations (71), with m_1=0.8.

In practice the modified sections are indistinguishable from the usual NACA sections at typical visualisation scales, but successfully serve their purpose of keeping the panel matrix well-conditioned with increasing numbers of panels. These modified sections are used in the computation of all the results which follow.

5.6.2 Overview of Results

The model was tested with a range of aerofoil shapes, angles of attack and Reynolds numbers; the most interesting results were of the low-speed stall process. Even using thin sections at a shallow angle of attack with a high Reynolds number, the boundary layer does not remain attached around the entire lifting surface. Rather, we found that the separation point moves from the trailing edge forward along the surface of the aerofoil and the flow becomes unstable.

This process is illustrated in Figure 224 and Figure 225. The section shown is a NACA 4412 at 5° angle of attack at Reynolds number 10,000. The computation was performed using 200 boundary points and a time step of 0.05.

In Figure 224, the flow at τ=1.0 is shown. The starting vortex can be seen approximately one chord length downstream from the trailing edge. The boundary layer remains attached to the lifting surfaces of the aerofoil and the separation point is at the trailing edge.

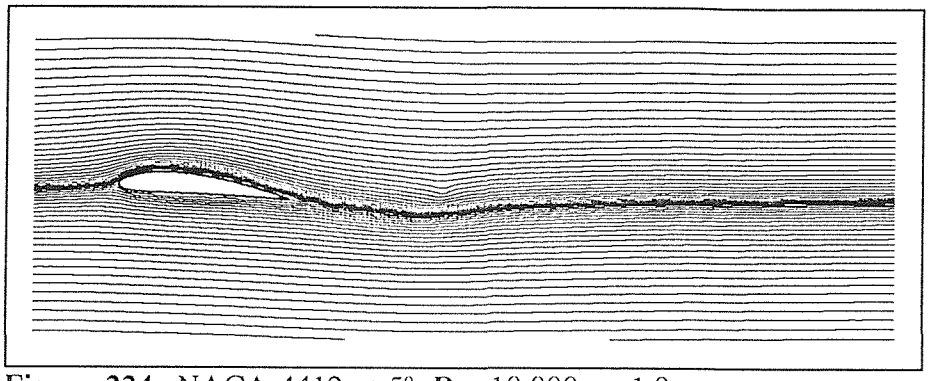


Figure 224 NACA 4412 at 5°, **Re**=10,000, τ=1.0

As time progresses, the separation point starts to oscillate near the trailing edge of the wing section, mainly on the upper lifting surface. This results in eddies, whose size increases with time, being shed into the wake. Figure 225 shows the computed streamlines at τ=7.0, where the boundary layer has become detached from the lifting surface on the upper side at the rear.

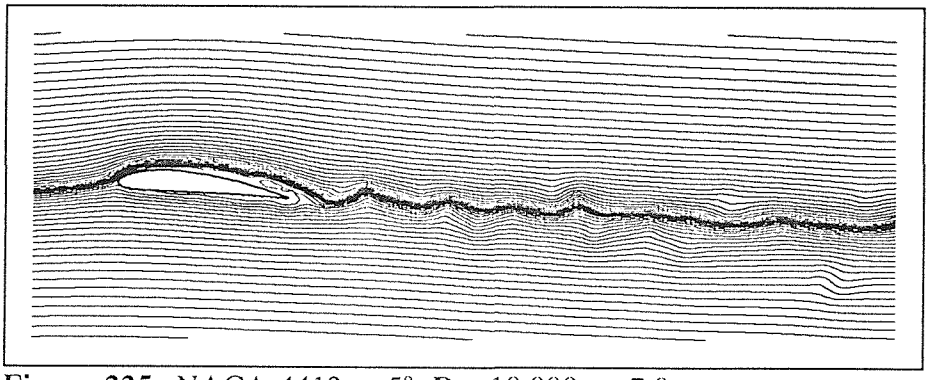


Figure 225 NACA 4412 at 5°, **Re**=10,000, τ=7.0

This instability increases the drag and reduces the lift on the wing section. We have also tested the model with the Reynolds number set at extremely high values, such as 10^6. It was found that the lift coefficient is calculated to be much smaller than that from thin aerofoil theory in inviscid flow and that the phenomena of detachment of the boundary layer and separation from lifting surfaces still occur. This happens because there is no model for the turbulent boundary layer and because the effective maximum Reynolds number, caused by

time-stepping errors, has been exceeded. Anderson (1992) presents similar results for laminar and turbulent finite difference calculations. In the laminar model, separation similar to that seen in our solutions occurs; in the turbulent case, the flow remains attached.

5.6.3 NACA 0012 at 5°, Re=21,000

Two sample visualisations are now given of the computed flow past a NACA 0012 at an angle of attack of 5° and Reynolds number 21,000. The calculation was performed using 200 boundary panels and a time step of 0.05. Four experimental visualisations of this flow are presented in McAlister and Carr (1979) (Fig. 3(a)). Some facts should be kept in mind while making a comparison. Firstly, the experiment uses short exposure photographs of hydrogen bubbles suspended in the fluid, thus the line joining chronologically consecutively generated bubbles is a streakline; our numerical plots show streamlines, thus in the unsteady regions a qualitative comparison must be used. However, large structures such as eddies can usually be seen in the experimental flows. Secondly, no times are given for the experimental photographs, again necessitating a qualitative comparison. Thirdly, the wing section used in the experiments has been modified by extending the leading edge and reducing the nose radius, in order to promote leading edge stall.

Despite these reservations, the experimental visualisations are in reasonable qualitative agreement with Figure 226 and Figure 227, which show the computed streamlines at τ=5.0 and τ=11.5 respectively. In particular, the experimentalists report that the leading separation point moves around on the upper surface; this behaviour is clearly seen in our results. The discrete vortex results, after the initial stall, always show one to three eddies on the upper surface of the aerofoil; there is evidence of the presence of eddies on the upper surface in the experiments. Finally, the wake in both computed and experimental flows contains an irregular mixture of large and small vortices of both rotational senses.

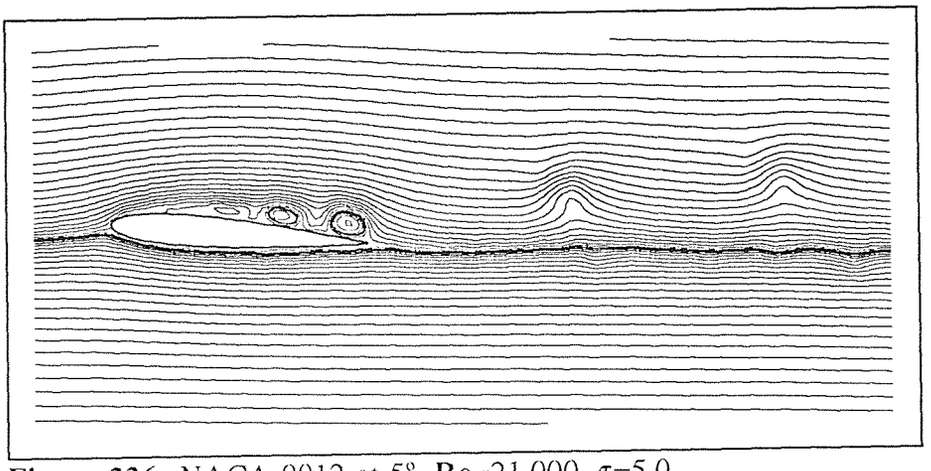


Figure 226 NACA 0012 at 5°, **Re**=21,000, τ=5.0

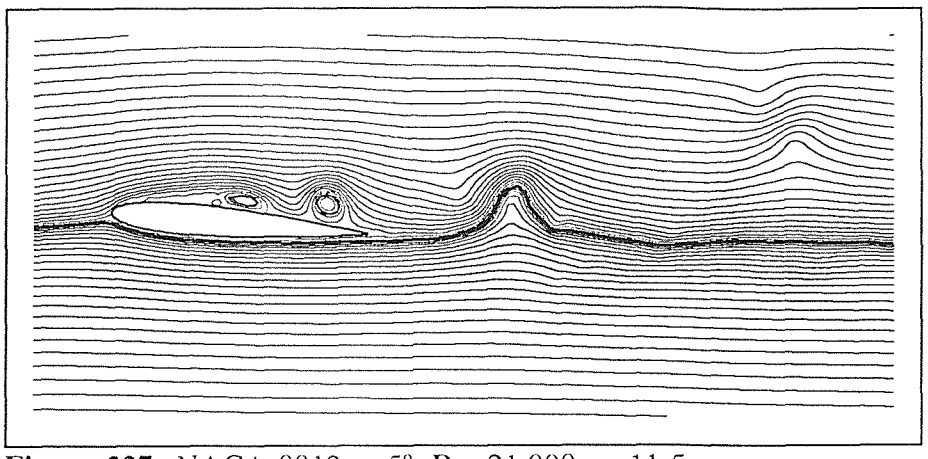


Figure 227 NACA 0012 at 5°, Re=21,000, τ=11.5

McAlister and Carr emphasize that the boundary layer is laminar in the experiments at Reynolds number 21,000. They also mention that results of repeated experiments under the same conditions can produce different results at these Reynolds numbers. This is further evidence of the strongly non-deterministic behaviour discussed in section 5.4.10.

5.6.4 NACA 0012 at 12°, Re=5000

The flow past a NACA 0012 at an angle of attack of 12° and Reynolds number 5000 is now presented in detail. This run used 200 boundary points and a time step of 0.05. The aerofoil immediately stalls and the flow then develops into a near-periodic state, with vortices shedding into the wake. Physically, the flow will be in the laminar regime at this Reynolds number, so these solutions should have an accuracy similar to those for the circular cylinder. For force results, we are now dependent on the Eulerian technique, so we expect early-time force results to be inaccurate and the oscillations in drag to be larger than calculated, but the average value of drag should be reasonably accurate using a time step of 0.05.

The following streamline plots show the initial evolution of the flow in time intervals of 2.5. Then the near-periodic flow is shown in more detail, using a time interval of 0.5 between frames.

Figure 228 shows the flow at τ=2.5, where the wing has already stalled. At earlier times a single eddy formed on the upper surface; there are now two distinct eddies. The wake is still reasonably straight as, at this stage, the only significant large eddy to have shed is the starting vortex.

