C.A.J. Fletcher

Computational Techniques for Fluid Dynamics 1

Fundamental and General Techniques

Second Edition
With 138 Figures

Springer-Verlag
Berlin Heidelberg New York London
Paris Tokyo Hong Kong Barcelona

Springer Series in Computational Physics

Editors: R. Glowinski M. Holt P. Hut H. B. Keller J. Killeen S. A. Orszag V. V. Rusanov

A Computational Method in Plasma Physics
F. Bauer, O. Betancourt, P. Garabedian
Implementation of Finite Element Methods for Navier-Stokes Equations
F. Thomasset
Finite-Difference Techniques for Vectorized Fluid Dynamics Calculations
Edited by D. Book
Unsteady Viscous Flows. D. P. Telionis
Computational Methods for Fluid Flow. R. Peyret, T. D. Taylor
Computational Methods in Bifurcation Theory and Dissipative Structures
M. Kubicek, M. Marek
Optimal Shape Design for Elliptic Systems. O. Pironneau
The Method of Differential Approximation. Yu. I. Shokin
Computational Galerkin Methods. C. A. J. Fletcher
Numerical Methods for Nonlinear Variational Problems
R. Glowinski
Numerical Methods in Fluid Dynamics. Second Edition
M. Holt
Computer Studies of Phase Transitions and Critical Phenomena
O. G. Mouritsen
Finite Element Methods in Linear Ideal Magnetohydrodynamics
R. Gruber, J. Rappaz
Numerical Simulation of Plasmas. Y. N. Dnestrovskii, D. P. Kostomarov
Computational Methods for Kinetic Models of Magnetically Confined Plasmas
J. Killeen, G. D. Kerbel, M. C. McCoy, A. A. Mirin
Spectral Methods in Fluid Dynamics. Second Edition
C. Canuto, M. Y. Hussaini, A. Quarteroni, T. A. Zang
Computational Techniques for Fluid Dynamics 1. Second Edition
Fundamental and General Techniques. C. A. J. Fletcher
Computational Techniques for Fluid Dynamics 2. Second Edition
Specific Techniques for Different Flow Categories. C. A. J. Fletcher
Methods for the Localization of Singularities in Numerical Solutions of Gas Dynamics Problems. E. V. Vorozhtsov, N. N. Yanenko
Classical Orthogonal Polynomials of a Discrete Variable
A. F. Nikiforov, S. K. Suslov, V. B. Uvarov
Flux Coordinates and Magnetic Field Structure: A Guide to a Fundamental Tool of Plasma Theory
W. D. D'haeseleer, W. N. G. Hitchon, J. D. Callen, J. L. Shohet

C. A. J. Fle[illegible]

Computational Techniques for Fluid Dynamics 1

Fundamental and General Techniques

Second Edition
With 138 Figures

Springer-Verlag
Berlin Heidelberg New York London
Paris Tokyo Hong Kong Barcelona

Preface to the First Edition

The purpose of this two-volume textbook is to provide students of engineering, science and applied mathematics with the specific techniques, and the framework to develop skill in using them, that have proven effective in the various branches of computational fluid dynamics (CFD). Volume 1 describes both fundamental and general techniques that are relevant to all branches of fluid flow. Volume 2 provides specific techniques, applicable to the different categories of engineering flow behaviour, many of which are also appropriate to convective heat transfer.

An underlying theme of the text is that the competing formulations which are suitable for computational fluid dynamics, e.g. the finite difference, finite element, finite volume and spectral methods, are closely related and can be interpreted as part of a unified structure. Classroom experience indicates that this approach assists, considerably, the student in acquiring a deeper understanding of the strengths and weaknesses of the alternative computational methods.

Through the provision of 24 computer programs and associated examples and problems, the present text is also suitable for established research workers and practitioners who wish to acquire computational skills without the benefit of formal instruction. The text includes the most up-to-date techniques and is supported by more than 300 figures and 500 references.

For the conventional student the contents of Vol. 1 are suitable for introductory CFD courses at the final-year undergraduate or beginning graduate level. The contents of Vol. 2 are applicable to specialised graduate courses in the engineering CFD area. For the established research worker and practitioner it is recommended that Vol. 1 is read and the problems systematically solved before the individual's CFD project is started, if possible. The contents of Vol. 2 are of greater value after the individual has gained some CFD experience with his own project.

It is assumed that the reader is familiar with basic computational processes such as the solution of systems of linear algebraic equations, non-linear equations and ordinary differential equations. Such material is provided by Dahlquist, Bjorck and Anderson in *Numerical Methods;* by Forsythe, Malcolm and Moler in *Computer Methods for Mathematical Computation;* and by Carnaghan, Luther and Wilkes in *Applied Numerical Analysis.* It is also assumed that the reader has some knowledge of fluid dynamics. Such knowledge can be obtained from *Fluid Mechanics* by Streeter and Wylie; from *An Indroduction of Fluid Dynamics* by Batchelor; or from *Incompressible Flow* by Panton, amongst others.

Computer programs are provided in the present text for guidance and to make it easier for the reader to write his own programs, either by using equivalent constructions, or by modifying the programs provided. In the sense that the CFD practitioner is as likely to inherit an existing code as to write his own from scratch, some practice in modifying existing, but simple, programs is desirable. An IBM-compatible floppy disk containing the computer programs may be obtained from the author.

The contents of Vol. 1 are arranged in the following way. Chapter 1 contains an introduction to computational fluid dynamics, designed to give the reader an appreciation of why CFD is so important, the sort of problems it is capable of solving and an overview of how CFD is implemented. The equations governing fluid flow are usually expressed as partial differential equations. Chapter 2 describes the different classes of partial differential equations and appropriate boundary conditions and briefly reviews traditional methods of solution.

Obtaining computational solutions consists of two stages: the reduction of the partial differential equations to algebraic equations and the solution of the algebraic equations. The first stage, called discretisation, is examined in Chap. 3 with special emphasis on the accuracy. Chapter 4 provides sufficient theoretical background to ensure that computational solutions can be related properly to the usually unknown "exact" solution. Weighted residual methods are introduced in Chap. 5 as a vehicle for investigating and comparing the finite element, finite volume and spectral methods as alternative means of discretisation. Specific techniques to solve the algebraic equations resulting from discretisation are described in Chap. 6. Chapters 3–6 provide essential background information.

The one-dimensional diffusion equation, considered in Chap. 7, provides the simplest model for highly dissipative fluid flows. This equation is used to contrast explicit and implicit methods and to discuss the computational representation of derivative boundary conditions. If two or more spatial dimensions are present, splitting techniques are usually required to obtain computational solutions efficiently. Splitting techniques are described in Chap. 8. Convective (or advective) aspects of fluid flow, and their effective computational prediction, are examined in Chap. 9. The convective terms are usually nonlinear. The additional difficulties that this introduces are considered in Chap. 10. The general techniques, developed in Chaps. 7–10, are utilised in constructing specific techniques for the different categories of flow behaviour, as is demonstrated in Chaps. 14–18 of Vol. 2.

In preparing this textbook I have been assisted by many people. In particular I would like to thank Dr. K. Srinivas, Nam-Hyo Cho and Zili Zhu for having read the text and made many helpful suggestions. I am grateful to June Jeffery for producing illustrations of a very high standard. Special thanks are due to Susan Gonzales, Lyn Kennedy, Marichu Agudo and Shane Gorton for typing the manuscript and revisions with commendable accuracy, speed and equilibrium while coping with both an arbitrary author and recalcitrant word processors.

