CLM - R 106

CLM - R 106

United Kingdom Atomic Energy Authority
RESEARCH GROUP

Report

VORTEX

A 2-DIMENSIONAL HYDRODYNAMICS SIMULATION CODE

J. P. CHRISTIANSEN

Culham Laboratory
Abingdon Berkshire

1970

Available from H. M. Stationery Office
FIVE SHILLINGS AND SIXPENCE NET

C.13

U.D.C.
532.5 : 681.3.06
681.3.06 : 532.5

VORTEX

A 2-DIMENSIONAL HYDRODYNAMICS SIMULATION CODE

by

J.P. CHRISTIANSEN

A B S T R A C T

Program VORTEX is a computer code which simulates the behaviour of an incompressible, inviscid homogeneous fluid in two dimensions by following the motion of a large number of point vortices. A clarified, documented listing of the program is available in the Culham Library. The 1st edition of the program was completed in November, 1969, whilst the 2nd edition used at present was completed in May, 1970.

This report is written in two parts. Part I describes the theory behind the program and Part II describes the program itself. Part II could conveniently be read in conjunction with the clarified listing.

U.K.A.E.A., Research Group,
Culham Laboratory,
Abingdon,
Berks.

July, 1970.

C O N T E N T S

INTRODUCTION

Although the basic equations of hydrodynamics have been studied for many years[1,2], their complexity sets a limit to the types of problems that can be solved analytically. In some cases relatively simple non-linear stationary flow patterns can be realized, in which the parameters that describe the physical systems are independent of time. In other cases the interest lies in the stability properties of such a stationary flow pattern, that is, whether or not it is able to sustain itself against small variations in certain of the characteristics parameters. To solve this type of problem one normally applies perturbation theory, in order to predict whether or not the original equilibrium is linearly stable. A theoretical analysis of this kind can follow the history of a fluid system in the linear regime, when the perturbation amplitudes are small, but because the equations of hydrodynamics are non-linear it is of considerable interest to study its history in the subsequent large amplitude regime. One way of doing this is to simulate the time behaviour of a mathematical model of the fluid on a computer. This report describes one of the various alternative methods by which a computer simulation can be carried out.

PART I: HYDRODYNAMIC MODEL

1. HAMILTONIAN EQUATIONS OF MOTION

The basic equations governing the time behaviour of an incompressible, inviscid, homogeneous fluid in purely 2-dimensional motion are

$$\underline{\nabla} \cdot \underline{u} = 0 \tag{1.1}$$

$$\frac{\partial \underline{u}}{\partial t} + \underline{u} \cdot \nabla \underline{u} = - \frac{1}{\rho_o} \nabla P \tag{1.2}$$

where $\underline{u}$ is the fluid velocity, ρ_o is the density and P the pressure. It has been pointed out by Onzager[10], and more recently by Morikawa[16], that these equations can be expressed in Hamiltonian form, with the spatial coordinates x, y playing the part of conjugate coordinates and momenta q, p. To see this we write

$$\underline{u} = \underline{\nabla} \times H \tag{1.3}$$

$$f = \underline{\nabla} \times \underline{u} \tag{1.4}$$

where the Hamiltonian H is the usual stream function, and f is the vorticity. Then eqs.(1.2) and (1.3) combined with (1.4) give

$$\nabla^2 H = - f \tag{1.5}$$

$$\frac{\partial f}{\partial t} + \underline{u} \cdot \nabla f \equiv \frac{\partial f}{\partial t} + [f, H] = 0 \tag{1.6}$$

Eq.(1.6) is Liouville's equation in Poisson-bracket form*, and the Hamiltonian H is to be obtained by solving Poisson's equation (1.5).

---

* For any two functions A, B, the Poisson bracket is defined by

$$[A, B] = \frac{\partial A}{\partial q} \frac{\partial B}{\partial p} - \frac{\partial B}{\partial q} \frac{\partial A}{\partial p}$$

2. INVARIANTS OF THE MOTION

Because the fluid system we are considering is not interacting with any other physical system there are no dissipative phenomena. Hence the total energy E is an invariant of the motion and this energy is purely kinetic

$$E = \tfrac{1}{2} \rho_o \int_R \underline{u} \cdot \underline{u} \, dS \tag{2.1}$$

or if we integrate by parts using (1.3 and (1.4)

$$E = - \tfrac{1}{2} \rho_o \int_R f \, H \, dS \tag{2.2}$$

The expression (2.2) is valid provided that

$$\int_C H \frac{\partial H}{\partial h} \, ds = 0 \tag{2.3}$$

where C is a contour bounding the region of integration R.
Because the scalar vorticity f is convected with the moving incompressible fluid, there is an infinite number of other invariants

$$A(f) \, df \tag{2.4}$$

expressing the area between moving contours of constant f, $f + df$. In most of the cases which we study, only part of the fluid has vorticity different from zero, and then this fluid has constant total area.

$$= \int_{f_{min}}^{\infty} A(f) \, df \tag{2.5}$$

Real space acts as a phase space for the system with the coordinates (x, y) or (q, p) conjugated through

$$\dot{x} = \dot{q} = \frac{\partial H}{\partial p} = \frac{\partial H}{\partial y} \quad , \quad \dot{y} = \dot{p} = - \frac{\partial H}{\partial q} = - \frac{\partial H}{\partial x} \tag{2.6}$$

Hamiltonian notation will be used throughout this report to emphasize the true structure of the equations, but another simplication that should be pointed out is that the Hamiltonian is invariant under rotations and translations of the (q, p) plane, which are of course just rotations and translations of the real 2-dimensional space. This degree of symmetry is unusual in Hamiltonian mechanics, and should introduce simplification. (as compared to Vlasov's equation in plasma physics, for example).

Our present understanding of the significance of this Hamiltonian formulation of 2D fluid dynamics may be explained in the following way. There are two great independent classical formalisms which involve the motion of incompressible fluids:

I. Hamiltonian theory, (phase fluids)

II. Vortex theory,[1,18] (real fluids).

Hamiltonian theory is intimately related to statistical mechanics, (both Maxwell-Boltzmann and Lynden-Bell[15]), vortex theory is related to the theory of fluid turbulence. Any connection between the two theories would be of great interest, however, and while the ideas of turbulence theory are currently being applied to phase fluids (e.g. in plasma physics), those of statistical mechanics are being applied to hydrodynamic turbulence.

(a) In the case of Vlasov's equation for particles interacting in n-dimensional configuration space, it is useful to study the incompressible motion of a self-interacting phase fluid[18] in an $m = 2n$-dimensional phase space, $n = 1, 2, 3$. The distribution function f is always a scalar, and m must be even. Classical statistical mechanics can be applied to the particles, and Lynden-Bell theory[15] to the phase fluid.

(b) In fluid dynamics, one can have incompressible inviscid motion in any number of dimensions $m = 1, 2, 3$, and the vorticity plays a significant role.

(c) For the particular case $m = 2$, the two types of formalism are in close correspondence, with the scalar vorticity f playing the part of the distribution function. Lynden-Bell theory can be applied to continuous distributions of vorticity, and Maxwell-Boltzmann theory to collections of point vortices.

(d) For the physically more significant case $m = 3$, ordinary Hamiltonian theory cannot be applied, because the number of dimensions is odd and the vorticity is a vector; one is studying the interaction of a collection of vortex tubes, rather than point vortices. Nevertheless it would be interesting to look for some generalization of Hamiltonian dynamics to this case, and the rewards might be great if statistical mechanics could also be successfully generalized.

3. POINT VORTICES

One way of following the motion of the fluid is to compute the time behaviour of two functions, the vorticity $f(q, p, t)$ and the Hamiltonian $H(q, p, t)$. These can be represented on a discrete mesh, as in the method of Fromm and Harlow[5]. Another technique is to calculate the motion of a limited number of point vortices, interacting with each other via their individual 2-body velocity potentials. This was done by Abernathy and Kronauer[6] in their study of the formation of a von Karman Vortex Street. A third possibility is to follow the motion of contours f = constant. This has been done successfully in plasma physics for the analogous case of the 1-dimensional Vlasov equation by Roberts and Berk[7], who used the 'waterbag' model earlier adopted by de Pack[8] and Dory[9], in which f is everywhere either a constant f_0, or zero. An assumption of a similar type, (vorticity constant or zero), has been made by many workers in hydrodynamics. Extensions of the Roberts-Berk method in the vortex case would involve changes in their program, but would be an interesting problem to try.

The method used in the VORTEX program, and reported briefly by Christiansen and Roberts[11,12] is more analogous to the particle methods used in plasma physics. It approximates f by a large number of point vortices:

$$f = \sum_{i=1}^{N} f_i \, \delta(q - q_i) \, \delta(p - p_i) \qquad (3.1)$$

where $f_i = +1$ or -1, interacting via a self-consistent velocity field. Each set of coordinates (q_i, p_i) is called a point vortex and satisfies the equations of motion

$$q_i = \frac{\partial H}{\partial p_i} \qquad p_i = -\frac{\partial H}{\partial q_i} \tag{3.2}$$

where H is the solution of Poissons equation (1.5) with f obtained from eq.(3.1);

$$H = -\sum_{\substack{i=1 \\ i\neq j}}^{N} \sum_{j=1}^{N} f_i f_j \log r_{ij} \tag{3.3}$$

with

$$r_{ij}^2 = (q_i - q_j)^2 + (p_i - p_j)^2 \,. \tag{3.4}$$

The infinite 'self energy' of a point vortex is excluded. The problem is thus reduced to solving eqs.(3.1), (3.2) and (3.3).

4. THE REGION OF CALCULATION AND ITS BOUNDARY CONDITIONS

For practical reasons we shall restrict ourselves to considering a square region in (q, p) or (x, y) space. Three types of boundary conditions are of interest:

A. H constant along a boundary, i.e. the boundary is a streamline.

B. $\frac{\partial H}{\partial n} = 0$ along a boundary, i.e. the boundary is perpendicular to all streamlines.

C. H is a periodic function.

These three conditions can be imposed separately on the x and y boundaries and therefore combine to give 9 possible sets of boundary conditions. The square region is covered by a square mesh of size 64×64.

A hydrodynamic fluid which is incompressible, inviscid and homogeneous is not entirely realistic in a physical sense, but it is mathematically convenient for two reasons; firstly because it provides a model for turbulent flows in the limit of infinite Reynolds number, and secondly because (in 2 dimensions) the system obeys classical Hamiltonian dynamics, so that a good deal of existing dynamical theory is likely to be applicable.

The basic hydrodynamic system, which obeys the partially differential equations (1.5) and (1.6), is a classical Hamiltonian system with an infinite number of degrees of freedom because of the continuous nature of the fluid. Such systems are well known, (the most familiar example being the classical electromagnetic field), and when interacting with systems with a finite number of degrees of freedom they give rise to interesting phenomena such as the ultra-violet catastrophe. By itself, however, a classical system with an infinite number of degrees of freedom behaves in a self-consistent way.

Compared to the exact physical system, the numerical model which is used to approximate it has three principal limitations:

I. The finite number of point vortices used to approximate f (eq.3.1).

II. The use of a discrete mesh.

III. The choice of boundary conditions.

I and II are of a practical nature and can be improved upon by increasing N (the number of point vortices) and M (the number of meshpoints). The simplified boundary conditions which are used facilitate the coding and speed up the program, but because of the square region of calculation, small m = 4 perturbations are produced in rotationally symmetric problems. A physically more realistic choice of boundary conditions would be problem dependence, more difficult to code and more time-consuming in execution.
The computer model described in this note has been used at Culham to simulate some 10 different problems, and there seems to be no reason why many other interesting hydrodynamic problems could not be solved in this way.

5. DIFFERENCE FORMULATION OF THE EQUATIONS

Since the calculation takes place on a mesh we approximate all space derivatives by finite differences.

Suppose that f is given at all the meshpoints (section 7). Instead of solving eq.(3.3) we solve Poisson's equation according to:

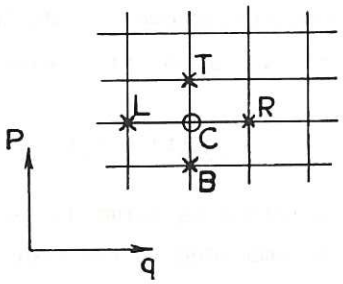


Fig.1.

$$\frac{H_T + H_B - 2H_C}{(\Delta p)^2} + \frac{H_R + H_L - 2H_C}{(\Delta q)^2} = - f_C \qquad (5.1)$$

where the suffices are explained in Fig.1. Equations (5.1) is solved by the "Hockney-Poisson solver" program[13,14] using a fast Fourier transform and recursive cyclic reduction technique and will not be dealt with further here.

Equation (3.2) becomes

$$\dot{q}_C = \frac{H_T - H_B}{2\Delta p} \qquad \dot{p}_C = \frac{H_L - H_R}{2\Delta q} \qquad (5.2)$$

6. LEAPFROG TIME INTEGRATION SCHEME

In order to integrate eq.(5.2) in time we simply write

$$\frac{q(t+\Delta t) - q(t-\Delta t)}{2\Delta t} = \dot{q}(t) \qquad (6.1)$$

and similarly for p. (A first-order integration would be too inaccurate, as explained in Section 11A). Thus if f, (and hence the Hamiltonian H and the velocity fields $\left(\frac{\partial H}{\partial p}, -\frac{\partial H}{\partial q}\right)$, as well as vortex positions q, p are known at time t, we can advance the positions from $t - \Delta t$ to $t + \Delta t$. This is done by introducing 2 sets of point vortices whose positions are defined at even times $t = 2n\Delta t$ and odd times $t = (2n+1)\Delta t$ respectively. Each set of vortices determines the values of f(t), H(t) and hence the velocity fields which are used to move the other set.

Referring to Fig.2, we can write eq.(6.1) for the even coordinates q_e as follows, (and similarly for p_e, and for q_o, p_o):

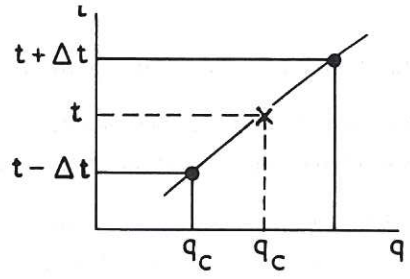


Fig.2.

$$q_e(t+\Delta t) = q_e(t-\Delta t) + \dot{q}_o(q_c, p_c, t) \cdot 2\Delta t \qquad (6.2)$$

where q_c, p_c are the vortex coordinates at the 'central' time t.

However, there are several ways of finding q_c and p_c in order to obtain a time and space - centred approximation.

The most straightforward and best known method is the leapfrog scheme, in which q_c, p_c are simply the coordinates of the odd set of particles, from which the velocity fields q_o, p_o themselves are determined. Another method used in program VORTEX (Section 9) is a Taylor-expansion scheme, in which q_c, p_c are computed from the even particle coordinates at time $t - \Delta t$, by the preliminary calculation

$$q_c(t) = q_e(t-\Delta t) + q_o(q_e(t-\Delta t), p_e(t-\Delta t), t)\ \Delta t \qquad (6.3)$$

The leapfrog method is normally used in VORTEX, but the more elaborate Taylor-expansion scheme can be embedded in the code without difficulty. Both schemes are weakly unstable because the odd and even vortices gradually get out of step. This problem is studied in Section 8, and a 'matching' procedure used to remove incipient instabilities is described in Section 9.

