J. Fluid Mech. (1986), *vol.* 173, *pp.* 15–41
Printed in Great Britain

The numerical experiment in fluid mechanics

By HASSAN AREF
University of California, San Diego, La Jolla, CA 92093, USA

(Received 24 March 1986)

Several aspects of the use of digital computers to generate solutions of equations of interest to fluid mechanics are discussed. The inter-disciplinary nature of the field of computational fluid dynamics (CFD) is emphasized: the dependence on strides in computer technology, the impact of advances in algorithm development, the continuous interaction with laboratory experiment and analytical theory. The particular role of that mode of computer usage usually referred to as the numerical experiment is highlighted. 'Experiments' of this type have played a central role in establishing concepts such as the soliton and the strange attractor as paradigms within fluid mechanics. The ambitious goal of providing digital counterparts to laboratory equipment such as the wind tunnel is considered. The possibility of abandoning the Eulerian representation of flow fields in favour of following swarms of Lagrangian particles on a computer is stressed. Issues arising from and results of using this methodology are reviewed. Computer simulations are contrasted with computer generated animation. The paper concludes with speculations on future developments.

1. Introduction

A hallmark of the 'scientific method' that we all espouse is the interdependence of analytical theory on one hand and laboratory experimentation and observation on the other. Theory at odds with (sufficiently careful) experiment and observation does not survive long regardless of internal, formal elegance. The introduction of experiment, attributable in large measure to Galileo, was a decisive epistemological advance from the pure contemplation of Aristotelian natural philosophy.

There are at least two reasons for bringing up this almost commonplace issue here. One is that many of the papers of G. I. Taylor mirror perfectly this counterpoint of theory and experiment of the scientific method. In an era of increasing specialization, where the common theorist is ill at ease in a laboratory and the common experimenter is often numbed by the formal mathematical jargon of his theorist colleagues, Taylor's papers provide a refreshing reminder that deep insight can frequently be stated in simple terms, and thus lends itself to simple experiments and simple theoretical explanation. In the best case both of these can be carried through by one individual.

A second reason for dwelling on the philosophy of the scientific method is that the duo of analysis and experiment is slowly but surely evolving into a trio of analysis, computation and experiment. The device responsible for this permeating change in the way we do science (as well as many other aspects of our society) is the digital computer. Computation asserts its independent stature as a worthy principal of the triumvirate particularly in that mode of inquiry designated the numerical experiment. As the name indicates this is an investigation that is closely allied with the

experimental probing typical of the laboratory, yet the substance that is probed and the methods used are the ethereal ones of theory.

Computational fluid dynamics, often abbreviated CFD, is a young subject in the sense that its more substantial implementations belong to generations following G. I. Taylor's. It is an ancient subject in the sense that the dream of rapid calculation goes back to the origins of scientific thought. It is a mature subject in the sense that many of the methods still in use originated when calculating machines were very much more primitive than they are today.

The methods of realizing rapid calculators have shifted with the technologies available, and the present prominence of digital calculation throughout much of the exact sciences owes its debt to breakthroughs in electronics, that promise to have as great an impact on society as a whole as, for example, the steam engine and its attendant 'industrial revolution'.

This paper is intended to survey the progress and status of computation as it relates to fluid mechanics. This young field, however, already has an enormous literature. Indeed, quality periodicals catering exclusively to computational methodology in science, such as the *Journal of Computational Physics* and *SIAM Journal of Scientific and Statistical Computing*, have appeared. Scores of review articles and monographs could be cited. To get a quick overview of the variety in methodology and approach within CFD the reader might consult the articles by Emmons (1970), Orszag & Israeli (1974), Patterson (1978), Zabusky (1981), Jameson (1983), Rogallo & Moin (1984), Van Dyke (1984), and Leonard (1985). It follows that I can provide at best a spotty and personal view of the subject.

Furthermore, CFD is an interdisciplinary field, where the next advance on any particular problem is just as likely to come from a numerical analyst or a computer scientist as from a fluid mechanician. CFD shares with laboratory experimentation a dependence on machinery, and since hardly any 'numerical experimenters' build their own equipment nowadays, the decisions and plans of major computer manufacturers influence the field. Most CFD practitioners do construct their own computer codes, but frequently well-defined, general, mathematical tasks are performed by special purpose 'packages', written by individuals that the computational fluid dynamicist may never know. Hence, the availability (commercial or otherwise) of specific quality software for specific machines can be an important factor.

I shall not attempt, nor would it be appropriate, to cover all these issues. In §2 there is a brief survey of rather standard material on the evolution of computer hardware and the corresponding scale of fluid mechanical calculations. In §3 I attempt to differentiate modes of usage of computers in fluid mechanics. I argue that there are hierarchies of computation: function evaluation, steady-state calculation, initial-value calculation, symbol manipulation, etc. My own preferences are for the category of initial-value problems, which seem to come closest to the laboratory experiment counterpart, and so I briefly discuss in §3 examples of discoveries that have been made in this particular mode. The paradigms of integrable behaviour in a system with infinitely many degrees of freedom, associated with solitons, and chaotic behaviour in systems with just a few degrees of freedom, are highlighted. In §4 my topic is the algorithm, the set of rules to be effected in generating a numerical flow solution. This again is a topic that can easily take up several book volumes (as indeed it has in the study by Knuth 1973), and so after some general remarks I proceed with unconcealed bias towards my own research interests in Lagrangian vortex methods. In §5 I comment on the relation between numerical flow simulation, on one hand, and computer generated animation, on the other. Finally, §6 discusses

the outlook for computational studies in fluid mechanics including the excitement generated by new developments in computer architecture, the enhanced availability of computing resources, and the impact computers will have on instruction and self-study.

Will more sophisticated computing equipment generate deeper insight? My assessment is that it will, but so many examples from G. I. Taylor's opus show us that elaborate facilities are not a necessary condition for profound advances.

2. On the evolution of computing machines

The desire to calculate quickly and accurately goes back to the origins of arithmetic. Indeed, the word 'calculate' derives from the Latin *calculare*, meaning a small stone used for reckoning. The post Second World War era is, however, the first to have computing machinery on such a scale that discoveries in science can be (and have been) made that probably would not have happened otherwise. To understand this development a brief reminder of the highlights in the evolution of computing machines may be in order. More thorough discussions can be found in standard references such as the *Encyclopædia Britannica*. Metropolis, Howlett & Rota (1980) contains many enlightening articles on this topic.

Simple mechanical devices such as the abacus and the calculating machine are still in use today although they have been superseded by electronic calculators. Eminent scientists of the seventeenth and eighteenth century, Pascal and Leibniz in particular, contributed to the design of mechanical calculators. Leibniz wrote on this topic. He envisioned calculators being used for the computation of tables in astronomy. The tedious nature of computing by hand is stressed by Leibniz in his 1685 manuscript *Machina arithmetica*, which contains the opinion: 'For it is unworthy of excellent men to lose hours like slaves in the labour of calculation, which could be safely relegated to anyone else if the machine were used.'

The calculators of Pascal and Leibniz and many of their modern electronic counterparts demand active participation of the operator. During the nineteenth century several important developments took place that we now recognize as seminal to the modern computer: the ideas and 'Analytical Engine' models of Charles Babbage (*c.* 1830), the symbolic logic developed by George Boole (1815–1864), and the introduction of punched cards with data and instructions by Joseph-Marie Jacquard and Herman Hollerith.

The Mark I capable of just 5 flops (an acronym for floating point operations per second) was built in 1944 by Aiken and engineers from International Business Machines Corporation. The ENIAC, the first all-purpose, all-electronic digital computer built by Eckert and Mauchly, capable of 5000 flops appeared a few years later. A seminal 1947 study by Goldstine and von Neumann is usually credited with introducing the idea of the stored program, although similar notions had appeared in the work of Zuse and Turing (see the interesting contribution by Knuth & Pardo in Metropolis *et al.* 1980).

One can argue that the main ideas for the modern computer were in place by 1950. The further phenomenal developments in speed and performance are primarily due to breakthroughs in electronics such as the invention of the transistor in 1948 and the introduction of the integrated circuit in the late 1960s. In the 1980s we find ourselves with what is often referred to as the 'fourth generation' of computer hardware consisting of a plethora of versatile microcomputers at the low end of the performance spectrum and a select group of 'supercomputers' at the high end. Peak

performances approach 10^9 flops or 1 Gigaflop. Typical semilog graphs of computer power (measured in flops) versus time (measured in years) with data points that correspond to computing machines introduced by various manufacturers show exponential growth of computational speed with time.

It would seem that a Megaflop machine holds a tremendous amount of computing power. To get a feel for this number one can consider the solution of a partial differential equation of the type encountered in fluid mechanics using a spatial mesh that contains N nodes. At each node we need the value of the field in question, θ say, at least one number for a scalar field but usually more. To compute a local derivative in terms of spatial derivatives we require a fixed number of floating point operations associated with each node: some differencing to get spatial derivatives, some multiplications, normalizations etc. This number scales at least as N. Thus, the number of operations per timestep must be expected to grow at least as quickly as N. In three dimensions $N = n^3$, where n is the one-dimensional resolution. Hence, $n = 100$ is entirely feasible and, indeed, well-resolved computations of smooth, laminar three-dimensional flows are entirely within the realm of possibility.

Additional problems arise, however, when we consider flows that have a hierarchy of scales, in particular turbulent flows. Here, as is well known, the ratio of the largest to the smallest scale that is 'excited' depends strongly on Reynolds number. According to Kolmogorov's 1941 theory (cf. Landau & Lifshiftz 1959, §32) the range of excited scales grows as $(Re/Re_T)^{\frac{3}{4}}$ where Re_T is the Reynolds number corresponding to transition from laminar to turbulent flow. The total number of mesh points, N, grows as the cube of this, so that $N = N_0(Re/Re_T)^{\frac{9}{4}}$, where N_0 is the required mesh size for $Re = Re_T$. Again when chosing the timestep and the total integration time the hierarchy of scales enters: the total integration time to follow a reasonable period of evolution, T, of the flow is set by the large scales. The time step, τ, is set by the rapidly evolving small scales. The number of steps taken in the calculation, T/τ, is given by turbulence scaling as $(Re/Re_T)^{\frac{3}{4}}$. Hence, the overall operation count scales as $(Re/Re_T)^3$ (Orszag 1970).

Estimates of this type show that the accessible range of Reynolds numbers is tied to computer hardware performance levels. The required spatial resolution, given by the number of mesh nodes N, increases with Reynolds number as does the necessary total run time of the simulation. The operation count per timestep increases with N. (This increase is method dependent but must be at least $O(N)$.) Hence, the requirements for a computer to generate solutions to the Navier–Stokes equations increase dramatically with Reynolds number. The 'digital wind tunnel', dream of the aircraft designer, is still in the future as far as fully resolved turbulence calculations with realistic geometries go.

However, turbulence is not everything, and so apart from developing strategies to ameliorate the scaling just described, one can pursue other problems in fluid mechanics for which the powers of present computers are quite adequate but which, nevertheless, still defy analytical methods.

