TRANSONIC AIRFOIL CALCULATIONS USING THE EULER EQUATIONS *

Antony Jameson
Princeton University
Princeton, NJ 08544

I am going to tell you about some of the transonic aerofoil calculations I have been doing recently based on the Euler equations. During the past year I have been collaborating with Wolfgang Schmidt of Dornier in an effort to find a reasonably fast method for solving these equations for engineering applications. The calculations I shall describe have mostly been performed on an 'O'-grid, for a non-lifting case. They use a potential flow as a starting solution, although a direct solution starting with uniform flow is also possible.

I have actually been trying to do the Euler equations on and off for several years, but only extensively in the last year. The history of the particular scheme I am going to tell you about began last summer with a visit to Dornier where I spent several weeks. There was an Euler code there developed by Schmidt and Rizzi, and results from that code were presented at the Stockholm workshop a couple of years ago (Rizzi and Viviand, 1981). It was a finite-volume code based on a time-split version of the MacCormack scheme, so that it took separate account of the gradients in each coordinate direction. I took that code as a starting point, and converted it to a scheme which uses central differencing in space and a three-stage time-stepping process which is stable for Courant numbers up to 2.0. It seemed to work quite well. Dornier have gone on to produce an engineering version of it which handles lifting cases and incorporates boundary layers. Rizzi has now implemented the new differencing scheme in his three-dimensional wing calculations. I have also been cooperating with Eli Turkel in Tel Aviv, and have followed his advice on how to treat the outer boundaries.

For the past nine months I have been trying to get the scheme to work to my satisfaction. The problem with both the earlier Rizzi-Schmidt code and the first version that I developed from it was that they converged quite quickly down to residuals of about 10^{-2} and then tended to oscillate. After a very painful nine months I seem to have made it converge right down to 10-8 in about 2000 time steps. For engineering purposes you could probably stop after 200-300 steps but I do feel that to be really useful an Euler code should have the ability to converge as far as possible, in fact down to the kind of residual we are accustomed to in potential flow calculations. For example, imagine we are trying to study the behaviour of a supposedly shock-free design, like a Korn aerofoil, around its design conditions. We want to see that shock-free solution at the design Mach number, and when we go up or down 0.001 in Mach number we want to see if a shockwave forms, and if so how much drag it produces. I don't think you can resolve those questions with a code in which the far-field Mach number wanders around between 0.74 and 0.76.

I have already said that the code I am working with uses the finite volume formulation. The other general point I want to make is that I work with a four-equation model. That is to say, I solve the four conservation laws for mass, momentum and energy. I could discard the energy equation and replace it by a condition of constant enthalpy, because I know that condition must hold in the steady state, but I prefer to let the enthalpy

*Reprinted from Preceeding of I.M.A. Conference on Numerical Methods in Aeronautical Fluid Dynamics, March 1981, edited by P.L. Roe, Academic Press, 1982, pp.289–308 vary, because I believe that I can add a useful damping term proportional to the difference between the local value of enthalpy and its free stream value. However, I will say more about that later.

The Euler equations we want to solve are, in conservation form

$$\frac{\partial w}{\partial t}+\frac{\partial f}{\partial x}+\frac{\partial g}{\partial y}=0 \tag{1}$$

where

$$w=\begin{pmatrix}\rho\\ \rho u\\ \rho v\\ \rho E\end{pmatrix};\quad f=\begin{pmatrix}\rho\\ \rho u^2+p\\ \rho uv\\ \rho uH\end{pmatrix};\quad g=\begin{pmatrix}\rho v\\ \rho uv\\ \rho v^2+p\\ \rho vH\end{pmatrix} \tag{2}$$

in which E= total energy, H = total enthalpy, and pressure p is given by

$$p=(\gamma-1)\rho(E^{-1/2}(u^2+v^2)) \tag{3}$$

We can transform these to an arbitrary set of coordinates (X,Y) for which the transformation derivatives are contained in the matrix

$$H=\begin{pmatrix}x_X & x_Y\\ y_X & y_Y\end{pmatrix} \tag{4}$$

The result is

$$\frac{\partial W}{\partial t}+\frac{\partial F}{\partial X}+\frac{\partial G}{\partial Y}=0 \tag{5}$$

where

$$\begin{array}{lclcl} W &=& \det(H)w &=& hw\\ F &=& y_Y f - x_Y g && \\ G &=& x_X g - y_X f && \end{array} \tag{6}$$