It is a pleasure to acknowledge the thoughtful assistance and professional competence provided by Professor W. Beiglböck, Ms. Christine Pendl, Mr. R. Michels and colleagues at Springer-Verlag in the production of this textbook. Finally I express deep gratitude to my wife, Mary, who has been unfailingly supportive while accepting the role of book-widow with her customary good grace.

Sydney, October 1987 *C. A. J. Fletcher*

Contents

Dr. Clive A. J. Fletcher
Department of Mechanical Engineering, The University of Sydney
New South Wales 2006, Australia

ISBN 3-540-53058-4 2. Auflage Springer-Verlag Berlin Heidelberg New York
ISBN 0-387-53058-4 2nd edition Springer-Verlag New York Berlin Heidelberg

ISBN 3-540-18151-2 1. Auflage Springer-Verlag Berlin Heidelberg New York
ISBN 0-387-18151-2 1st edition Springer-Verlag New York Berlin Heidelberg

Library of Congress Cataloging-in-Publication Data. Fletcher, C. A. J. Computational techniques for fluid dynamics / C. A. J. Fletcher.– 2nd ed. p. cm.–(Springer series in computational physics) Includes bibliographical references and index. Contents: 1. Fundamental and general techniques. ISBN 3-540-53058-4 (Springer-Verlag Berlin, Heidelberg, New York).–ISBN 0-387-53058-4 (Springer-Verlag New York, Berlin, Heidelberg) 1. Fluid dynamics–Mathematics. 2. Fluid dynamics–Data processing. 3. Numerical analysis. I. Title. II. Series. QC 151.F58 1991 532'.05'0151–dc20 90-22257

Typesetting: Macmillan India Ltd., India
55/3140-543210 – Printed on acid-free paper

Preface to the Second Edition

The purpose and organisation of this book are described in the preface to the first edition (1988). In preparing this edition minor changes have been made, particularly to Chap. 1 to keep it reasonably current. However, the rest of the book has required only minor modification to clarify the presentation and to modify or replace individual problems to make them more effective. The answers to the problems are available in *Solutions Manual for Computational Techniques for Fluid Dynamics* by C. A. J. Fletcher and K. Srinivas, published by Springer-Verlag, Heidelberg, 1991. The computer programs have also been reviewed and tidied up. These are available on an IBM-compatible floppy disc direct from the author.

I would like to take this opportunity to thank the many readers for their usually generous comments about the first edition and particularly those readers who went to the trouble of drawing specific errors to my attention. In this revised edition considerable effort has been made to remove a number of minor errors that had found their way into the original. I express the hope that no errors remain but welcome communication that will help me improve future editions.

In preparing this revised edition I have received considerable help from Dr. K. Srinivas, Nam-Hyo Cho, Zili Zhu and Susan Gonzales at the University of Sydney and from Professor W. Beiglböck and his colleagues at Springer-Verlag. I am very grateful to all of them.

Sydney, November 1990 *C. A. J. Fletcher*

1. Computational Fluid Dynamics: An Introduction

This chapter provides an overview of computational fluid dynamics (CFD) with emphasis on its cost-effectiveness in design. Some representative applications are described to indicate what CFD is capable of. The typical structure of the equations governing fluid dynamics is highlighted and the way in which these equations are converted into computer-executable algorithms is illustrated. Finally attention is drawn to some of the important sources of further information.

1.1 Advantages of Computational Fluid Dynamics

The establishment of the science of fluid dynamics and the practical application of that science has been under way since the time of Newton. The theoretical development of fluid dynamics focuses on the construction and solution of the governing equations for the different categories of fluid dynamics and the study of various approximations to those equations.

The governing equations for Newtonian fluid dynamics, the unsteady Navier-Stokes equations, have been known for 150 years or more. However, the development of reduced forms of these equations (Chap. 16) is still an active area of research as is the turbulent closure problem for the Reynolds-averaged Navier-Stokes equations (Sect. 11.5.2). For non-Newtonian fluid dynamics, chemically reacting flows and two-phase flows the theoretical development is at a less advanced stage.

Experimental fluid dynamics has played an important role in validating and delineating the limits of the various approximations to the governing equations. The wind tunnel, as a piece of experimental equipment, provides an effective means of simulating real flows. Traditionally this has provided a cost-effective alternative to full-scale measurement. In the design of equipment that depends critically on the flow behaviour, e.g. aircraft design, full-scale measurement as part of the design process is economically unavailable.

The steady improvement in the speed of computers and the memory size since the 1950s has led to the emergence of computational fluid dynamics (CFD). This branch of fluid dynamics complements experimental and theoretical fluid dynamics by providing an alternative cost-effective means of simulating real flows. As such it offers the means of testing theoretical advances for conditions unavailable experimentally. For example wind tunnel experiments are limited to a certain range of Reynolds numbers, typically one or two orders of magnitude less than full scale.

Computational fluid dynamics also provides the convenience of being able to switch off specific terms in the governing equations. This permits the testing of theoretical models and, inverting the connection, suggests new paths for theoretical exploration.

The development of more efficient computers has generated the interest in CFD and, in turn, this has produced a dramatic improvement in the efficiency of the computational techniques. Consequently CFD is now the preferred means of testing alternative designs in many branches of the aircraft, flow machinery and, to a lesser extent, automobile industries.

Following Chapman et al. (1975), Chapman (1979, 1981), Green (1982), Rubbert (1986) and Jameson (1989) CFD provides five major advantages compared with experimental fluid dynamics:

(i) Lead time in design and development is significantly reduced.
(ii) CFD can simulate flow conditions not reproducible in experimental model tests.
(iii) CFD provides more detailed and comprehensive information.
(iv) CFD is increasingly more cost-effective than wind-tunnel testing.
(v) CFD produces a lower energy consumption.

Traditionally, large lead times have been caused by the necessary sequence of design, model construction, wind-tunnel testing and redesign. Model construction is often the slowest component. Using a well-developed CFD code allows alternative designs (different geometric configurations) to be run over a range of parameter values, e.g. Reynolds number, Mach number, flow orientation. Each case may require 15 min runs on a supercomputer, e.g. CRAY Y-MP. The design optimisation process is essentially limited by the ability of the designer to absorb and assess the computational results. In practice CFD is very effective in the early elimination of competing design configurations. Final design choices are still confirmed by wind-tunnel testing.

Rubbert (1986) draws attention to the speed with which CFD can be used to redesign minor components, if the CFD packages have been thoroughly validated. Rubbert cites the example of the redesign of the external contour of the Boeing 757 cab to accommodate the same cockpit components as the Boeing 767 to minimise pilot conversion time. Rubbert indicates that CFD provided the external shape which was incorporated into the production schedule before any wind-tunnel verification was undertaken.

Wind-tunnel testing is typically limited in the Reynolds number it can achieve, usually short of full scale. Very high temperatures associated with coupled heat transfer fluid flow problems are beyond the scope of many experimental facilities. This is particularly true of combustion problems where the changing chemical composition adds another level of complexity. Some categories of unsteady flow motion cannot be properly modelled experimentally, particularly where geometric unsteadiness occurs as in certain categories of biological fluid dynamics. Many

geophysical fluid dynamic problems are too big or too remote in space or time to simulate experimentally. Thus oil reservoir flows are generally inaccessible to detailed experimental measurement. Problems of astrophysical fluid dynamics are too remote spatially and weather patterns must be predicted *before* they occur. All of these categories of fluid motion are amenable to the computational approach.