3. Modi operandi

The tables of special functions, that we today take for granted, were not readily available a century or two ago. In principle such tables can be worked out from various expansion and approximation formulae. In practice the computational effort involved can be quite large. This evaluation of specific numbers from specific formulae to a specified accuracy is the most basic mode of numerical work. I shall refer to it

as function evaluation. The format of a problem in this category is that one has some closed form expression to be evaluated as an independent variable takes on a succession of values. The elucidation of a dispersion relation arising from a linearized stability analysis or the evaluation of functions given by integral representations are examples of numerical computations in this mode. Frequently the results of a function evaluation are known in qualitative terms, maybe even with a few asymptotic results, and the main purpose of the work is to refine the numerical accuracy in a quantitative description.

Much of the numerical work that comes under the heading of function evaluation does not have the exploratory character that one would demand of a 'numerical experiment'. However, once we begin solving systems of nonlinear equations, such as those that may arise in calculating the equilibrium of a simple dynamical system, the analytical structure is frequently so rich, and the solutions obtained so varied and often unexpected, that a genuine sense of search and discovery prevails. In this case the programmed computer does indeed take on a role similar to that of the laboratory apparatus.

Let me elaborate on just one example of fluid mechanical significance: the equations of motion for a system of parallel line vortices, established by Helmholtz, are

$$\dot{z}_\alpha^* = \frac{1}{2\pi \mathrm{i}} \sum_{\beta=1}^{N}{}' \frac{\Gamma_\beta}{z_\alpha - z_\beta} \quad (\alpha = 1, \ldots, N). \tag{1}$$

I have used a notation in which each vortex is represented by its point of intersection with an orthogonal, complex z-plane, the circulations of the vortices are denoted by Γs, and the asterisk denotes complex conjugation. (Because of this representation of the vortices by points, they are usually referred to as point vortices.)

The dynamics described by (1) is considered in several references (see Aref 1983, 1985 for review). For now I want to focus on steady states of an assembly of point vortices. This problem arises, for example, in the hydrodynamics of superfluid ^{4}He. It turns out that vortices in this fluid take the form of concentrated line filaments, and quantum mechanics then demands that each vortex has a circulation $\Gamma = h/m$, where h is Planck's constant and m is the mass of a ^{4}He atom (Onsager 1949). Laboratory experiments have succeeded in visualizing states with patterns of several such identical vortices in solid-body rotation (Yarmchuk, Gordon & Packard 1979). In principle this allows the fundamental natural constant h/m to be determined from macroscopic quantities characterizing a hydrodynamic, steady flow situation!

The analytical problem is to solve a system of coupled, nonlinear, algebraic equations of the form

$$\Lambda z_\alpha^* = \sum_{\beta=1}^{N}{}' \frac{1}{z_\alpha - z_\beta} \quad (\alpha = 1, \ldots, N), \tag{2}$$

where Λ is the parameter

$$\Lambda = \frac{2\pi\Omega}{\Gamma}. \tag{3}$$

Here Γ is the common circulation of the identical point vortices and Ω is the angular frequency of rotation of the pattern. Standard numerical methods exist for solving such systems of equations.

This problem occupied William Thomson (Lord Kelvin) when he was attempting a theory of atoms based on vortex dynamics. He referred to the subject (which extended to three dimensions at the hands of P. G. Tait led to the initiation of the theory of knots), as vortex statics. In spite of the name there is much dynamical

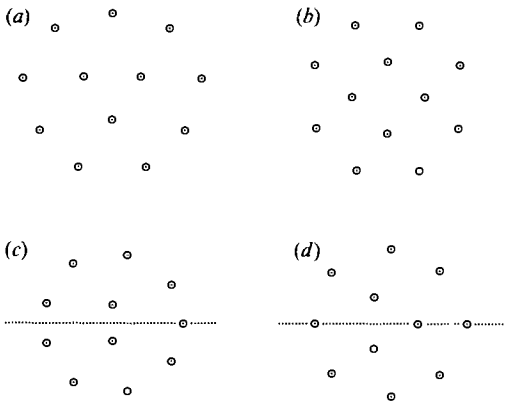

FIGURE 1. Steadily rotating point vortex patterns as found by Campbell & Ziff (1978) through numerical solution of (2). In (*a*) and (*b*) the partitions of 12 vortices into rings have commensurate parts ($3+3+6$ for a, $4+8$ for b), and the rotational symmetry is exact. In (*c*) and (*d*) the partitions have incommensurate parts ($11 = 2+9$ in c, $3+8$ in d) and the symmetry is reduced to a reflection symmetry in the axis shown. (These statements are conjectures based on an examination of the 'Los Alamos Catalog'.)

information to be gained from such states. It is interesting to note that although this problem has been known for many years, the only configurations found and studied analytically in two dimensions have been very simple, symmetrical ones: identical vortices arranged in a regular polygon with or without a central vortex were considered by J. J. Thomson in 1883 and by T. H. Havelock in 1931. A curious special case discovered by Stieltjes (see Calogero 1978) has the vortices all on a line at positions given by the roots of the Nth Hermite polynomial.

A detailed investigation partly analytical and partly numerical of the stability of the 'vortex polygons' was performed by Morikawa & Swenson (1971), but only recently has a full-scale search for steady states been launched. Campbell & Ziff (1978) produced their 'Los Alamos Catalog' of stable patterns, claimed to be complete for $N = 1, \ldots, 30$.

The general impression of these patterns is that the vortices are arranged on concentric circles. This is indeed the case when the number of vortices can be partitioned into a sum of commensurate integers. For example, $12 = 4+8 = 3+3+6$ and stable configurations exist with vortices on either two or three concentric circles (see figure 1*a*, *b*; these configurations are labelled 12_1 and 12_3 by Campbell & Ziff 1978). For 11 vortices, however, one finds states that look like two-circle configurations corresponding to the partitions $11 = 3+8$ and $11 = 2+9$ (see figure 1*c*, *d*). Closer scrutiny reveals that these configurations do not have rotational symmetry at all. In fact, both have an axis of symmetry, shown dotted in figure 1(*c*, *d*).

Two intriguing numerical discoveries have thus been made: one is that a pattern of vortices rotating as a rigid body need not have circular symmetry. The other is a tantalizing hint of a deep connection with the theory of partitions. Many details of the numerical results, however, remain without comprehensive theoretical understanding.

To conclude this brief discussion of a particular kind of steady-state calculation I quote a little known result of Tkachenko (1964) that, hopefully, will whet the readers' appetite: consider the point vortex configuration in figure 2(*a*). It has three

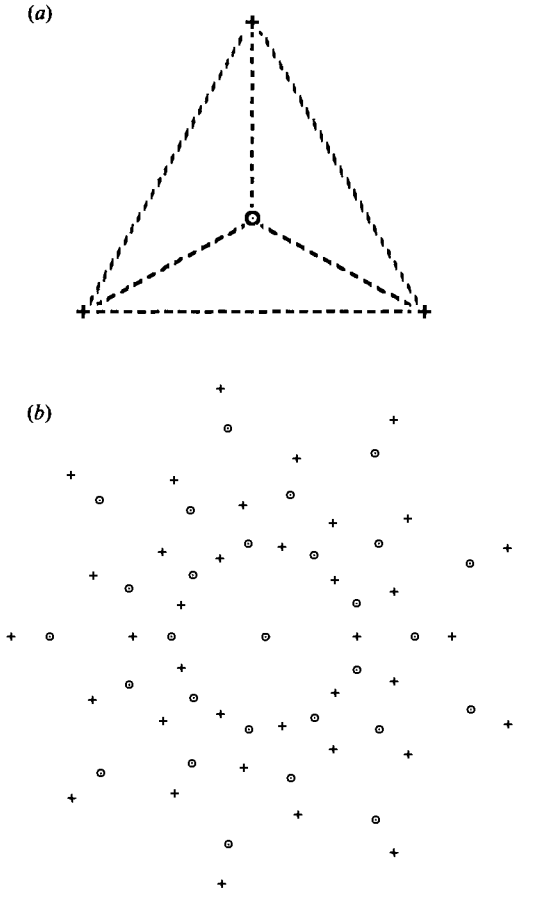

FIGURE 2. Point vortex 'equilibria', i.e. completely stationary configurations of point vortices. In (*a*) a simple case with just four vortices, three of one sign (+), one opposite (o). In (*b*) a more complicated case with 64 vortices (36 of one sign, 28 of the other) found recently by Campbell and Kadtke (1986) using a method suggested by work of Tkachenko (1964).

identical vortices at the vertices of an equilateral triangle with a vortex of opposite strength at the centre. It is easily verified that this configuration is completely stationary, i.e. the velocity of each component vortex vanishes. Tkachenko considers the generalization of this result. Consider a pattern with $N_+ \equiv \frac{1}{2}n(n+1)$ point vortices of circulation $+\Gamma$, $N_- \equiv \frac{1}{2}n(n-1)$ of circulation $-\Gamma$, for integer n. Let the positions of the positive vortices, z_α, $\alpha = 1, \ldots, N_+$, in the complex plane be the roots of a polynomial

$$P(z) = \prod_{\alpha=1}^{N_+} (z - z_\alpha). \tag{4a}$$

Similarly, let the positions of the negative vortices, ζ_β, $\beta = 1, \ldots, N_-$, be the roots of a polynomial

$$Q(z) = \prod_{\beta=1}^{N_-} (z - \zeta_\beta). \tag{4b}$$

Then Q and P satisfy the ordinary differential equation

$$QP'' + PQ'' = 2P'Q', \tag{5}$$

where the primes denote differentiation with respect to z. A derivation of this result (different from that given by Tkachenko 1964) appears in the Appendix. Tkachenko considered the solution analytically for the first four pairs of 'magic numbers' $\{N_+, N_-\} = \{1,0\}$, $\{3,1\}$, $\{6,3\}$ and $\{10,6\}$. A single stationary point vortex is the lowest-order solution in this scheme. The next one is the configuration in figure 2(*a*).

Recently, Campbell & Kadtke (1986) have generalized Tkachenko's results to higher n using computer algebra to solve for the hierarchy of coefficients arising from the polynomial equation (5), and then finding the roots in a conventional way, e.g. using Newton's method. They have also generalized the approach to configurations with more than two species of point vortices. They find one-parameter families of completely stationary configurations. Figure 2(*b*) provides just one example, a configuration with 36 vortices of one circulation, 28 of the opposite circulation.

We come next to the solution of an ordinary differential equation (ODE). One ODE discretized in some fashion will yield a system of coupled algebraic equations. Many of the 'special functions' found in tables satisfy ODEs, but frequently one has other information about them that is better to use for tabulation. For some of the functions of interest to fluid mechanics, however, all we have is an ODE. Typically this equation arises from making a similarity solution *ansatz* in the partial differential equation (PDE) provided by fluid mechanics. Usually these solutions are associated with steady flow, or at least with flows where the time dependence enters via a similarity variable. I shall refer to this mode of computation as steady state calculation. I have in mind the determination of functions such as the Blasius boundary-layer profile and its offspring, the similarity solutions of Falkner and Skan, the similarity flow between rotating disks of von Kármán, etc. We find in much of the early computational work precision calculations of the key functions involved in such solutions: Hartree's (1937) work on the boundary-layer equations, Cochran's (1934) evaluation of the von Kármán similarity solution, and so on.