7. AREA-WEIGHTING OR NEAREST-GRID-POINT APPROXIMATIONS

From the vector N_{qp} containing N sets of particle coordinates (q, p) we construct a mesh or grid function f_{ij}. To do this, the 'charge' associated with each point vortex must be allocated to the surrounding grid points in a prescribed way. Two standard methods are mentioned here:

Nearest-grid-point approximation

The region is divided up into square cells (Fig.3). All the charge belonging to a point vortex is allocated to the grid-point at the centre of the cell in which the vortex lies.

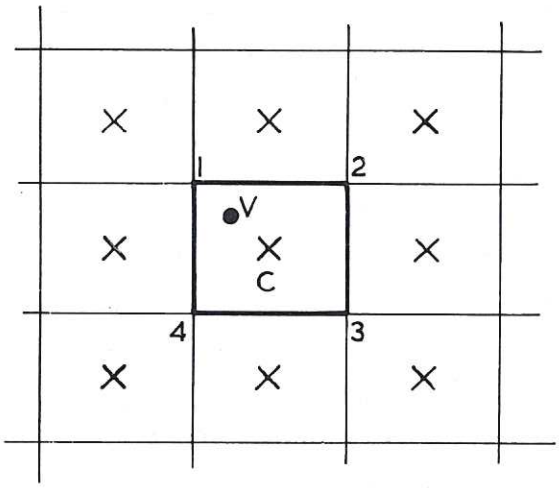


Fig.3. All the charge associated with a vortex V inside the cell 1234 is credited to the grid point C.

Area-weighting approximation

The point vortex is imagined to be replaced by a space vortex of uniform vorticity, whose size and orientation are the same as those if the grid cells. It will then overlap 1, 2, 3, or 4 grid cells. (Fig.4). Its charge is allocated to the centres of the cells which it overlaps, the weighting field being given by the areas of overlap. If the cells have unit dimension and (α,β) are the coordinates of V relative to D, then the weighting factors are:

$$\left.\begin{array}{ll} A & (1-\alpha)\beta \\ B & \alpha\beta \\ C & \alpha(1-\beta) \\ D & (1-\alpha)(1-\beta) \end{array}\right\} \qquad (7.1)$$

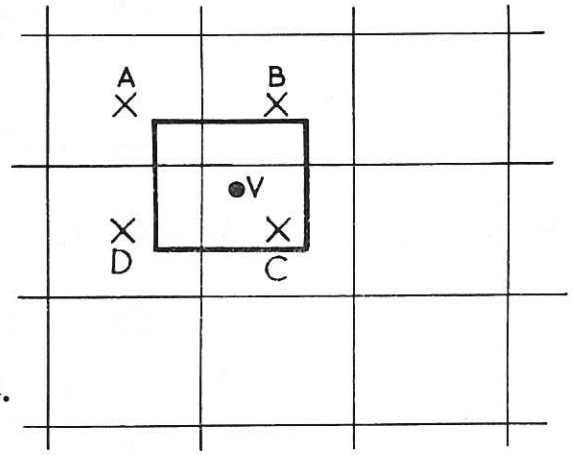


Fig.4.

Program VORTEX uses this second method which provides a higher accuracy, although at an increased cost in computer time.

8. STABILITY OF THE DIFFERENCE SCHEME

Care is needed to prevent the odd and even vortices getting out of step. Fig.5. shows a streamline on which the "correct" position of point vortices are marked. $\underline{\underline{\bullet}}$ is an "even" point, i.e. at a time $t + 2n\Delta t$.
$\underline{\underline{o}}$ is an "odd" point, i.e. at a time $t + (2n+1)\Delta t$.

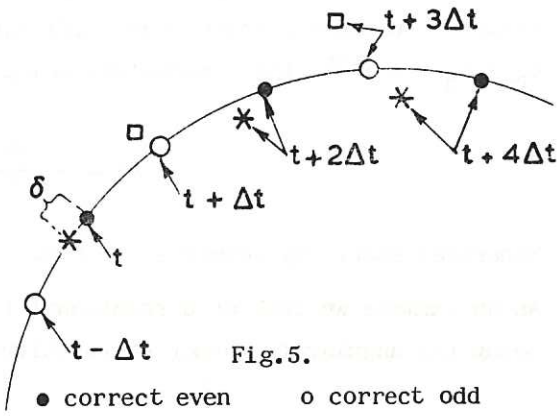


Fig.5.

● correct even o correct odd
□ calculated even * calculated odd

In order to examine the numerical stability of the calculation we write eq.(6.1) for both the odd and even positions and form the limit $\Delta t \to 0$. Setting $\underline{r} = (q, p)$ we get

$$\frac{d\underline{r}_e}{dt} = \dot{\underline{r}}\,(\underline{r}_C) \qquad \frac{d\underline{r}_o}{dt} = \dot{\underline{r}}\,(r_C) \qquad (8.1)$$

where $\underline{r}_C$ is the position at which the velocities are evaluated. This $\underline{r}_C$ is assumed to deviate from the correct value R_C.

Leapfrog scheme

$$\text{Suppose} \quad \underline{r}_e = \underline{R}_C + \underline{\delta} \quad \text{and} \quad \underline{r}_o = \underline{R}_C - \underline{\delta} \qquad (8.2)$$

and consider the motion in a specified velocity field, independent of the vortex particles.

Inserting (8.2) in eq.(8.1) we have:

$$\frac{d(\underline{R}_C + \underline{\delta})}{dt} = \dot{\underline{r}}\ (\underline{R}_C - \underline{\delta}), \quad \frac{d(\underline{R}_C - \underline{\delta})}{dt} = \dot{\underline{r}}\ (\underline{R}_C + \underline{\delta})\ . \qquad (8.3)$$

Subtraction gives

$$\frac{d\underline{\delta}}{dt} = - \frac{\partial \dot{\underline{r}}}{\partial t} \underline{\delta} \ , \tag{8.4}$$

or

$$\left.\begin{aligned} \frac{d\delta_q}{dt} &= - \frac{\partial \dot{q}(\underline{R}_C)}{\partial q} \delta q - \frac{\partial \dot{q}(R_C)}{\partial p} \delta p \ , \\ \frac{d\delta_p}{dt} &= - \frac{\partial \dot{p}(R_C)}{\partial q} \delta q - \frac{\partial \dot{p}(R_C)}{\partial p} \delta p \ . \end{aligned}\right\} \tag{8.5}$$

Setting $u_q = \dot{q}$, $u_p = \dot{p}$, and using eqs.(1.1) and (1.3), eq.(8.5) becomes

$$\left.\begin{aligned} \frac{d\delta_q}{dt} &= - \frac{\partial u_q}{\partial q} \delta q - \frac{\partial u_q}{\partial p} \delta p \ , \\ \frac{d\delta_p}{dt} &= - \left(f + \frac{\partial u_q}{\partial p} \right) \delta q + \frac{\partial u_q}{\partial q} \delta p \ . \end{aligned}\right\} \tag{8.6}$$

At this stage we must specify the kind of velocity field which is being studied. If we assume a stationary state, eqs.(8.6) can be integrated in time. We can set $\delta_p = \delta_q = \delta \, e^{i\delta t}$, the eigenvalues being given by

$$\delta^2 = - \left(\frac{\partial u_q}{\partial q} \right)^2 - \left(\frac{\partial u_q}{\partial p} \right)^2 - f \frac{\partial u_q}{\partial p} \tag{8.7}$$

Numerical stability requires $\delta^2 \geqslant 0$.

As an example we look at a rotational flow where $\underline{u} = (u_q , u_p) = \omega(\underline{r}) \times \underline{r}$, $\omega = (0,0,\omega)$ being the angular frequency of rotation, and $\underline{r}$ the radius vector.

Since $f = 2\,\omega(\underline{r}) + |\underline{r}| \frac{d\omega}{dr}$, eq.(8.7) gives

$$\omega^2 \left[1 + \frac{1}{\omega} r \frac{d\omega}{dr} \right] \geqslant 0 \quad \text{or} \quad r \frac{d \, \ell_n \omega}{dr} \geqslant - 1 \tag{8.8}$$

Hence for $\omega(r)$ varying as r^n the stability condition is

$$n \geqslant - 1 \tag{8.9}$$

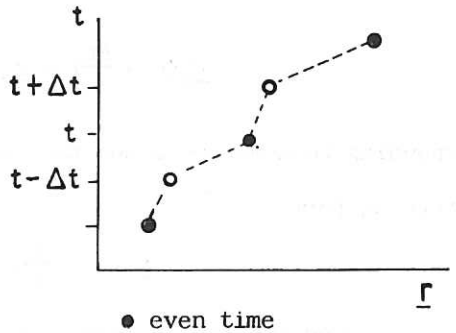
corresponding to

$$\left.\begin{aligned} \delta &= \pm \omega \sqrt{n+1} \ , \qquad n \geqslant - 1 \ , \\ \delta &= \pm \omega \sqrt{|n| - 1} \ , \quad n \leqslant - 1 \ . \end{aligned}\right\} \tag{8.10}$$

The amplification of δ during one rotation is $\exp(2\pi \sqrt{|n| - 1})$. From this we conclude that since in regions where $f = 0$ ω will very often vary as r^{-2}, whilst in regions where $f \neq 0$ ω will predominently vary as r^2, numerical instabilities will occur at the border between these regions over a time period in which $\frac{\partial u_q}{\partial q}$, $\frac{\partial u_q}{\partial p}$ and f do not change significantly. It is therefore necessary to prevent numerical instabilities from dominating the motion of the point vortices.

9. THE MATCHING PROCEDURE

Figure 6 shows, (in an exaggerated way), how the odd and even positions can "get out of step".
To prevent this from becoming too serious we stop the integration periodically, and readjust the positions of the vortices.



● even time
o odd time Fig.6.

If the calculation is stopped at time t we know $\underline{r}_e(t)$ and $\underline{r}_o(t - \Delta t)$. We form

$$\underline{r}_C(t - \tfrac{\Delta t}{2}) = \tfrac{1}{2}(\underline{r}_e(t) + \underline{r}_o(t - \Delta t)). \tag{9.1}$$

We then calculate $f(t - \frac{\Delta t}{2})$ as explained in Section 7, and solve for $H(t - \frac{\Delta t}{2})$ using the potential-solver. From eqs.(5.2) we find $\underline{u}_C(t - \frac{\Delta t}{2})$ and define a <u>new</u> set of positions at time $t - \Delta t$ and t respectively:

$$\tilde{\underline{r}}_o(t - \Delta t) = \underline{r}_C(t - \tfrac{\Delta t}{2}) - \underline{u}_C(t - \tfrac{\Delta t}{2}, \underline{r}_C)\,\tfrac{\Delta t}{2} \tag{9.2}$$

$$\tilde{\underline{r}}_e(t) = \underline{r}_C(t - \tfrac{\Delta t}{2}) + \underline{u}_C(t - \tfrac{\Delta t}{2}, \underline{r}_C)\,\tfrac{\Delta t}{2} \tag{9.3}$$

So that the positions are displaced symmetrically about $\underline{r}_C$ as shown in Fig.7.

We notice that the procedure is only correct to second order in t, because the chosen central velocity $\underline{u}_C$ is <u>not</u> the space-time averaged value over the interval $\frac{\Delta t}{2}$.

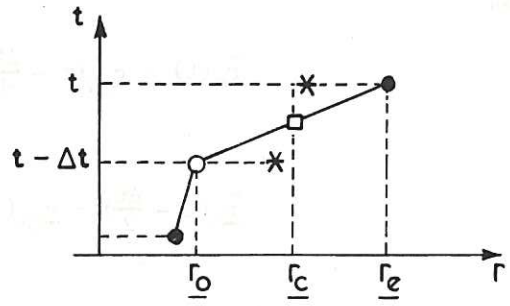


Fig.7. * new positions
□ calculated centre position

We can illustrate the use of this procedure by estimating the number of timesteps M between successive "matching of positions" which will achieve adequate accuracy. We noticed in Section 8 that amplification during one rotation using the leapfrog scheme was

$$\exp(2\pi\sqrt{|n| - 1}) = \exp(2\pi)$$

for $n = -2$. As an example we consider a circular cloud of vorticity of radius R, so that the time for one rotation is $2\pi R/V_R$. For reasons of accuracy we require $\delta M\,\Delta t \approx 1$, so that the time between successive matchings is $\approx R/V_R$. Typically V_R is chosen to be $\lambda/\Delta t$ where $\lambda = \frac{1}{4} - \frac{1}{2}$ the mesh spacing, and R is say 8, so that M is of order

$$M \approx \frac{R}{\lambda} \tag{9.4}$$

or say 16-32.

In the second edition of VORTEX the centring and matching procedures have been improved to reduce the amplitudes of computational modes.

Suppose as before that at time t we know $\underline{r}_C(t)$, $\underline{r}_o(t - \frac{\Delta t}{2})$ and $\underline{u}(t - \frac{\Delta t}{2})$. We then form:

$$\underline{r}_o'(t - \tfrac{\Delta t}{4}) = \underline{r}_o(t - \tfrac{\Delta t}{2}) + \underline{u}(t - \tfrac{\Delta t}{2}\ \ \underline{r}_o)\ \ \tfrac{\Delta t}{4} \tag{9.5}$$

corresponding to a motion along the tangent at $t = t - \frac{\Delta t}{2}$.

As before we form

$$\underline{r}_C(t - \tfrac{\Delta t}{4}) = \tfrac{1}{2}(\underline{r}_e(t) + \underline{r}_o(t - \tfrac{\Delta t}{2})) \tag{9.6}$$

having preserved the set positions $\underline{r}_o(t - \frac{\Delta t}{2})$. As a centre position we then average $\underline{r}_o'$ and $\underline{r}_C$ according to

$$\underline{r}_{CN}(t - \tfrac{\Delta t}{4}) = \tfrac{1}{2}(\underline{r}_C(t - \tfrac{\Delta t}{4}) + \underline{r}_o'(t - \tfrac{\Delta t}{4})) \tag{9.7}$$

From $\underline{r}_{CN}$ we calculate $f_{CN}(t - \frac{\Delta t}{4})$, $H_{CN}(t - \frac{\Delta t}{4})$ and $\underline{u}_{CN}(t - \frac{\Delta t}{4})$. To separate odd and even positions from $\underline{r}_{CN}$ we now use the Taylor-Expansion scheme (eq.6.3), that is we find

$$\underline{r}_1(t - \tfrac{\Delta t}{8}) = \underline{r}_{CN}(t - \tfrac{\Delta t}{4}) + \underline{u}_{CN}(t - \tfrac{\Delta t}{4}, \underline{r}_{CN}) \cdot \tfrac{\Delta t}{8} \tag{9.8}$$

$$\underline{r}_2(t - \tfrac{3\Delta t}{8}) = \underline{r}_{CN}(t - \tfrac{\Delta t}{4}) - \underline{u}_{CN}(t - \tfrac{\Delta t}{4}, \underline{r}_{CN})\ \tfrac{\Delta t}{8} \tag{9.9}$$

The new set of positions at $t = t$ (even) and $t = t - \frac{\Delta t}{2}$ (odd) are then in turn found from

$$\tilde{\underline{r}}_e(t) = \underline{r}_{CN}(t - \tfrac{\Delta t}{4}) + \underline{u}_{CN}(t - \tfrac{\Delta t}{4}, \underline{r}_1 . \tfrac{\Delta t}{4} \tag{9.10}$$

$$\tilde{\underline{r}}_o(t - \tfrac{\Delta t}{2}) = \underline{r}_{CN}(t - \tfrac{\Delta t}{4}) - \underline{u}_{CN}(t - \tfrac{\Delta t}{4}, \underline{r}_2) . \tfrac{\Delta t}{4} \tag{9.11}$$

These two improvements have proved useful when testing the program on a single circular vortex. Because the positions found from eqs.(9.2), (9.3) are correct to first order only they will introduce an expansion δ_i of a circular vortex when applied for the first order time ($\underline{r}_C$ being the initial position). For a circular vortex (see Fig.8) this expansion is approximately

$\delta_i \approx R \cdot \frac{1}{2}\,(u\,\frac{\Delta t}{4} \cdot \frac{1}{2\pi R})^2$. With $R = 7.2$, $\mu \cdot \Delta t = 0.6$, (the values used in the numerical experiment), we get $\delta_i \simeq 1.4 \cdot 10^{-4}$, which is in agreement with the measured value $1.45 \cdot 10^{-4}$. The new centring (eq.9.7) puts however $\delta_i = 0$.