Experimental facilities, such as wind tunnels, are very effective for obtaining global information, such as the complete lift and drag on a body and the surface pressure distributions at key locations. However, to obtain detailed velocity and pressure distributions throughout the region surrounding a body would be prohibitively expensive and very time consuming. CFD provides this detailed information at no additional cost and consequently permits a more precise understanding of the flow processes to be obtained.

Perhaps the most important reason for the growth of CFD is that for much mainstream flow simulation, CFD is significantly cheaper than wind-tunnel testing and will become even more so in the future. Improvements in computer hardware performance have occurred hand in hand with a decreasing hardware cost. Consequently for a given numerical algorithm and flow problem the relative cost of a computational simulation has decreased significantly historically (Fig. 1.1). Paralleling the improvement in computer hardware has been the improvement in the efficiency of computational algorithms for a given problem. Current improvements in hardware cost and computational algorithm efficiency show no obvious sign of reaching a limit. Consequently these two factors combine to make CFD increasingly cost-effective. In contrast the cost of performing experiments continues to increase.

The improvement in computer hardware and numerical algorithms has also brought about a reduction in energy consumption to obtain computational flow simulations. Conversely, the need to simulate more extreme physical conditions, higher Reynolds number, higher Mach number, higher temperature, has brought about an increase in energy consumption associated with experimental testing.

The chronological development of computers over the last thirty years has been towards faster machines with larger memories. A modern supercomputer such as

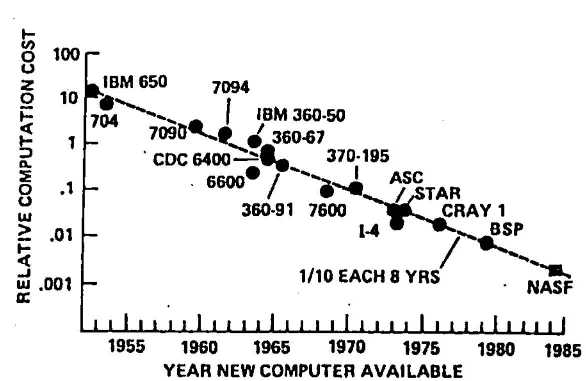


Fig. 1.1. Relative cost of computation for a given algorithm and flow (after Chapman, 1979; reprinted with permission of AIAA)

the CRAY Y-MP is capable of operating at more than 2000 Megaflops (Dongarra 1989). A Megaflop is one million floating-point arithmetic operations per second. More recent supercomputers, e.g. the NEC SX3, are capable of theoretical speeds of 20000 Megaflops. The speed comes partly from a short machine cycle time, that is the time required for each cycle of logic operations. The CRAY Y-MP has a cycle time of 6 nanoseconds (6×10^{-9} s) whereas the NEC SX3 has a cycle time of 2.9 ns.

A specific operation, e.g. a floating point addition, can be broken up into a number of logic operations each one of which requires one machine cycle to execute. If the same operation, e.g. floating point addition, is to be applied sequentially to a large number of elements in a vector, it is desirable to treat each logic operation sequentially but to permit different logic operations associated with each vector element to be executed concurrently. Thus there is a considerable overlap and a considerable speed-up in the overall execution time if the computational algorithm can exploit such a pipeline arrangement.

Modern supercomputers have special vector processors that utilise the pipeline format. However vector processors have an effective "start-up" time that makes them slower than scalar processors for very short vectors. One can define a break-even vector length, N_b, for which the vector processor has the same speed as a scalar processor. For very long vectors ($N=\infty$) the theoretical vector processor speed is achieved.

To compare the efficiency of different vector-processing computers it is (almost) standard practice to consider $N_{1/2}$ (after Hockney and Jesshope 1981), which is the vector length for which half the asymptotic peak vector processing performance ($N=\infty$) is achieved. The actual $N_{1/2}$ is dependent on the specific operations being performed as well as the hardware. For a SAXPY operation ($S=AX+Y$), $N_{1/2}=37$ for a CRAY X-MP and $N_{1/2}=238$ for a CYBER 205. For most modern supercomputers, $30 \leqq N_{1/2} \leqq 100$.

The speed-up due to vectorisation is quantifiable by considering Amdahl's law which can be expressed as (Gentzsch and Neves 1988)

$$G=[(1-P)+P/R]^{-1} \quad \text{and} \quad R=V(N)/S \tag{1.1}$$

where G is the overall gain in speed of the process (overall speed-up ratio)
$V(N)$ is the vector processor speed for an N component vector process
S is the scalar processor speed for a single component process
P is the proportion of the process that is vectorized and
R is the vector processor speed-up ratio.

As is indicated in Fig. 1.2 a vector processor with a theoretical ($N=\infty$) vector speed-up ratio, $R=10$, must achieve a high percentage vectorisation, say $P>0.75$, to produce a significant overall speed-up ratio, G. But at this level $\partial G/\partial P \gg \partial G/\partial R$. Thus modification of the computer program to increase P will provide a much bigger increase in G than modifying the hardware to increase V and hence R. In addition unless a large proportion of the computer program can be written so that vector lengths are significantly greater than $N_{1/2}$, the overall speed-up ratio, G, will not be very great.

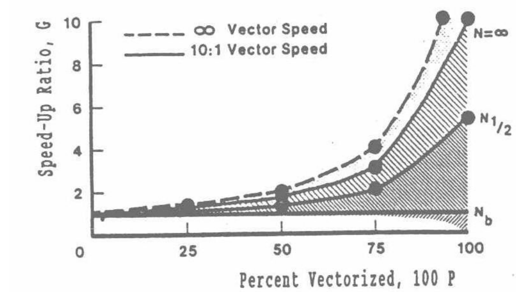


Fig. 1.2. Amdahl's Law

The ability to increase the overall execution speed to the limit set by the hardware depends partly on the ability of the operating system and compiler to vectorise the computational algorithm and partly on choosing computational algorithms that are inherently vectorisable (Ortega and Voigt 1985). The structuring of computational algorithms to permit vectorisation is an important research topic but is beyond the scope of this book (see Gentzsch and Neves 1988). The long term trend would appear to be towards making the operating system and compiler take care of the vectorisation with less emphasis on the user having to manipulate the basic algorithm.

With a pipeline architecture, an efficient vector instruction set and as small a cycle time as possible the major means of further increasing the processing speed is to introduce multiple processors operating in parallel. Supercomputers are typically being designed with up to sixteen processors in parallel. Theoretically this should provide up to a factor of sixteen improvement in speed. Experiments by Grassl and Schwarzmeier (1990) with an eight-processor CRAY Y-MP indicate that 84% of the theoretical improvement can be achieved for a typical CFD code such as ARC3D (Vol. 2, Sect. 18.4.1).

The concept of an array of processors each operating on an element of a vector has been an important feature in the development of more efficient computer architecture (Hockney and Jesshope 1981). The Illiac IV had 64 parallel processors and achieved an overall processing speed comparable to the CRAY-1 and CYBER-205 even though the cycle time was only 80 ns. However Amdahl's law, (1.1), also applies to parallel processors if R is replaced by N_P, the number of parallel processors, and P is the proportion of the process that is parallelisable. The relative merits of pipeline and parallel processing are discussed in general terms by Levine (1982), Ortega and Voigt (1985) and in more detail by Hockney and Jesshope (1981) and Gentzsch and Neves (1988).