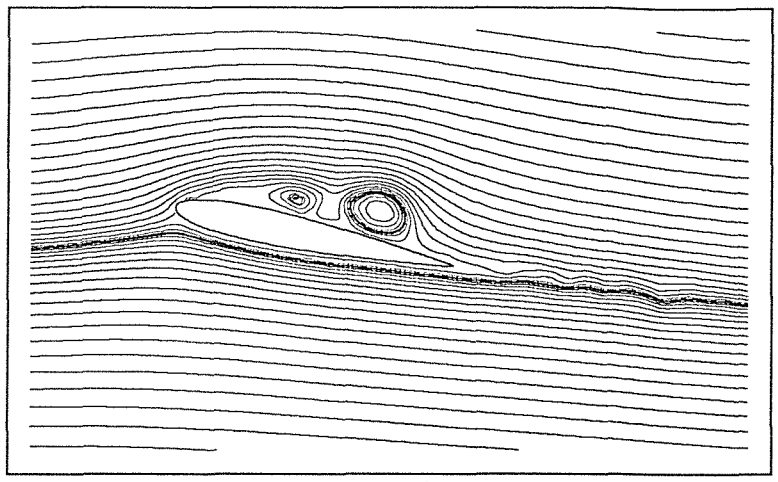


Figure 228 NACA 0012 at 12°, **Re**=5000, τ=2.5

In Figure 229-Figure 232, the flow moves towards the near-periodic shedding pattern which eventually develops. At early times, some very large eddies develop on the upper surface and then shed into the wake. During this early phase the lift, shown in Figure 240, drops significantly but only for a brief period near τ=6.0. Subsequently, the eddies become smaller and cause smaller oscillations in the lift, which remains positive although it undergoes large oscillations as the eddies shed.

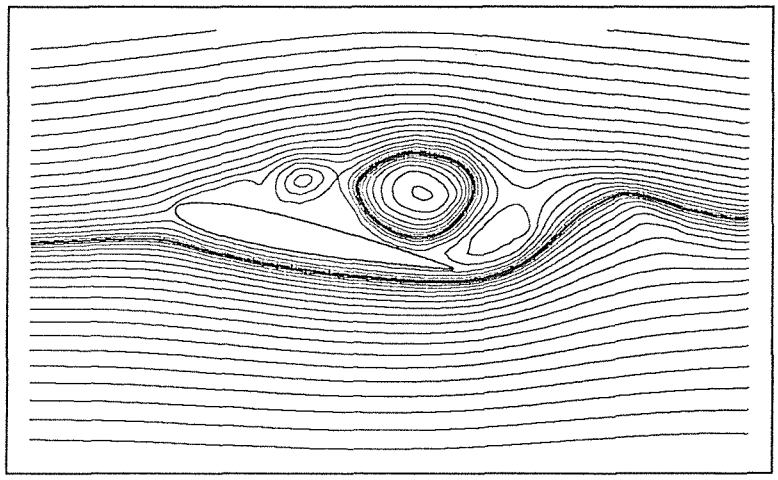


Figure 229 NACA 0012 at 12°, **Re**=5000, τ=5.0

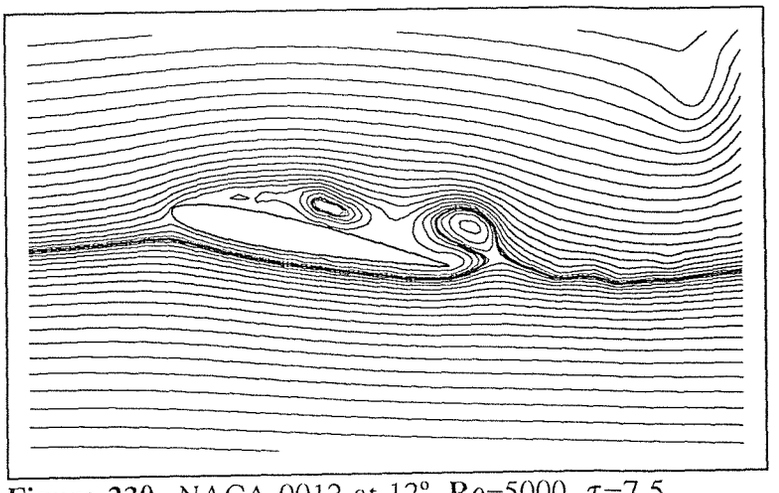


Figure 230 NACA 0012 at 12°, Re=5000, τ=7.5

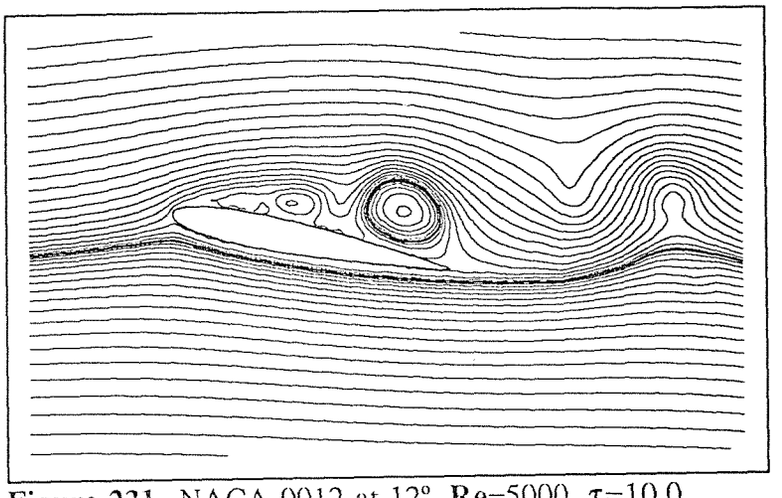


Figure 231 NACA 0012 at 12°, Re=5000, τ=10.0

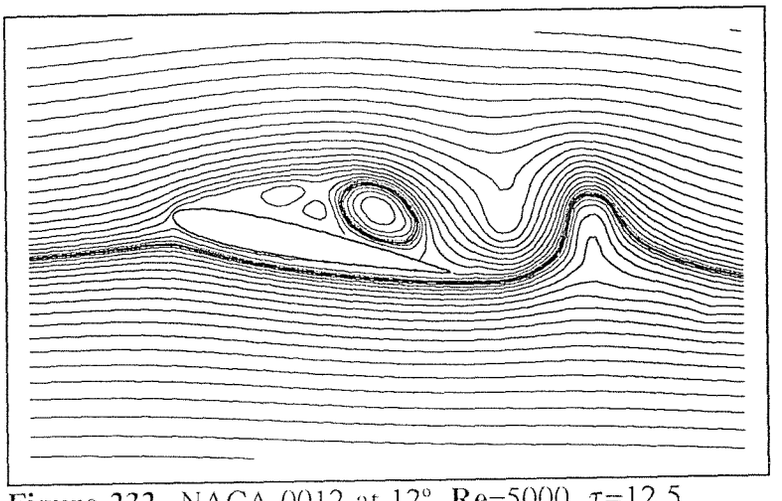


Figure 232 NACA 0012 at 12°, Re=5000, τ=12.5

Figure 233-Figure 239 show two cycles of the near-periodic flow in more detail. Notice that the flow is *not* precisely periodic and the states tend to be similar, but not identical to those on the previous cycle. The period, as can be seen from the lift in Figure 240, is

approximately 1.5 non-dimensional time units. To examine the periodicity, we can compare visualisations separated by a time interval of 1.5 units. In this example the middle part of the cycle shows slightly different behaviour, with eddies which were distinct on previous cycles being joined, probably because the flow is changing very rapidly and the period is not precisely 1.5 time units.

The flow is driven primarily by vorticity separating from the upper lifting surface near the leading edge, which forms into eddies attached to the upper surface. These eddies grow to a variety of sizes and travel the length of the aerofoil to eventually shed from the trailing edge, with complex interactions taking place between the attached eddies.