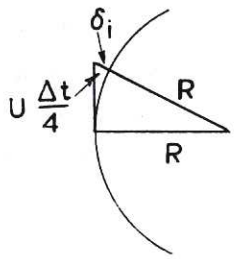


Fig.8.

10. INACCURACIES IN THE DIFFERENCE SCHEME

Although the time integration can be made sufficiently accurate for $\Delta t \to 0$, the integration of eqs.(3.2) and (3.3) as approximated by eqs.(5.1) and (5.2) introduces a

significant overall inaccuracy.

Consider a vorticity distribution

$$f = \cos \frac{2\pi kq}{N} \tag{10.1}$$

where N is the number of meshpoints in the q-direction. The analytical solution to eq.(3.2) is

$$H = \left(\frac{N}{2\pi k}\right)^2 \cos \frac{2\pi kq}{N} \tag{10.2}$$

Inserting (10.2) in (5.1) we get for $\Delta q = 1$,

$$\frac{\Delta^2 H}{(\Delta q)^2} = \left(\frac{N}{2\pi k}\right)^2 \left(\cos\left(-\frac{2\pi k}{N}\right) + \cos\frac{2\pi k}{N} - 2\right)\Big) = f_\Delta \tag{10.3}$$

Comparing f and f_Δ we see that

$$D = \frac{f_\Delta}{f} = \left(\frac{N}{2\pi k}\right)^2 2\ (1 - \cos\frac{2\pi k}{N}) \tag{10.4}$$

Program VORTEX uses N = 64. From (10.4) we can calculate the truncation of a mode and as examples we get

k-value	ratio	error
k = 1	D = 0.999	-
k = 4	D = 0.99	1 %
k = 8	D = 0.955	4.5 %
k = 16	D = 0.815	19.5 %
k = 32	D = 0.405	59.5 %

The approximation (5.2) will truncate the higher harmonics in a similar way.

The truncation errors become significant in regions where $\frac{\partial f}{\partial q}$, $\frac{\partial f}{\partial p}$ are large. In program VORTEX we normally consider distributions in f which are Heaviside functions. This means that at the borders where ∇f is very large (theoretically infinite) care must be taken to observe whether or not "diffusion" occurs because of the numerical instabilities which are likely to arise there.

11. TEST RUNS WITH VORTEX

It is important to carry out controlled numerical experiments on simulation codes, both to eliminate programming errors and also to determine the accuracy of the difference approximations. Testing a computer simulation code is often very difficult because of the number of parameters involved, as well as the range of values which they can take. A meaningful series of tests would comprise a suite of "like" runs in which each parameter is varied independently. It therefore seems appropriate to record our experience with the VORTEX code, for a number of successful and unsuccessful runs. This section will deal with a number of numerical tests, and the starting point is a very simple problem.

One Circular Vortex

Consider a circular vortex of radius R. The velocity field due to this vortex is

$$V_\theta = \tfrac{1}{2} F_0 r \ , \ V_r = 0, \ (0 \leqslant r \leqslant R),$$

$$V_\theta = \tfrac{1}{2} F_0 \frac{R^2}{r} \ , \ V_r = 0, \ (R \leqslant r < \infty).$$

where F_0 is the total vorticity defined by $F_0 = \pi R^2 f_0$, f_0 being the constant vorticity density. This system is in stable equilibrium. Program VORTEX has simulated this situation in a variety of ways.

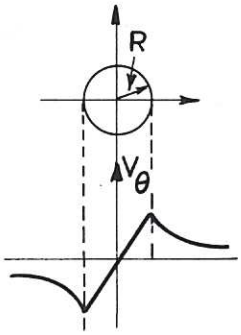


Fig.9.

A. First order time integration, nearest grid point approximation

The vorticity distribution is set up by distributing point vortices at random within a circle of radius R. After approximately 200 timesteps with $\Delta t = 0.50$ the circle has expanded to twice its original radius, thus showing the necessity of using a better scheme.

B. Second order scheme, (Taylor-expansion scheme), nearest grid point approximation

The vorticity distribution is set up as in A. After approximately 250 timesteps with $\Delta t = 0.25$ (2nd order) the original circular shape is still retained. However the energy has decreased by 6%, and the enstrophy (vorticity squared) by a similar amount. The nearest grid point approximation cannot conserve energy sufficiently well and it introduces fluctuations which can be interpreted as "numerical viscosity". Particle trajectories are not exactly circles.

C. Second order scheme (leapfrog), with area-weighting

The vorticity distribution is set up as in A. The circular shape is still retained after 100 timesteps with $\Delta t = 0.25$. The energy shows a variation of about 0.6%, a considerable improvement on B. The random distribution however still introduces fluctuations although the area-weighting implies a smoothing of the distribution, (truncation of higher harmonics).

D. Second order scheme (leapfrog), area-weighting

In this case a uniform f_0 is set up by distributing points on equidistantly spaced concentric circles. 1000 timesteps with $\Delta t = 0.25$ have been calculated. Energy varies by $\sim 0.02\%$ between its maximum and minimum values, an improvement from C, while the enstrophy varies by 0.06%. Each circle of points retains its shape virtually unaltered until 600-700 timesteps have elapsed. The area and circumference of the outer circle have been examined by an analyser program. The result is that the area as function of time oscillates with an amplitude of order 10^{-5} and a period of order 0.1 - 0.2 T, where T is the rotation time for the vortex ($T = \frac{4\pi}{F_0}$). The circumference as a function of time oscillates in a similar way. The conclusion is that a number of "false" m modes ($e^{im\theta}$) are being superimposed by the approximations which have been made. Because of the square shaped region of calculation (see Fig.I) the equilibrium streamlines will not be circles. All modes $m = 4, 8, 12, 16, \ldots\ldots$ will therefore be present as false modes and a quantitive estimate by an analysis program (MODANA) reveals that the amplitudes of these modes grow from 0.02 ($t = 0$) to 0.3 ($t = 2T$) in units of one mesh cell. Other modes $m = 1, 2, 3, 5, 6, 7, 9$ have amplitude less than 0.001. A comparison between the initial state (Fig.II)

and the state after 750 timesteps (Fig.III), which is approximately 5 revolutions of the circular vortex, shows how the circumference has changed. Since the amplitudes of a mode m varies as r^m the effect should be more pronounced at larger r values. A boundary particle will be displaced δr from r = R because the hodograph for the present problem is not a circle. δr is initially of order 0.002 if R = 7.2, i.e. $\frac{\delta r}{R} \approx 3\ 10^{-4}$. δr will however increase with time (Section 8) until matching is applied. If this is done every 16 timesteps $\delta\ r/R$ might on average be of order 10^{-3}. After 10^3 timesteps a particle is therefore likely to have arrived at a position which is off by an amount up to 1 mesh cell. Experience with this run, as well as with a number of related simulations, suggests that the code can be run for 500 - 750 timesteps with $\langle \underline{u} \cdot \frac{\Delta t}{2} \rangle = \frac{1}{4}$ before accumulation of errors become too significant. In other words when a particle on average has travelled a distance of the order 2 - 3 period the accumulated errors can be of the order of cell-length. In the example with the circular vortex $\langle \underline{u} \cdot \frac{\Delta t}{2} \rangle = 0.3$, so that $(\underline{u} \cdot \frac{\Delta t}{2})_{max} = 0.6$ at the boundary of the vortex.

Two Circular Vortices

A more complex problem is the flow generated by two finite vortices of the same or opposite polarity, both being initially circular. Depending on parameters such as the relative polarity, radius, separation distance etc, a situation can arise in which the system takes up a stable large-amplitude oscillatory state. Such a case has been simulated by setting up a distribution of vorticity similar to that of D. It is found that, while energy is conserved to within 10^{-4} over 700 - 1000 timesteps, the enstrophy and the energy of the point vortices oscillate with a definite period, the period of the system. This latter effect is due to the finite number of point vortices

12. CONCLUDING REMARKS

Program VORTEX has at present simulated 10 different hydrodynamic problems with a variety of data. These problems are being examined analytically and results generated by the program as well as by a suite of analyser programs are being studied to explain the behaviour of the computer model used. The outcome of this work will be published elsewhere, but it may be mentioned that the following physical situations have been studied:

1. Kelvin - Helmholtz instability.
2. Formation of a Karman Vortex street.
3. Interaction between two circular clouds of vorticity.
4. Simulation of turbulence problems.
5. Kirchoffs elliptic vortex.
6. Mode - mode coupling on the surface of a circular vortex.

The program has also been used by J.B. Taylor and B. McNamara to simulate the diffusion of plasma in two dimensions using the guiding centre model.

PART II. THE COMPUTER CODE

13. INTRODUCTION

This second part is best read in conjunction with the clarified documented listing which is available in the Culham Library, or from the author. Lists of subroutines and variables as well as numbering of routines will not be given here. It is thus assumed that

the reader is familiar with the notation of the program, and names and symbols will be used without any further explanation.

In the following two sections the structure of the program is explained by diagrams which should not need further comments.

Section 16 explains the printed output which is normally produced by the program, while Section 17 explains briefly the input required. Section 18 discusses the techniques used for coding, mainly the Usercode portion of the program (KDF9 assembler language), and Section 20 describes a suite of analyser programs some of which have been clarified. Listings are available in the Culham Library, or from the author.

14. BLOCK STRUCTURE

Broadly speaking program VORTEX can be divided into 8 blocks or sections as shown on the next page apart from Block VIII. These blocks are centred around subroutines with the following names:

I	MAIN
II	F DATA
III	INITAL
IV	CHECK
V	START
VI	COTROL
VII	CONVEC

Block VIII includes several segments.

15. FLOW DIAGRAMS

On the following pages we show the logical structure of each block by means of flow diagrams. On the left are written the names and numbers of the routines concerned, together with the page number of each routine in the clarified listing. Routines are either written in FORTRAN, ALGOL or KDF9 USERCODE. Two segments need further explanation. POT1 is the Hockney-Poisson solver program which is described elsewhere[14]. MOVIE is the high-speed movie-making package written by J.P. Boris and P. Hodges. Neither of these are included in the listing of the program. At the end of the clarified listing there is a subroutine map or 'program tree' which will help the understanding of the flow diagrams.

16. OUTPUT FROM PROGRAM

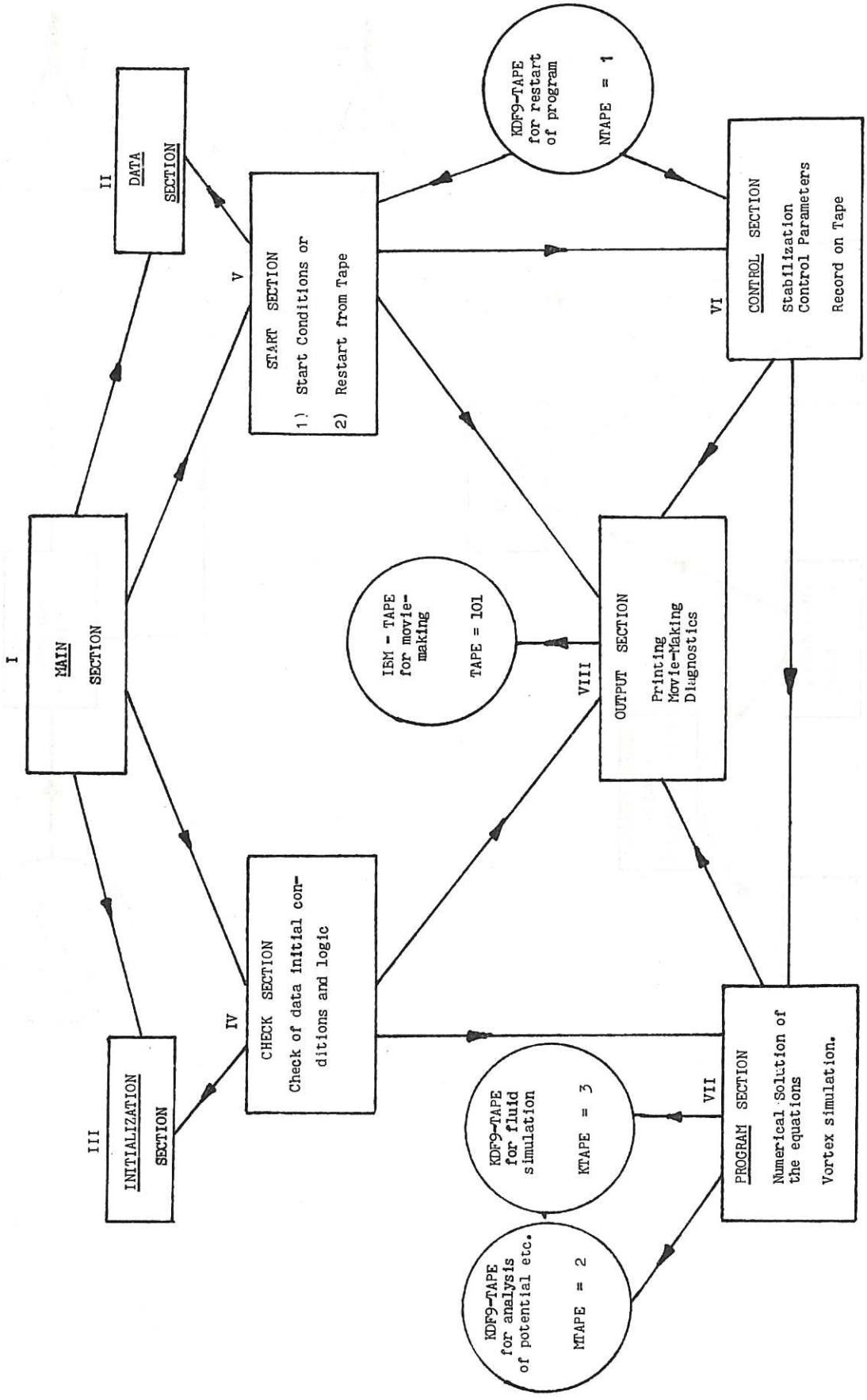
The clarified listing includes a test run with typical output generated by the code. This output can be divided into 2 parts:

I. Diagnostic comments during initialisation, starting, and closing down of a run.

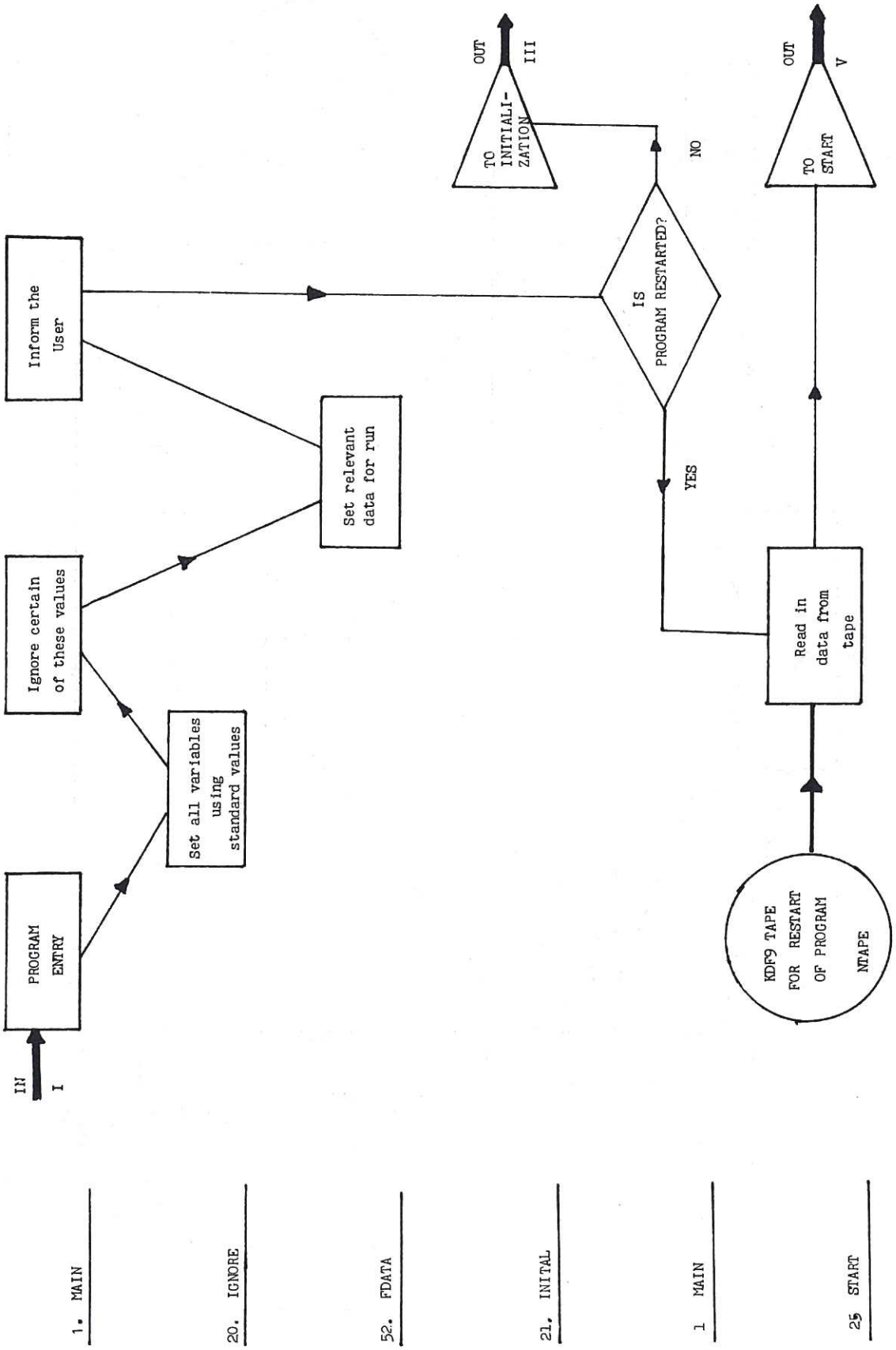
II. Diagnostic results as the simulation proceeds.

Part I consists of labelling the job, checking the number of point vortices and checking the value of the timestep DT. 5 pages with results from initialisation, a test run (automatically performed) and the starting conditions follow, and a page with the values of certain common variables completes the first part of the output.
Part II consists of information which is printed before and after each matching has occurred. This printout is performed by routine MPRINT which also produces the 5 pages mentioned above.

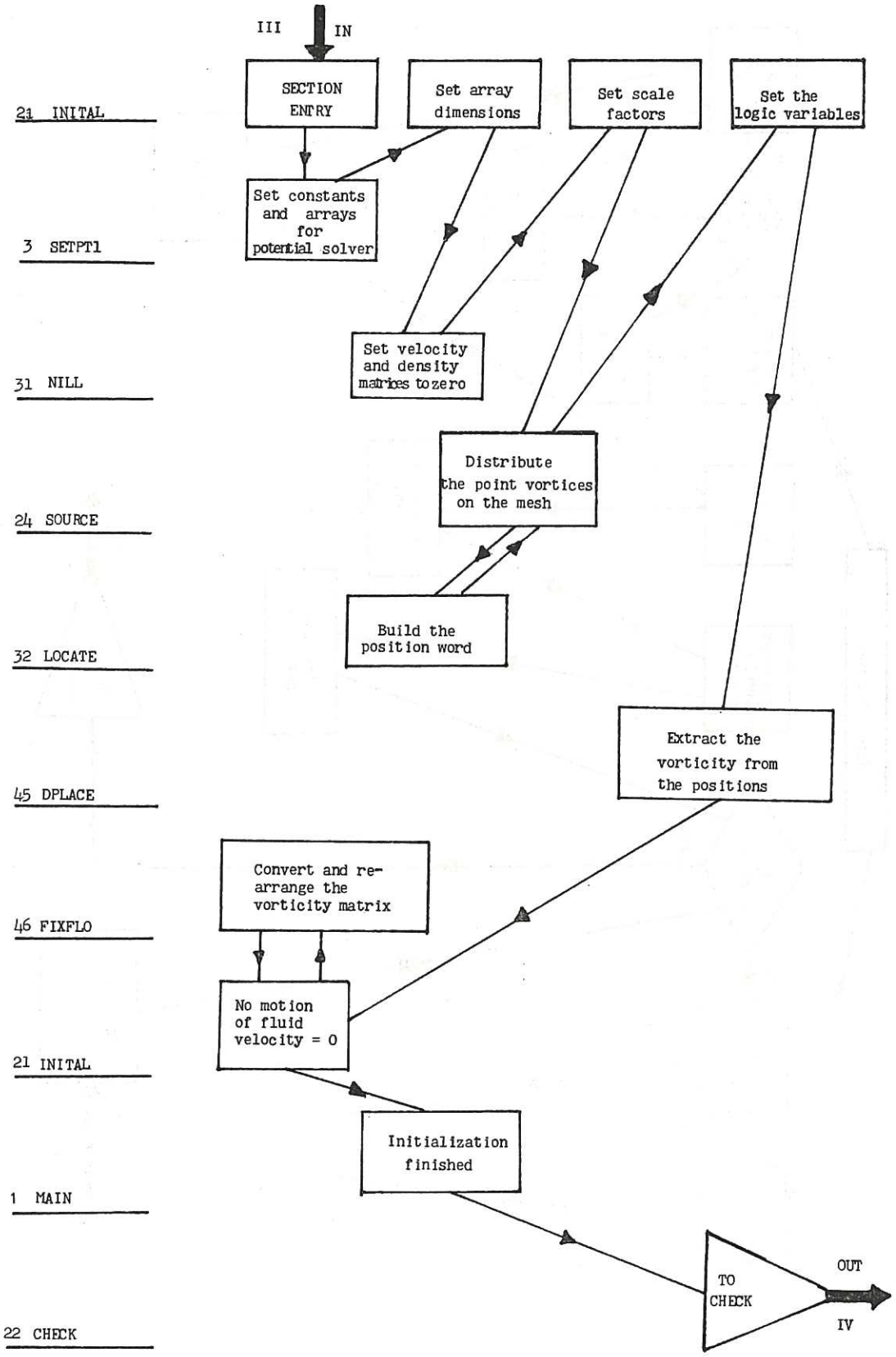
I. OVERALL FLOW DIAGRAM



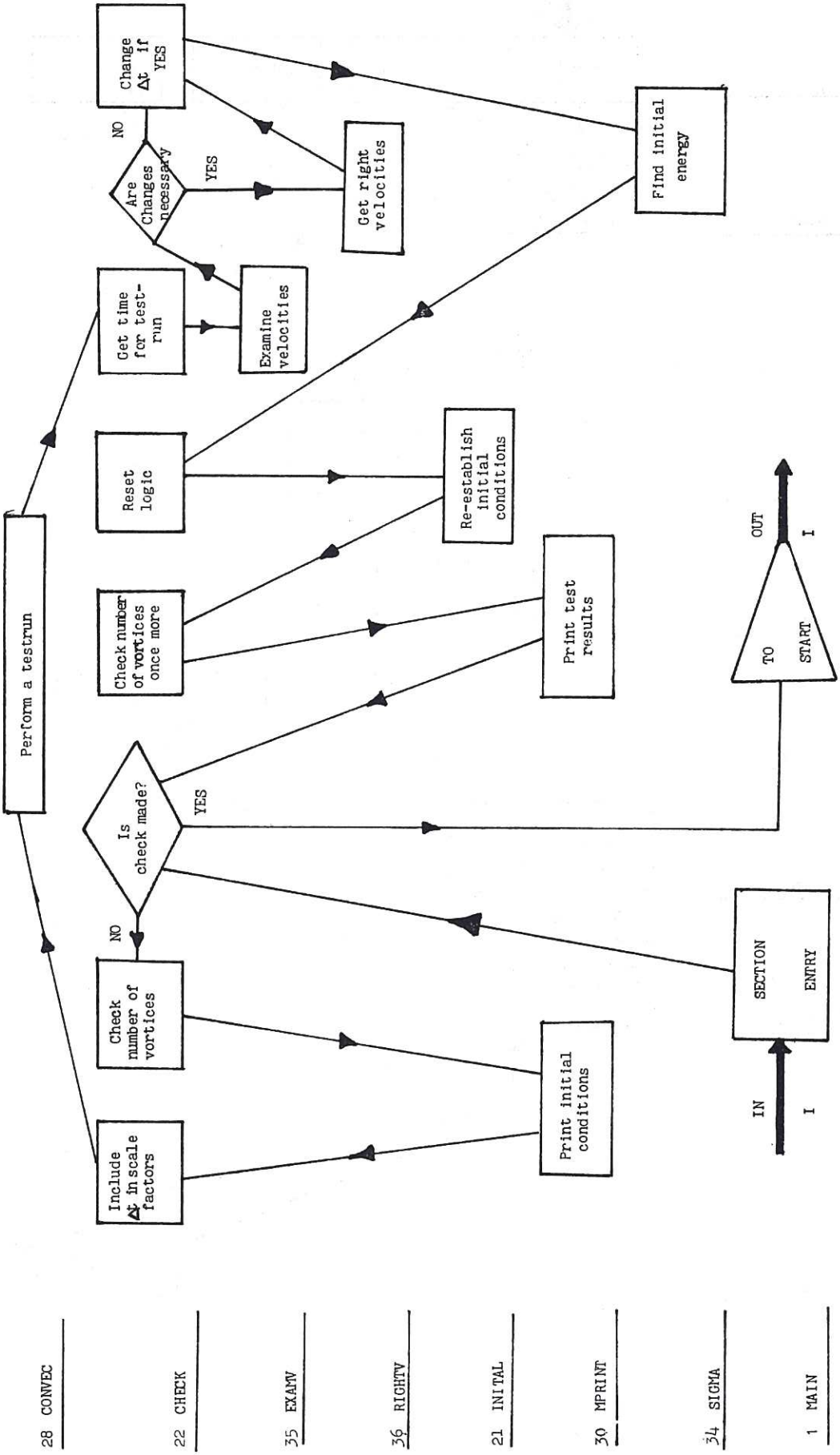
II. DATA


IN
I
PROGRAM ENTRY
Set all variables using standard values
Ignore certain of these values
Inform the User
Set relevant data for run
IS PROGRAM RESTARTED?
NO
YES
OUT
III
TO INITIALI-ZATION
KDF9 TAPE FOR RESTART OF PROGRAM
NTAPE
Read in data from tape
OUT
V
TO START
1. MAIN
20. IGNORE
52. FDATA
21. INITAL
1 MAIN
25 START