We shall identify the lines X,Y = constant with the boundaries of our finite volumes. Note that $h=\det(H)$ is the volume in (x,y) space of a cell having unit volume in (X,Y) space. If we define the contravariant velocities

$$\begin{pmatrix}U\\ V\end{pmatrix}=H^{-1}\begin{pmatrix}u\\ v\end{pmatrix} \tag{7}$$

(which are simply the velocity components normal to the cell sides) we can write

$$w=\begin{pmatrix}\rho h\\ \rho uh\\ \rho vh\\ \rho Eh\end{pmatrix};\quad F=\begin{pmatrix}U\rho h\\ U\rho uh+y_Y p\\ U\rho vg-x_Y p\\ U\rho Hh\end{pmatrix};\quad G=\begin{pmatrix}V\rho h\\ V\rho uh-y_X p\\ V\rho vh+x_X p\\ V\rho Hh\end{pmatrix} \tag{8}$$

The discretisation scheme which I have made out of this uses a finite volume form with cell-centered quantities (Fig. 1), so that (ρ, u, v, E) are supposed to represent values at some central point in each cell. You calculate a "flux velocity" across each cell face, according to

$$Q_k=\Delta y_k u_k-\Delta x_k v_k \tag{9}$$

where k=1,2,3,4, and u_2, for example, is the average velocity for the two cells (i,j) and $(i+1,j)$.

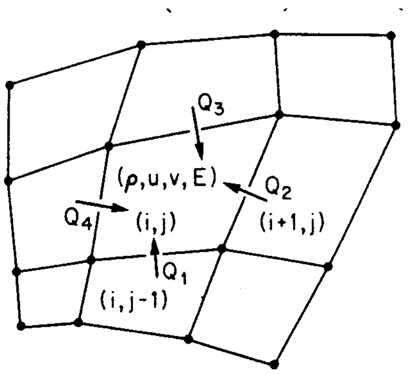

Figure 1: A typical finite-volume cell

The equation which gives me, let us say, the local rate of change of x-momentum is

$$\frac{\partial}{\partial t}(\rho u h) + \sum_{k=1}^{k=4}(Q_k(\rho u)_k + \Delta y_k p_k) = 0 \tag{10}$$

where, as before, $(\rho u)_k$ and p_k are the appropriate average values.

So that is how I do the flux balancing. If you follow it through you will see that it amounts to carrying out a central differencing operation on the quantities F and G. As a digression I might say that in the fifty or so Euler codes I have written by now a lot of alternatives have been tried, and some of them may yet be made to work. I still haven't found out whether it is really better to store things at the centers of cells or at the corners, but the present scheme uses the cell centers, and in effect does central differencing on them. However, it is well known that central differencing is unstable unless you add some dissipation terms, so what I actually solve is

$$\frac{\partial}{\partial}(\rho u h) + Qw - Dw = 0 \tag{11}$$

where Q is my flux balancing operator (not to be confused with the Q_k defined in equation (9)) and D is some dissipative operator. A point that I want to emphasize is that I have kept the time-stepping scheme quite independent from the spatial differencing, so that my final steady state should be one in which $Qw - Dw = 0$. With other differencing schemes this does not always happen; for example with a MacCormack scheme the 'steady state' may depend on Δt.