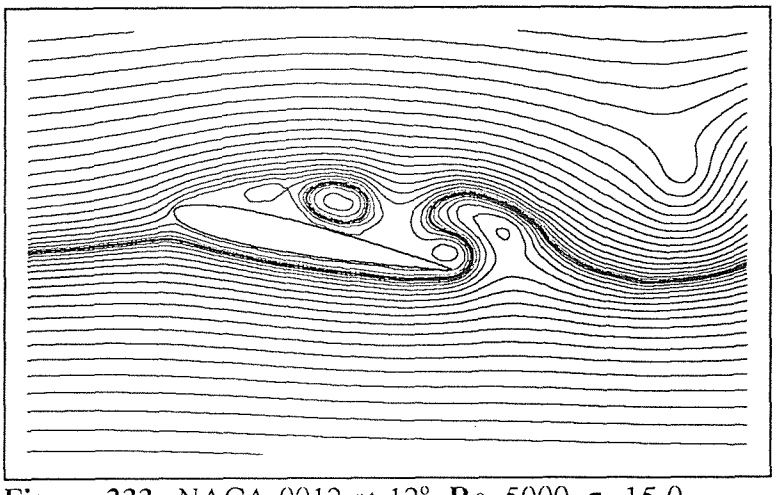


Figure 233 NACA 0012 at 12°, **Re**=5000, τ=15.0

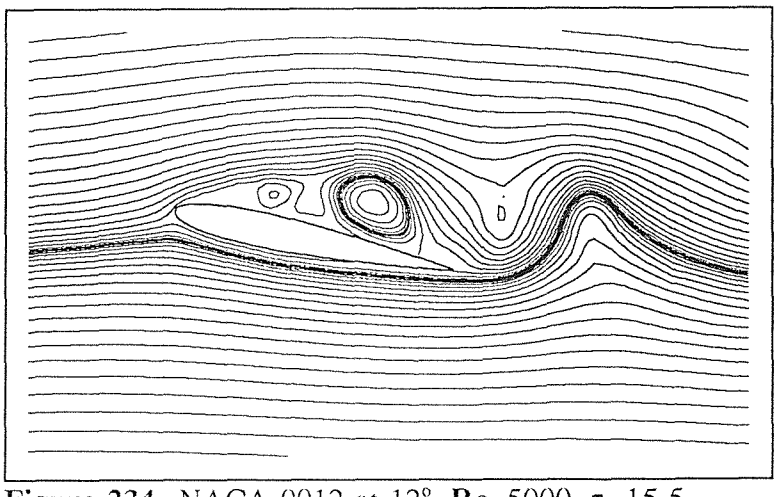


Figure 234 NACA 0012 at 12°, **Re**=5000, τ=15.5

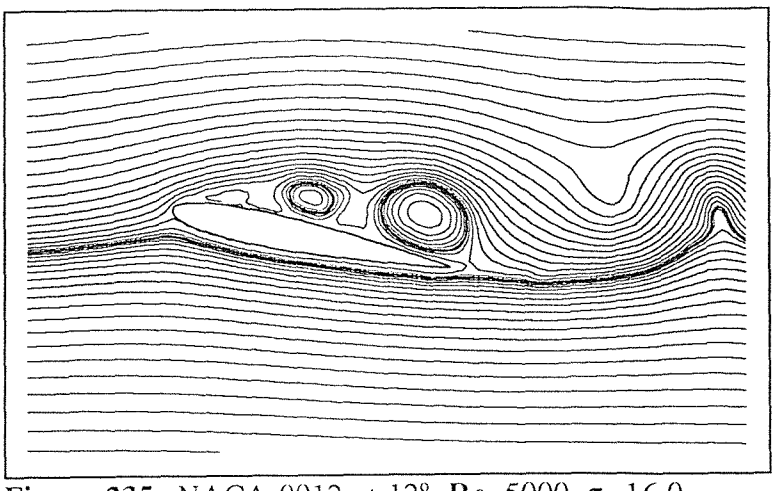


Figure 235 NACA 0012 at 12°, **Re**=5000, τ=16.0

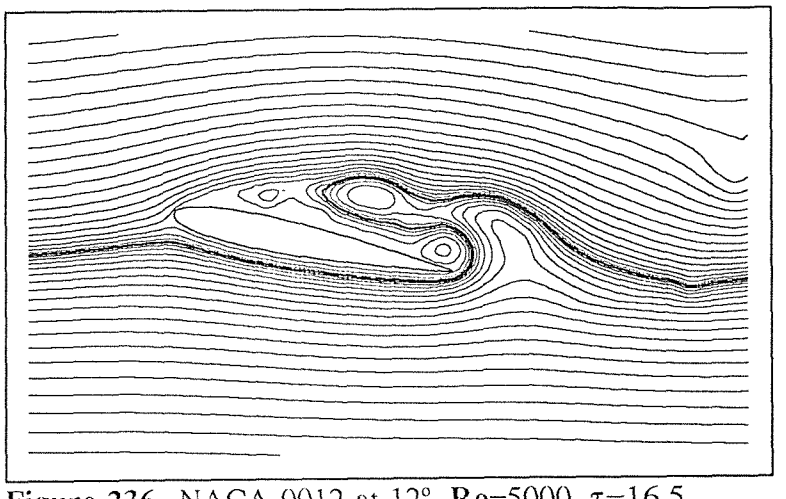


Figure 236 NACA 0012 at 12°, **Re**=5000, τ=16.5

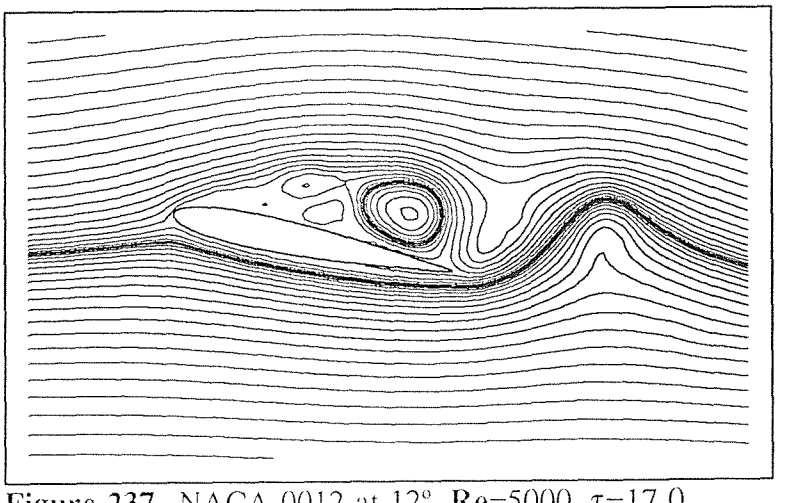


Figure 237 NACA 0012 at 12°, **Re**=5000, τ=17.0

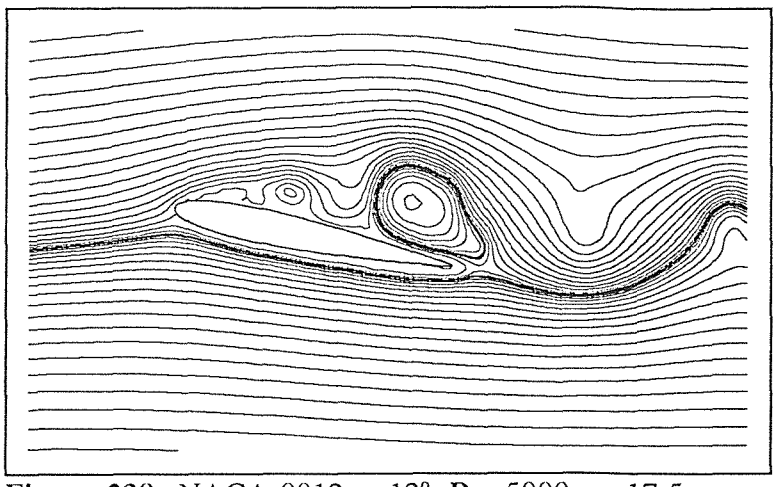


Figure 238 NACA 0012 at 12°, **Re**=5000, τ=17.5

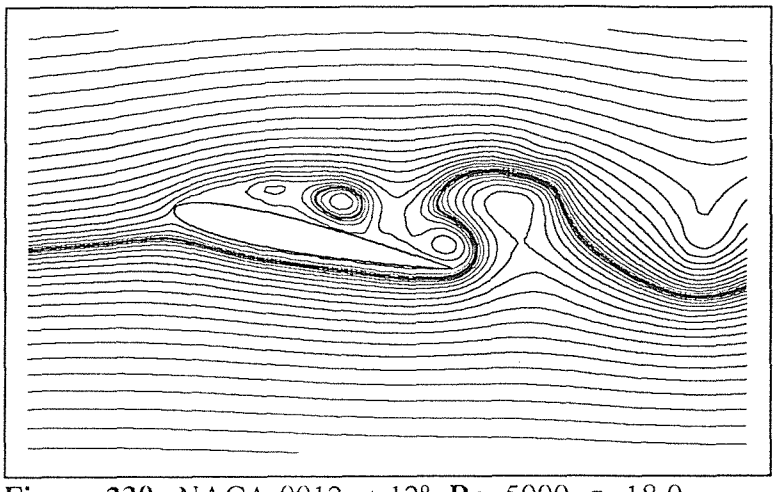


Figure 239 NACA 0012 at 12°, **Re**=5000, τ=18.0

Figure 240 shows the drag and lift coefficients for the wing section for the same run. The drag quickly settles quickly to a near constant value, with mean value 0.23. The lift is initially erratic but eventually develops a behaviour with a clear periodic component. The period is measured to be 1.53, leading to a Strouhal number of 0.65. The mean long term lift, measured from τ=8.0 onwards, is 0.63. So the lift is lower than the value predicted from lifting line theory, although its value does remain positive.

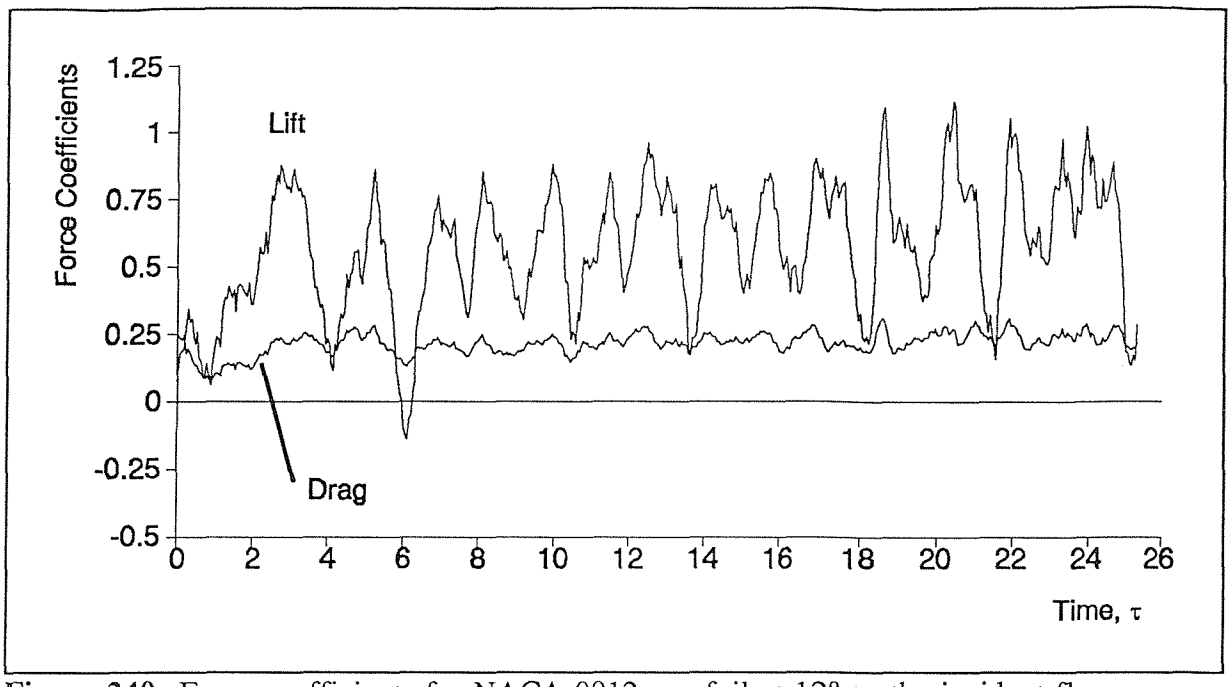


Figure 240 Force coefficients for NACA 0012 aerofoil at 12° to the incident flow

5.6.5 NACA 4412 at 12°, Re=5000

Now we present flow past a NACA 4412 wing section in precisely the same conditions as used in the previous section, an angle of attack of 12° and Reynolds number 5000. The run was conducted using 200 boundary points and a time step of 0.05. The purpose of this run is to examine the stall process and to isolate the flow differences caused by using a cambered aerofoil. Two visualisations at early times are shown in order to illustrate the stall itself, then the near-periodic flow is shown in detail.

Figure 241 shows the flow at τ=2.5, where the stall has started with the formation of a recirculation zone on the upper aerofoil surface. The process is not so far developed as in the case of the symmetric wing section (see Figure 228), as the eddy on the upper surface is still intact. Thus in these particular conditions, the camber has delayed the onset of the stall.