Numerous computations in this vein have been performed to determine shapes of steadily rotating or translating vortices, steadily propagating bubbles, uniformly propagating waves, etc. Solutions of essentially analytical precision can be obtained, yet the closed form expression in terms of known functions remains beyond our grasp. The steady-state solutions can be supplemented by a linearized stability analysis (again performed numerically) to give some hints regarding dynamical behaviour. A full comprehension of dynamics must, however, await an initial-value calculation.

In terms of numerical methodology there is not much of a step from integrating an ODE to performing an initial-value calculation for a PDE, which when discretized in some fashion (see §4) yields a system of coupled ODEs. And it is in this mode that two seminal discoveries were made, both of which illustrate the numerical experiment at its best: the discovery of solitons in the solutions of certain PDEs (Zabusky & Kruskal 1965), and the discovery of the phase space structure now known as a strange attractor in a small system of coupled ODEs (Lorenz 1963).

I stress that both these discoveries arose from numerical initial-value calculations. This is the mode of computation that most closely parallels the laboratory experiment. The experimenter prepares a set-up that appears to be of interest. Then the flow is set in motion and relevant results and diagnostics are gathered.

The discovery of solitons and a flavour of the interplay between numerical calculations and analytical theory has been reviewed recently by Zabusky (1981). Some sense of the exchanges between numerical experiment and theory in connection with the concept of the strange attractor and other aspects of chaotic behaviour in nonlinear dynamics may be gained from the articles by Ford (1978) and Feigenbaum

(1980). The interactions on this topic between theory and computation on one hand, and laboratory experiment on the other, are still unfolding in the literature.

There is little need for me to reiterate much of this material here, but a couple of comments may be in order: it is worth noting that both discoveries were made with computing equipment that by today's standards is primitive, and that both investigations used numerical techniques that one would label as conventional. Adequate facilities for the graphical display of solutions as they unfolded and a work environment in which yesterday's calculation could influence tomorrow's calculation appear as important ingredients for success. These are logistical matters familiar to any laboratory experimenter, but whereas the laboratory experimenter usually controls his own equipment, the numerical experimenter must often run his experiments on equipment set up and maintained by someone else. The commonly used term 'computer facility' conjures up an image of something that is easy to interact with and use. However, as most practitioners will know, making a computer system easy to use for a heterogeneous user community is a very difficult task, and only after considerable resources have been expended and much experience gained does the result approximate the desires.

The other comment is that although one would expect something quantitative with many-digit accuracy as the logical outcome of a numerical experiment, these two discoveries, that have since become paradigms for much of modern science, are primarily discoveries of qualitative features. I am not suggesting that the calculations were inaccurate, although they undoubtedly were less accurate than some subsequent verifications. But the main lesson learned was on the qualitative behaviour of differential equations.

This is a common feature of numerical experiments in fluid mechanics. The equations have a solution space that is so rich and varied that almost any realization of a solution, be it analog or digital, is bound to turn up something of interest. The immediate challenge, of course, is to discover features that are common to several realizations. This leads naturally to a desire to perform parameter studies, long sequences of essentially identical calculations using slightly altered initial configurations and/or control parameters. As computer resources become more abundant any single computation ceases to be an entity worth reporting. It is the collection of several computations covering some range of parameters that is becoming the entity of interest.

The computer is an almost ideal instrument for this kind of survey. Once a computer code has been written and debugged† it has few useful functions but to be run again and again with slightly altered input data each time. Similar statements can be made, I assume, about a piece of laboratory equipment such as a wind tunnel. The computation has enormous advantages over the laboratory experiment in the precision with which fine-tuning of control parameters can be done, and in the accuracy with which initial conditions can be set. The degree to which a flow field can be monitored in a computation is also unrivalled by laboratory realizations. On the other hand, long-time records of Eulerian flow data in a complicated flow are not easily attainable in a computation due to the limitations mentioned in §2. The strengths and weaknesses of 'analog and digital wind tunnels' complement each other in rather remarkable ways!

† The origin of the term '(computer)bug', which is now widely used to designate situations where computer hardware or software fails to operate according to expectations, seems to be a malfunction in the Mark II vacuum tube computer (*c.* 1945). A moth had been trapped within causing electrical malfunctions.

Another *forte* of numerical experimentation is the ability to simulate fluid flow under conditions that may be difficult to realize in any real laboratory. Two-dimensional hydrodynamics, for example, has been largely an analytical and computational subject, extending from the classical solutions of potential flow theory to the subject of two-dimensional turbulence, although geophysical observations continually provide important inspiration. It has been difficult to realize such flows in the laboratory. Invariably the fluid would discover the third dimension. On the computer, however, this is clearly not a problem. There appear to be important differences between the evolution of a given flow configuration in two dimensions and the evolution of that same flow configuration when it is allowed to yield to three-dimensional instabilities. The theory of isotropic turbulence in two dimensions is a case in point (Batchelor 1969; Kraichnan 1967; Leith 1968). But also mixing layers and wakes appear to evolve differently when computations of two-dimensional hydrodynamics are compared to laboratory flows with three-dimensional features (see Aref 1983 for discussion).

For many years laboratory realization of two-dimensional hydrodynamics was confined to the case of irrotational flows using the ingenious device named after Hele-Shaw (1898). Recently Couder and collaborators (see Couder & Basdevant 1986) have pursued a clever technique using large soap films that appears to allow laboratory visualization of two-dimensional rotational flows. Among other things they have realized flow in the wake of a cylinder, and they have observed in the breakdown of such a wake the formation of pairs of opposite eddies that propagate outward from the wake region. This particular mechanism had been noticed previously in computations (Aref & Siggia 1981), in point vortex model solutions (Aref 1982), and even in essentially isotropic two-dimensional turbulence calculations (McWilliams 1984). Since the phenomenology is so different from what is observed in laboratory flows with three-dimensional small scales, one can consider the pair formation mechanism to be a numerical discovery now confirmed by laboratory experiment.

The ability of a computer code to simulate flow that is difficult to realize in the laboratory has as its unfortunate extension the ability of a computational solution to be altogether unphysical. The computer is in this respect a forgiving piece of equipment. It will willingly solve equations that have no counterpart in the world of physical experience. Verification of a computer code is an extremely important aspect of the process of scientific computation. Analytical solutions of the equations, the more intricate the better, play a very important role here. Precision laboratory experiments also may be essential to verify the calculations in regimes where analytical solutions are not available.

Apart from this cross-referencing so pivotal to the success of the scientific method (with or without computation), numerical experiments usually have their own built-in consistency checks. One varies the grid size, the size of a timestep, the limit on accepting an iteration as having converged, etc. Hopefully, the solution eventually becomes insensitive to such changes of parameters and can be said to have been checked for convergence. This is not, however, the end of the story. The conscientious numerical experimenter must still ascertain that the converged object is indeed the solution sought.

I suggested that the concepts of a soliton and chaotic behaviour in few-degree-of-freedom systems were essential paradigms to emerge from numerical experimentation. As we now know from the analytical characterization of these paradigms a partial differential equation displaying solitons corresponds, generally speaking, to a Hamiltonian system with a denumerable infinity of independent integrals of

the motion. Thus, one would expect integration of a soliton-bearing equation to be reasonably forgiving. The perturbations incurred by truncation errors of a discretization will be bounded from growing arbitrarily by all the integrals of motion. Hence, even the early calculations performed on soliton-bearing systems showed up an unusual phenomenon with great clarity.

For systems with chaotic dynamics the situation is at first glance completely different. In this case one can give simple examples where arbitrarily small changes in initial conditions amplify tremendously in a finite number of integration steps. All the aspects of this behaviour that are so appealing when using chaos as a paradigm for the transition to turbulence in fluid flows can be turned around to become vicious criticisms of any computation that claims to follow a system with chaotic dynamics. On a finite computer any initial condition is given to finite precision. Hence, it corresponds in reality to a small interval of initial conditions. In a system where small uncertainties in initial state amplify explosively, a small interval can quickly evolve into an extremely complicated structure. Yet, unless special programming considerations are taken, the computer will continue representing this complicated, evolved interval as a single point. This interaction between intrinsic stochasticity (chaos) of a system of differential equations and the representation of it on a digital computer raises several vexing questions.

The general paradox of chaos, viz. that stochastic behaviour can arise in a system that evolves according to completely deterministic laws, has a further twist when the computer simulation of such systems is considered. We must first appreciate that the computer, the ultimate deterministic automaton, is capable of producing essentially stochastic output. The computer scientist calls the required software, which is readily available, a random number generator. We thus have a deterministic procedure on a deterministic machine to solve a deterministic problem which leads to answers that in principle and in practice can only be characterized as stochastic. The problem that arises is that the chaos intrinsic to the system being computed quickly renders the computation itself inaccurate.

A complete resolution of these difficulties is not yet available, but as a pragmatic 'fix' one can point out that chaotic regions in phase space are generally open sets, and so one can hope that although a given phase space orbit is not being followed accurately for all time, whatever is being computed is at least representative of the chaotic region. In particular, it is unlikely that an integrable system would display the 'fuzziness' in a Poincaré section expected for a non-integrable system. Alternatively, quantities such as Lyapunov exponents that demand only short forward time integrations can probably be computed reliably. Hence, we are led to conclude – and a substantial body of numerical experiments would tend to verify this – that chaotic and ultimately turbulent behaviour can be diagnosed computationally, although following any particular realization accurately for a long time is a non-trivial task.

From this vantage point the possibility available to those who study equilibrium statistical mechanics of abandoning time integration of differential equations of motion (molecular dynamics) in favour of a direct simulation of statistics via the Monte Carlo method looks even more appealing. Unfortunately, for chaotic systems and for turbulent flows we are limited at present to following initial-value calculations.

Maybe the most pernicious example of this influence of system chaos on computing ability arises when one considers following the motion of passively advected particles. The advection equations for particle motion in a prescribed flow $\boldsymbol{V}(\boldsymbol{x}, t)$, viz.

$$\dot{x} = u(x, y, z, t), \quad \dot{y} = v(x, y, z, t), \quad \dot{z} = w(x, y, z, t), \tag{6a–c}$$

where u, v, w are the components of $\boldsymbol{V}$, are generally non-integrable. This has important consequences when discussing stirring and mixing (see Aref 1984 or Aref & Balachandar 1986 for a lead to the burgeoning literature on chaotic advection and/or Lagrangian turbulence). It (again) has apparently devastating consequences when discussing the accuracy and reliability of numerical methods involving Lagrangian particles. Harlow and collaborators experienced this while pursuing computer codes based on a hybrid Eulerian–Lagrangian technique known as the Marker-and-Cell (MAC) method. In one paper Amsden & Harlow (1964) refer to the striking 'relative orderliness of Eulerian representation over Lagrangian.'