The dissipation operator which I have found to work is actually a combination of second and fourth differences. These are done separately in the X and Y directions so we will just consider the one-dimensional case and show how it applies to the density equation. I smooth the density according to

$$D\rho = D_X\rho + D_Y\rho$$

where

$$D_X\rho = d_{i+\frac{1}{2},j} - d_{i-\frac{1}{2},j} \tag{12}$$

and

$$d_{i+\frac{1}{2},j} = \frac{h_{i+\frac{1}{2},j}}{\Delta t^*_{i+\frac{1}{2},j}} \Big[\epsilon^{(2)}_{i+\frac{1}{2},j}(\rho_{i+1,j} - \rho_{i,j}) - \epsilon^{(4)}_{i+\frac{1}{2},j}(\rho_{i+2,j} - 3\rho_{i+1,j} + 3\rho_{i,j} - \rho_{i-1,j}) \Big] \tag{13}$$

where Δt^* is the time step which would bring about a local Courant number of unity. It varies from cell to cell, and I have computed its average value on the interface between cells $(i,j),(i+1,j)$. Likewise $h_{i+\frac{1}{2},j}$, is an average of two cell volumes. Dimensionally, eqn (13) is a rate of change of mass (since the E are numerical constants) which is as it should be for use in eqn (11). The factor outside the square brackets in eqn (13) is of order (Δx), and the terms inside would be respectively of order (Δx), $(\Delta x)^3$ if $\epsilon^{(2)}$, $\epsilon^{(4)}$ were each of order unity. In fact I am going to make $\epsilon^{(2)}$ of order $(\Delta x)^2$, so the whole expression will be of order $(\Delta x)^4$, it adds to the spatial differencing operator Q a relative error which is merely third order. You can easily see that the added terms are actually a mixture of second and fourth differences.

The way that these ϵ coefficients are chosen is to adapt them to the local flow gradients. For the second-order coefficient I take

$$\epsilon^{(2)}_{i+\frac{1}{2},j} = \max(V_{i+1,j}, V_{i,j}) \tag{14}$$

where

$$V_{i,j} = K^{(2)} \frac{|p_{i+1,j} - 2p_{i,j} + p_{i-1,j}|}{|p_{i+1,j}| + 2\,|p_{i,j}| + |p_{i-1,j}|} \tag{15}$$

In eqn (15), $K^{(2)}$ is a constant, and I am not sure whether the normalisation factor in the denominator is needed. However, the outcome is that $\epsilon^{(2)}$ is positive, of order $(\Delta x)^2$, and proportional to the second difference of pressure. It does seem to do a nice job suppressing the oscillations around shocks.

However, what I found was that the second-order dissipation was not enough by itself; the solution would still go down to moderately small residuals and then oscillate. The oscillations were most noticeable in the small cells near the trailing edge, presumably because a given increment in mass looks like a big increment in density if you have a small cell, and it was density that I was monitoring. The total excursion in density. would be about $\pm 1\%$ and there might be a limit cycle of around 130 time steps. It turned out that those fluctuations could be eliminated by adding the fourth difference term, but then overshoots returned near the shockwave; the obvious answer was to switch off that term near the shock, which was quite easy because that was where $\epsilon^{(2)}$ was being turned on. In fact, I set

$$\epsilon^{(4)}_{i+\frac{1}{2},j} = \max\left\{0, (K^{(4)} - \epsilon^{(2)}_{i+\frac{1}{2},j})\right\} \tag{16}$$

where $K^{(4)}$ is a second constant, and that does the trick. What it comes down to is that the second difference is usually multiplied by a factor of order $(\Delta x)^2$, rising to order unity near a shock; the fourth difference is multiplied by a factor which is usually of order unity, but which falls to zero through a shock.

Now we go on to how the time-stepping is done. I began by devising a three-step scheme which can be written as follows. We call the complete space operator, flux balance and dissipation, P, so that

$$Pw = Qw - Dw \tag{17}$$

and then we go from time level n to time level (n+1) by means of

$$\begin{array}{rcl} w^{(0)} & = & w^n \\ w^{(1)} & = & w^{(0)} - \Delta t P w^{(0)} \\ w^{(2)} & = & w^{(0)} - \frac{\Delta t}{2} P w^{(0)} - \frac{\Delta t}{2} P w^{(1)} \\ w^{(3)} & = & w^{(0)} - \frac{\Delta t}{2} P w^{(0)} - \frac{\Delta t}{2} P w^{(2)} \\ w^{n+1} & = & w^{(3)} \end{array} \tag{18}$$

That was the scheme I used at Dornier last summer, and at the time I thought it was rather neat, although it turned out not to be new, having been proposed by Gary in 1964. If you do the usual stability analysis you find that it is stable for Courant numbers less than 2.0, and you get that rate of advance for the price of evaluating the spatial operator three times. For comparison, the MacCormack scheme gives you a Courant number of 1.0 for two evaluations.