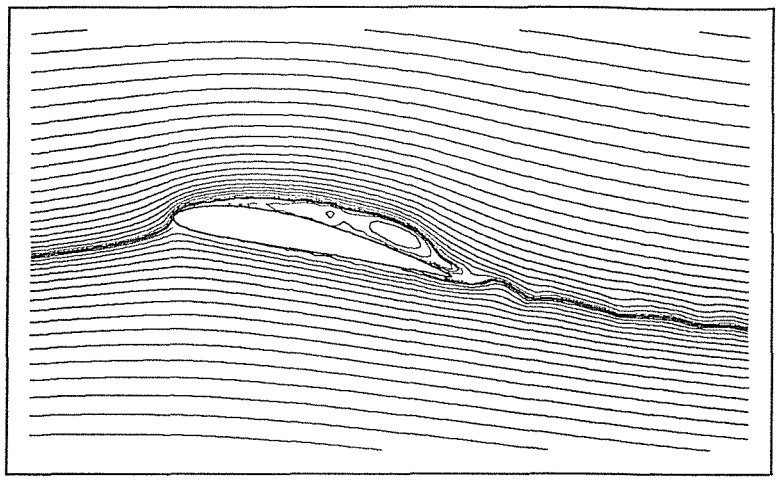


Figure 241 NACA 4412 at 12°, **Re**=5000, τ=2.5

Figure 242 shows the flow at τ=5.0 and is similar to the NACA 0012 at the same time, as shown in Figure 229. Thus, although the initial stall is delayed with the cambered section, the eddy shedding ultimately develops to a similar state. The large drop in lift occurs slightly after this time, near τ=6.0, which is almost the same time as occurred with the symmetric aerofoil.

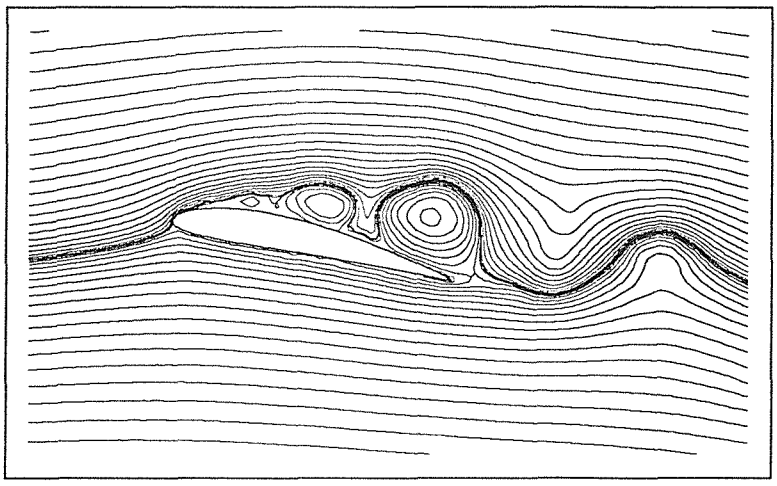


Figure 242 NACA 4412 at 12°, **Re**=5000, τ=5.0

Figure 243-Figure 249 show the flow at time intervals of 0.5 for the near-periodic state which eventually develops. The period of the cycle is approximately 2 non-dimensional time units in this case, longer than with the symmetric aerofoil. Visualisations one cycle later (eg. Figure 245 and Figure 249) are clearly very similar, but again the flow is not precisely periodic.