This quick synopsis would be incomplete if I did not touch also on that intriguing mode of computer usage called computer algebra. As anyone who has done extensive analytical calculations will testify there comes a point when the ideas have been digested, and the plan of attack on the problem has been decided, where large amounts of algebra may be called for. The textbooks will use phrases such as 'straightforward but tedious' to cover over the several pages of manipulations that may be necessary in getting from one line to the next. This situation arises frequently in perturbation calculations where several terms of some series are required, and where the procedures for generating such terms involve iteration of a given set of steps. Human intelligence and capacity are such that the rules for doing calculations are generally easier to discover than actually doing all the operations manually and correctly. It follows that such calculations would be admirably suitable for an atomaton if the rules could be programmed. The applications of computer algebra to fluid mechanics have been comprehensively reviewed by Van Dyke (1984).

Whether the desired output is a single number, an array of numbers or an algebraic expression, the key to making an automaton deliver it is an algorithm. This is the subject of the next section.

4. On algorithms

The issue of the relation of man to machine encompasses at one extreme the deep analysis of Turing on the concept of computability (see Hopcroft 1984 for a popular account), and at the other the day to day considerations of almost anyone who uses a computer to derive answers to problems. Central to all these considerations is the concept of an algorithm, the set of rules or procedure by which the problem is to be solved.

At present most of these procedures come from human programmers, and involve in an essential way the accumulated analytical insight into the problem. This may change. For example, one of the more successful chess playing programs (HITECH) incorporates the ability to store an expandable volume of facts about the game including presumably lessons 'learnt' from previous experiences at the board. Maybe one day we shall want to 'teach' a computer fluid mechanics, in the sense that we let it try different approaches to problems, parametrized in some fashion, and assign a figure of merit to each solution.

What we have today in this direction are only embryonic forms such as automatic grid generation schemes for finite-element calculation, or packages that make a choice between different time integration methods based on characteristics of the equations to be solved. For now we perform, at best, numerical experiments with our computers, i.e. the numbers are experimental, the method of calculation, the algorithm, is under external, human control. Convincing numerical results usually do not arise until similar outcomes have been produced by different algorithms. (The requirement of

having to run the program on different computers has waned with the increased reliability of computing equipment.) One day the choice of algorithm and the generation of new algorithms may to a large extent be done by the automaton.

The focus on algorithm is not new to the computer era (cf. Euclid's algorithm). Indeed, we have always held in particular reverence the great algorists, such as Euler, who seemed to have an innate sense for formulating a calculable problem. There are, surprisingly maybe, examples in mathematics where this kind of consideration has little role to play. Existence and uniqueness proofs exist that give hardly a hint of how to find the object known to exist and to be unique. These are essentially useless for computation.

With the relative proliferation of powerful computers the motivation for thinking algorithmically has increased tremendously. Now one has the hope of actually implementing the algorithm in a computer program. Furthermore, with the increased emphasis on computation in the sciences, and novel developments in computer architecture, we now repeatedly ask questions of a new variety: Does this algorithm have a vectorizable implementation? Is the operation count for algorithm A smaller than the operation count for algorithm B? Can an algorithm for this problem be found that can be implemented efficiently on a concurrent processor? And so on.

When confronted with a calculational task originating from an analytical problem there are typically several modes of attack. For example, if the task is to compute the number π to a given accuracy, one could work in physical space and draw inscribed and circumscribed polygons for a given circle in the manner of Archimedes, one could use series expansions, some derived from the theory of Fourier series, or one could use considerations from probability theory and perform a Monte Carlo type calculation to simulate the outcome of Buffon's needle problem (Beckmann 1971).

Similarly in fluid mechanics the solution to the appropriate partial differential equation can be found by real space discretization and use of finite-difference or finite-element techniques. Or one can use an expansion of the unknown field in Fourier series (Gottlieb & Orszag 1977). Sometimes one can use discretizations that derive their legitimacy from very different considerations, such as solving the diffusion equation by using a random walk (see, for example, the recent paper by Ghoniem & Sherman 1985), or the recently proposed methods for solving the Navier–Stokes equation using 'cellular automata' (Frisch, Hasslacher & Pomeau 1986; d'Humiéres, Lallemand & Shimomura 1985; Orszag & Yakhot 1986; Margolus, Toffoli & Vichniac 1986). The variety of ways in which any given problem can be cast in a form suitable for programming on a digital computer is in principle infinite and in practice very large. From this vast collection of possible procedures the computational fluid dynamicist must select a few that meet the requirements of the problem.

Real space discretization is the most intuitive and probably the most common in fluid mechanical calculations. It has as its basis a body of analytical results relating differences between function values at certain grid points to derivatives of that function. For example, using a Taylor series expansion

$$F(x+h)+F(x-h)-2F(x) = F''(x)\,h^2+O(h^4), \tag{7}$$

where $O(h^4)$ signifies terms of order h^4 or higher. Dividing throughout by h^2 we see that the expression $\{F(x+h)+F(x-h)-2F(x)\}/h^2$ gives the second derivative $F''(x)$ up to an error of second order in h.

Similar considerations apply to differencing in time in an unsteady calculation. An amusing example is provided by the equations describing the point mass pendulum:

$$\ddot{\theta} = -\omega^2 \sin\theta, \tag{8}$$

where ω is a constant. Consider discretizing this equation for purposes of time integration. Let θ_n be the values of θ at uniformly spaced instants in time $n\Delta t$, $n = 0, 1, 2, \ldots$. Then to second order in the time interval Δt

$$\theta_{n+1} + \theta_{n-1} - 2\theta_n = -(\omega\Delta t)^2 \sin\theta, \tag{9}$$

according to (7). We can turn this into a two-dimensional mapping by defining a momentum-like variable

$$p = \dot{\theta}/\omega, \tag{10}$$

and considering the scheme

$$\theta_{n+1} = \theta_n + (\omega\Delta t)\, p_n, \tag{11a}$$

$$p_{n+1} = p_n - (\omega\Delta t) \sin\theta_{n+1}. \tag{11b}$$

This scheme defines a mapping of the (θ, p) 'phase plane' onto itself. If it is to be faithful to the integrable dynamical system from which it came, it should display solutions the successive iterates of which fall on a curve

$$\tfrac{1}{2}p^2 - \cos\theta = \text{constant}, \tag{12}$$

corresponding to the conservation of total mechanical energy. The mapping (11), however, is an often treated example from the literature on chaotic dynamics, where it is known as the standard map, and it is well known to display breakdown of KAM tori and attendant symptoms of chaos as the parameter that we have written here as $\omega\Delta t$ is increased (cf. Lichtenberg & Lieberman 1983). Note that this is an example of chaos in an area-preserving mapping. True to its origins as a non-dissipative system the mapping (11) does conserve phase space area in the sense that

$$\partial(\theta_{n+1}, p_{n+1})/\,\partial(\theta_n, p_n) = 1. \tag{13}$$

This connection between chaotic behaviour and numerical methods has been explored further by Yamaguti & Oshiki (1981). Other examples of this type exist where a mapping displaying chaotic behaviour arises as the result of some *prima facie* sensible discretization made for computational purposes. This is 'numerical instability' in one of its more insidious forms!

For convenience and brevity consider Burgers' equation

$$u_t + uu_x = \nu u_{xx} \tag{14}$$

as a prototype of the field equations that we wish to deal with in fluid mechanics. This is a useful test case since we can reduce it to the diffusion equation (which in turn can be solved analytically) using the so-called Cole–Hopf transformation

$$u = -2\nu(\log\phi)_x. \tag{15}$$

Real space discretization using finite differences on a regular mesh will yield ODEs for the field amplitudes at the mesh points with quadratic coupling terms. The same is true for Fourier series methods, although ingenious manipulations and the fast Fourier transform (FFT) algorithm allow substantially reduced operation counts relative to what one would expect by direct inspection of the equations (cf. Orszag 1972; Gottlieb & Orszag 1977). In both cases one incurs a truncation error by restricting to a finite number of mesh points or Fourier modes.

However, Burgers' equation allows also a set of solutions of a particular form for which the PDE can be exactly transcribed to a set of ODEs describing a kind of

particle dynamics (Chuudnovsky & Chuudnovsky 1977). The details are as follows: if

$$u(x,t) = -2\nu \sum_{\alpha=1}^{N} (x - z_\alpha(t))^{-1}, \tag{16}$$

where the complex $z_\alpha(t)$, of which there may be any finite number N, solve the system of ordinary differential equations

$$\dot{z}_\alpha = -2\nu \sum_{\beta=1}^{N} (z_\alpha - z_\beta)^{-1}, \tag{17}$$

then $u(x,t)$ gives a complex-valued field solving Burgers' equation. (Real solutions arise when the poles $z_\alpha(t)$ occur as complex conjugate pairs.) This kind of solution is usually referred to as a 'pole decomposition'. Similar results hold for the Korteweg–de Vries equation (KdV), modified KdV, the Benjamin–Ono equation, the Sivashinsky equation and others.

While the possibility of subjecting a PDE to finite difference or spectral discretizations is always available, the existence of an essentially exact reduction to a finite set of ODEs is a relatively rarer occurrence, and one that virtually begs for exploitation in a numerical method. The most important example of this situation may be the two-dimensional Euler equation

$$(\Delta\psi)_t + \psi_x(\Delta\psi)_y - \psi_y(\Delta\psi)_x = 0, \tag{18}$$

here written for the stream function ψ, where one has a 'decomposition' in terms of an assembly of point vortices: if we consider the possibility

$$\psi(x,y,t) = \sum_{\alpha=1}^{N} \Gamma_\alpha G[x,y;x_\alpha(t),y_\alpha(t)], \tag{19}$$

where $G[\boldsymbol{x};\boldsymbol{x}']$ is a generalized Green function, discussed in detail by Lin (1943), and the Γs are again point vortex circulations, then the ansatz (19) solves Euler's equation (in a 'weak' sense, since the vortices themselves are singularities) if the vortex coordinates x_α, y_α evolve according to the equations

$$\Gamma_\alpha \dot{x}_\alpha = -\frac{\partial H}{\partial y_\alpha}, \quad \Gamma_\alpha \dot{y}_\alpha = \frac{\partial H}{\partial x_\alpha}, \tag{20}$$

where

$$H = \sum_{1 \leqslant \alpha < \beta \leqslant N} \Gamma_\alpha \Gamma_\beta G[x_\alpha, y_\alpha; x_\beta, y_\beta] + \frac{1}{2} \sum_{\alpha=1}^{N} \Gamma_\alpha^2 g[x_\alpha, y_\alpha; x_\alpha, y_\alpha], \tag{21}$$

with G the Green function from before and

$$g[\boldsymbol{x};\boldsymbol{x}'] = G[\boldsymbol{x};\boldsymbol{x}'] - (2\pi)^{-1} \log|\boldsymbol{x}-\boldsymbol{x}'|. \tag{22}$$

On the infinite plane $G[\boldsymbol{x};\boldsymbol{x}']$ is just $(2\pi)^{-1}\log|\boldsymbol{x}-\boldsymbol{x}']$, $g = 0$ and H is

$$H = (2\pi)^{-1} \sum_{1 \leqslant \alpha < \beta \leqslant N} \Gamma_\alpha \Gamma_\beta \log|\boldsymbol{x}_\alpha - \boldsymbol{x}_\beta|. \tag{23}$$

Representing the vortex positions by complex variables, z_α, then gives (1), which for identical vortices are in fact surprisingly similar to (17) above.