Then I realised that (18) belongs to the class of Runge-Kutta methods, and that a lot is known about the properties of such methods, especially in the context of ordinary differential equations, such as

$$\frac{\partial w}{\partial t} + Aw = 0 \tag{19}$$

The most useful concept seems to be that of the stability region. If we apply some specified algorithm to eqn (19) it turns out that there are values of $A\Delta t$, with A in the complex plane, for which the algorithm will be stable, and that region of the complex plane is the stability region. There is a book by Stetter (1973), in which plots of these regions are given for a variety of schemes.

The way that we can apply this concept to partial differential equations such as

$$\frac{\partial w}{\partial t} + A\frac{\partial w}{\partial x} = 0 \tag{20}$$

is just to take the Fourier transform of w, say $\hat{(w)}$ and to realize that the central difference operator multiplies $\hat{(w)}$ by a factor which is purely imaginary, but that the dissipative terms add a purely real component to that factor.

In the plotted stability regions that I am going to show you (Fig. 2) the imaginary part is merely what we call Courant number, and the real part is a measure of the added dissipation. You can now see that the first order scheme is not stable anywhere on the real axis but if we add enough dissipation (for example by using one sided differences) we can get ourselves inside the circle. In fact the second-order two-step scheme is also unstable everywhere on the imaginary axis, but the thirdorder, three-step scheme does include a segment of the imaginary axis. The three step scheme which I showed you just now (eqn(18)) goes up to a Courant number of 2.0, and that is more efficient than MacCormack or Lax-Wendroff, in terms of time advanced for each evaluation of the spatial operator P. In fact we can do even better by going to the standard fourth-order scheme thus

$$\begin{array}{rcl} w^{(0)} & = & w^n \\ w^{(1)} & = & w^{(0)} - \frac{\Delta t}{2} P w^{(0)} \\ w^{(2)} & = & w^{(0)} - \frac{\Delta t}{2} P w^{(1)} \\ w^{(3)} & = & w^{(0)} - \Delta t P w^{(2)} \\ w^{(4)} & = & w^{(0)} - \frac{\Delta t}{6} P w^{(0)} - \frac{\Delta t}{3} P w^{(1)} - \frac{\Delta t}{3} P w^{(2)} - \frac{\Delta t}{6} P w^{(3)} \\ w^{n+1} & = & w^{(4)} \end{array} \tag{21}$$

and this gives us a Courant number of $2\sqrt{(}2) = 2.8$ for four evaluations of the operator. Also I have got quite a lot of room to add dissipative terms if I need to, near the shocks. That is shown by the shaded region to the left of the axis.