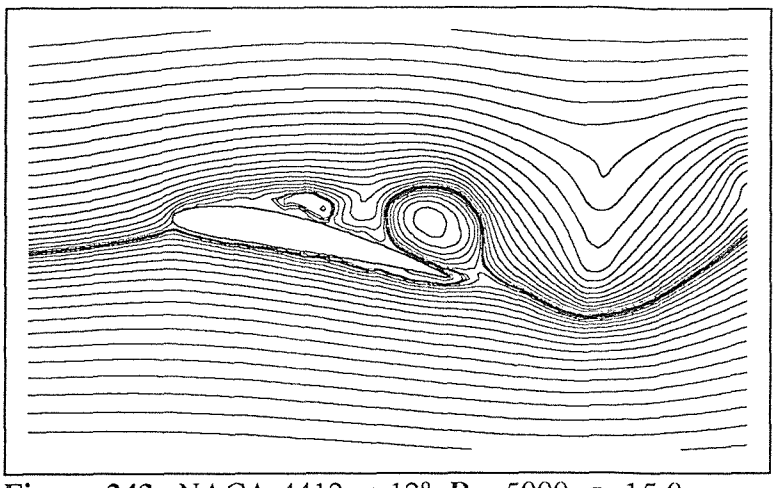


Figure 243 NACA 4412 at 12°, Re=5000, τ=15.0

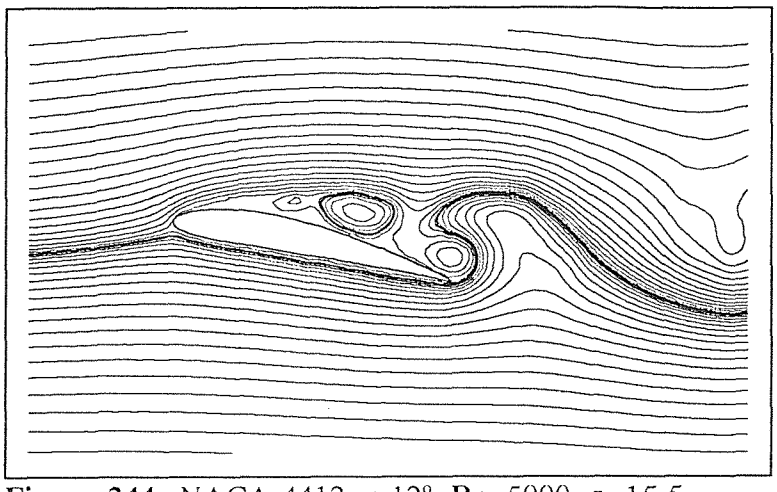


Figure 244 NACA 4412 at 12°, Re=5000, τ=15.5

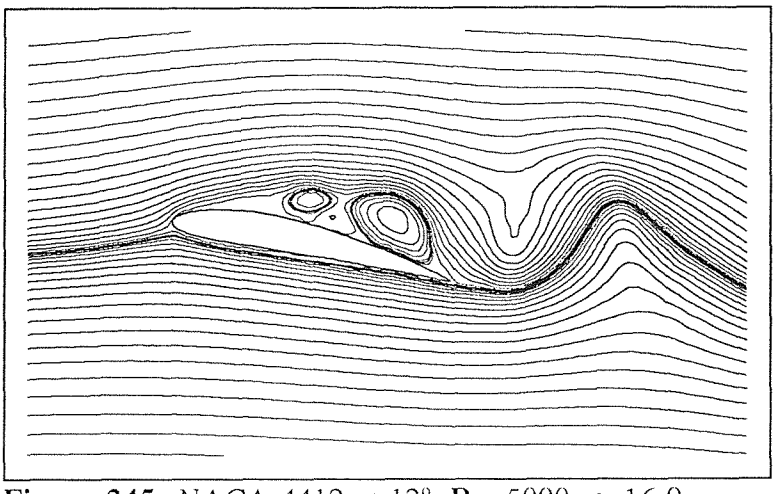


Figure 245 NACA 4412 at 12°, Re=5000, τ=16.0

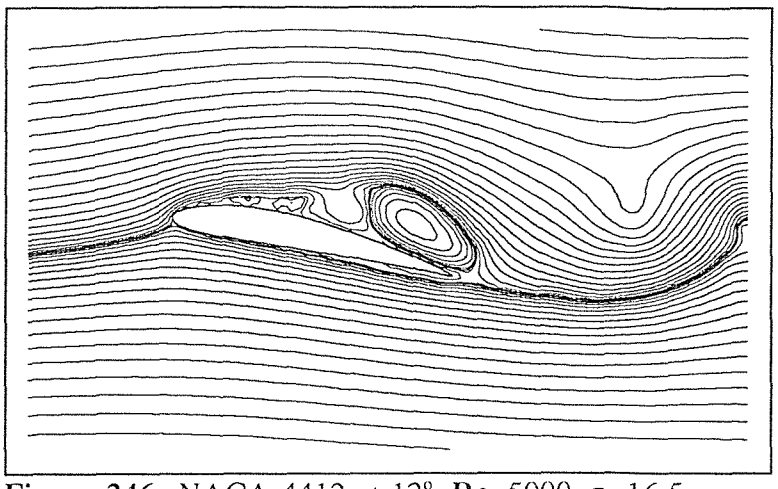


Figure 246 NACA 4412 at 12°, **Re**=5000, τ=16.5

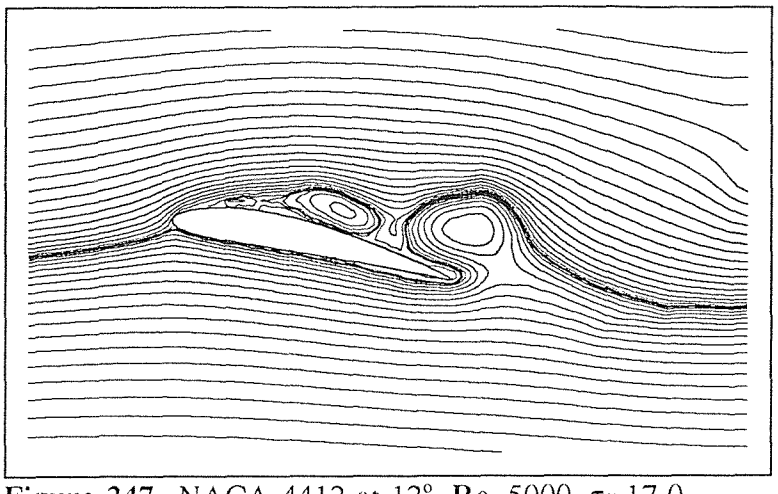


Figure 247 NACA 4412 at 12°, **Re**=5000, τ=17.0

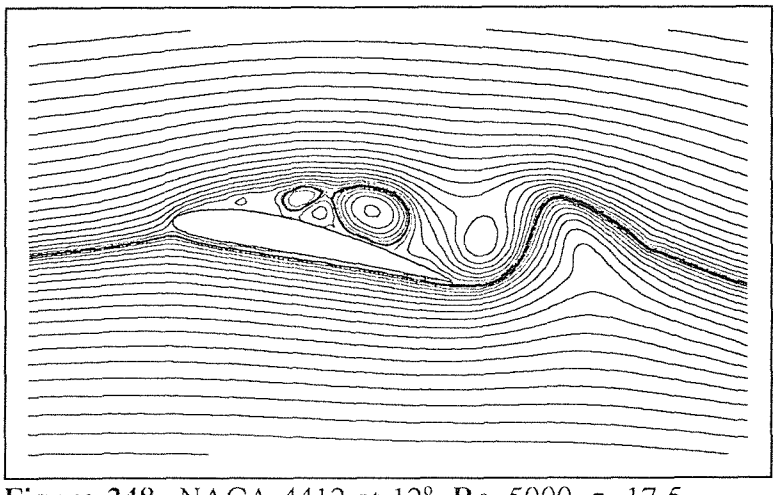


Figure 248 NACA 4412 at 12°, **Re**=5000, τ=17.5

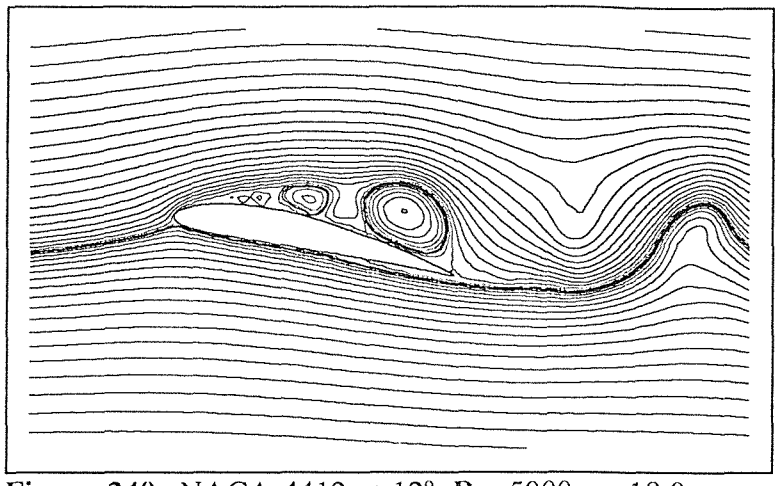


Figure 249 NACA 4412 at 12°, **Re**=5000, τ=18.0

Figure 250 shows the lift and drag coefficients as calculated for this run. The graph is qualitatively similar to that for the NACA 0012 in the same conditions, shown in Figure 240. The behaviour of the lift at early times is likely to be inaccurate; an erratic phase follows where the lift is quite high. Next lift undergoes a sharp drop before the near-periodic cycle is established. The period of the cycle is measured to be 1.88, giving a Strouhal number of 0.53. The average value of the long term lift (measured for $\tau \geq 8.0$) is 0.71, higher than for the NACA 0012. The drag stays at a near constant value, with a mean of 0.23. In conclusion, this flow behaves in a similar manner to that using a symmetric aerofoil. The camber has the effect of slowing the initial development of the stall eddy and the near-periodic phase which develops has a lower frequency.

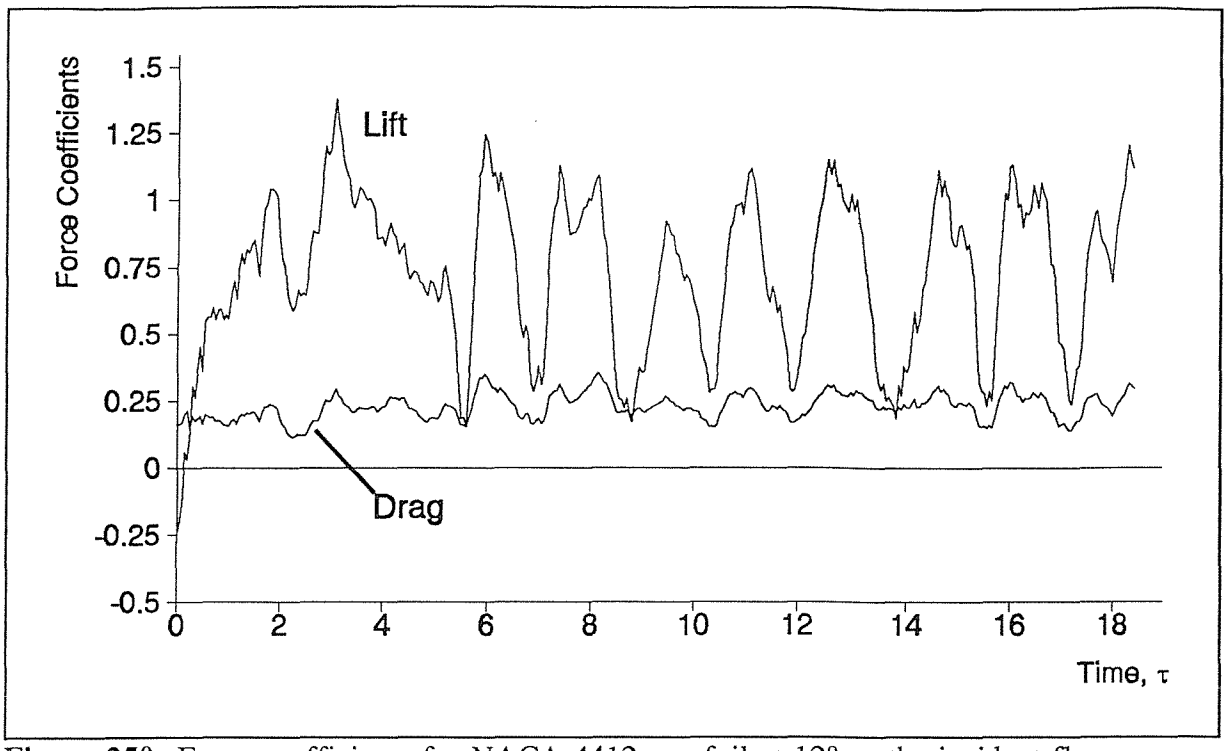


Figure 250 Force coefficients for NACA 4412 aerofoil at 12° to the incident flow

5.6.6 Flow Past NACA Aerofoils - Conclusions

Unfortunately our model is not suitable for simulation of wings in flying conditions and so cannot be used to predict flight performances. However, the model will allow a detailed study of the flow features and forces during the low speed stall process.

We have shown aerofoils stalling in a variety of conditions. At these Reynolds numbers the flows are driven by separation of the boundary layer from lifting surfaces and consequent instability. In particular, separation occurs near the leading edge causing eddies to form and travel down the lifting surfaces to ultimately shed from the trailing edge. Other runs have shown us that these eddies are larger in size when using aerofoils at higher angles of attack and lower Reynolds numbers, as expected. We do not know how far this trend will continue but it may be possible to stabilise the flow in a highly accurate two-dimensional calculation. This would be an interesting piece of theoretical fluid mechanics.

6. Conclusion

6.1 Summary and Discussion

The discrete vortex method has been thoroughly investigated. Working from the extensive existing research and injecting some original ideas, a numerical solution technique has been formulated. In the literature, many additions to the basic model which are claimed to improve results are suggested. We have tried to construct a model containing as few arbitrary parameters as possible, although it was not possible to eliminate them entirely. The resulting technique can be used to solve the two-dimensional Navier-Stokes equations for an incompressible homogeneous Newtonian fluid. The specific case of impulsively started flow past a bluff body has been considered here.

The mathematics of the method have been investigated. The technique builds from a model of an unbounded inviscid fluid, introducing a viscosity model and using potential theory to create boundaries. We have developed a new panel method to represent the body; results show that the boundary condition at the body surface is satisfied more accurately than using a traditional method. The ad-hoc additions to the model have also been considered.

The computational cost of the method has been reduced by the use of a zonal decomposition/summation algorithm and by utilisation of MIMD parallel architecture computers. Timings of runs confirm that the improved algorithm has near linear dependence on the number of vortices used to simulate the fluid motion. In addition, performance scales well with increases in the number of processors used to run the computation. This powerful combination of modern computational and numerical techniques has allowed us to investigate the properties of the model using very large runs. A large number of vortices may be used to increase the spatial resolution of the model for short time, high accuracy results; fewer vortices may be used to take a run further in non-dimensional time. It would not have been practical to perform such a computationally intensive study as this one using ordinary sequential/vector computers, because of the limited computer time available.

The first test problem for the discrete vortex code was the impulsively started flow past a circular cylinder. An experimental test of the convergence of the method revealed a systematic dependence of results upon the choice of input parameters, such as the time step. The main result of this sensitivity is that a 'time-shift' can occur such that the solutions produced by the method compare to experiments at different times. Solutions are otherwise qualitatively independent of the choice of the model parameters. Excellent comparisons with experiments can be produced using a small time step and a large number of boundary points, provided the time-shift is taken into account.

Results for the short time flow past the non-rotating cylinder at a range of Reynolds numbers compare well with experiments and another numerical scheme. The majority of these test-computations were conducted using a moderately large time step in order to test the validity of such solutions for extension to long runs. During these tests the lower limit of Reynolds number for the method was found to be approximately 300, below which the relatively large diffusion of discrete vortices causes an undesirably large level of noise in solutions.

Long time solutions were presented for flow past a non-rotating cylinder at three Reynolds numbers. Global flow quantities, such as Strouhal number and mean drag, are in good agreement with published experimental and numerical figures. At Reynolds number 550 the early near-symmetrical flow becomes unstable, then eddies shed to form a regular Karman vortex street. As Reynolds number increases, the eddy shedding pattern becomes more irregular. At Reynolds numbers 5000 and 31,700, eddies are seen to shed in pairs, which then travel slightly in the transverse direction as they are carried downstream. Regular shedding modes are also possible, which can form a near-regular vortex street in some cases. Another possible phase is similar to the near-symmetric flow at early times; a pair of eddies grow behind the cylinder accompanied by a constant lift and a large drop in drag. These findings are consistent with discussions given in the literature concerning the irregularity of high Reynolds number flows. This non-determinism might be explained in terms of the nonlinearity and the consequent bifurcation of the Navier-Stokes equations, possibly leading to chaos.

An alternative test problem was also considered — flow past the rotating cylinder, at a variety of rotational speeds. The discrete vortex method compares well with experimental visualisations and another numerical method at Reynolds number 1000. The flows at low rotational rates are broadly similar to those using a non-rotating cylinder, with a periodic eddy shedding behaviour. As the rotation rate increases, a wide layer of fluid forms at the boundary which rotates with the cylinder; there also appears to be a trend towards long term pseudo-steady behaviour — at the highest rotation rate considered, a mean steady flow is established which is subject to small amplitude variations. The qualitative behaviour of the computed force coefficients is in agreement with another numerical study, but our long term values are systematically smaller in magnitude.