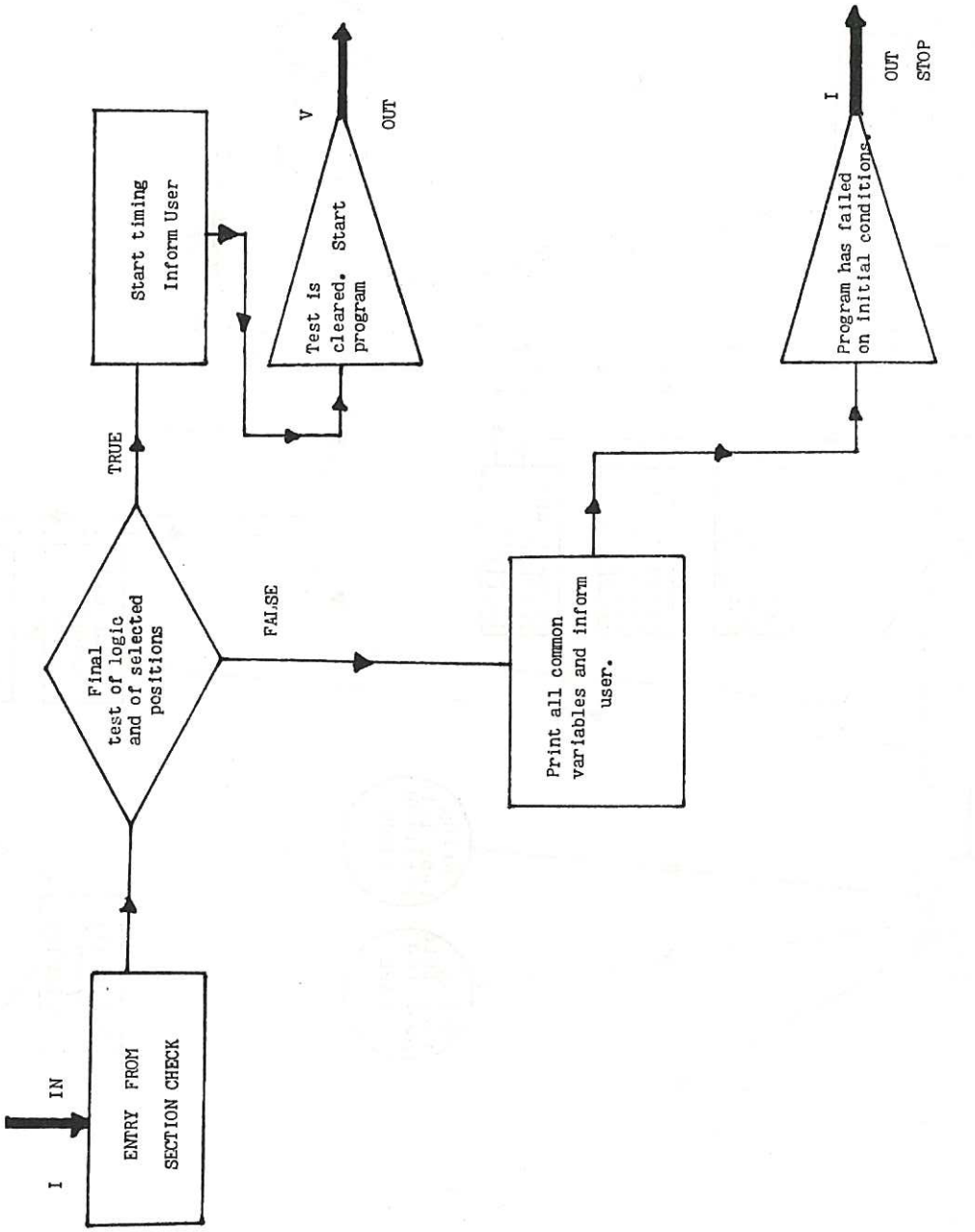
III. INITIALIZATION



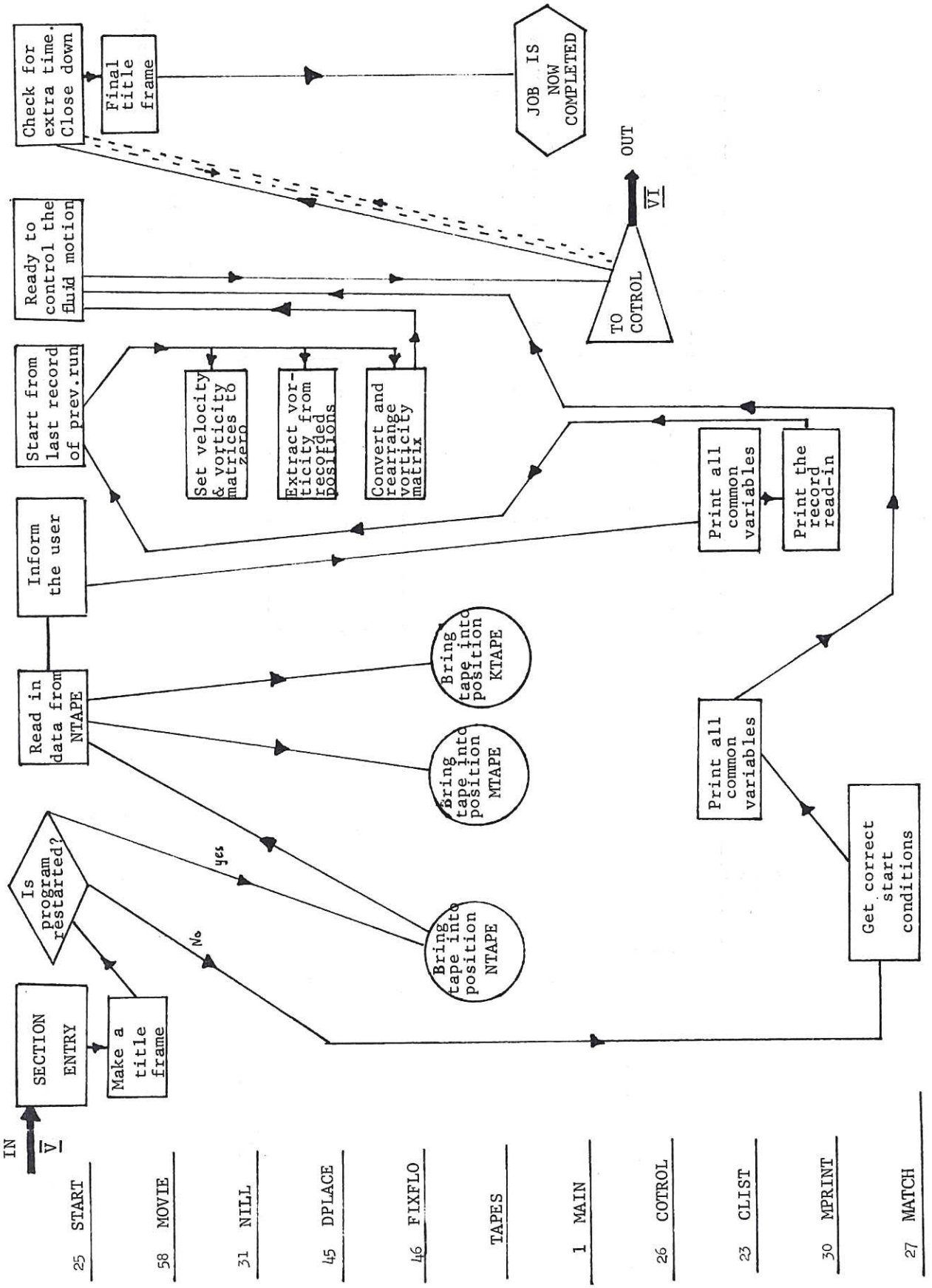
IV. CHECK 'a'


28 CONVEC
22 CHECK
35 EXAMV
36 RIGHTV
21 INITAL
30 MPRINT
34 SIGMA
1 MAIN
IN
I
SECTION
ENTRY
OUT
I
TO
START
Perform a testrun
Include Δt in scale factors
Check number of vortices
Is check made?
NO
YES
Check number of vortices once more
Reset logic
Get time for test-run
Change Δt if YES
Are Changes necessary
NO
YES
Examine velocities
Get right velocities
Find initial energy
Re-establish initial conditions
Print test results
Print initial conditions

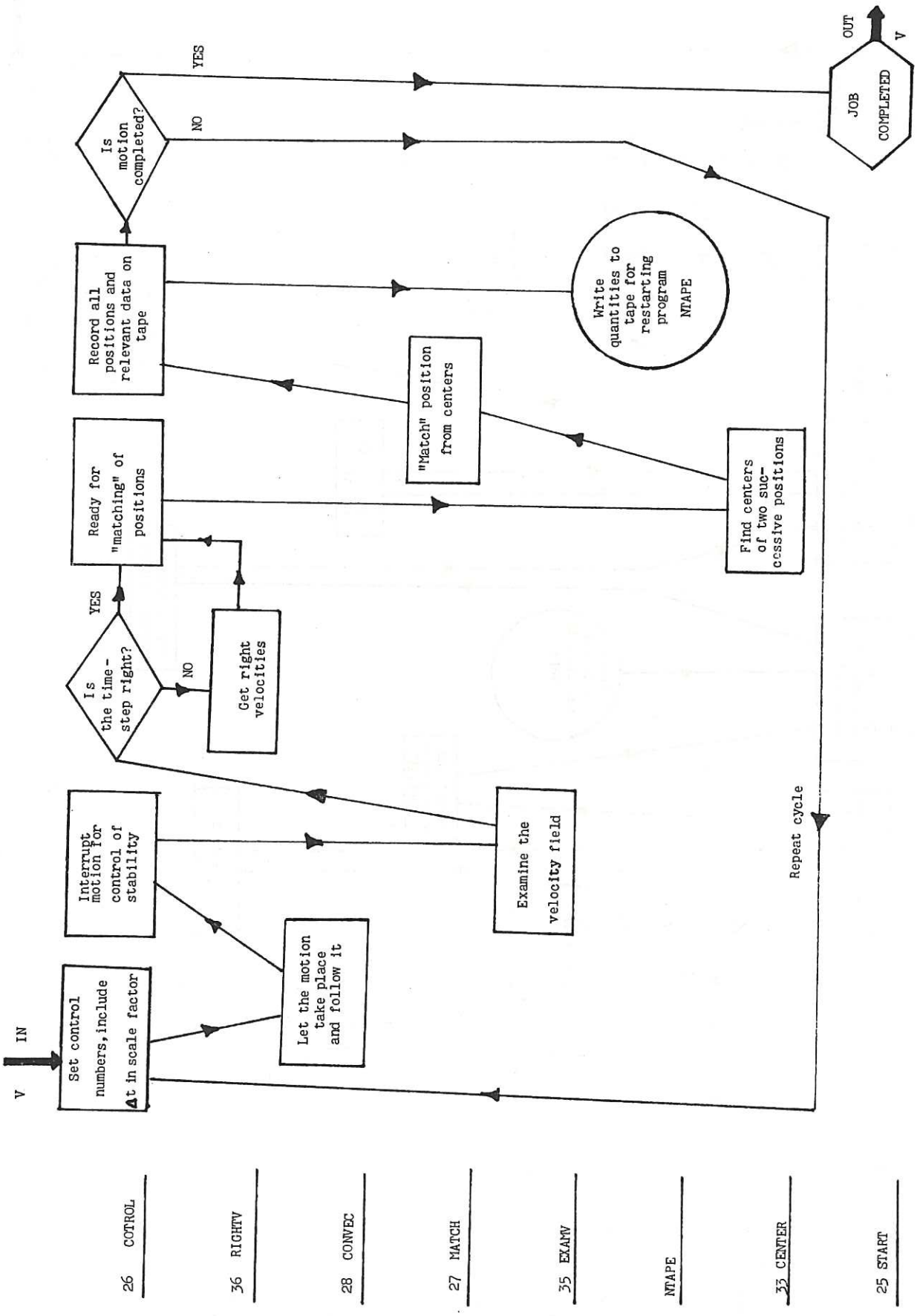
IV. CHECK 'b'


I
IN
ENTRY FROM
SECTION CHECK
Final
test of logic
and of selected
positions
TRUE
FALSE
Start timing
Inform User
V
OUT
Test is
cleared. Start
program
Print all common
variables and inform
user.
I
OUT
STOP
Program has failed
on initial conditions.
1. MAIN
25. START
23. CLIST
EXIT STOP

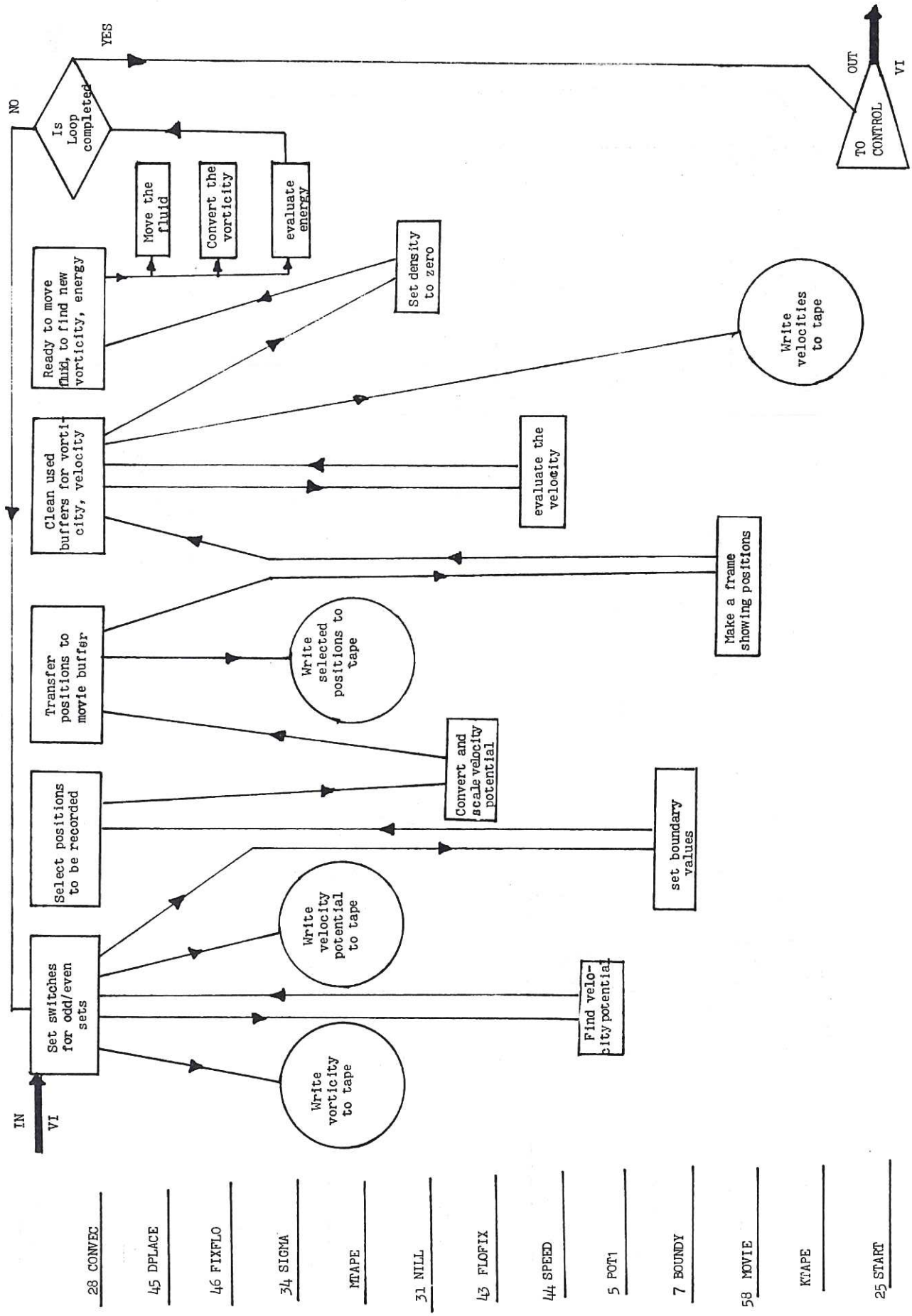
V. START


IN
V
SECTION ENTRY
Make a title frame
Is program restarted?
yes
No
Bring tape into position NTAPE
Bring tape into position MTAPE
Bring tape into position KTAPE
Read in data from NTAPE
Inform the user
Start from last record of prev.run
Set velocity & vorticity matrices to zero
Extract vorticity from recorded positions
Convert and rearrange vorticity matrix
Ready to control the fluid motion
Check for extra time. Close down
Final title frame
JOB IS NOW COMPLETED
OUT
VI
TO COTROL
Print all common variables
Print the record read-in
Print all common variables
Get correct start conditions
25 START
58 MOVIE
31 NILL
45 DPLACE
46 FIXFLO
TAPES
1 MAIN
26 COTROL
23 CLIST
30 MPRINT
27 MATCH