Point vortices are not as pleasant as the poles of Burgers' equation since they represent singularities within the flow field itself. However, the observations just summarized besides being of independent theoretical interest are useful for computation.

The point vortex 'decomposition' of Euler's equation in two dimensions has been made the basis of a family of numerical procedures usually referred to as vortex

methods. The first numerical experiment using such a method was performed by Rosenhead (1931) who reported hand calculations in which an assembly of point vortices was advanced in time in an attempt to elucidate the mechanics of vortex sheet roll-up. The methodology now has a large and growing literature (see the reviews by Aref 1983 and Leonard 1980, 1985 where many further references may be found). Extensions to three dimensions have met with some success, although the appropriate three-dimensional analog of the two-dimensional point vortex is still a controversial topic (Novikov 1983; Saffman & Meiron 1986).

Charney (1963), in a paper entitled 'Numerical experiments in atmospheric hydrodynamics,' begins:

> Numerical methods are being applied to an increasing variety of problems in atmospheric hydrodynamics .... If the object is to make the best possible forecast of a meteorological event, a large number of degrees of freedom in the form of grid values are required to approximate the partial differential equations of the continuous flow by a finite set of algebraic difference equations. If, on the other hand, the object is to arrive at physical understanding, it is desirable to represent the motion with as few degrees of freedom as possible. But merely to decrease the number of grid points is to increase the truncation error .... The use of functional expansions offers better possibilities. Lorenz ... has shown that the representation in terms of finite series of orthogonal functions can preserve the conservation laws ... even though the computed motions are not strictly those of possible physical systems. An alternative approach is to represent the motions by discrete vortex elements. Thus the continuous vorticity distribution in two-dimensional flow may be approximated by a finite set of parallel rectilinear vortex filaments of infinitesimal cross-section and finite strength, whose motion is governed by a set of ordinary differential equations .... This is analogous to replacing a continuous mass distribution by a set of gravitating mass points. It has the virtue that mass, energy, linear and angular momentum continue to be conserved and that the motions represented are those of conceivable, though idealized, physical systems. It is, in a sense, the dual of Lorenz's functional representation, the Green's function being the dual of the eigenfunction, or the 'particle' the dual of the 'wave'. Which representation is the more suitable depends on the nature of the field of motion to be approximated. Fields with wave-like properties are more amenable to functional representation, whereas those with discontinuities or vortex-like properties are more naturally represented by discrete vortices.

The discovery of chaos in the problem of four point vortices by numerical experiments (see Aref 1983, 1985) attests to the validity of Charney's remarks. The further fact that according to (20) phase space and configuration space coincide for vortices, as emphasized by Onsager (1949), promises to be of profound importance.

On the algorithmic side an interesting extension of vortex methods has been made to certain problems in stratified flows. Consider the flow situation sketched in figure 3, where a sharp interface separates two homogeneous but immiscible fluid regions individually in potential flow: that interface can instantaneously be represented as a vortex sheet. At the interface the normal component of velocity is continuous. The tangential component, however, can have a jump. The latter gives the interface a local vortex sheet strength or circulation per unit arclength. Due to stratification, circulation is not conserved by Kelvin's law, but is governed by the circulation theorem of Bjerknes, which, depending on the problem, leads to a subsidiary integral or integro-differential equation. Examples of this situation include the 'fingering' seen in stratified Hele-Shaw flow (also known as the Taylor–Saffman instability), the Rayleigh–Taylor problem and several well-established theoretical models of waves on interfaces between fluid layers of different densities. The idea that such

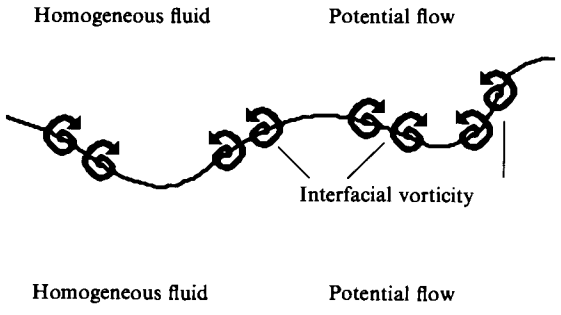

FIGURE 3. The 'generic' sharp interface problem that is amenable to vortex methods. The homogeneous fluid regions are assumed to be in potential flow. The normal component of velocity is continuous at the interface. The tangential component, however, may have a discontinuity. This slippage corresponds to interfacial vorticity.

stratified flow problems are amendable to vortex methods arose well before serious implementations were feasible. Birkhoff discussed the Rayleigh–Taylor case in 1954 (Birkhoff 1954, 1962). A lucid recent statement is given by Baker (1982). De Josselin de Jong (1959, 1960) showed how to cast the Taylor–Saffman problem in this form a few years later. Representative implementations can be found in the work of Baker, Meiron & Orszag (1980), Tryggvason & Aref (1983, 1985), Pullin (1982), and DeGregoria & Schwartz (1986). A host of other applications in this genre exist, including of particular interest to fluid mechanics certain models of wave motion (Longuet-Higgins & Cokelet 1976) and finite area vortices (Deem & Zabusky 1978).

It is interesting to observe the relative ease with which these intrinsically Lagrangian methods have been adapted for computation. The Lagrangian representation of fluid mechanical equations is usually considered to be too complicated, and most analytical studies work with the equations of motion in the Eulerian form. However, once a computer is going to do the calculations, the issue of what is easy and what is complicated changes character. If we can find an algorithm for a given problem in which we track a number of distinguishable particles, then there may be definite advantages to be realized by doing the calculation in this way. For example, in stratified flow problems with sharp interfaces it is possible to concentrate all the degrees of freedom of the calculation on the interface itself, thereby realizing considerable gains in resolution for a given expenditure of resources. Furthermore, the theoretical formulation implies that we have a sharp non-diffusing interface and this feature is guaranteed by the Lagrangian representation. (For this reason the methodology is sometimes referred to as being 'naturally adaptive'.) Methods in which a field with a sharp jump across a moving interface is represented on an Eulerian grid incur a multitude of numerical problems.

The excerpt from Charney's (1963) article mentioned an analogy between mass points and point vortices. In either problem the number of operations necessary to advance one configuration to the next apparently scales as N^2, where N is the number of particles. But this is a crude estimate, and ways can be found to better the operation count and, thus, increase the number of point particles that can be considered. The 'vortex-in-cell' method described by Christiansen (1973) uses a hybrid Lagrangian–Eulerian technique to effectively reduce the operation count from $O(N^2)$ to $O(N \log_2 N)$. This possibility is enormously important since it allows one to probe statistical flow regimes that would otherwise be inaccessible to computation. For example, in the Taylor–Saffman problem of stratified Hele-Shaw flow it has been

possible to simulate violently deforming interfaces with a multitude of interacting structures (waves, 'fingers', and 'bubbles') leading to a flow that is essentially statistical (Tryggvason & Aref 1983). Indeed, simple scaling laws expected to govern such flows can be checked from the computations. In this particular example the work of several investigators (Tryggvason & Aref 1983, 1985; DeGregoria & Schwartz 1986; Bensimon 1986) has provided a collection of consistent methods that allow one to consider the concept of a 'digital Hele-Shaw cell'. This is not an immediate counterpart of the laboratory version since a number of effects present in real experiments (e.g. wetting of the plates, and effects of surface tension due to interface curvature in the transverse direction) are not included in the purely two-dimensional Hele-Shaw equations. Nevertheless, the dialogue between experimental observation and calculations has been extremely useful in elucidating several aspects of the fluid mechanics in this case, including the role of viscosity contrast across the interface and the instability of fingers in the 'single fluid case' for small lateral surface tension. For more detail on this topic see the general reports by Robinson (1985*a*, *b*), the brief review by Aref (1986), the article by Saffman (1986), and the review by Homsy (1987).

The relation between analytical formulation and computational algorithm is an important one. I have concentrated here on topics close to my own interests, but corresponding accounts could have been given using examples from other areas in fluid mechanics, such as turbulent flow or gas dynamics.

5. Simulation versus animation

So far in our discussion we have implicitly assumed that the best computational fit to a laboratory experiment or field observation in fluid mechanics will arise by subjecting the Navier–Stokes equations (or some subset of these equations) to the methods of numerical analysis. This will yield an algorithm and ultimately an executable program on some device, and once debates on the relative merits of individual techniques have died down, a trustworthy set of solutions will remain. We refer to such a set of solutions as a flow simulation.

This definition is considerably more rigorous than one would use in other areas of science. For example, if the objective were to simulate cell division or plant growth on a computer, a program would be written that advanced the process according to some set of model equations, entirely in analogy with what we do in CFD. But in the biological example the model would be rather more flexible. If the output from a given model did not agree with laboratory experiment or field observation, the evolutionary biologist would usually be willing to change the model and try the simulation again. This mode of work in which the model generating the simulation is tuned to make the outcome agree more closely with observation is rather the inverse of what we are used to in CFD. It may be referred to as the hypothetico-deductive method, and its objective is to arrive at a mathematical model that will produce a certain kind of output deemed desirable on the basis of other considerations.

There is computational work in fluid mechanics that has a little of this flavour in the area of rheology, where the constitutive equation for the medium is unknown, and where part of the objective of a numerical experiment is to compare results for different constitutive laws in the hope of finding a best fit for some particular fluid.

But for all cases in fluid mechanics where the governing equations are known, the numerical experiment is aimed at exploring the solution space of those equations. However, if the objective is simply to make a computer produce a flow pattern of some realism, there are clearly several possibilities, some of which are considerably

simpler than solving discretizations of PDEs! It is of some interest to contemplate these, in particular since we have noted that the refinement in accuracy and resolution necessary for turbulent flow simulations pushes the limits of present computer technology.

To distinguish such other methods of flow generation by computer from the simulations (which are based on solving discretizations of the basic differential equations of fluid mechanics), I shall refer to them as computer generated flow animation. I am interested here in exploring briefly the relationships between simulation and animation.

The early literature on animation is to a large extent wound up with that on high-speed photography. Animation in its primitive form arises by passing still images, which differ only slightly one from the next, past an observer. Human persistence of vision does the rest. High-speed photography produces in essence the reverse process: a progression perceived as continuous is artificially broken down into a sequence of stills. The well-known pioneering work of Eadweard Muybridge consisted in the then painstaking effort of producing a sequence of stills of a phenomenon (e.g. a horse in gallop) considered rapid by ordinary human timescales. Advances in high-speed ciné photography now make such sequences routine.