The development of bigger and cheaper memory modules is being driven by the substantial commercial interest in data storage and manipulation. For CFD applications it is important that the complete program, both instructions and variable storage, should reside in main memory. This is because the speed of data transfer from secondary (disc) storage to main memory is much slower than data transfer rates between the main memory and the processing units. In the past the main memory size has typically limited the complexity of the CFD problems under investigation.

The chronological trend of increasing memory capacity for supercomputers is impressive. The CDC-7600 (1970 technology) had a capacity of 4×10^5 64-bit words. The CYBER-205 (1980 technology) has a capacity of 3×10^7 64-bit words and the CRAY-2 (1990 technology) has a capacity of 10^9 64-bit words.

Significant developments in minicomputers in the 1970s and microcomputers in the 1980s have provided many alternative paths to cost-effective CFD. The relative cheapness of random access memory implies that large problems can be handled efficiently on micro- and minicomputers. The primary difference between microcomputers and mainframes is the significantly slower cycle time of a microcomputer and the simpler, less efficient architecture. However the blurring of the distinction between microcomputers and personal workstations, such as the SUN Sparcstation, and the appearance of minisupercomputers has produced a *price/performance continuum* (Gentzsch and Neves 1988).

The coupling of many, relatively low power, parallel processors is seen as a very efficient way of solving complex CFD problems. Each processor can use fairly standard microcomputer components; hence the potentially low cost. A typical system, QCDPAX, is described by Hoshino (1989). This system has from 100 to 1000 processing units, each based on the L64132 floating point processor. Thus a system of 400 processing units is expected to deliver about 2000 Megaflops when operating on a representative CFD code.

To a certain extent the relative slowness of microcomputer-based systems can be compensated for by allowing longer running times. Although 15 mins on a

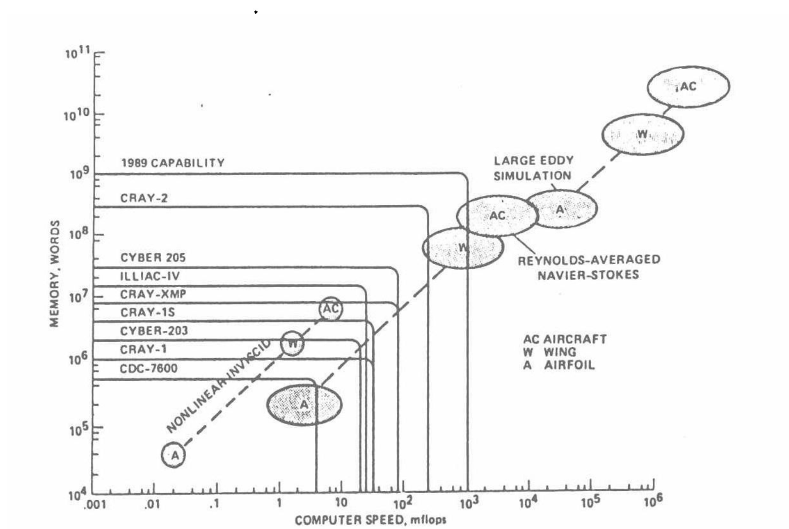


Fig. 1.3. Computer speed and memory requirements for CFD (after Bailey, 1986; reprinted with permission of Japan Society of Computational Fluid Dynamics)

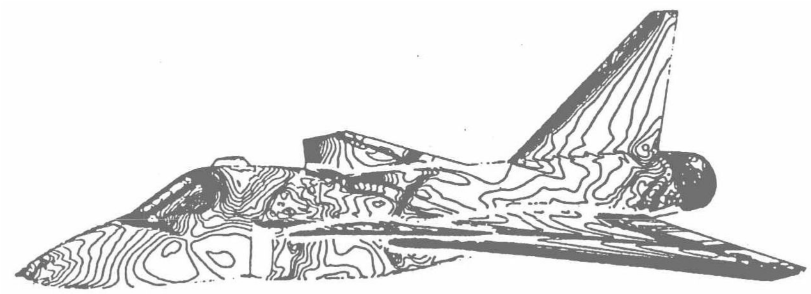


Fig. 1.4. Surface pressure distribution on a typical military aircraft. Surface pressure contours, $\Delta C_p = 0.02$ (after Arlinger, 1986; reprinted with permission of Japan Society of Computational Fluid Dynamics)

supercomputer appears to be the accepted norm (Bailey 1986) for routine design work, running times of a few hours on a microcomputer may well be acceptable in the research and development area. This has the advantage of allowing the CFD research worker adequate time to interpret the results and to prepare additional cases.

The future trends for computer speed and memory capacity are encouraging. Predictions by Simon (1989) indicate that by 2000 one may expect sustained computer speeds up to 10^6 Megaflops and main memory capacities of 50000 Megawords. This is expected to be adequate (Fig. 1.3) for predictions of steady viscous (turbulent) compressible flow around complete aircraft and to allow global design optimisation to be considered seriously.

1.2 Typical Practical Problems

Computational fluid dynamics, particularly in engineering, is still at the stage of development where "problems involving complex geometries can be treated with simple physics and those involving simple geometry can be treated with complex physics" (Bailey 1986). What is changing is the accepted norm for simplicity and complexity. Representative examples are provided below.

1.2.1 Complex Geometry, Simple Physics

The surface pressure distribution on a typical supersonic military aircraft is shown in Fig. 1.4. The freestream Mach number is 1.8 and the angle of attack is 8°. The aircraft consists of a fuselage, canopy, engine inlets, fin, main delta wing and forward (canard) wings. In addition control surfaces at the trailing edge of the delta wing are deflected upwards 10°. Approximately 19000 grid points are required in each cross-section plane at each downstream location. The complexity of the geometry places a considerable demand on the grid generating procedure. Arlinger (1986) uses an algebraic grid generation technique based on transfinite interpolation (Sect. 13.3.4).

The flow is assumed inviscid and everywhere supersonic so that an explicit marching scheme in the freestream direction can be employed. This is equivalent to the procedure described in Sect. 14.2.4. The explicit marching scheme is particularly efficient with the complete flowfield requiring 15 minutes on a CRAY-1. The finite volume method (Sect. 5.2) is used to discretise the governing equations. Arlinger stresses that the key element in obtaining the results efficiently is the versatile grid generation technique.

1.2.2 Simpler Geometry, More Complex Physics

The limiting particle paths on the upper surface of a three-dimensional wing for increasing freestream Mach number, M_∞, are shown in Fig. 1.5. The limiting particle paths correspond to the surface oil-flow patterns that would be obtained experimentally. The results shown in Fig. 1.5 come from computations (Holst et al. 1986) of the transonic viscous flow past a wing at 2° angle of attack, with an aspect ratio of 3 and a chord Reynolds number of 8×10^6.

For these conditions a shock wave forms above the wing and interacts with the upper surface boundary layer causing massive separation. The region of separation changes and grows as M_∞ increases. The influence of the flow past the wingtip makes the separation pattern very three-dimensional. The terminology, spiral node, etc., indicated in Fig. 1.5 is appropriate to the classification of three-dimensional separation (Tobak and Peake 1982).