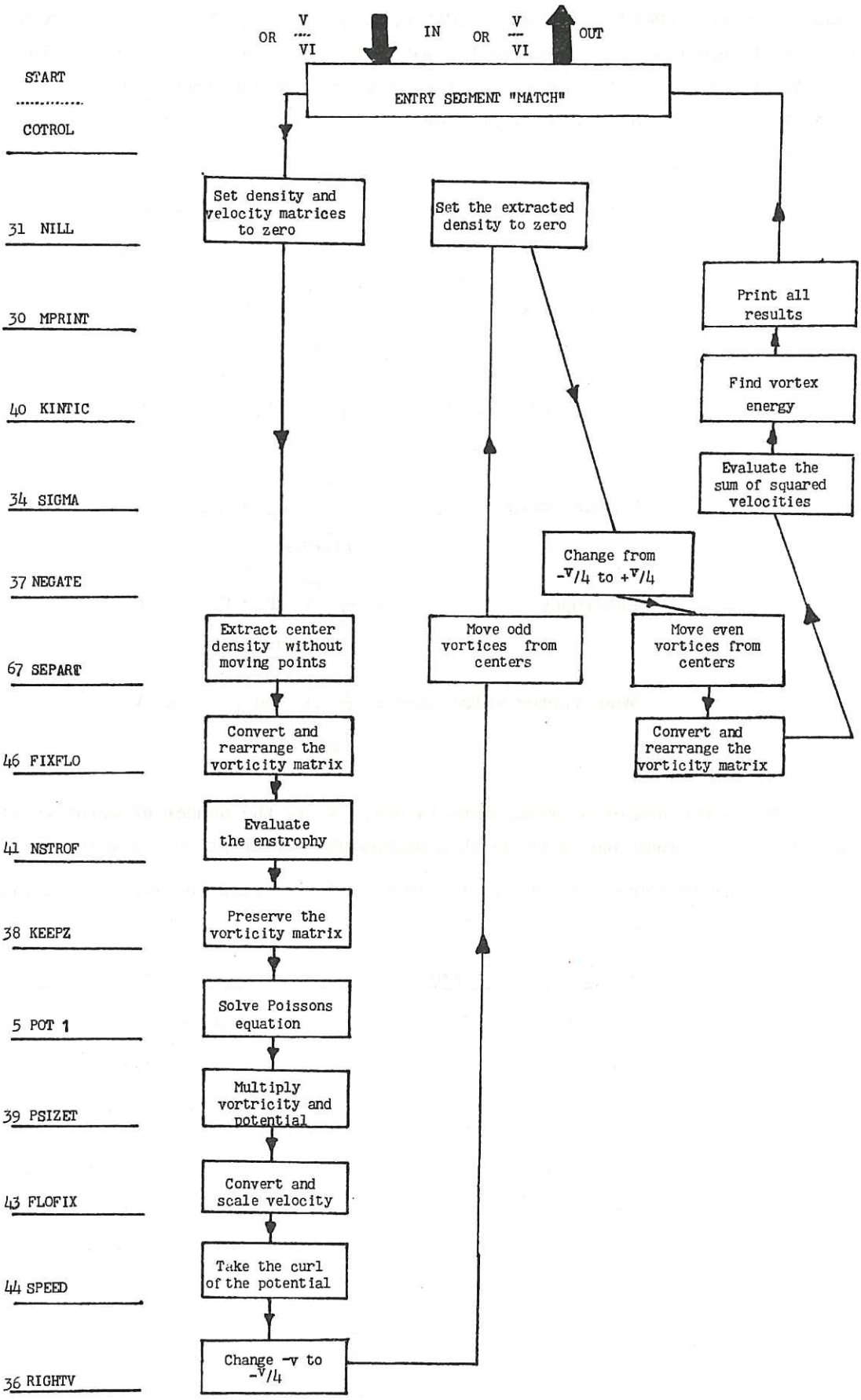
VI. CONTROL


IN
Set control numbers, include Δt in scale factor
Let the motion take place and follow it
Interrupt motion for control of stability
Examine the velocity field
Is the time-step right?
YES
NO
Get right velocities
Ready for "matching" of positions
"Match" position from centers
Find centers of two successive positions
Record all positions and relevant data on tape
Write quantities to tape for restarting program NTAPE
Is motion completed?
YES
NO
JOB COMPLETED
OUT
Repeat cycle
26 CONTROL
36 RIGHTV
28 CONVEC
27 MATCH
35 EXAMV
NTAPE
33 CENTER
25 START

VII. FLUID MOTION



VIII. MATCH


OR V/VI
IN
OR V/VI
OUT
START
COTROL
ENTRY SEGMENT "MATCH"
31 NILL
Set density and velocity matrices to zero
Set the extracted density to zero
30 MPRINT
Print all results
40 KINTIC
Find vortex energy
34 SIGMA
Evaluate the sum of squared velocities
37 NEGATE
Change from -V/4 to +V/4
67 SEPART
Extract center density without moving points
Move odd vortices from centers
Move even vortices from centers
46 FIXFLO
Convert and rearrange the vorticity matrix
Convert and rearrange the vorticity matrix
41 NSTROF
Evaluate the enstrophy
38 KEEPZ
Preserve the vorticity matrix
5 POT 1
Solve Poissons equation
39 PSIZET
Multiply vortricity and potential
43 FLOFIX
Convert and scale velocity
44 SPEED
Take the curl of the potential
36 RIGHTV
Change -v to -V/4

The information comprises the positions of selected odd/even vortices and selected velocities. Printing can be done in octal or decimal format according to whether the common variables LPRINT is .TRUE. or .FALSE. respectively. The fluid energy has been calculated at each timesteps prior to the matching and an array with these values is printed.

After the matching process, the fluid energy, the sum (zeta x psi), mean fluid velocities, vortex energy, enstrophy, and mean vortex velocities are printed. These are calculated as follows:

$$\text{Fluid energy} = \sum_{\text{Mesh}} u_m \, u_m = E_1$$

$$\text{Sum (zeta x psi)} = \sum_{\text{Mesh}} f_m \cdot H_m = E_2$$

$$\text{Mean fluid velocity} = \frac{1}{M} \sum_{\text{Mesh}} u_m = U$$

$$\text{Vortex energy} = \sum_{\text{particles}} u_{pt} \cdot u_{pt} = L$$

$$\text{Enstrophy} = \frac{1}{M} \sum_{\text{Mesh}} f_m \cdot f_m = F$$

$$\text{Mean Vortex velocities} = \frac{1}{N} \sum_{\text{particles}} u_{pt} = V$$

where M is the number of meshpoints (4096), N is the number of point vortices, subscript 'm' indicates a value at a meshpoint, subscript 'pt' a value for a point vortex.

In order to convert to physical values, i.e. to find the scaling, we imagine a system of units as follows:

Quantity	symbol	value	computer model value
length	L	1	Mesh spacing
vorticity	T^{-1}	1	1 vortex particle
density	ML^{-2}	1	Mass of fluid within a cell.

If a particle is meant to simulate a charged particle in a plasma, (electron, ion), then that particle possesses a unit charge which is either positive or negative.

In order to calculate the value of the timestep in physical units we assume that the average velocity of the particles is known denoting this by $\langle \mu \rangle$ (in the right units) we get

$$\Delta t = \frac{\sqrt{L}}{2 \langle \mu \rangle} \frac{M}{N^3}$$

where L is the vortex energy as printed. This relation will give Δt in physical units, which can be compared with the value of Δt in program units.

In addition to the printed output program VORTEX can produce 4 magnetic tapes. 3 of these tapes contain information to be processed by analyser programs. The 4th tape is an IBM-tape which when run on the Benson-Lehner microfilm recorder will produce film or hard-copy showing the motion of the point vortices.

17. INPUT REQUIRED FOR A RUN

The input required by program VORTEX is the vorticity distribution f as approximated by eq.(3.1). This approximation is made by specifying the coordinates (q , p) of each point vortex. All coordinates are initially calculated in floating point form such that $0. \leqslant (q , p) \leqslant 64. 0.$ The (q , p) positions of a point vortex are then packed into one fixed-point word in the array NXY. This array has to be filled up as indicated in Fig.10 assuming the number of point vortices to be NREAL.

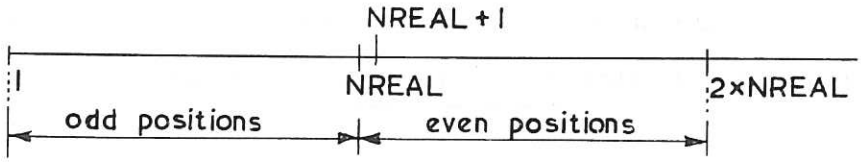


Fig.10.

The array is to be filled with position words starting from the 1st element. Element i and NREAL + i represent the positions of the same physical vortex and initially they are identical. The program checks this as well as the number of positions filled in and if there is an error an abnormal exit will occur.

In the clarified listing as well as in the User Manual (Sec.21) an example is given showing how to distribute points uniformly within a circle. The routine which the user supplies is called SOURCE.

The data for program VORTEX is supplied by the user in a routine called FDATA and the reader is referred to the User Manual.

In case an experienced user wants to change part of the program while maintaining the basic structure, a dummy routine called EXPERT is called for frequently from the routine CONVEC, which does the actual simulation sequence. By introducing his own version of EXPERT, the user is able to control the run or introduce new facilities.

18. PROGRAMMING TECHNIQUES

The most prominent feature of the VORTEX code is its vector integration technique; integer arithmetic has been used wherever possible. The (q , p) components of the positions and velocities are packed into one word NXY as illustrated (NPSI is the velocity word).

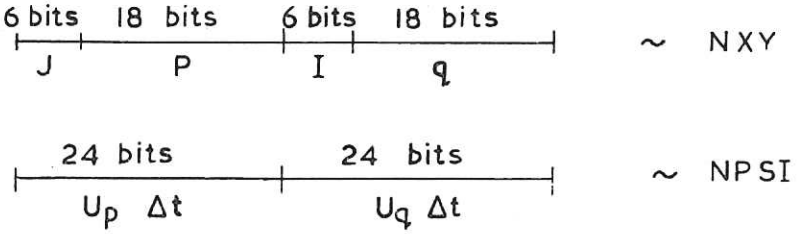


Fig.11.

With this structure most of the coding is done in KDF9 usercode. The 6 bits in a position word containing the mesh coordinate are easy to extract by shift operations. The most obvious advantage of this packing is the simultaneous time integration of (q, p) by adding the two words shown in the figure.

The potential-solver program POT1 works in floating point form, however, so that conversion routines FIX FLO, FLO FIX are used as indicated in the structure diagrams.

A careful study of the execution times for different instructions in KDF9 usercode has led to optimum design of the high speed loops in the routines which perform the time integration.

The next section briefly describes how a relatively simple routine is coded.

19. EXAMPLES OF OPTIMUM CODING

Suppose that the $2N$ positions of the odd/even point vortices are stored in an array called NXY. The first half of NXY contains N values of $\underline{r}_o$, the second values of $\underline{r}_e$.

Suppose that we wish to find $\underline{r}_C = \frac{1}{2}(\underline{r}_e + \underline{r}_o)$ the socalled centre position encountered in Section 9.

The following three criteria have been set for this subroutine (CENTER):

1. Although we write $\underline{r}_C = \frac{1}{2}(\underline{r}_e + \underline{r}_o)$ it is necessary to calculate $\underline{r}_C = \frac{1}{2}\underline{r}_e + \frac{1}{2}\underline{r}_o$. Referring to Fig.11 we see that if $IY_{o,e} \geqslant 32$ the y-interval will be lost because of over flow if addition is done before division.

2. Furthermore as each component has 24 bits we must make sure that division does not involve "flow" of bits from Y to IX.

3. To optimise the speed we examine the actual times for KDF9 instructions. Integer addition or division take 1 or 47 μsec respectively if done in the nesting store. A shift instruction (~ x 2" or /2") takes 5-6 μsec. A loop which scans over all N = NREAL point vortices must be as fast as possible.

The routine which performs this calculation is called CENTER. When the call occurs the addresses of its 2 arguments are left in the top two cells of the KDF9 nesting store, N1 and N2.

CENTER (NXY(1), NREAL)

```
= M15,      (Put address of NREAL in modifier of Q15      )
MOM15,      (Bring value of NREAL to nesting store N1     )
DUP,        (Make a duplicate in N2                       )
= RC15,     (Put NREAL in counter of Q15 and 1 in         )
            (incrementer, 0 in modifier                   )
= RM12,     (Put the other value of NREAL in the          )
            (modifier of Q12                              )
DUP,        (Make a duplicate of the address of           )
            (NXY(1) in N2                                 )
= M15,      (Put one copy of the NXY(1) in the modifier   )
            (of Q15                                       )
= + M12,    (Add the other copy of the address of MXY(1)  )
            (to the current contents in the modifier of Q12)
```

This section of code takes about 55 μsec and it is only executed once for one call of CENTER.

It results in the following contents of the Q-stores

	Counter	Incrementer	Modifer
Q12	0	1	Address of NXY(1+NREAL)
Q15	NREAL	1	Address of NXY(1)

THE LOOP IN CENTER

Execution time in μsec	Command	Number of used cells in nesting store	Explanation
9	1,MOM12,	1	Bring value of NXY(JE) (evenvortex) to top cell of nesting store
2	DUP,	2	Make a duplicate of this value
9	MOM15,	3	Bring value of NXY(JO) (odd vortex) to top of cell of nesting store
1	- ,	2	Make NXY(JE) - NXY(JO) ~ N1
5	SHA-1,	2	Shift the bit pattern in N1 arithmetic 1 bit down preserving the sign bit
7	SHC-23,	2	Shift the bit pattern in N1 (cyclic) 23 bits, i.e. y-part into x-part and vice-versa
5	SHA-1,	2	Shift the bit pattern in N1 arithmetic 1 bit down, i.e. x-part/2
6	SHC-24,	2	Shift the bit pattern in N1 (cyclic) 24 bits, i.e. y-part/2 into old position and x-part/2 into old position
1	- ,	1	Form NXY(JE) - (NXY(JE)-NXY(JO))/2 This gives $\underline{r}_C$
2	DUP,	2	Duplicate the value of $\underline{r}_C$
10	= MOM12Q,	1	Store this value and update address to NXY(JE + 1)
10	= MOM15Q,	0	Store this value, update address to NXY(JO + 1) and subtract 1 from counter in Q15
11	J1C15NZ,	0	Jump to label 1 if counter in Q15 is not zero
	VR,	0	Reset overflow register.
	EXIT1,END,	0	Return to the calling routine

We notice that the total execution time for the instructions in this loop is 78 μsec for one vortex particle. (0.4 sec ~ for 5000 particles).

The most "expensive" commands are the fetches and stores.

The 4 shift instructions are necessary because the first shift, SHA-1, may result in the flow of a non-zero bit from Y to IX.

The number of instructions words for this loop is $3\frac{4}{6}$, $\frac{4}{6}$ meaning 4 syllables out of the 6 syllables in one word. (1 syllable = 8 bits). Written in FORTRAN or ALGOL the same code would be rather complicated, time consuming, but perhaps easier to interpret.

20. SUITE OF ANALYSER PROGRAMS

The analyser programs developed so far are:

A. MODANA, mode analyser program
B. SPECTRUM, spectrum values of matrix
C. TELESCOPE, examine Vortex structures under magnification
D. FERMI, statistical properties
E. MARKER, exhibit the fluid motion
F. SURFACE, 3-D surface plots

MODANA examines m-modes excited on the surfaces of a vortex. It produces either phase-plane diagrams or growth rates.

SPECTRUM evaluates the frequency distribution of a mesh function and is meant to explain clustering phenomena occurring during simulation.

TELESCOPE magnifies the pictures produced by MOVIE thus enabling the user to see how a given distribution develops in time.

FERMI evaluates the functional relationship between f and H, and is used for examining the statistical properties of the computer model.

MARKER shows the motion of the physical fluid by introducing dummy particles moving with the velocities calculated by VORTEX.

SURFACE was written by Dr J.P. Boris and is able to produce pictures of 3-D surfaces for either f or H.

21. USERS MANUAL FOR PROGRAM VORTEX

Contents of Users Manual

21.1 Input Deck

The input deck for the Culham KDF9 is quite simple. It can be divided into 3 groups of cards:

I. Control cards

II. Fortran cards which formulate the problem

III. Data cards, (including Fortran data)

In order to illustrate this an example is given below, in which we assume that the problem to be solved is that of a single finite circular vortex placed in the centre of a square region, the boundaries of which have no normal component of the fluid velocity. To set up a circular region of constant vorticity we distribute point vortices in equidistantly spaced rings.