The code was developed to solve for flow past an arbitrary body shape, so that we could study the flow past four-digit NACA aerofoils. The lack of a turbulence model, the fundamental limitation of the model to incompressible fluids and an upper limit on Reynolds number for valid solutions, precludes the study of aerofoils at typical flying speeds. The model is suited to simulating the dynamics of the low-speed stall process. Results for a NACA 0012 at a 5° angle of attack and Reynolds number 21,000 are in good qualitative agreement with experimental visualisations. The stall process is clearly visualised; eddies are

formed on the upper lifting surface that move along the aerofoil to shed from the trailing edge. A comparison of two flows over aerofoils at 12° angle of attack shows that the cambered NACA 4412 inhibits the development of low-speed stall more than the symmetric NACA 0012. Using a cambered aerofoil, the initial stall takes longer to develop and the periodic flow which follows has a lower frequency. Detailed visualisations are presented in both cases.

In conclusion, the discrete vortex method has been demonstrated to be capable of producing worthwhile solutions of the two-dimensional Navier-Stokes equations. The formulation of the method is complex, especially in the choice of the numerous ad-hoc modelling parameters which are necessary. Computations are not especially cheap, except at short times, but are highly suited to exploitation of parallel architectures.

The method can produce results of comparable accuracy to other standard computational fluid dynamics techniques, such as finite-difference or finite-volume methods and, although it is unlikely to ever replace these mainstream methods, is a useful and complimentary addition to the modeller's toolkit. In particular, the discrete vortex method can provide a completely independent check on results because it adopts a fundamentally different approach to the problem. In particular, it does not suffer from the grid-related errors common to all grid-based methods. If the discrete vortex results are in close agreement with another method, the level of confidence of those results is significantly increased, which could be valuable in a situation where the flow is unsuitable for experiments but where the accuracy of results is extremely important.

6.2 Ideas for Further Research

This work has demonstrated the construction of a robust numerical solver using the discrete vortex method. However, we believe that further work could improve results for the same set of problems or generalise the method to solve additional flow problems.

We believe that the main source of numerical errors in the current method is the boundary model. Future numerical schemes could significantly benefit from a careful study of the boundary representation. Using some simple boundary problems, comparisons should be made with either analytic solutions or trusted numerical results to ensure that the small-scale details of the boundary model are correct.

The vortex-in-cell scheme is favoured by many authors. It would be possible to develop a parallel algorithm to use this technique. The finite-volume or spectral-element schemes both parallelise efficiently and could be used to solve the Poisson equation on the grid. Both the

grid and the vortices could be stored in a distributed manner across the network of processors and thus the workload balanced. The computational benefits of parallel processing would allow fine grids and large numbers of vortices to be used, increasing the accuracy of the results. Most of the techniques needed to combine the Lagrangian and Eulerian approaches are either known or should be relatively simple to formulate. However, a considerable amount of software development would be required.

The current model will only solve for stationary boundaries, apart from the specific case of the rotating cylinder. If a different panel method were used, it would be possible to consider an arbitrarily moving boundary. This would allow, in particular, the problem of dynamic stall of a wing section to be studied in detail. The most obvious approach is to construct a boundary model which allows an arbitrary velocity to be assigned to each control point. The panel method required to construct such a model would necessarily be more complicated, as our method does not have a sufficient number of degrees of freedom.

Current three-dimensional discrete vortex methods allow accurate solution of some fundamental fluid dynamics problems, such as propagation of a vortex ring. However, with advances in modelling techniques and the trend towards ever decreasing computational cost, the point will soon be reached where larger problems can be tackled. A similar approach to that used here could be adopted for the three-dimensional method, ie. develop a cheap, parallel algorithm. The induction law for vortex elements has a similar form to that in two dimensions so we expect to be able to formulate a zonal decomposition/summation algorithm for the three-dimensional problem. The zonal structure would be a hierarchy of cuboids containing equal numbers of elements. The influence of an element in three dimensions decays with the inverse square of distance from it, so the convergence condition should be more favourable. It should be possible to construct an efficient parallel algorithm on a similar basis to the one used here. The data dependency of the calculation remains the same — each processor needs access to the local data stored on each of the others. However, the geometric aspects of the algorithm, such as sorting and spreading, may become significantly more complicated for an arbitrary three-dimensional geometry.

Appendix A : Zonal Decomposition Analysis

In this appendix, the mathematical derivation of the zonal decomposition theorem (equation (47)) is given. The proof is constructive and is used to derive the required convergence condition, which is used in the algorithm to determine when the series may be used.

Start by assuming a situation similar to that shown in Figure 17. Assume that there are N_P point vortices of strengths ω_k situated at the complex points $z_k=x_k+iy_k \in D \subseteq \mathbb{C}$, for $k=1,...N_P$. The aim is to evaluate the velocity field induced by the vortices contained within D at an arbitrary point $z=x+iy$.

The velocity field induced by the N_P vortices at z is,

$$\begin{aligned} u &= -\frac{1}{2\pi}\sum_{k=1}^{N_P}\frac{(y-y_k)\,\omega_k}{|z-z_k|^2} \\ v &= \frac{1}{2\pi}\sum_{k=1}^{N_P}\frac{(x-x_k)\,\omega_k}{|z-z_k|^2} \end{aligned} \qquad (75)$$

Now let $\xi=u-iv$, then we can rewrite equations (75) in the form,

$$\xi = \frac{-1}{2\pi}\left[\sum_{k=1}^{N_P}\frac{(y-y_k)\omega_k + i(x-x_k)\omega_k}{|z-z_k|^2}\right] = \frac{1}{2\pi i}\sum_{k=1}^{N_P}\frac{(z-z_k)^*\,\omega_j}{|z-z_k|^2} = \frac{1}{2\pi i}\sum_{k=1}^{N_P}\frac{\omega_j}{z-z_k} \qquad (76)$$

Take some arbitrary point, λ, within D. Rewrite all the vortex positions relative to the centre point, so let $\delta z_k=z_k-\lambda$. Thus,

$$\xi = \frac{1}{2\pi i}\sum_{k=1}^{N_P}\frac{\omega_k}{z-\lambda-\delta z_k} = \frac{1}{2\pi i}\sum_{k=1}^{N_P}\frac{\omega_k}{z-\lambda}\left[1-\frac{\delta z_k}{z-\lambda}\right]^{-1} \qquad (77)$$

So, provided $|\delta z_k|<|z-\lambda|\ \forall$ k, we can expand this as a series,

$$\xi = \frac{1}{2\pi i}\sum_{k=1}^{N_P}\frac{\omega_k}{z-\lambda}\left[1+\frac{\delta z_k}{z-\lambda}+\left(\frac{\delta z_k}{z-\lambda}\right)^2+\ldots+\left(\frac{\delta z_k}{z-\lambda}\right)^{n_t}+R_{n_t,k}\right]$$

$$= \frac{1}{2\pi i(z-\lambda)}\left[\sum_{k=1}^{N_P}\omega_k+\frac{1}{(z-\lambda)}\sum_{k=1}^{N_P}\omega_k\delta z_k+\frac{1}{(z-\lambda)^2}\sum_{k=1}^{N_P}\omega_k(\delta z_k)^2+\ldots\right.$$

$$\left.+\frac{1}{(z-\lambda)^{n_t-1}}\sum_{k=1}^{N_P}\omega_k(\delta z_k)^{n_t-1}+\sum_{k=1}^{N_P}R_{n_t,k}\right] \qquad (78)$$

now we rewrite this series as

$$\xi = \frac{1}{2\pi i(z-\lambda)}\left[\sum_{j=1}^{n_t}\frac{a_j}{(z-\lambda)^{j-1}}+R_{n_t}\right] \qquad (79)$$

where $a_j = \sum_{k=1}^{N_P}\omega_k(\delta z_k)^{j-1}$ and $R_{n_t} = \sum_{k=1}^{N_P}R_{n_t,k}$

which is in the standard form of a series of n_t terms and a truncation error. The above analysis shows the derivation of the series coefficients and the zonal summation formula, we now move on to estimate the truncation error from an assumed convergence condition. Return to the truncation term,

$$R_{n_t,k} = \frac{\omega_k(\delta z_k)^{n_t}}{(z-\lambda)^{n_t}}\left[1+\frac{\delta z_k}{z-\lambda}+\left(\frac{\delta z_k}{z-\lambda}\right)^2+\ldots\right]$$

$$= \frac{\omega_k(\delta z_k)^{n_t}}{(z-\lambda)^{n_t}}\left[\frac{1}{1+\left(\frac{\delta z_k}{z-\lambda}\right)}\right] \qquad (80)$$

$\because \quad |\delta z_k| < |z-\lambda|.$

We introduce the convergence condition and then show that it bounds the error in the series approximation.

if $|z-\lambda| > h.sup\{|\delta z_k| : k=1,\ldots N_P\} \geq \delta z_k$, for any k

then $\left|1+\frac{\delta z_k}{z-\lambda}\right| \geq 1-\frac{1}{h}$ (by the triangle rule) (81)

$$\Rightarrow \quad \frac{1}{\left|1+\frac{\delta z_k}{z-\lambda}\right|} \leq \frac{1}{1-\frac{1}{h}}$$

Now use this result and equation (80) to bound the total error, ϵ.

$$\epsilon \equiv |R_{n_t}| \le \sum_{k=1}^{N_P} |R_{n_t,k}| \qquad \text{(by the triangle rule)}$$

$$|R_{n_t,k}| = \frac{|\omega_k|\,|\delta z_k|^{n_t}}{|z-\lambda|^{n_t}}\left|1+\frac{\delta z_k}{z-\lambda}\right|^{-1} \le \frac{|\omega_k|}{h^{n_t-1}(h-1)} \tag{82}$$

$$\therefore \quad |R_{n_t}| \le \frac{\sum_{k=1}^{N_P} |\omega_k|}{h^{n_t-1}(h-1)}$$

To this point the analysis is strict. We have shown that the error is bounded and decreases exponentially with the number of terms in the series. The common multiplicative terms have been factorised out of both the series and the error. Since we are using floating point arithmetic, common multiplicative terms will not cause inaccuracy because they can only affect the magnitude of the calculation and not the number of significant figures. The error can be approximated to

$$\epsilon \sim \frac{\sum_{k=1}^{N_P} |\omega_k|}{h^{n_t}} \sim \frac{N_P\overline{\omega}}{h^{n_t}} \tag{83}$$

$$\therefore \quad \log(h) \sim \frac{\log(N_P\overline{\omega}) - \log(\epsilon)}{n_t}$$

where $\overline{\omega}$ is the 'mean' vorticity of a blob.