The solutions require a three-dimensional grid of approximately 170 000 points separated into four partially overlapping zones. The two zones immediately above and below the wing have a fine grid in the normal direction to accurately predict the severe velocity gradients that occur. In these two zones the thin layer Navier-Stokes equations (Sect. 18.1.3) are solved. These equations include viscous terms only associated with the normal direction. They are an example of reduced Navier-Stokes equations (Chap. 16). In the two zones away from the wing the flow is assumed inviscid and governed by the Euler equations (Sect. 11.6.1).

The grid point solutions in all zones are solved by marching a pseudo-transient form (Sect. 6.4) of the governing equations in time until the solution no longer changes. To do this an implicit procedure is used similar to that described in Sect. 14.2.8. The zones are connected by locally interpolating the overlap region, typically two cells. Holst indicates that stable solutions are obtained even though severe gradients cross zonal boundaries.

By including viscous effects the current problem incorporates significantly more complicated flow behaviour, and requires a more sophisticated computational algorithm, than the problem considered in Sect. 1.2.1. However, the shape of the computational domain is considerably simpler. In addition the computational grid is generated on a zonal basis which provides better control over the grid point locations.

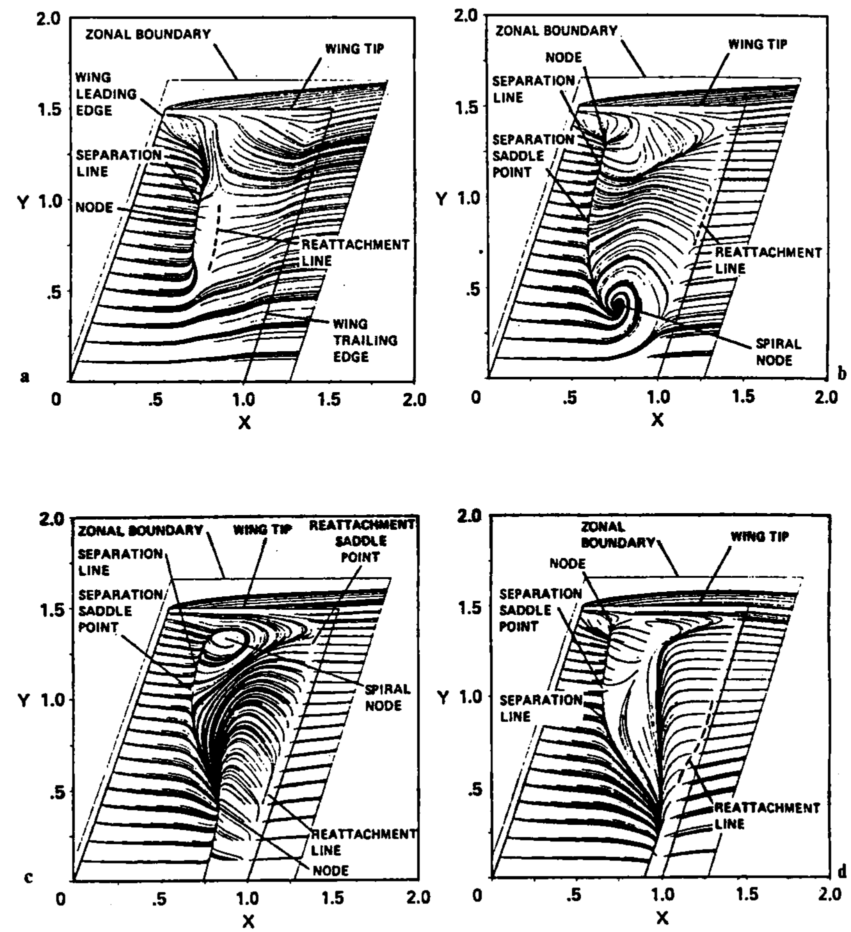
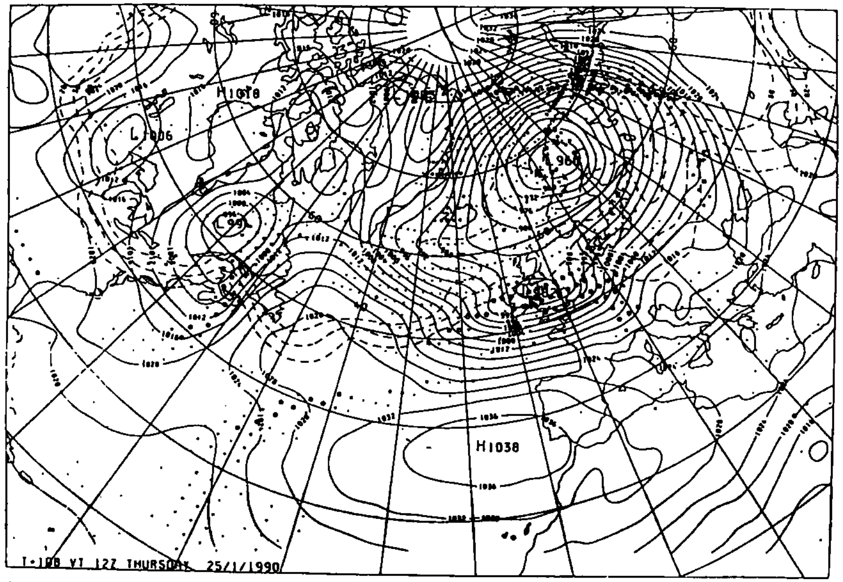


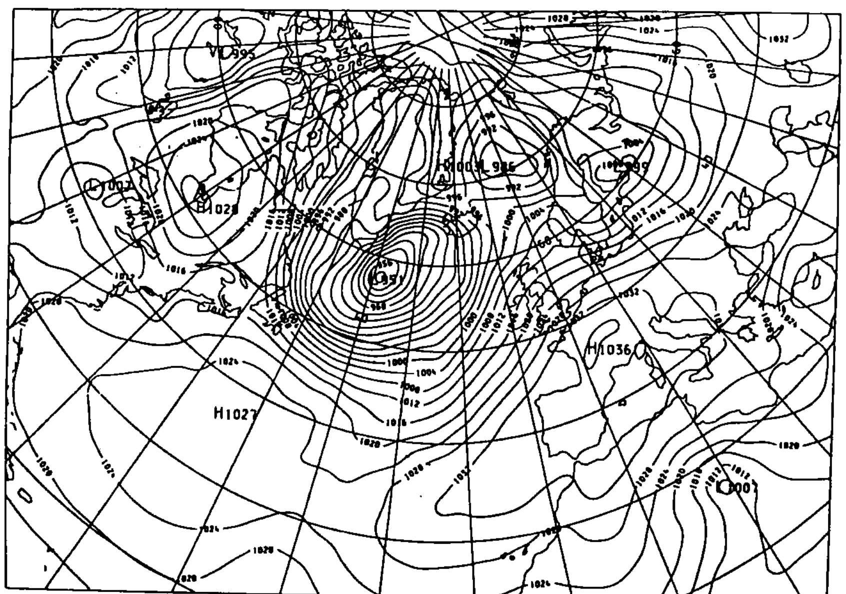
Fig. 1.5a–d. Particle paths for upper wing surface flow. (**a**) $M_\infty = 0.80$ (**b**) $M_\infty = 0.85$ (**c**) $M_\infty = 0.90$ (**d**) $M_\infty = 0.95$ (after Holst et al., 1986; reprinted with permission of Japan Society of Computational Fluid Dynamics)

1.2.3 Simple Geometry, Complex Physics

To illustrate this category a meteorological example is used instead of an engineering example. Figure 1.6 shows a four-day forecast (b) of the surface pressure compared with measurements (a). This particular weather pattern was associated with a severe storm on January 29, 1990 which caused substantial property damage in the southern part of England. The computations predict the developing weather pattern quite closely.