In the job deck listed below all cards underlined are necessary for any problem, that is they are independent of our particular example.

GROUP I

```
1.   * JOB .....
2.   * IOD TAPE 1/COMMON/SAVE * FOR USER
3.   * IOD TAPE 2/  "  /  "      "   "
4.   * IOD TAPE 3/  "  /  "      "   "
5.   * SUBSTITUTE VORTEX  //  CHRIS J
6.   * SUBSTITUTE DELSQPHI  //  CP GROUP
```

GROUP II

```
7.   * IDENTIFIER SOURCE
8.   * FORTRAN
9.   x SUBROUTINE SOURCE
10.  * SUBSTITUTE VORCOM
11.      IODD = NINDXY
12.      DR = 0.3
13.      NDR = 24
14.      NPR = 10
15.      XC  = 32.0
16.      YC  = 32.0
17.      DO 102 JR = 1, NDR
18.      R = FLOAT (JR) * DR
19.      NTHETA = JR * NPR
20.      DTHETA = 2.0 * PI/FLOAT (NTHETA)
21.      DO 101 JJ = 1, NTHETA
22.      THETA = FLOAT (JT) * DTHETA
```

```
23.             X = XC + R * COS (THETA)
24.             Y = YC + R * SIN (THETA)
25.         CALL LOCATE (X,Y, NXY(1), IODD)
26.         IEVEN = IODD + NREAL + 1
27.         NXY (EVEN) = NXY (IODD + 1)
28.     101 IODD = IODD + 1
29.     102 CONTINUE
30.         NI NDXY = IODD
31.         RETURN
32.         END
```

GROUP III

```
33.     *   IDENTIFIER FDATA
34.     *   FORTRAN
35.     *   SUBROUTINE FDATA
36.     *   SUBSTITUTE VORCOM
37.         NRUN = 192
38.         RETURN
39.         END
40.     *   DATA
41.     T.H.I.S.  T.H.E.  L.A.B.E.L.L.I.N.G.  O.F.
42.     A  S.P.E.C.I.A.L.  T.E.S.T.  R.U.N.  T.O.
43.     D.E.M.O.N.S.T.R.A.T.E.  T.H.E.  U.S.E.
44.     O.F.  L.A.B.E.L.S.
45.     R.U.N.  1.2.(3).2  9./.1.2./1.9.6.9.
46.     *   END JOB
```

21.2 "Default mode" as opposed to "users mode"

The inclusion of cards 2-4 and 7-39 constitutes a deviation from the default mode, which comprises for the three parts:

I. Cards 1,5,6 SUPPLIED BY USER

II. Cards 7,8,9, 31 and 32. DUMMY SOURCE

III. Cards 33,34,35, 38 and 39. DUMMY FDATA
Cards 40-46. SUPPLIED BY USER

The program initially sets all common variables to default values (section 21.7), some of which are overwritten by the user in Groups II and III. Unless this is done, the default mode runs without any formulation of the problem to be solved, and so abnormal exit from the program will occur.

It is thus necessary to deviate from the default mode. In the following three sections we shall briefly describe the consequences of the deviations which have been made in our example. (DM "Default Mode").

21.3 Comments on Group I: "Control Cards"

A. Card 1 is an obvious necessity

B. Cards 2-4 supply the program with three magnetic tapes, because DM requires this.

C. Card 5 calls in the following cards:

```
47.   *   SUBFILE FLUID  //  CHRIS J
48.   *   SUBFILE OF LIBRARY  //  CP GROUP
49.   *   SUBFILE EG 3 LIBRARY
50.   *   SUBFILE CUL LIB
51.   *   TABLES
52.   *   CARD LIST
53.   *   STORAGE /6912
54.   *   IODTAPE 101/IBM-SC/SAVE
55.   *   XEQ
56.   *   PRELUDE
57.   *   RLB * ..... 3/12/1969 .. PRELUDE
58.   *   CHAIN 1
59.   *   RLB * ..... 3/12/1969 ... DUMMY. MAIN. PROGRAM
```

D. Card 6 calls in the Hockney-Poisson solver program.

21.4 Comments on Group II: "Formulation of Problem"

In the example discussed here, the particle positions are defined in loops. No matter how the user formulates the problem, the card 25 is used to define each single set of coordinates (x, y) in floating point form, with $0.0 \leqslant (x, y) \leqslant 64.0$, and cards 26, 27 and 28 must follow immediately after.

21.5 Comments on Group III: "Data"

In this example we have accepted all DM values except NRUN, the number of timesteps.

On cards 41-45 we have labelled our run by symbols starting in column 1 and proceeding in every other column until column 55.

21.6 Epilogue

Because we have accepted DM data it seems appropriate to explain what this is. In the following section the name of a variable, its significance and its DM value are shown. For logical variables the value .TRUE. causes action, the value .FALSE. causes the opposite. The table also shows the range of values;

A-B meaning that the variable varies continuously from A to B.

I1/12/13 meaning that the variable varies discontinuously from I1 to I2 in steps of I3.

Blank meaning no variation and

\+ meaning arbitrary variation.

21.7 TABLE OF DEFAULT VALUES

NAME	SIGNIFICANCE	RANGE	DEFAULT VALUE
DT	Value of delta t	+	1.0
HX	Mesh spacing in x	+	1.0
HY	Mesh spacing in y	+	1.0
IBCX	Parameter for boundary conditions in x	1/3/1	1
IBCY	Parameter for boundary conditions in y	1/3/1	1
KTAPE	Channel number of tape containing the velocities		3
MTAPE	Channel number of tape containing the potential		2
MTRY	Frequency of printing/matching	2/32/2	16
MVELOC	Number of bits representing V (average)	1/23/1	17
NHDPY	Frequency of hardcopy production	2/NRUN/1	16
NMINUS	Number of negative vortices	0/3200/1	0
NP1	Starting point of position printing	1/3199/1	1
NP2	End point of position printing	2/3200/1	8
NP3	Increment between NP1 and NP2	1/3200/1	1
NREAL	Total number of point vortices	1/3200/1	3200
NRUN	Number of timesteps to be done	1/200/1	2
NSKIP	Number of blocks to be skipped for restart of program from NTAPE	2/ /1	0
NTAPE	Channel number of tape for restarting program		1
NV1	Starting point of velocity printing	1/4224/1	1820
NV2	End point of velocity printing	2/4225/1	1827
NV3	Increment between NV1 and NV2	1/4225/1	64
POTCON	Potential value on boundaries	+	0.0
ZAD	Switch for adjusting the charges	0.0/1.0/1.0"	0.0
ZFAC	Scale Factor	+	64.0
LANA	Potential/vorticity to MTAPE		.TRUE.
LBEGIN	Start from initial conditions		.TRUE.
LCOPY	Make hard copies		.TRUE.
LFILM	Make film		.TRUE.
LMATCH	Stabilise the numerics		.TRUE.
LPRINT	Print diagnostic results in octal or decimal		.TRUE.
LRECOR	Record history on NTAPE		.TRUE.
LTROL	Control DT		.TRUE.
LVELOC	Eulerian velocities to KTAPE		.TRUE.

21.8 Comments on table

I. The two magnetic tapes MTAPE and KTAPE contain information dumped from core during simulation. They are meant to be used by a suite of analyser programs.

II. The tape NTAPE contains all the information required for a restart of the program from a problem time T_p

where $T_p = N * MTRY * MTRY * DT$, N arbitrary

At time T_p the tape NTAPE has recorded 2N blocks. A restart from time T_p is thus implemented by the following two cards (if N is say 30).

```
LBEGIN = FALSE
NSKIP  = 2 * 30
```

If a tape is not required the appropriate control card can be omitted.

III. The variables IBCX and IBCY determine 1 set out of 9 possible sets of boundary conditions in (x , y) according to:

IBCX = 1	VX = 0	at x-boundaries
IBCX = 2	VY = 0	at x-boundaries
IBCX = 3	(VX,VY)	are periodic in X,

and similarly for IBCY at the Y-boundaries.

IV. The variable DT is automatically changed by the program according to the value of MVELOC, if LTROL = .TRUE.
With LTROL = .FALSE. the value of DT is not changed. MVELOC is equal to the number of bits occupied by the average velocity, that is 17 ~ ¼, 18 ~ ½, 19 ~ 1 etc.

V. The indexes NP1, NP2, NP3, NV1, NV2, NV3 can be set by the user according to which particles/which meshpoints he wants to examine with diagnostics.

VI. POT CON is the potential value at the boundaries. It is only required if either IBCX or IBCY or both are equal to 1.

VII. ZAD is a switch for adjusting the charges in double periodic geometry in such a way that the total charge is zero. ZAD is normally equal to 0.0, but if IBCX = IBCY = 3 (doubly periodic geometry) and if NMINUS ≠ NREAL/2, then ZAD must be set to 1.Ø by the user.

21.9 Users control of program VORTEX

The users control of a particular simulation consists of 3 distinct parts:

A. He has to define the problem concerned by setting up a distribution of point vortices in routine SOURCE (Section 21.1 and Section 21.4).

B. He has to provide the program with data relevant to his problem in routine FDATA (Section 21.5 and Section 21.7).

C. He can if he wishes interrupt the simulation cycle at 14 stages in the program, at which a routine EXPERT(I) is called. EXPERT(I), I = 1,14 is a dummy routine in the DM mode, but the user can write

his own version, allowing for the appropriate entries. EXPERT(I) is called at stages 1-9 from subroutine CONVEC, at stage 10 from subroutine START (Section 2.6) and at stages 11-14 from subroutine MATCH.

21.10 Conclusion of manual

Timing shows that a timestep performed by program VORTEX when run with appropriate DM - DATA takes approximately 20 secs. By cutting out all options this figure can be reduced to 12 secs. Relocation, initialisation and implementation of 300 timesteps for a typical run with appropriate DM - DATA takes 70 minutes.

22. CONCLUSION

A number of different goals have been aimed at in project VORTEX. In order of importance, these are briefly:

I. Production of physical results

II. Experience in programming and numerical technique

III. Documentation

It is left to the reader to judge the success of III and the author will welcome criticism of this topic.

ACKNOWLEDGEMENTS

This report was originally written in June 1969. Because substantial improvements to the VORTEX code have been made since, it was decided to publish the report after results had been obtained.

The Author is grateful to Dr K.V. Roberts who is the originator and supervisor of the project. The Author would also like to thank Dr Roberts and Mr B. McNamara for reading this report and suggesting some modifications.

REFERENCES

1. TRUESDELL, C., The Kinematics of vorticity. Indiana University Press (1954).
2. LAMB, H., Hydrodynamics. 6th Ed. Cambridge University Press (1932).
3. CHANDRASEKHAR, S., Hydrodynamics and Hydrodynamic stability. Oxford University Press (1961).
4. LIN, C.C., The theory of hydrodynamic stability. Cambridge University Press (1955).
5. FROMM, J.E., HARLOW, F.H., Phys. Fluids, 6, 975 (1963).
6. ABERNATHY, F.H., KRONAUER, R.E., J. Fluid Mech., 13, 1 (1962).
7. BERK, H.L., ROBERTS, K.V., Phys. Fluids, 10, 1595 (1967).
8. De PACK, D.C., J. Electronics Control, 13, 417 (1962).
9. DORY, R.A. J. Nuclear Energy (Pt.C), 6, 511 (1964).
10. ONSAGER, L., "Nuovo Cim. Suppl. VI, 130 (1949).
11. CHRISTIANSEN, J.P., ROBERTS, K.V., Proc. of Computational Physics Conference Culham Laboratory, paper 40. (1969).
12. CHRISTIANSEN, J.P., Roberts, K.V., Proc. of Computational Physics Conference Culham Laboratory, paper 52. (1969).
13. HOCKNEY, R.W., J.A.C.M., 12, No.1, (1965).
14. CHRISTIANSEN, J.P., HOCKNEY, R.W., Delsqphi, A. 2-dimensional Poisson solver program. Computer Physics Communications (to be submitted).
15. LYNDEN-BELL, D., M.N.R.A.S. 136, 101, (1967).
16. MORIKAWA, G.K., J. of Meteorlogy, 17, (1960).
17. TRUESDELL, C., TOUPIN, R.A. The classical field theories. Handbuch der Physik, vol.III/1 Springer Verlag, Berlin 1960.
18. BERK, H.L., NIELSEN, C.E., ROBERTS, K.V., Phys. Fluids. 13, 980, (1970).