The computer is an almost ideal instrument for the generation of animation. Long sequences of pictures that only differ slightly can be generated with relative ease using a base picture and algorithms effecting the simple geometric changes of translation and rotation. Such sequences, referred to as computer generated animation, play an ever greater role in commercially produced film sequences for entertainment, education and advertisement.

Fluid mechanics has an intriguing relation to this somewhat diffuse area of human enterprise. Some phenomena, such as a splash or the motion of a supersonic projectile, are within the realm of fluid mechanics, yet they are so rapid that direct visual observation is of limited use. The famous images of projectiles with shock waves due to Mach from 1886 (see Reichenbach 1983) and the pictures of splashes produced by Worthington (1908) and later Edgerton are contributions both to fluid mechanics and to high-speed photography.

With the computer power of the present one can take a sequence of such pictures from an experiment, digitize them and, using simple algorithms, interpolate between them. When pasted together the outcome is a computer generated flow animation that can have resolution and realism beyond the most ambitious simulations.

So what has been forfeited? If the objective is to provide flow pictures by computer, is this not the obvious route rather than negotiating the pitfalls of a numerical treatment of PDEs? The answer to such questions is that an image of the flow generated on the computer by this method cannot contain any information not already present in the original, experimental film record. A piece of computer generated animation may obey some simple rules of geometrical scaling, but it can hardly obey the more subtle type of scaling associated, for example, with Reynolds number similarity. The essence of the simulation is that it embodies at a basic level the dynamical laws of mass, momentum and energy on which fluid mechanics is based. The animation does not, as witnessed by the widespread use of animation to create extranatural evolution sequences as in fictional cartoons. On the other hand, animation of the kind just described does have its place. For example, it may be useful in some cases to subject flow images generated both by simulation and animation to the same processing for comparison purposes. It is interesting to observe that realistic animation of a fluid dynamical process based on a few images is difficult to

achieve. Rotation and translation of rigid objects are easily strung together to produce high quality animation. But the essence of fluid motion is change of form, and to capture that correctly one needs, in effect, most of the information embodied in the differential equations of motion.

One can now sharpen the argument by asking about the relevance to fluid mechanics of the output from turbulence models, for example, or by asking what role fractal sets (Mandelbrot 1977), created by models that bear little relation to the equations of fluid mechanics, can possibly have to the evolution of fluid motion. The output from a turbulence model, by which I mean any of the variety obtained by closing the hierarchy of moment equations, can never be a simulation of turbulence in the sense I used the term above: regardless of how fine a spatial resolution is used, the numerical solution will at best approximate the solution of the turbulence model, and the theoretical–analytical problem of showing that the model represents Navier–Stokes turbulence remains even after massive computations have been done. And what are we to make of fractal sets that look exactly like real clouds, to take an example in the realm of fluid flow, yet are generated according to simple iterative rules that bear no obvious relation to the governing equations? (See in this connection also the recent, brief commentary by Kadanoff 1986.)

The constructive answer to such questions is that computational discoveries in the hypothetico-deductive mode are being made. If a high quality calculation using a turbulence model shows up a feature that is also seen in experimental measurement, then this reflects well on the turbulence model, and the calculation has brought out an aspect of the model that one might not have noticed analytically. Similarly, the fact that fractal sets appropriately tuned can sometimes produce images that correlate well with reality is a computational hint at the type of solution we should be looking for analytically. In fact, there is a tenuous bond from the chaos that can be present in the Lagrangian motion of particles advected by fluid flow, that was mentioned in §3, to the omnipresence of fractal sets in chaotic systems, on to the emergence of fractal sets in advection–diffusion problems such as those that must govern the dynamics of clouds.

The tradition in CFD has been to pursue the vast set of discretizations of PDEs to produce ever finer and faster simulations of fluid flow. This is a noble endeavour that should certainly continue. It will remain the most convincing way of producing flow solutions for the forseeable future. But it does lack an inherent quality of originality. The simulation derives its legitimacy from those instances in which it can produce reliable results more quickly or accurately than theory or laboratory experiment. Indeed, simulation results exist that may never be captured by an analytical formula derived from first principles.

However, the computer is capable of so many other things, from solving model problems to manipulating symbols. Algorithmic developments that are not in the vein of discretization followed by simulation – numerical implementations of the ideas of the renormalization group (Wilson 1983) being a prime example – should be kept constantly in mind.

6. Outlook

It is clear that computers will continue to become more powerful. It is also clear that computers are now so powerful that much useful science can be done using them. And while there exist computer codes to do many standard tasks, including solving the Navier–Stokes equations at some level in a variety of circumstances, vast areas

in all branches of science have not been subjected to the computational scrutiny that they deserve.

In this final section I shall comment on three areas where advances are being made and focus on the impact on fluid mechanics. I ask to be excused for basing these comments on the somewhat parochial perspective of activities at my own institution and my own research group, but these are the developments with which I am most familiar.

First of all, there is an initiative underway in the United States to ease access to computer resources, in particular to supercomputer resources on machines such as the CRAY-1, Cyber 205, CRAY-XMP and CRAY-2. Workers at academic institutions have by and large had limited access to such machines, and this has rightly been diagnosed as unproductive. There is a pragmatic effect that one's vision with regard to what constitutes a feasible computation is very much dictated by what computing hardware is at one's disposal. By making high grade computing available to a broad base of academic scientists the hope is that ambitious problems will be undertaken more readily.

After a nationwide call for proposals the National Science Foundation (NSF) awarded high-speed computing centres to five universities in the USA: Illinois, Cornell, Princeton, UCSD and Pittsburgh. The main computer at the San Diego Supercomputer Center (SDSC) is a CRAY-XMP 48 with four processors and 64 Megabytes of memory. Average calculational rates are in the range 50–100 Mflops with peak performance levels at 420 Mflops.

SDSC has been operational for only a few months. Project grants of computer time range from ten CPU hours per year to a few hundred. The initiative to make extremely powerful computers available to a broad segment of the academic community is so new that no assessments of success or failure can yet be made. One can hope, on one hand, that profound discoveries will occur. One can fear, on the other hand, that resources diverted to computing will hurt advances in theoretical analysis, laboratory experiment, and field observation. One can hardly argue with the fact that if any general technological achievement is to influence science in the second half of the twentieth century, the revolution in the power of computers is the prime candidate.

The focus on computers and their use in science has rekindled an interest in the construction of computers for specific scientific tasks reminiscent of that seen in the early years of computing (cf. several articles in Metropolis *et al.* 1980), but potentially of much greater power. Various computationally simple problems, usually with discrete states and a simple algorithm (Monte Carlo calculation of spin systems, for example), have led to the construction of dedicated computers that perform spectacularly on one problem or a small class of problems. Considerable anticipation attaches to the issue of whether the Navier–Stokes equation is such a problem when viewed from the perspective of cellular automata (see the references given in §4).

One idea that seems to have a broad range of applicability is that of concurrency, i.e. the notion that one can speed up a complicated calculation by apportioning it to several processors working on it in parallel. For the past three years or so a group at the California Institute of Technology have been developing algorithms and codes for scientific computation on a uniquely designed computer employing a parallel architecture with up to 64 processors. This machine is modestly called the Cosmic Cube. It is reported to achieve calculational speeds of up to ten times a standard mini-computer, the Digital Equipment Corp. (DEC) VAX 11/780 at a fraction of the cost (Fox & Otto 1984; Seitz 1985).

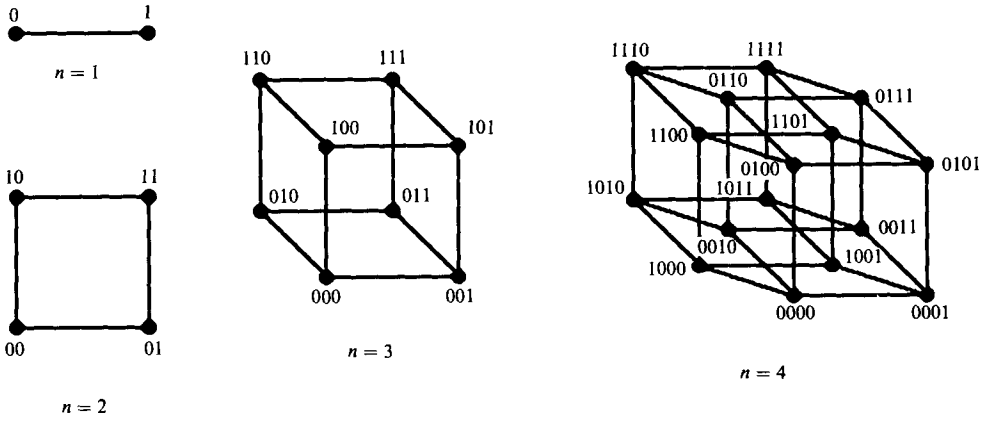

FIGURE 4. Topology of cubes in n dimensions for $n = 1, 2, 3, 4$. The $2^4 = 16$ processors in the AMETEK 'hypercube' machine mentioned in the text are interconnected according to the topology of the four-dimensional cube shown. Note the labelling of processors by binary numbers. An $(n+1)$-cube can be thought of as made up of two n-cubes connected at corresponding vertices.

The 'Cube' is based on an interesting idea. Instead of having one powerful central processing unit (CPU) sequentially accessing shared data and instructions, it has several lesser CPUs (nodes) each with their own storage and with capabilities for message passing. The nodes are physically interconnected so that the entire arrangement with n nodes has the topology of the vertices and edges of a cube in 2^n-dimensional space (cf. figure 4). Hence the name '(hyper)cube' for this type of machine.

The Cosmic Cube architecture is currently being put into commercial production by several vendors. A configuration that has recently been delivered to my laboratory consists of an AMETEK System 14 machine (with just 16 concurrent processors) and a host mini-computer, in this case a MicroVax II from DEC. Both machines are housed in boxes that fit at the sides of a desk. The two machines use standard electrical outlets and require no special ventilation. One can easily extrapolate to the day when the powers of both are contained in a single box of smaller size than either using VLSI technology.

In operation each of the nodes of the Cube can run its own program using its own data and passing messages to other nodes. By clever concurrent use of the processors at the nodes algorithms can be speeded up by a factor that is close to the number of nodes. Implementations of vortex methods (§4) using this methodology are currently underway in our laboratory. This hardware/software combination is not competitive with corresponding codes written for and run on present day supercomputers. However, the concurrent processor methodology is likely to be implemented on future generations of supercomputers, and so it is important to explore how one can adapt and develop efficient algorithms that take advantage of this new feature in computer architecture.