(a)



(b)

Fig. 1.6a, b. Surface pressure comparison. (a) Measurements; (b) Predictions (after Cullen, 1990; reprinted with permission of the Meteorological Office, U.K.)

The governing equations (Cullen 1983) are essentially inviscid but account for wind, temperature, pressure, humidity, surface stresses over land and sea, heating effect, precipitation and other effects (Haltiner and Williams 1980). The equations are typically written in spherical polar coordinates parallel to the earth's surface and in a normalised pressure coordinate perpendicular to the earth's surface. Consequently difficulties associated with an irregular computational boundary and grid generation are minimal.

Cullen (1990) indicates that the results shown in Fig. 1.6 were obtained on a $192 \times 120 \times 15$ grid and used a split explicit finite difference scheme to advance the solution in time. This permits the complete grid to be retained in main memory. 432 time steps are used for a $4\frac{1}{2}$ day forecast and require 20 minutes processing time on a CYBER 205.

Cullen (1983) reports that the major problem in extending accurate large-scale predictions beyond 3 to 4 days is obtaining initial data of sufficient quality. For more refined local predictions further difficulties arise in preventing boundary disturbances from contaminating the interior solution and in accurately representing the severe local gradients associated with fronts.

For global circulation modelling and particularly for long-term predictions the spectral method (Sect. 5.6) is well suited to spherical polar geometry. Spectral methods are generally more economical than finite difference or finite element methods for comparable accuracy, at least for global predictions. The application of spectral methods to weather forecasting is discussed briefly by Fletcher (1984) and in greater detail by Bourke et al. (1977). Chervin (1989) provides a recent indication of the capability of CFD for climate modelling.

The above examples are indicative of the current status of CFD. For the future Bailey (1986) states that "more powerful computers with more memory capacity are required to solve problems involving both complex geometries and complex physics". The growth in human expectations will probably keep this statement current for a long time to come.

1.3 Equation Structure

A connecting feature of the categories of fluid dynamics considered in this book is that the fluid can be interpreted as a continuous medium. As a result the behaviour of the fluid can be described in terms of the velocity and thermodynamic properties as continuous functions of time and space.

Application of the principles of conservation of mass, momentum and energy produces systems of partial differential equations (Vol. 2, Chap. 11) for the velocity and thermodynamic variables as functions of time and position. With boundary and initial conditions appropriate to the given flow and type of partial differential equation the mathematical description of the problem is established.

Many flow problems involve the developing interaction between convection and diffusion. A simple example is indicated in Fig. 1.7, which shows the temperature distribution of fluid in a pipe at different times. It is assumed that the fluid

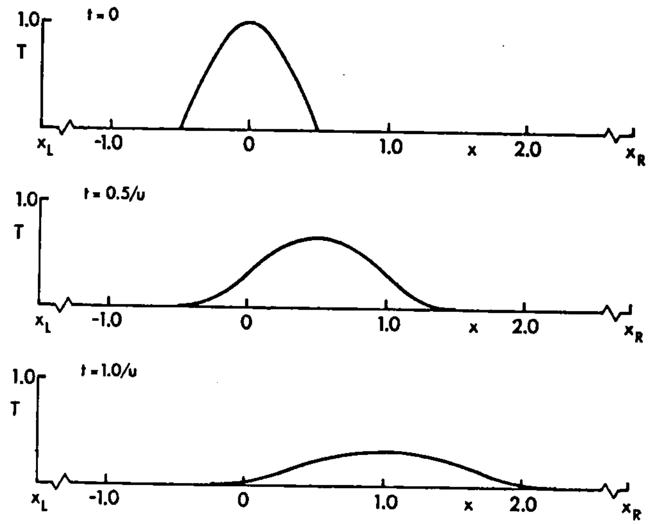


Fig. 1.7. One-dimensional temperature distribution

is moving to the right with constant velocity u and that the temperature is constant across the pipe.

The temperature as a function of x and t is governed by the equation

$$\frac{\partial T}{\partial t}+u\frac{\partial T}{\partial x}-\alpha\frac{\partial^2 T}{\partial x^2}=0 \quad \text{for } x_L \leqq x \leqq x_R \quad \text{and} \quad t>0 \ . \tag{1.2}$$

With a suitable nondimensionalisation, appropriate boundary and initial conditions are

$$T(x_L, t)=T(x_R, t)=0 \quad \text{and} \tag{1.3}$$

$$\begin{aligned} T(x, 0) &= \cos \pi x, \quad -0.5 \leqq x \leqq 0.5 \\ &= 0 \ , \qquad x<-0.5 \quad \text{and} \quad x>0.5 \ . \end{aligned} \tag{1.4}$$

Equations (1.2–4) provide a mathematical description of the problem. The term $\alpha\partial^2 T/\partial x^2$ is the diffusion term and α is the thermal diffusivity. This term is responsible for the spread of the nonzero temperature both to the right and to the left; if α is small the spread is small. Computational techniques for dealing with equations containing such terms are dealt with in Chaps. 7 and 8.

The term $u\partial T/\partial x$ is the convection term and is responsible for the temperature distribution being swept bodily to the right with the known velocity u. The treatment of this term and the complete transport equation (1.2) are considered in Chap. 9. In more than one dimension convective and diffusive terms appear associated with each direction (Sect. 9.5).

Since u is known, (1.2) is linear in T. However, when solving for the velocity field it is necessary to consider equations with nonlinear convective terms. A prototype

for such a nonlinearity is given by Burgers' equation (Sect. 10.1)

$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}-\alpha\frac{\partial^2 u}{\partial x^2}=0 \ . \tag{1.5}$$

The nonlinear convective term, $u\partial u/\partial x$, permits very steep gradients in u to develop if α is very small. Steep gradients require finer grids and the presence of the nonlinearity often necessitates an additional level of iteration in the computational algorithm.

Some flow and heat transfer problems are governed by Laplace's equation,

$$\frac{\partial^2 \phi}{\partial x^2}+\frac{\partial^2 \phi}{\partial y^2}=0 \ . \tag{1.6}$$

This is the case for a flow which is inviscid, incompressible and irrotational. In that case ϕ is the velocity potential (Sect. 11.3). Laplace's equation is typical of the type of equation that governs equilibrium or steady problems (Chap. 6). Laplace's equation also has the special property of possessing simple exact solutions which can be added together (superposed) since it is linear. These properties are exploited in the techniques described in Sect. 14.1.

For many flow problems more than one dependent variable will be involved and it is necessary to consider systems of equations. Thus one-dimensional unsteady inviscid compressible flow is governed by (Sect. 10.2)

$$\frac{\partial \varrho}{\partial t}+\frac{\partial(\varrho u)}{\partial x}=0 \ , \tag{1.7a}$$

$$\frac{\partial(\varrho u)}{\partial t}+\frac{\partial}{\partial x}(\varrho u^2+p)=0 \ , \tag{1.7b}$$

$$\frac{\partial E}{\partial t}+\frac{\partial}{\partial x}[u(p+E)]=0 \ , \tag{1.7c}$$

where p is the pressure and E is the total energy per unit volume given by

$$E=\frac{p}{\gamma-1}+0.5\,\varrho u^2 \ , \tag{1.8}$$

and γ is the ratio of specific heats. Although equations (1.7) are nonlinear the structure is similar to (1.5) without the diffusive terms. The broad strategy of the computational techniques developed for scalar equations will also be applicable to systems of equations.