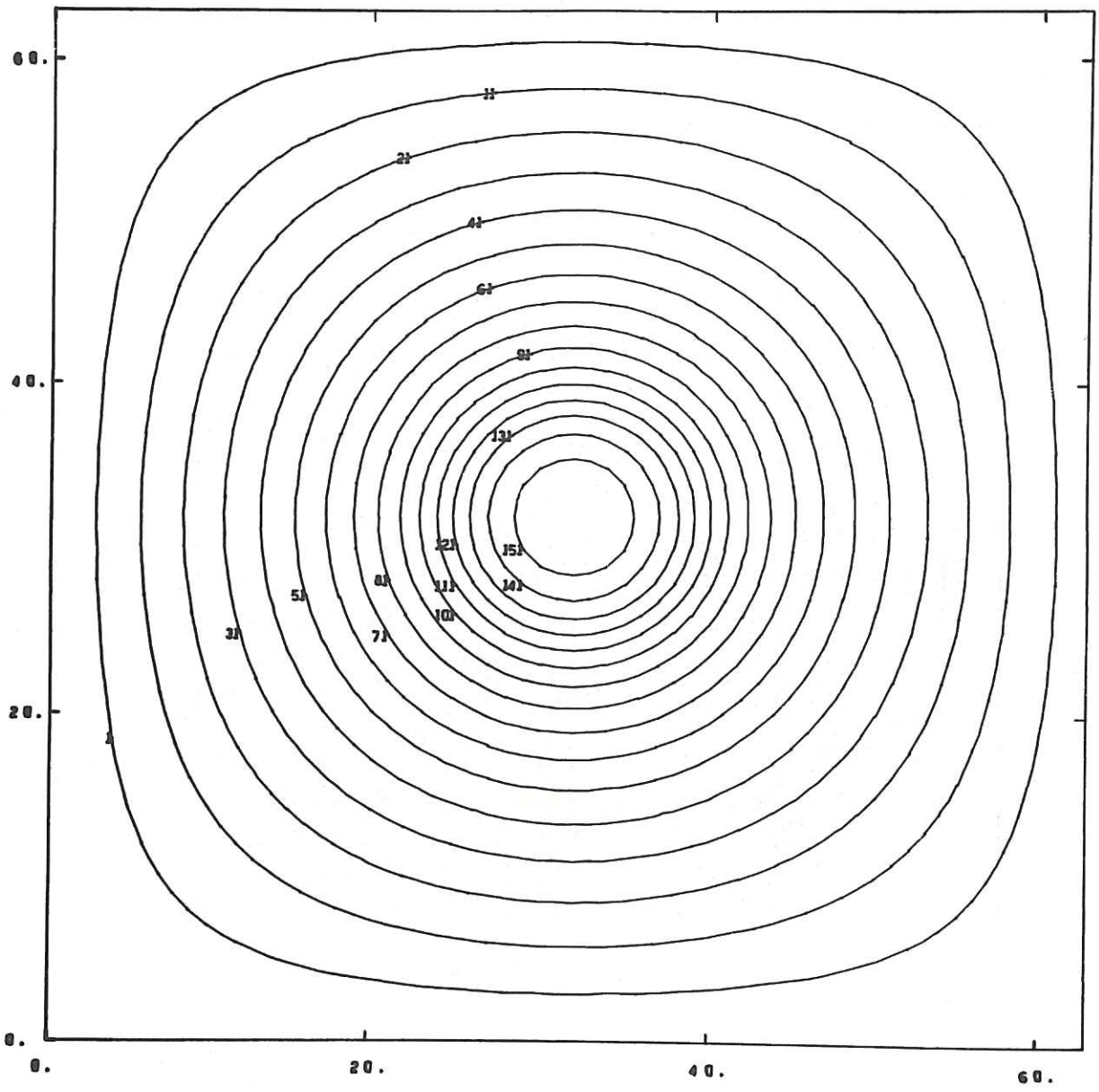


Fig. I Streamlines arising from a single circular vortex. Mesh size 64 x 64

CLM-R106

THE PICTURE IS PRODUCED BY PROGRAM TELESCOPE

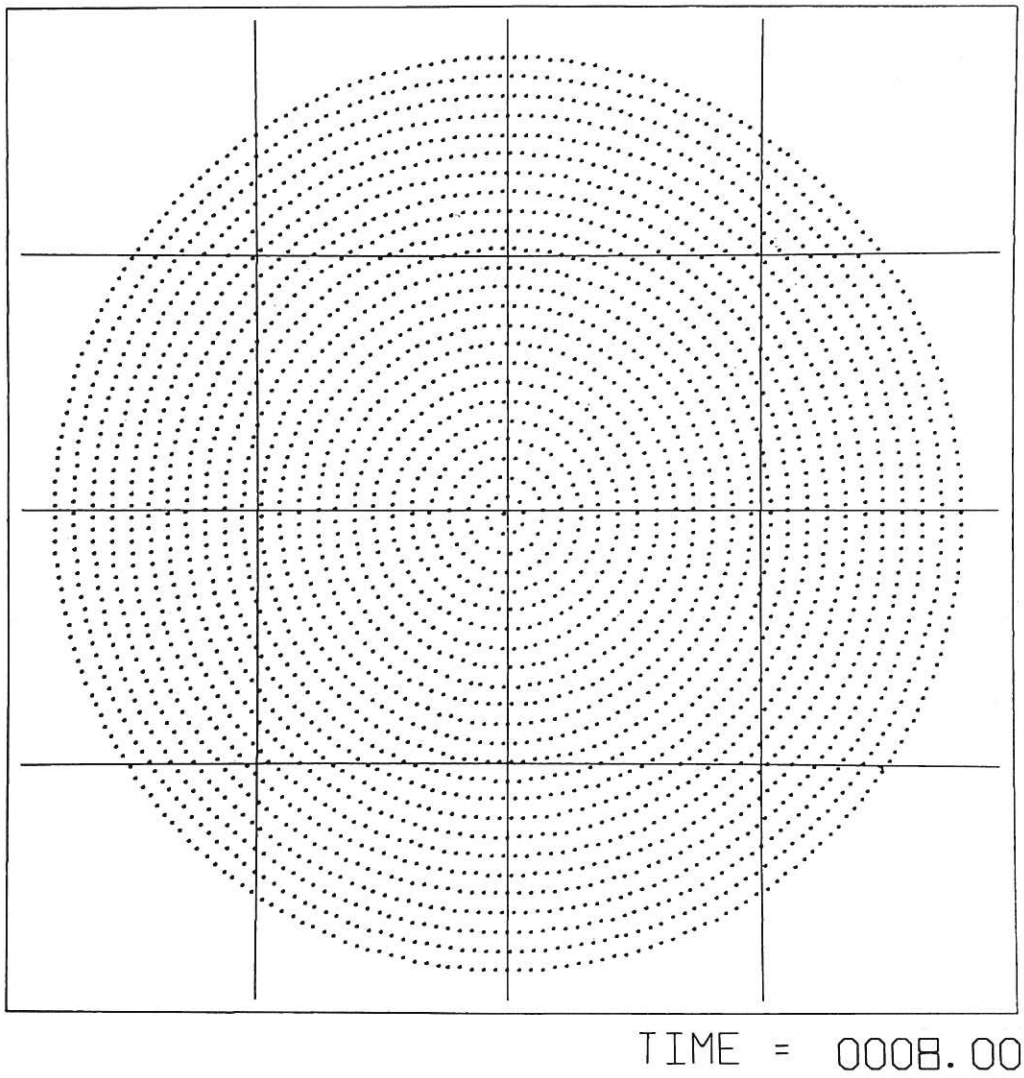


Fig. II Single circular vortex made up of point vortices. Mesh size 16 x 16

THE PICTURE IS PRODUCED BY PROGRAM TELESCOPE

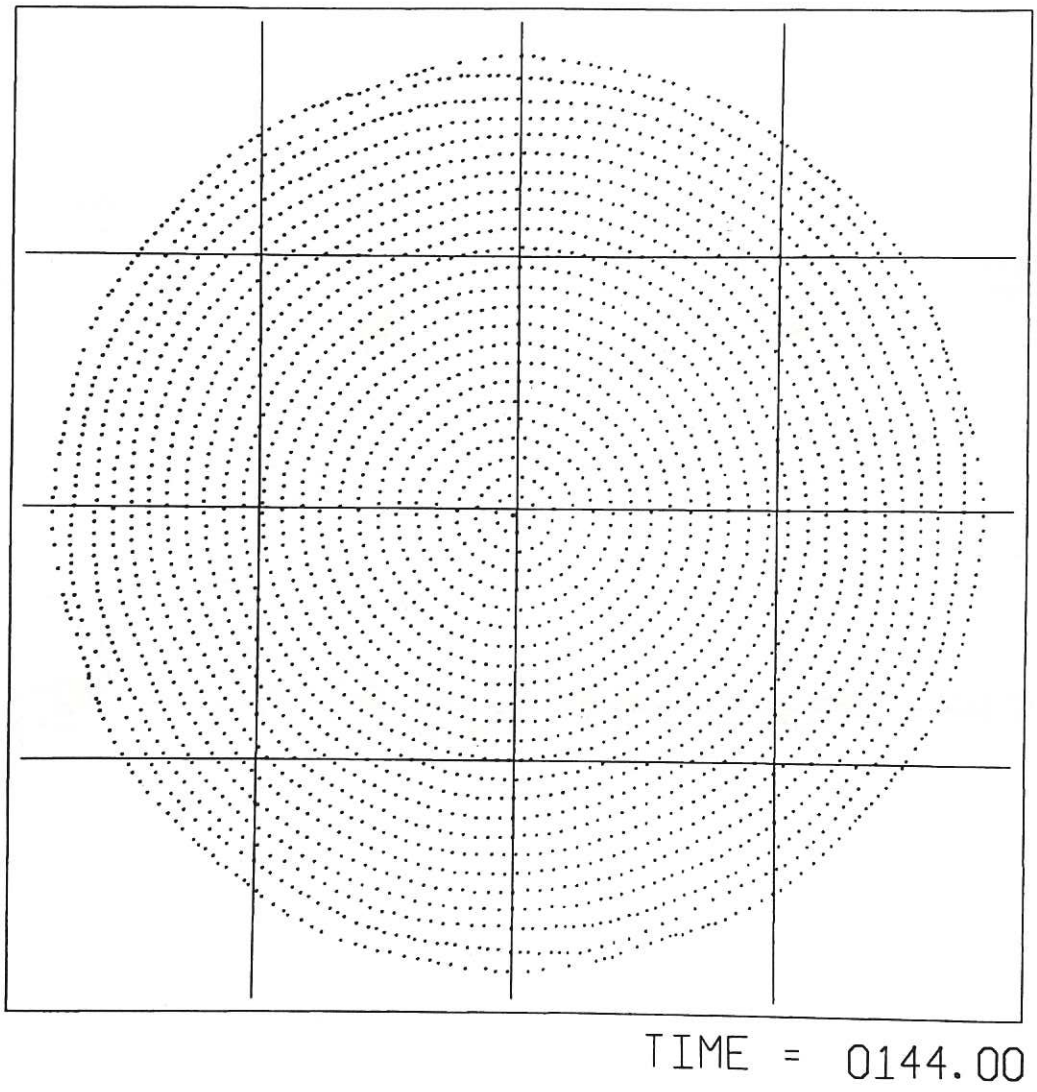
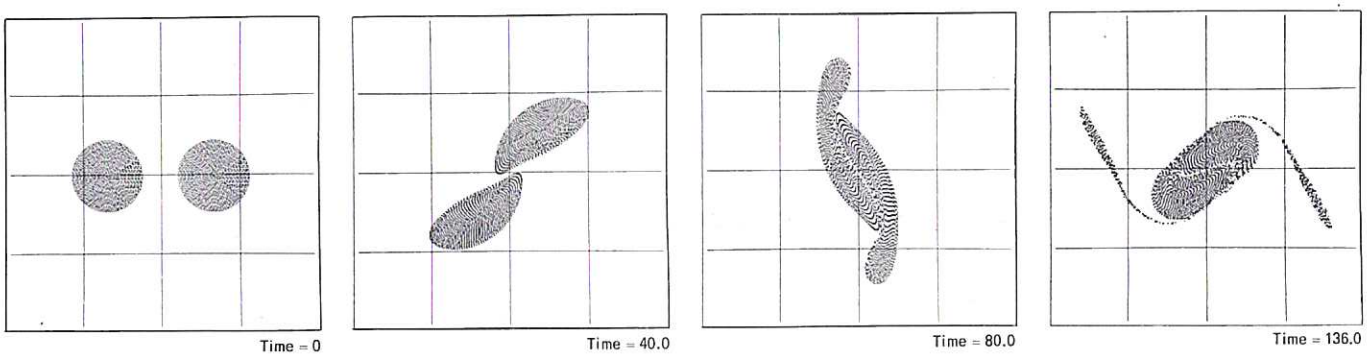
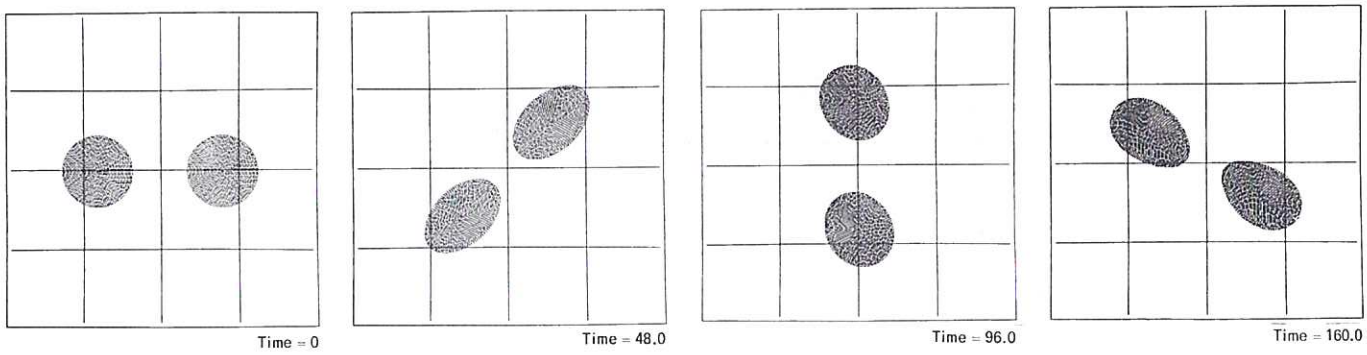


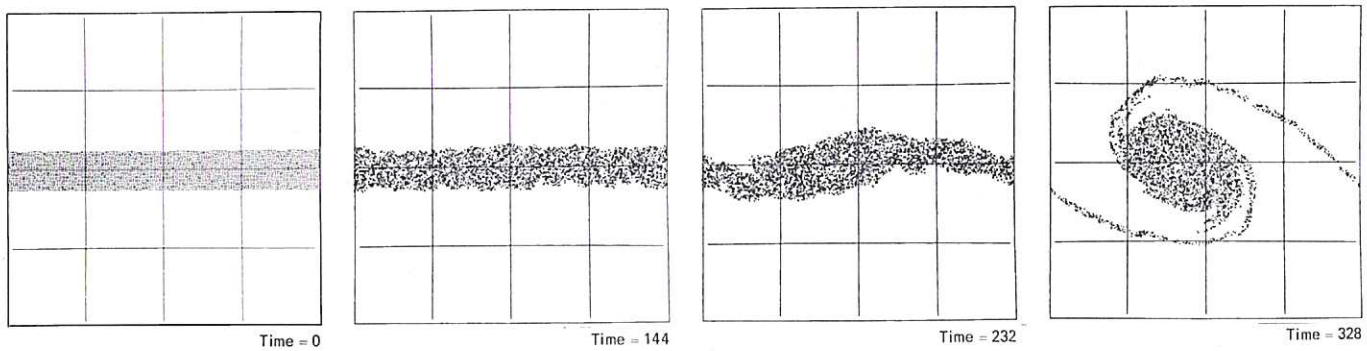
Fig. III Single circular vortex made up of point vortices. Mesh size 16 x 16



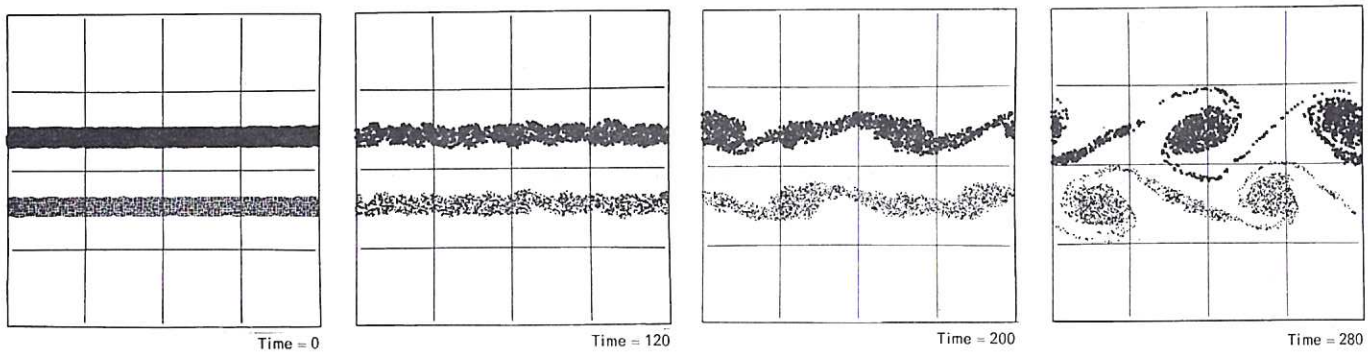
Two vortices coalescing because of sufficient initial proximity



Two vortices precessing around each other. Large amplitude oscillations on their surfaces



The onset of the Kelvin-Helmholtz instability



The formation of the von Karman vortex street

Fig. IV Flow problems simulated by program VORTEX

Available from

HER MAJESTY'S STATIONERY OFFICE

49 High Holborn, London, W.C.1
13a Castle Street, Edinburgh 2
109 St. Mary Street, Cardiff CF1 1JW
Brazennose Street, Manchester M60 8AS
50 Fairfax Street, Bristol BS1 3DE
258 Broad Street, Birmingham 1
7-11 Linenhall Street, Belfast BT2 8AY

or through any bookseller.