The final topic that I want to mention is the explosive growth in personal computers or workstations. Computing power is today by no means limited to the very large mainframe computers. Small portable machines with powers that rival the mightiest computers of a few decades ago are today readily available. Although in the sciences such computers are heavily used for text processing, program editing, post-processing of data, producing graphics, communicating with other (often larger)

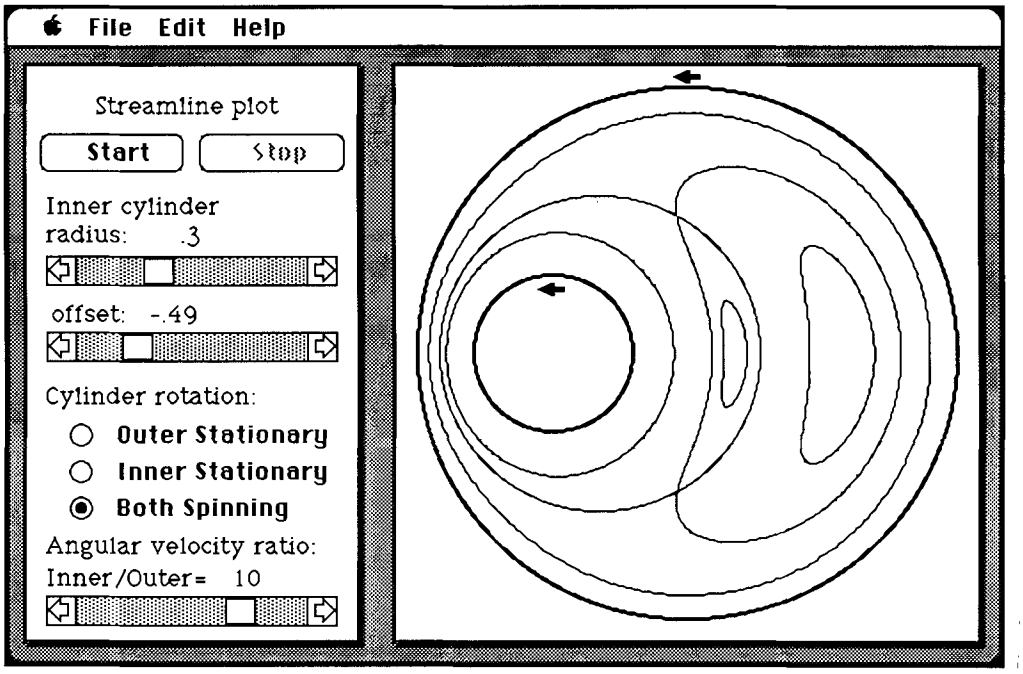

FIGURE 5. Basic screen in a demonstration program aimed at visualizing the streamlines for Stokes flow between eccentric rotating cylinders. The user sets parameters on the left and the streamline pattern is drawn on the right. The program is based on an analytic solution described by Ballal & Rivlin (1976).

computers, etc., and not much actual calculation is done on them, they do have tremendous potential for smaller problems including, in particular, illustrative examples of use in instruction. Just as a laboratory experiment of a few decades ago deemed to be particularly valuable frequently reappears as a component of a graduate or undergraduate laboratory course, so should computations originally done for research purposes be reworked to provide practice in numerical experimentation. There is some of this in fluid mechanics (see e.g. Olfe 1986), but not nearly enough. The tremendous incentive of wide markets that attract writers of word processors and spreadsheets does not seem to exist for advanced educational software.

In figure 5 I show the basic screen in a recent program written by K. Baird, S. Reed and myself for the Apple Macintosh™ personal computer. The program is written using the Aztec C development system. The problem that we chose to rework in this format arose as part of the recent investigation by Aref & Balachandar (1986) mentioned in §3. The flow being displayed is the two-dimensional Stokes flow between eccentric rotating cylinders (with the familiar Couette flow as a special limiting case). The basic screen contains various controls on the left (buttons, scroll bars) for the user to select the geometry (radius of inner cylinder, offset of centres) and the angular velocities of the two cylinders. By using such controls the programmer is assured that the input from the user always falls within required bounds. In more conventional dialogues extensive checking of input must be performed. Once the desired parameters have been set the user 'clicks' the start button and the program calculates the corresponding set of streamlines displayed on the right.

As the reader of the recent detailed analysis of this flow by Ballal & Rivlin (1976) will know the steady flow is extremely rich. Separation regions can appear adjacent to either cylinder when it is stationary, and internal saddle stagnation points (as shown in figure 5) appear for both co- and counter-rotating cylinders. Streamline patterns can be saved, and saved patterns can be read in and plotted up.

The student can spend several sessions with this kind of program. Initially one can discuss the solution of the standard Couette flow problem. Then the offset can be varied and the variety of the solution space explored. The criteria for separation can form the topic of another session. And when discussing the internal hyperbolic stagnation points, one can pursue the difference between the stagnation point for a cylinder with circulation in potential flow theory. In this Stokes flow example the self-intersections of the streamline are not in general at right angles because there is vorticity at the stagnation point.

We are currently developing a few projects of this type for the use of our students. The essential challenge to fluid mechanics and its practitioners is to provide ideas on what kind of flow situations make good demonstrations given the limitations in computing speed of most desktop computers. It is, in a new guise, the familiar query of how to provide much insight into fluid mechanics with relatively simple means, a quest very much in the spirit of G. I. Taylor.

Research support from the US Department of Energy (Office of Basic Energy Sciences) under grant DE-FG-03-85ER13349, by the National Science Foundation (NSF), Presidential Young Investigator (PYI) Award MSM-8451107, by the sponsors of the Petroleum Research Fund of the American Chemical Society under grant PRF15888-AC5, and by the University of California Microelectronics Innovation and Computer Research Opportunities (MICRO) program under grant 85-201 is gratefully acknowledged. Industrial matching funds for the PYI have been generously donated by the General Electric Company and by AMETEK Computer Research Division. Supercomputing resources provided through NSF at the National Centre for Atmospheric Research (NCAR), and the San Diego Supercomputer Center have been invaluable. NCAR is sponsored by NSF.

Appendix. Derivation of Tkachenko's (1964) polynomial relation

Since all the vortices are stationary, we have (by (1)) the conditions:

$$\sum_{\beta}{}' (z_\alpha - z_\beta)^{-1} = \sum_{\mu} (z_\alpha - \zeta_\mu)^{-1}, \tag{A 1}$$

$$\sum_{\alpha} (\zeta_\mu - z_\alpha)^{-1} = \sum_{\lambda}{}' (\zeta_\mu - \zeta_\lambda)^{-1}, \tag{A 2}$$

for each α and each μ. The primes signify omission of singular terms. By direct differentiation of (4a) we also have

$$P'(z) = P(z) \sum_{\alpha} (z - z_\alpha)^{-1}, \tag{A 3}$$

and then

$$\begin{aligned} P''(z) &= P(z) \{ \sum_{\alpha,\beta} (z-z_\alpha)^{-1}(z-z_\beta)^{-1} - \sum_{\alpha} (z-z_\alpha)^{-2} \} \\ &= P(z) \sum_{\alpha,\beta}{}' (z-z_\alpha)^{-1}(z-z_\beta)^{-1}. \end{aligned} \tag{A 4}$$

Now, when α and β are different, we can write

$$(z-z_\alpha)^{-1}(z-z_\beta)^{-1} = \{(z-z_\alpha)^{-1}-(z-z_\beta)^{-1}\}(z_\alpha-z_\beta)^{-1}. \tag{A 5}$$

Thus,
$$Q(z)\,P''(z) = 2P(z)\,Q(z)\sum_{\alpha,\beta}{}' (z-z_\alpha)^{-1}(z_\alpha-z_\beta)^{-1}. \tag{A 6}$$

Similarly,
$$Q''(z)\,P(z) = 2P(z)\,Q(z)\sum_{\mu,\lambda}{}' (z-\zeta_\mu)^{-1}(\zeta_\mu-\zeta_\lambda)^{-1}. \tag{A 7}$$

From (A 3) and the analogous expression for $Q'(z)$ we get

$$2P'(z)\,Q'(z) = 2P(z)\,Q(z)\sum_{\alpha,\mu}(z-z_\alpha)^{-1}(z-\zeta_\mu)^{-1},$$

or, using, a transcription similar to (A 5)

$$2P'(z)\,Q'(z) = 2P(z)\,Q(z)\sum_{\alpha,\mu}\{(z-z_\alpha)^{-1}-(z-\zeta_\mu)^{-1}\}(z_\alpha-\zeta_\mu)^{-1}. \tag{A 8}$$

Comparing (A 6), (A 7) and (A 8), and using the relations (A 1) and (A 2), immediately gives (5).

REFERENCES

Amsden, A. A. & Harlow, F. H. 1964 Slip instability. *Phys. Fluids* **7**, 327–334.

Aref, H. 1982 Point vortex motions with a center of symmetry. *Phys. Fluids* **25**, 2183–2187.

Aref, H. 1983 Integrable, chaotic, and turbulent vortex motion in two-dimensional flows. *Ann. Rev. Fluid Mech.* **15**, 345–389.

Aref, H. 1984 Stirring by chaotic advection. *J. Fluid Mech.* **143**, 1–21.

Aref, H. 1985 Chaos in the dynamics of a few vortices – fundamentals and applications. In *Theoretical and Applied Mechanics, Proc. 16th Intl Congr. Theor. Appl. Mech.* (ed. F. I. Niordson & N. Olhoff), pp. 43–68. Elsevier.

Aref, H. 1986 Finger, bubble, tendril, spike. An essay on the morphology and dynamics of interfaces in fluids. *Polish Acad. Sci. Fluid Dyn. Trans.* **13**, (in press).

Aref, H. & Balachandar, S. 1986 Chaotic advection in a Stokes flow. *Phys. Fluids* (in press).

Aref, H. & Siggia, E. D. 1981 Evolution and breakdown of a vortex street in two dimensions. *J. Fluid Mech.* **109**, 435–463.

Baker, G. R. 1982 Generalized vortex methods for free-surface flows. In *Waves on Fluid Interfaces* (ed. R. E. Meyer). Publ. MRC Univ. Wisconsin-Madison, vol. 50. Academic.

Baker, G. R., Meiron, D. I. & Orszag, S. A. 1980 Vortex simulation of the Rayleigh–Taylor instability. *Phys. Fluids* **23**, 1485–1490.

Ballal, B. Y. & Rivlin, R. S. 1976 Flow of a Newtonian fluid between eccentric rotating cylinders: Inertial effects. *Arch. Rat. Mech. Anal.* **62**, 237–294.

Batchelor, G. K. 1969 Computation of the energy spectrum in homogeneous two-dimensional turbulence. *Phys. Fluids* **12** (Suppl. 2), 233–239.

Beckmann, P. 1971 *A History of* π. The Golem Press.

Bensimon, D. 1986 Stability of viscous fingering. *Phys. Rev.* A **33**, 1302–1308.

Birkhoff, G. 1954 Taylor instability and laminar mixing. *Los Alamos Sci. Lab. Rep. No.* LA-1862; appendices in *Rep.* LA-1927.