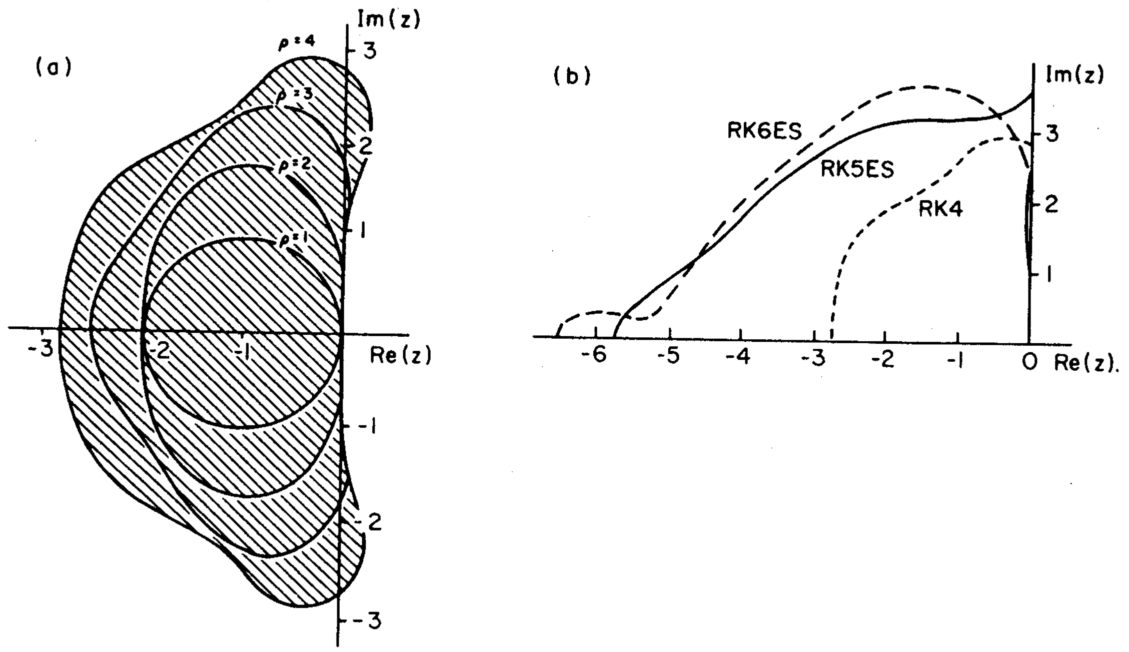

Figure 2: Stability regions for various Runge-Kutta time-stepping schemes: (a) standard schemes or order 1,2,3,4 (Stetter, 1973), (b) non-standard schemes of order 4,5,6 (Lawson, 1966)

A very interesting possibility is that when you add extra stages to a Runge-Kutta scheme you can do so either with the intention of gaining more speed or of gaining more accuracy. And although the standard four-step method is fourth-order accurate in time there is not very much point in that when you are only second-order accurate in space. So perhaps by backing off in accuracy you could gain even more speed. Lawson (1966) has done some work on these lines, and he has a six-stage scheme which is stable out to Courant numbers of 3.50, with a huge amount of room in the real direction, and I think there is scope for a lot of investigation here, to find the most efficient possibilities.

Something else that I have been thinking about a lot recently is the point I mentioned earlier about how to make use of the fact that we expect to have constant enthalpy in the steady state. That is obvious when you look at the equations (2) and notice that when the time derivatives vanish the fourth equation is just H times the first equation. We can try to exploit that in two different ways. We could assert that H will always be H and then solve the non-physical system given by the first three equations (see the chapter by Lerat, for example). The alternative is something for which I do not yet have a real mathematical justification, but for which I can offer analogies based on experience in potential flow calculations.

Suppose we set out to solve the potential equation by means of line relaxation. Then it turns out that the relaxation operator effectively adds something like a (Pt term, turning the unsteady potential equation into a kind of telegraph equation which has damped waves as its fundamental solutions. On the other hand, the true linear wave equation

$$\varphi_{tt} = \varphi_{xx} + \varphi_{yy} \tag{22}$$

is undamped, as you can see if you multiply by φ_t and integrate

$$\iint \varphi_t \varphi_{tt} dxdy = \iint \varphi_{xx} \varphi_t dxdy + \iint \varphi_{yy} \varphi_t dxdy \tag{23}$$