Equation (83) can be used to calculate an upper limit on the number of terms needed to converge the series to a particular accuracy (an upper limit for the individual circulations of the vortices must also be provided). In practice, we have found that it overestimates by about 50%. This is because the proof uses some weak theory, such as the triangle rule, which will always overestimate the error. See section 4.5.4 for numerical tests of the accuracy of the method.

Appendix B : Pseudo Code for Algorithms

In this appendix, pseudo code for three of the main algorithms is presented. The pseudo code is C-like, but uses Pascal-like constructs where the correct C would have been non-intuitive. It is intended that anyone familiar with a sequential programming language will be able to read the code.

1. Point in Polygon

The following code will take the vortex at `(x,y)`, to the point `(x+u,y+v)` unless it would have entered the polygon whose corners are the first `N` elements of `(px[i],py[i])`, in which case it is taken in the same direction, but only to a point just off the surface. The routine returns 1 if the vortex would have entered the body, allowing it to be optionally deleted.

```
int stepWithReflect(real x, y, u, v, const real px[], py[], const int N)
 {
 real minT, hitS, dot, mod, sx, sy, ux, uy, s, t, x1, x2, y1, y2, dx, dy, epsilon;
 int hitI, hit, i;

 // brings a vortex up to the surface of a polygon
 minT:=1; hit:=0;
 ux:=u; uy:=v;
 for i:=0 to (N-1) {                          // find smallest value of t, the intersection point
   x1:=px[i]; x2:=px[(i+1)%n];
   y1:=py[i]; y2:=py[(i+1)%n];
   dx:=x2-x1; dy:=y2-y1;
   if (abs((ux*dy)-(uy*dx))>1.0E-10) {        // not travelling near-parallel to the wall
     t:=(((y-y1)*dx)-((x-x1)*dy))/bot;        // calculate intersection point at the wall
     if (abs(dx)>abs(dy)) s:=((x-x1)+(t*ux))/dx;
     else                 s:=((y-y1)+(t*uy))/dy;
     if (t>=0 && s>=0 && s<=1 && t<=1) {      //  intersection is a wall collision ?
       mod:=sqrt((dx*dx)+(dy*dy));
       hit:=1; hitI:=i; hitS:=s; minT:=t;
       sx:=dx/mod; sy:=dy/mod;
       }
     }
   }
 if (hit) {                                   // adjust the velocity so that vortex does not enter body
   epsilon:=mod*0.001;                        // just off the surface
   dot:=(ux*sx)+(uy*sy);                      // would be needed if we were zeroing normal component
   u:=(minT*ux)+(epsilon*sy);                 // move to intersection point + slightly along normal
   v:=(minT*uy)-(epsilon*sx);
   }
 x:=x+u;
 y:=y+v;
 return(hit);
 }
```

2. Pigeon Sort

The following code implements the pigeon sort algorithm, as discussed in section 4.7. Upon entry the array `index[]` should contain a permutation of the integers `0,...n-1`.

```
psort(real data[], int index[], const int n) {
  int i, itmp, start, length, noInPig[n], noOfSlot[n], workIndex[n],
      noInSlot[n+1], npig[n+2];
  real min,max,tmp,slotLength;

  for i:=0 to n { noInSlot[i]:=0; }                          // zero in each slot at start
  start:=0;
  noInPig[start]:=n;
  while start<(n-1) {                                        // main loop through data
    while noInPig[start]>1 {                                 // loop to subdivide slots
      length:=noInPig[start];
      max:=data[index[start]];
      min:=max;
      for i:=start to (start+length-1) {                     // loop to find min/max
        tmp:=data[index[i]];
        if (tmp>max) max:=tmp;
        else if (tmp<min) min:=tmp;
        }
      if (max<>min) {                                        // are all values equal ?
        slotLength:=( real(length) )/(max-min);
        for i:=start to (start+length-1) {
          itemp:=int( (data[index[i]]-min)*slotLength );
          noOfSlot[i]:=itemp;
          noInSlot[itemp]:=noInSlot[itemp]+1;
          }
        npig[0]:=start;
        for i:=0 to length {
          npig[i+1]:=npig[i]+noInSlot[i];
          if (noInSlot[i]<>0) noInPig[npig[i]]:=noInSlot[i];
          noInSlot[i]:=0;
          }
        for i:=start to (start+length-1) {
          workIndex[npig[noOfSlot[i]]]:=index[i];
          npig[noOfSlot[i]]:=npig[noOfSlot[i]]+1;
          }
        for i:=start to (start+length-1) { index[i]:=workIndex[i]; }
        }
      else {
        start:=start+(length-1);                             // all equal -> skip this slot
        noInPig[start]:=1;
        }
      }
    start:=start+noInPig[start];
    }
  }
```

3. Zone Splitting

The following code splits an array of real data into two halfs, each containing an equal number of data values. The example here is one-dimensional and so appears rather trivial since a sorted array can be trivially partitioned. However, precisely the same algorithm is used in two dimensions where the recursive splitting can take place on different dimensions. Upon entry the array `sortOrder[]` contains the sorted index as output from the pigeon sort, `zoneOrder[]` contains the vortex indeces lying in the zone to be split, `divide` is the number of vortices to be put into one of the split zones and `N` is the number of vortices to be split.

```
split(int zoneOrder[], const int sortOrder[], const real data[], const int divide, const int N) {
  int tempOrder[maxNumberVortices/2+1];
  real divideValue;
  int pos1,pos2,val;

  divideValue:=data[sortOrder[divide]];                    // value at "mid point"
  pos1:=pos2:=0;
  for i:=0 to (N-1) {
    val:=zoneOrder[i];
    if (data[val]<divide.value) {                          // move to lower zone segment
      zoneOrder[pos1]:=val;
      pos1:=pos1+1;
      }
    else {                                                 // move to temp segment
      tempOrder[pos2]:=val;
      pos2:=pos2+1;
      }
    }
  for i:=0 to (N-divide-1) { zoneOrder[divide+i]:=tempOrder[i]; }   // move temp to upper
  }
```

Appendix C : Streamfunction Analytic Formulae

1. Gaussian Vortex

The streamfunction of a gaussian vortex, at distance r from the vortex, is given by the following integral,

$$\psi(r) = \int_{s=0}^{r} \frac{1}{2\pi s}\left(1 - e^{-\frac{s^2}{\sigma^2}}\right) ds \tag{84}$$

which cannot be evaluated in terms of standard functions. We compute the value of this function using three matched functions, as described below.

1. $r \le \sigma$.

   Use the Taylor series expansion for small r, which gives the following series. Use the first ten terms.

$$\psi(r) = \frac{1}{4\pi}\left[\frac{r^2}{\sigma^2} - \frac{1}{2.2!}\frac{r^4}{\sigma^4} + \frac{1}{3.3!}\frac{r^6}{\sigma^6} - \ldots\right] \tag{85}$$

2. $\sigma < r \le 5\sigma$.

   Use the following fitted polynomial.

$$r_1 = \frac{r-\sigma}{2\sigma} - 1$$

$$\psi(r) = \frac{1}{2\pi}\left[a_0 + a_1 r_1 + a_2 r_1^{\,2} + a_3 r_1^{\,3} + a_4 r_1^{\,4} + a_5 r_1^{\,5} + a_6 r_1^{\,6}\right] \tag{86}$$

where a_0= 1.387160490008288lE+000

a_1= 6.6330940309551012E-001

a_2=-2.1849769095599672E-001

a_3= 1.2261743591800506E-001

a_4=-6.2088524621613669E-002

a_5=-3.5609712665060425E-002

a_6= 4.1523804866750204E-002.

3. $r>5\sigma$

Use the streamfunction of a point vortex.

$$\psi(r) = \frac{1}{2\pi}\ln\left(\frac{r}{5\sigma}\right) + \psi(5\sigma) \tag{87}$$

2. Curved Vortex Sheet

The streamfunction of the new panel element, whose velocity distribution is given in equation (31), can be calculated analytically. In the following equation, the quantities a, b, c, ζ_1, s_0 and s_1 are the same as those described in the velocity equation. The streamfunction at the point z, $\psi(z)$ is given by,

$\psi(z) = Im\{W\}$, where

$$W = \frac{1}{2}\left[\log(\zeta_1 - z) - 3 - \frac{b}{a}\right] + \frac{1}{a(s_0 - s_1)}\left[(As_0 + B)\log\left(\frac{s_0 - 1}{s_0}\right) - (As_1 + B)\log\left(\frac{s_1 - 1}{s_1}\right)\right] \tag{88}$$

$$A = z + b - c + \frac{b^2}{2a}$$

$$B = (c - z)\left(\frac{b}{2a} + 2\right) .$$

References

ANDERSON, J. D., 1984, *Fundamentals of Aerodynamics* (McGraw-Hill).

ANDERSON, J. D., 1992, 'Explicit Finite Difference Methods ...', in *Computational Fluid Dynamics*, Editor Wendt, J. F., (Springer-Verlag).

AREF, H. and FLINCHEM, E. P., 1984, 'Dynamics of a Vortex Filament in a Shear Flow', *J. Fluid Mech.*, Vol. 148, pp. 477-497.

ASHURST, W. T., 1981, 'Vortex Ring Instability', *Bull. Am. Phys. Soc.*, Vol. 26, p. 1267.

ASHURST, W. T., 1983, 'Large Eddy Simulation via Vortex Dynamics', AIAA-83-1879-CP.

BADR, H. M., COUTANCEAU, M., DENNIS, S. C. R. and MÉNARD, C., 1990, 'Unsteady Flow Past a Rotating Circular Cylinder at Reynolds Numbers 10^3 and 10^4', *J. Fluid Mech.*, Vol. 220, pp. 459-484.

BAKKER, P. G., 1991, *Bifurcations in Flow Patterns* (Kluwer).

BAKER, G. R., 1979, 'The Cloud-in-Cell Technique Applied to the Roll-up of Vortex Sheets', *J. Comput. Phys.*, Vol. 31, pp. 76-95.

BATCHELOR, G. K., 1967, *An Introduction to Fluid Dynamics* (Cambridge University Press).

BEALE, J. T. and MAJDA, A., 1981, 'Rates of Convergence for Viscous Splitting of the Navier-Stokes Equations', *Math. of Computation*, Vol. 37, pp. 243-259.

BEALE, J. T. and MAJDA, A., 1982, 'Vortex Methods I and II', *Math. of Computation*, Vol. 39, No.159, pp. 1-52.

BIRCH, P., 1988, 'Pigeon Sort', *Personal Computer World*, Vol. 11, No. 11 (Nov. 1988), pp. 208-213.