Birkhoff, G. 1962 Helmholtz and Taylor instability. In *Hydrodynamic Instability, Proc. Symp. Appl. Maths*, vol. 13, pp. 55–76. Providence: Am. Math. Soc.

Calogero, F. 1978 Motion of poles and zeros of special solutions of nonlinear and linear partial differential equations and related 'solvable' many-body problems. *Nuovo Cimento* B **43**, 177–241.

Campbell, L. J. & Kadtke, J. B. 1986 Stationary configurations of point vortices and other logarithmic objects in two dimensions. *Los Alamos Nat. Lab.* preprint.

CAMPBELL, L. J. & ZIFF, R. M. 1978 A catalog of two-dimensional vortex patterns. *Los Alamos Sci. Lab. Rep. No.* LA-7384-MS, 40 pp.

CHARNEY, J. G. 1963 Numerical experiments in atmospheric hydrodynamics. In *Experimental Arithmetic, High Speed Computing and Mathematics, Proc. Symp. Appl. Maths*, vol. 15, pp. 289–310. Providence: Am. Math. Soc.

CHRISTIANSEN, J. P. 1973 Numerical simulation of hydrodynamics by the method of point vortices. *J. Comput. Phys.* **13**, 363–379.

CHUUDNOVSKY, D. V. & CHUUDNOVSKY, G. V. 1977 Pole expansions of nonlinear partial differential equations. *Nuovo Cimento* B **40**, 339–353.

COCHRAN, W. G. 1934 The flow due to a rotating disk. *Proc. Camb. Phil. Soc.* **30**, 365–375.

COUDER, Y. & BASDEVANT, C. 1986 Experimental and numerical study of vortex couples in two-dimensional flows. *J. Fluid Mech.* **173**, 225–251.

DEEM, G. S. & ZABUSKY, N. J. 1978 Vortex waves: Stationary 'V states', interactions, recurrence, and breaking. *Phys. Rev. Lett.* **40**, 859–862.

DEGREGORIA, A. J. & SCHWARTZ, L. W. 1986 A boundary integral method for two-phase displacement in Hele-Shaw cells. *J. Fluid Mech.* **164**, 383–400.

DE JOSSELIN DE JONG, G. 1959 Vortex theory for multiple phase flow through porous media. *Water Res. Ctr. Contr. No.* 23, UC Berkeley, 80 pp.

DE JOSSELIN DE JONG, G. 1960 Singularity distributions for the analysis of multiple-fluid flow through porous media. *J. Geophys. Res.* **65**, 3739–3758.

EMMONS, H. W. 1970 Critique of numerical modelling of fluid-mechanics phenomena. *Ann. Rev. Fluid Mech.* **2**, 15–37.

FEIGENBAUM, M. J. 1980 The metric universal properties of period doubling bifurcations and the spectrum for a route to turbulence. In *Nonlinear Dynamics* (ed. R. H. G. Helleman), *Ann. N.Y. Acad. Sci.* **357**, 330–336.

FORD, J. 1978 A picture book of stochasticity. In *Topics in Nonlinear Dynamics: A Tribute to Sir Edward Bullard* (ed. S. Jorna), *AIP Conf. Proc.* **46**, 121–146.

FOX, G. C. & OTTO, S. W. 1984 Algorithms for concurrent processors. *Phys. Today* **37**, 53–59.

FRISCH, U., HASSLACHER, B. & POMEAU, Y. 1986 A lattice gas automaton for the Navier–Stokes equation. *Phys. Rev. Lett.* **56**, 1505–1508.

GHONIEM, A. F. & SHERMAN, F. S. 1985 Grid-free simulation of diffusion using random walk methods. *J. Comput. Phys.* **61**, 1–37.

GOTTLIEB, D. & ORSZAG, S. A. 1977 Numerical analysis of spectral methods: theory and application. *CBMS-NSF Reg. Conf. Ser. Appl. Maths* vol. 26, 170 pp. SIAM, Philadelphia.

HARTREE, D. R. 1937 On an equation occurring in Falkner and Skan's approximate treatment of the equations of the boundary layer. *Proc. Camb. Phil. Soc.* **33**, 223–239.

HELE-SHAW, J. H. S. 1898 The flow of water. *Nature* **58**, 34–36.

HOMSY, G. M. 1987 Viscous fingering in porous media. *Ann. Rev. Fluid Mech.* (in press).

HOPCROFT, J. E. 1984 Turing machines. *Sci. Am.* **250**, 86–98.

D'HUMIERES, D., LALLEMAND, P. & SHIMOMURA, T. 1985 Lattice gas cellular automata, a new experimental tool for hydrodynamics. *Los Alamos Nat. Lab. Preprint* LA-UR-85-4051.

JAMESON, A. 1983 The evolution of computational methods in aerodynamics. *J. Appl. Mech.* **50**, 1052–1070.

KADANOFF, L. P. 1986 Fractals: Where's the physics? *Phys. Today* **39**, 6–7.

KNUTH, D. E. 1973 *The Art of Computer Programming.* Addison-Wesley.

KRAICHNAN, R. H. 1967 Inertial ranges in two-dimensional turbulence. *Phys. Fluids* **10**, 1417–1423.

LANDAU, L. D. & LIFSHIFTZ, E. M. 1959 *Fluid Mechanics.* Pergamon. 536 pp.

LEITH, C. E. 1968 Diffusion approximation for two-dimensional turbulence. *Phys. Fluids* **11**, 671–673.

LEONARD, A. 1980 Vortex methods for flow simulation. *J. Comput. Phys.* **37**, 289–335.

LEONARD, A. 1985 Computing three-dimensional incompressible flows with vortex elements. *Ann. Rev. Fluid. Mech.* **17**, 523–559.

LICHTENBERG, A. J. & LIEBERMAN, M. A. 1983 *Regular and Stochastic Motion.* Springer.

LIN, C. C. 1943 *On the Motion of Vortices in Two Dimensions.* Toronto University Press. 39 pp.

LONGUET-HIGGINS, M. S. & COKELET, E. D. 1976 The deformation of steep surface waves on water II. Growth of normal mode instabilities. *Proc. R. Soc. Lond.* A **364**, 1–28.

LORENZ, E. N. 1963 Deterministic nonperiodic flow. *J. Atmos. Sci.* **20**, 130–141.

MANDELBROT, B. B. 1977 *Fractals, Form, Chance, Dimension.* W. H. Freeman & Co.

MARGOLUS, N., TOFFOLI, T. & VICHNIAC, G. 1986 Cellular – automata supercomputer for fluid-dynamics modelling. *Phys. Rev. Lett.* **56**, 1694–1696.

METROPOLIS, N., HOWLETT, J. & ROTA, G-C. (eds) 1980 *History of Computing in the Twentieth Century – A Collection of Essays.* 659 pp. Academic.

MCWILLIAMS, J. C. 1984 The emergence of isolated coherent vortices in turbulent flow. *J. Fluid Mech.* **146**, 21–43.

MORIKAWA, G. K. & SWENSON, E. V. 1971 Interacting motion of rectilinear geostrophic vortices. *Phys. Fluids* **14**, 1058–1073.

NOVIKOV, E. A. 1983 Generalized dynamics of three-dimensional vortical singularities (vortons). *Sov. Phys., J. Exp. Theor. Phys.* **57**, 566–569.

OLFE, D. B. 1986 *Fluid Mechanics for the IBM PC.* McGraw Hill.

ONSAGER, L. 1949 Statistical hydrodynamics. *Nuovo Cimento* **6** (Suppl.), 279–287.

ORSZAG, S. A. 1970 Analytical theories of turbulence. *J. Fluid Mech.* **41**, 363–386.

ORSZAG, S. A. 1972 Comparison of pseudospectral and spectral approximation. *Stud. Appl. Maths* **51**, 253–259.

ORSZAG, S. A. & ISRAELI, M. 1974 Numerical simulation of viscous incompressible flows. *Ann. Rev. Fluid Mech.* **6**, 281–318.

ORSZAG, S. A. & YAKHOT, V. 1986 Reynolds number scaling of cellular – automaton hydrodynamics. *Phys. Rev. Lett.* **56**, 1691–1693.

PATTERSON, G. S. 1978 Prospects for computational fluid mechanics. *Ann. Rev. Fluid Mech.* **10**, 289–300.

PULLIN, D. I. 1982 Numerical studies of surface-tension effects in nonlinear Kelvin–Helmholtz and Rayleigh–Taylor instability. *J. Fluid Mech.* **119**, 507–532.

REICHENBACH, H. 1983 Contributions of Ernst Mach to fluid mechanics. *Ann. Rev. Fluid Mech.* **15**, 1–28.

ROBINSON, A. L. 1985*a* When are viscous fingers stable? *Science* **228**, 834–836.

ROBINSON, A. L. 1985*b* Fractal fingers in viscous fluids. *Science* **228**, 1077–1079.

ROGALLO, R. S. & MOIN, P. 1984 Numerical simulation of turbulent flows. *Ann. Rev. Fluid Mech.* **16**, 99–137.

ROSENHEAD, L. 1931 The formation of vortices from a surface of discontinuity. *Proc. R. Soc. Lond.* A **134**, 170–192.

SAFFMAN, P. G. 1986 Viscous fingering in Hele-Shaw cells. *J. Fluid Mech.* **173**, 73–94.

SAFFMAN, P. G. & MEIRON, D. I. 1986 Difficulties with three-dimensional weak solutions for inviscid incompressible flow. *Phys. Fluids* **29**, 2373–2375.

SEITZ, C. S. 1985 The cosmic cube. *Commun. ACM* **28**, 22–33.

TKACHENKO, V. K. 1964 Dissertion. Inst. Phys. Probl., Moscow, USSR.

TRYGGVASON, G. & AREF, H. 1983 Numerical experiments on Hele–Shaw flow with a sharp interface. *J. Fluid Mech.* **136**, 1–30.

TRYGGVASON, G. & AREF, H. 1985 Finger interaction mechanisms in stratified Hele–Shaw flow. J. Fluid Mech. **154**, 287–301.

VAN DYKE, M. 1984 Computer-extended series. *Ann. Rev. Fluid Mech.* **16**, 287–309.

WILSON, K. G. 1983 The renormalization group and critical phenomena. *Rev. Mod. Phys.* **55**, 583–600.

WORTHINGTON, A. M. 1908 *A Study of Splashes.* Longmans Green.

YAMAGUTI, Y. & OSHIKI, S. 1981 Chaos in numerical analysis of ordinary differential equations. *Physica* D **3**, 618–626.

YARMCHUK, E. J., GORDON, M. J. V. & PACKARD, R. E. 1979 Observation of stationary arrays in rotating superfluid Helium. *Phys. Rev. Lett.* **43**, 214–217.

ZABUSKY, N. J. 1981 Computational synergetics and mathematical innovation. *J. Comput. Phys.* **43**, 195–249.

ZABUSKY, N. J. & KRUSKAL, M. D. 1965 Interaction of 'solitons' in a collisionless plasma and the recurrence of initial states. *Phys. Rev. Lett.* **15**, 240–243.