For flow problems where the average properties of the turbulence need to be included the conceptual equation structure could be written as follows

$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}-\frac{\partial}{\partial x}\left(\alpha\frac{\partial u}{\partial x}\right)=S \ , \tag{1.9}$$

where "α" is now a function of the dependent variable u, and S is a source term containing additional turbulent contributions. However, it should be made clear (Sects. 11.4.2 and 11.5.2) that turbulent flows are at least two-dimensional and often three-dimensional and that a system of equations is required to describe the flow.

1.4 Overview of Computational Fluid Dynamics

The total process of determining practical information about problems involving fluid motion can be represented schematically as in Fig. 1.8.

The governing equations (Chap. 11) for flows of practical interest are usually so complicated that an exact solution is unavailable and it is necessary to seek a computational solution. Computational techniques replace the governing partial differential equations with systems of algebraic equations, so that a computer can be used to obtain the solution. This book will be concerned with the computational techniques for obtaining and solving the systems of algebraic equations.

For local methods, like the finite difference, finite element and finite volume methods, the algebraic equations link together values of the dependent variables at adjacent grid points. For this situation it is understood that a grid of discrete points is distributed throughout the computational domain, in time and space. Consequently one refers to the process of converting the continuous governing equations

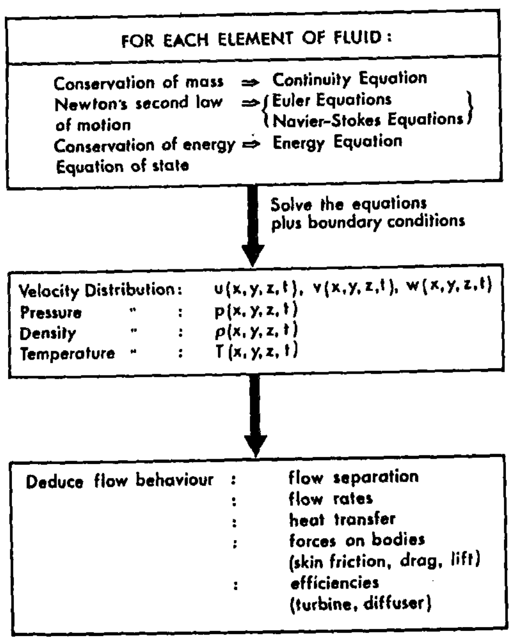


Fig. 1.8. Overview of computational fluid dynamics

to a system of algebraic equations as discretisation (Chap. 3). For a global method, like the spectral method, the dependent variables are replaced with amplitudes associated with different frequencies, typically.

The algebraic equations produced by discretisation could arise as follows. A typical finite difference representation of (1.2) would be

$$\frac{T_j^{n+1}-T_j^n}{\Delta t}+\frac{u(T_{j+1}^n-T_{j-1}^n)}{2\Delta x}=\frac{\alpha(T_{j-1}^n-2T_j^n+T_{j+1}^n)}{\Delta x^2}\;, \tag{1.10}$$

where $x=j\,\Delta x$ and $t=n\,\Delta t$.

If the solution is known at all grid points x_j at time level n, (1.10) can be used to provide an algorithm for T_j^{n+1}, i.e.

$$T_j^{n+1}=T_j^n-\left(\frac{u\Delta t}{2\Delta x}\right)(T_{j+1}^n-T_{j-1}^n)+\left(\frac{\alpha\Delta t}{\Delta x^2}\right)(T_{j-1}^n-2T_j^n+T_{j+1}^n)\;. \tag{1.11}$$

Repeated use of (1.11) generates the solution at all interior grid points, x_j, at time level $n+1$. Incrementing n and substituting the values T^{n+1} into the right-hand side of (1.11) allows the discrete solution to be marched forward in time.

For a local method, e.g. the finite difference method, the required number of grid points for an accurate solution typically depends on the dimensionality, the geometric complexity and severity of the gradients of the dependent variables. For the flow about a complete aircraft a grid of ten million points might be required. At each grid point each dependent variable and certain auxiliary variables must be stored. For turbulent compressible three-dimensional flow this may require anywhere between five and thirty dependent variables per grid point. For efficient computation all of these variables must be stored in main memory.

Since the governing equations for most classes of fluid dynamics are nonlinear the computational solution usually proceeds iteratively. That is, the solution for each dependent variable at each grid point is sequentially corrected using the discretised equations. The iterative process is often equivalent to advancing the solution over a small time step (Chap. 6). The number of iterations or time steps might vary from a few hundred to several thousand.

The discretisation process introduces an error that can be reduced, in principle, by refining the grid as long as the discrete equations, e.g. (1.10), are faithful representations of the governing equations (Sect. 4.2). If the numerical algorithm that performs the iteration or advances in time is also stable (Sect. 4.3), then the computational solution can be made arbitrarily close to the true solution of the governing equations, by refining the grid, if sufficient computer resources are available.

Although the solution is often sought in terms of discrete nodal values some methods, e.g., the finite element and spectral methods, do explicitly introduce a continuous representation for the computational solution. Where the underlying physical problem is smooth such methods often provide greater accuracy per unknown in the discretised equations. Such methods are discussed briefly in Chap. 5.

1.5 Further Reading

The purpose of the present text is to provide an introduction to the computational techniques that are appropriate for solving flow problems. More specific information is available in other books, review articles, journal articles and conference proceedings.

Richtmyer and Morton (1967) construct a general theoretical framework for analysing computational techniques relevant to fluid dynamics and discuss specific finite difference techniques for inviscid compressible flow. Roache (1976) examines viscous separated flow for both incompressible and compressible conditions but concentrates on finite difference techniques. More recently, Peyret and Taylor (1983) have considered computational techniques for the various branches of fluid dynamics with more emphasis on finite difference and spectral methods. Holt (1984) describes very powerful techniques for boundary layer flow and inviscid compressible flow. Book (1981) considers finite difference techniques for both engineering and geophysical fluid dynamics where the diffusive mechanisms are absent or very small.

Thomasset (1981), Baker (1983) and Glowinski (1984) examine computational techniques based on the finite element method and Fletcher (1984) provides techniques for the finite element and spectral methods. Canuto et al. (1987) analyse computational techniques based on spectral methods. Haltiner and Williams (1980) discuss computational techniques for geophysical fluid dynamics.

The review articles by Chapman (1975, 1979, 1981), Green (1982), Krause (1985), Kutler (1985) and Jameson (1989) indicate what engineering CFD is currently capable of and what will be possible in the future. These articles have a strong aeronautical leaning. A more general review is provided by Turkel (1982). Cullen (1983) and Chervin (1989) review the current status of meteorological CFD. Review papers on specific branches of computational fluid dynamics appear in Annual Reviews of Fluid Dynamics, in the lecture series of the von Karman Institute and in the monograph series of Pineridge Press. More advanced computational techniques which exploit vector and parallel computers will not be covered in this book. However Ortega and Voigt (1985) and Gentzsch and Neves (1988) provide a comprehensive survey of this area.