BOUARD, R. and COUTANCEAU, M., 1980, 'The Early Stage of Development of the Wake Behind an Impulsively Started Circular Cylinder for $40<Re<10^4$' *J. Fluid Mech.*, Vol. 101, p. 583.

CHEER, A. Y., 1989, 'Unsteady Separated Wake Behind an Impulsively Started Circular Cylinder in Slightly Viscous Fluid', *J. Fluid Mech.*, Vol. 201, pp. 485-505.

CHORIN, A. J., 1973, 'Numerical Study of Slightly Viscous Flow' *J. Fluid Mech.*, Vol. 57, pp. 785-796.

CHORIN, A. J., 1978, 'Random Vortices and Random Vortex Sheets' *SIAM-AMS proceedings*, Vol. 11, pp. 19-31.

CHORIN, A. J., 1981, 'Estimates of Intermittency, Spectra and Blow-Up in Developed Turbulence', *Commun. Pure Applied Math.*, Vol. 34, pp. 853-866.

CHORIN, A. J., 1982, 'Evolution of Turbulent Vortex', *Commun. Math. Phys.*, Vol. 83, pp. 517-535.

CHRISTIANSEN, J. P., 1973, 'Numerical Simulation of Hydromechanics by the Method of Point Vortices', *J. Comput. Phys.*, Vol. 13, pp. 363-379.

COLLINS, W. M. and DENNIS, S. C. R., 1973, 'The Initial Flow Past an Impulsively Started Circular Cylinder', *Q. J. Mech. Appl. Maths*, Vol. 26, p. 53.

COUDER, Y., CHOMAZ, J. M., and RABAUD, M., 1989, 'On the Hydrodynamics of Soap Films', *Physica D*, Vol. 37, pp. 384-405.

EINSTEIN, A., 1956, *Investigation of the Theory of Brownian Motion* (Dover).

ECE, M. C., WALKER, J. D. and DOLIGALSKI, T. L., 1984, *Phys. Fluids*, Vol. 27, pp. 1077.

FINK, P. T. and SOH, W. K., 1978, 'A New Approach to Roll-up Calculations of Vortex Sheets', *Proc. Roy. Soc. London A*, Vol. 362, pp. 195-209.

FRANKE, R., RODI, W. and SCHÖNUNG, B., 1990, 'Numerical Calculation of Laminar Vortex-Shedding Flow Past Cylinders', *J. Wind End. and Ind. Aero.*, Vol. 35, pp. 237-257.

GRAHAM, J. M. R., 1988, 'Computation of Viscous Separated Flow Using a Particle Method', *Numerical Methods for Fluid Dynamics III* (Oxford Science Pub.).

GREENGARD, C. A., 1985, 'The Core-Spreading Approach Approximates the Wrong Equation', *J. Comp. Phys.*, Vol. 61, pp. 345.

HALD, O. H., 1979, 'Convergence of Vortex Methods for Euler's Equations : II', *SIAM J. Numer. Anal.*, Vol. 16, pp. 726-755.

HALD, O. H., 1985, 'Convergence of Vortex Methods for Euler's Equations : III', PAM-270, Pure and Appl. Math. Ctr., Univ. CA, Berkeley.

HALD, O. H., 1986, 'Convergence of a Random Method with Creation of Vorticity', *SIAM J. Sci. Stat. Comput.*, Vol. 7, pp. 1373-1386.

HALD, O. H. and DEL PRETE, M. V., 1978, 'Convergence of Vortex Methods for Euler's Equations', *Math. of Computation*, Vol. 32, pp. 791-809.

HAM, N. D., 1968, 'Aerodynamic Loading on a Two-Dimensional Airfoil During Dynamic Stall', *AIAA J.*, Vol. 6, pp. 1927-1934.

HARLOW, F. H., 1964, 'The Particle-in-Cell Computing Method for Fluid Dynamics', *Methods in Computational Phys.*, Vol. 3, pp. 319-343.

HELMHOLTZ, H., 1867, 'On Integrals of the Hydrodynamical Equations which Express Vortex-Motion', Transl. P. G. Tait, in *Phil. Mag.* (4), Vol. 33, pp. 485-512.

HOARE, C. A. R., 1978, *Communicating Sequential Processes* (Prentice Hall).

ISHII, K., KUWAHARA, K., OGAWA, S. and CHYU, W. J., 1985, 'Computation of Flow Around a Circular Cylinder in a Supercritical Regime', *AIAA*, paper 85-1660, pp. 125-132.

KRASNY, R., 1986, 'A Study of Singularity Formation in a Vortex Sheet by the Point Vortex Approximation', *J. Fluid Mech.*, Vol. 167, pp. 65-93.

LEONARD, A., 1980, 'Vortex Methods for Flow Simulation', *J. Comput. Phys.*, Vol. 37, pp. 289-335.

LEONARD, A., 1985, 'Computing Three-Dimensional Incompressible Flows with Vortex Elements', *Ann. Rev. Fluid Mech.*, Vol. 17, pp. 523-559.

LIGHTHILL, M. J., 1963, 'Introduction: Boundary Layer Theory', Chapter 2 in *Laminar Boundary Layers* (Oxford Univ. Press), pp. 72-82.

McALISTER, K. W., and CARR, L. W., 1979, 'Water Tunnel Visualisations of Dynamic Stall', *Trans. ASME, J. Fluids Eng.*, Vol. 101, pp. 376-380.

MOORE, D. W., 1976, 'The Stability of an Evolving Two-Dimensional Vortex Sheet', *Mathematika*, Vol. 23, pp. 35-44.

MOORE, D. W., 1981, 'On the Point Vortex Method', *SIAM J. Sci. Stat. Comput.*, Vol. 2, No. 1, pp. 65-84.

MOORE, D. W., 1984, 'Numerical and Analytical Aspects of Helmholtz Instability', *Theoretical and Applied Mechanics*, (IUTAM), pp. 263-274.

MORTON, B. R., 1984, 'The Generation and Decay of Vorticity', *Geophys. Astrophys. Fluid Dynamics*, Vol. 28, pp. 277-308.

OGAMI, Y. and AKAMATSU, T., 1991, 'Viscous Flow Simulation Using the Discrete Vortex Model — the Diffusion Velocity Method', *Computers and Fluids*, Vol. 19, pp. 433-441.

PATERSON, A., 1983, *A First Course in Fluid Dynamics* (Cambridge University Press).

PIERCE, D., 1961, 'Photographic Evidence of the Formation and Growth of Vorticity Behind Plates Accelerated from Rest in Still Air', *J. Fluid Mech.*, Vol. 11, pp. 460-464.

POUNTAIN, D. and MAY, D., *A Tutorial Introduction to Occam Programming* (McGraw-Hill).

QUARTAPELLE, L., and NAPOLITANO, M., 1983, 'Force and Moment in Incompressible Flows' *AIAA J.*, Vol. 21, pp. 911-913.

RAO, P. M. and KUWAHARA, K., 1991, 'Numerical Solution of Unsteady Flow Around a Circular Cylinder', *Sādhanā*, Vol. 16, pp. 47-58.

ROSENHEAD, L., 1931, 'The Formation of Vortices from a Surface of Discontinuity', *Proc. Roy. Soc.* Series A, Vol. 134, pp. 170-192.

SAFFMAN, P. G., 1974, 'The Structure and Decay of Trailing Vortices', *Arch. Mech.*, Vol. 26, pp. 423-439.

SARPKAYA, T., 1963, 'Lift, Drag and Added-Mass Coefficients for a Circular Cylinder immersed in a time-Dependent Flow', *ASME J. Appl. Mechanics*, Vol. 85, pp. 13-15.

SARPKAYA, T., 1989, 'Computational Methods with Vortices - The 1988 Freeman Scholar Lecture', *J. Fluids Eng. (Trans. ASME)* Vol. 111, pp. 5-52.

SARPKAYA, T. and IHRIG, C. J., 1986, 'Impulsively Started Flow About Rectangular Prisms: Experiments and Discrete Vortex Analysis', *AMSE Journal of Fluids Engineering*, Vol. 108, pp. 47-54.

SARPKAYA, T. and SHOAFF, R. L., 1979, 'An Inviscid Model of Two-Dimensional Vortex Shedding for Transient and Asymptotically Steady Separated Flow Over a Cylinder', *AIAA J.*, Vol. 17, pp. 1193-1200.

SCHLICHTING, H., 1960, *Boundary Layer Theory* (McGraw-Hill), p. 16.

SETHIAN, J. A. and GHONIEM, A. F., 1987, 'Validation Study of Vortex Methods', *J. Comp. Phys.*, Vol. 74, pp. 283-317.

SMITH, P. A. and STANSBY P. K., 1988, 'Impulsively Started Flow Around a Circular Cylinder by the Vortex Method', *J. Fluid Mech.*, Vol. 194, pp. 45-77.

SPALART, P. R., 1988, 'Vortex Methods for Separated Flows', *Von Karman Inst. for Fluid Mech., Lecture series 1988-05.*

TA PHUOC LOC and BOUARD, R., 1985, 'Numerical Solution of the Early Stage of the Unsteady Viscous Flow Around a Circular Cylinder : a Comparison with Experimental Visualisation and Measurements', *J. Fluid Mech.*, Vol. 160, p. 583.

TAMURA, T., OHTA, I. and KUWAHARA, K., 1990, 'On the Reliability of Two-Dimensional Simulation for Unsteady Flows Around a Cylinder-Type Structure', *J. Wind Eng. and Ind. Aero.*, Vol. 35, pp. 275-298.

TIEMROTH, E. C., 1986, *Simulation of the Viscous Flow Around a Cylinder by the Random Vortex Method*, Ph.D. thesis, Univ. Of Calif., Berkeley.

VAN DER VEGT, J. J. W. and DE BOOM, W. C., 1985, 'Numerical Simulation of Flow Around Circular Cylinders at High Reynolds Numbers', *Proc. BOSS '85*, pp. 227-238.

VAN DER VEGT, J. J. W. and HUIJSMANS, R. H. M., 1984, 'Numerical Simulation of Flow Around Bluff Bodies at High Reynolds Numbers', *15th Symp. Naval Hydro.*, pp. 569-585.

VAN DOMMELEN, L. and RUNDENSTEINER, E. A., 1989, 'Fast, Adaptive Summation of Point Forces in the Two-Dimensional Poisson Equation', *J. Comp. Phys.*, Vol. 83, pp. 126-147.

WAX, N., 1954, *Selected Papers on Noise and Stochastic Processes* (Dover).