Relevant journal articles appear in AIAA Journal, Journal of Computational Physics, International Journal of Numerical Methods in Fluids, Computer Methods in Applied Mechanics and Engineering, Computers and Fluids, Applied Mathematical Modelling, Communications in Applied Numerical Methods, Theoretical and Computational Fluid Dynamics, Numerical Heat Transfer, Journal of Applied Mechanics and Journal of Fluids Engineering. Important conferences are the International Conference series on Numerical Methods in Fluid Dynamics, International Symposium series on Computational Fluid Dynamics, the AIAA CFD conference series, the GAMM conference series, Finite Elements in Flow Problems conference series, the Numerical Methods in Laminar and Turbulent Flow conference series and many other specialist conferences.

2. Partial Differential Equations

In this chapter, procedures will be developed for classifying partial differential equations as elliptic, parabolic or hyperbolic. The different types of partial differential equations will be examined from both a mathematical and a physical viewpoint to indicate their key features and the flow categories for which they occur.

The governing equations for fluid dynamics (Vol. 2, Chap. 11) are partial differential equations containing first and second derivatives in the spatial coordinates and first derivatives only in time. The time derivatives appear linearly but the spatial derivatives often appear nonlinearly. Also, except for the special case of potential flow, systems of governing equations occur rather than a single equation.

2.1 Background

For linear partial differential equations of second-order in two independent variables a simple classification (Garabedian 1964, p. 57) is possible. Thus for the partial differential equation (PDE)

$$A\frac{\partial^2 u}{\partial x^2}+B\frac{\partial^2 u}{\partial x\,\partial y}+C\frac{\partial^2 u}{\partial y^2}+D\frac{\partial u}{\partial x}+E\frac{\partial u}{\partial y}+Fu+G=0\ , \tag{2.1}$$

where A to G are constant coefficients, three categories of partial differential equation can be distinguished. These are

$$\begin{aligned}&\text{elliptic PDE:} && B^2-4AC<0\ ,\\ &\text{parabolic PDE:} && B^2-4AC=0\ ,\\ &\text{hyperbolic PDE:} && B^2-4AC>0\ .\end{aligned} \tag{2.2}$$

It is apparent that the classification depends only on the highest-order derivatives in each independent variable.

For two-dimensional steady compressible potential flow about a slender body the governing equation, similar to (11.109), is

$$(1-M_\infty^2)\frac{\partial^2 \phi}{\partial x^2}+\frac{\partial^2 \phi}{\partial y^2}=0\ . \tag{2.3}$$

Applying the criteria (2.2) indicates that (2.3) is elliptic for subsonic flow ($M_\infty < 1$) and hyperbolic for supersonic flow ($M_\infty > 1$).

If the coefficients, A to G in (2.1), are functions of x, y, u, $\partial u/\partial x$ or $\partial u/\partial y$, (2.2) can still be used if A, B and C are given a local interpretation. This implies that the classification of the governing equations can change in different parts of the computational domain.

The governing equation for steady, compressible, potential flow, (11.103), can be written in two-dimensional natural coordinates as

$$(1-M^2)\frac{\partial^2\phi}{\partial s^2}+\frac{\partial^2\phi}{\partial n^2}=0 \ , \tag{2.4}$$

where s and n are parallel and perpendicular to the local streamline direction, and M is the local Mach number. Applying conditions (2.2) on a local basis indicates that (2.4) is elliptic, parabolic or hyperbolic as $M<1$, $M=1$ or $M>1$. A typical distribution of local Mach number, M, for the flow about an aerofoil or turbine blade, is shown in Fig. 11.15. The feature that the governing equation can change its type in different parts of the computational domain is one of the major complicating factors in computing transonic flow (Sect. 14.3).

The introduction of simpler flow categories (Sect. 11.2.6) may introduce a change in the equation type. The governing equations for two-dimensional steady, incompressible viscous flow, (11.82–84) without the $\partial u/\partial t$ and $\partial v/\partial t$ terms, are elliptic. However, introduction of the boundary layer approximation produces a parabolic system of PDEs, that is (11.60 and 61).

For equations that can be cast in the form of (2.1) the classification of the PDE can be determined by inspection, using (2.2). When this is not possible, e.g. systems of PDEs, it is usually necessary to examine the characteristics (Sect. 2.1.3) to determine the correct classification.

The different categories of PDEs can be associated, broadly, with different types of flow problems. Generally time-dependent problems lead to either parabolic or hyperbolic PDEs. Parabolic PDEs govern flows containing dissipative mechanisms, e.g. significant viscous stresses or thermal conduction. In this case the solution will be smooth and gradients will reduce for increasing time if the boundary conditions are not time-dependent. If there are no dissipative mechanisms present, the solution will remain of constant amplitude if the PDE is linear and may even grow if the PDE is nonlinear. This solution is typical of flows governed by hyperbolic PDEs. Elliptic PDEs usually govern steady-state or equilibrium problems. However, some steady-state flows lead to parabolic PDEs (steady boundary layer flow) and to hyperbolic PDEs (steady inviscid supersonic flow).

2.1.1 Nature of a Well-Posed Problem

Before proceeding further with the formal classification of partial differential equations it is worthwhile embedding the problem formulation and algorithm construction in the framework of a well-posed problem. The governing equations

and auxiliary (initial and boundary) conditions are well-posed mathematically if the following three conditions are met:

i) the solution exists,
ii) the solution is unique,
iii) the solution depends continuously on the auxiliary data.

The question of existence does not usually create any difficulty. An exception occurs in introducing exact solutions of Laplace's equation (Sect. 11.3) where the solution may not exist at isolated points. Thus it does not exist at the location of the source, $\boldsymbol{r}=\boldsymbol{r}_s$ in (11.53). In practice this problem is often avoided by placing the source outside the computational domain, e.g. inside the body in Fig. 11.7.

The usual cause of non-uniqueness is a failure to properly match the auxiliary conditions to the type of governing PDE. For the potential equation governing inviscid, irrotational flows, and for the boundary layer equations, the appropriate initial and boundary conditions are well established. For the Navier-Stokes equations the proper boundary conditions at a solid surface are well known but there is some flexibility in making the correct choice for farfield boundary conditions. In general an underprescription of boundary conditions leads to non-uniqueness and an overprescription to unphysical solutions adjacent to the boundary in question.

There are some flow problems for which multiple solutions may be expected on physical grounds. These problems would fail the above criteria of mathematical well-posedness. This situation often arises for flows undergoing transition from laminar to turbulent motion. However, the broad understanding of fluid dynamics will usually identify such classes of flows for which the computation may be complicated by concern about the well-posedness of the mathematical formulation.

The third criterion above requires that a small change in the initial or boundary conditions should cause only a small change in the solution. The auxiliary conditions are often introduced approximately in a typical computational algorithm. Consequently if the third condition is not met the errors in the auxiliary data will propagate into the interior causing the solution to grow rapidly, particularly for hyperbolic PDEs.

The above criteria are usually attributed to Hadamard (Garabedian 1964, p. 109). In addition we could take a simple parallel and require that for a well-posed computation:

i) the computational solution exists,
ii) the computational solution is unique,
iii) the computational solution depends continuously on the approximate auxiliary data.

The process of obtaining the computational solution can be represented schematically as in Fig. 2.1. Here the specified data are the approximate implementation of the initial and boundary conditions. If boundary conditions are placed on derivatives of u an error will be introduced in approximating the boundary conditions. The computational algorithm is typically constructed from