If you integrate by parts and neglect the boundary terms, or consider an infinite domain, then

$$\iint \varphi_t \varphi_{tt} dxdy + \iint \varphi_x \varphi_{xt} dxdy + \iint \varphi_y \varphi_{yt} dxdy = 0$$

and so

$$\frac{\partial P}{\partial t} = 0 \tag{24}$$

where

$$P = \frac{1}{2} \iint (\varphi_t^2 + \varphi_x^2 + \varphi_y^2) dxdy \tag{25}$$

so the quantity P, which is a kind of energy, does not decay. The equation which mimics the relaxation process is

$$\varphi_{tt} + \alpha \varphi_t + \varphi_{xx} + \varphi_{yy} \tag{26}$$

and then the same piece of algebra gives

$$\frac{\partial P}{\partial t} + \alpha \iint \varphi_t^2 dxdy = 0 \tag{27}$$

Therefore, if a is positive, the positive quantity P must decay, until we reach a steady state with $\varphi_t = 0$ everywhere. The coefficient α is actually proportional to $(2/\omega - 1)$ where ω is the relaxation factor, and since we all know that you have $\omega < 2$ for stability, that makes the whole argument fairly believable.

Now I would like to get that sort of damping into the Euler equations if I possibly can, but unfortunately I no longer have a potential. However, you may recall that the unsteady Bernoulli equation you get from the potential equation tells you that $\varphi_t + H =$ const, so maybe adding some terms proportional to $(H - H_\infty)$ will put some damping into the Euler equations. I feel that will be the case if you do it right, but I am not yet sure of the details. What I am sure of is that you had better be certain that H = constant really is a solution, not merely of the differential equations, but of the difference equations you use to solve them. And that is not easy to ensure in something like a MacCormack scheme, with mixtures of backward and forward differences. But it is quite easy to ensure in the kind of central difference scheme I have told you about, with this proviso, that the dissipation operator in the energy equation is applied to ρH rather than to ρE.

What I tried was to add a term proportional to $(H - H_\infty)$ in the continuity equation, thus

$$\frac{\partial \rho}{\partial t} + \rho \nabla . q + \alpha \rho (H - H_\infty) = 0 \tag{28}$$

and then I found that I needed to add corresponding terms to all other three equations, so as to retain conservation. In the first place I tried the extra terms out in a code which only had the second-order dissipation and really was not converging very well and I found that the enthalpy damping would make it converge. Subsequently I added the fourth-order dissipation, and then the scheme would converge whether or not I included the enthalpy damping. The enthalpy damping does produce quicker convergence but I still have to find the best combination, and to decide whether the extra improvement is worthwhile.

For non-lifting flow past the NACA 0012 at M = 0.08 I can get an adequate engineering solution within 600 time steps if I start by creating the aerofoil impulsively in a uniform stream, or I can bring that down to 300 if I start from the potential solution. At higher Mach numbers with a stronger shock the potential solution is less useful. It is interesting that the method appears to give very respectable shockwaves without the need to switch differences, or to get involved with any complicated logic depending on the signs of eigenvalues. I have not attempted any kind of multigrid solution yet because until now the basic code was not coverging to my satisfaction. But now it does, and so maybe this is the time to try and implement a multigrid version. What needs to be observed, though, is that multigrid depends on having some element of smoothing, and does not fit readily with problems where the basic mechanism is wave propagation.

The last thing I want to tell you about is boundary conditions. At the outer boundary what I do is to calculate the characteristic variables, on the basis of a local linearisation. These turn out to be $(p - c^2\rho)$, $(p + cu\rho)$, $(p - cu\rho)$ and v, where v is the tangential velocity component, and u is the normal velocity component relative to the outer boundary. The first and last of these are carried along the streamline, with velocity u, and the second and third are carried along Mach waves with velocities (u + c), (u - c). Now the theory says that if you are at a subsonic outflow boundary, the third quantity may be imposed, but the other three have to be determined from the interior, whereas if you are at a subsonic inflow boundary the second must be determined from the interior, but the other three may be imposed. And basically that is what I do, but I convert all these conditions so that they relate to the conservation variables I am actually using. Some people say that you can get away with over-specifying the conditions at the outer boundary, simply fixing everything at its free-stream value, but when I have tried that it has sometimes gone badly wrong. With a more dissipative difference scheme it seems less critical what you do.

To apply a boundary condition on the body is very easy for steady flows if you look back to eqn (10). All you have to do is to set $Q_k = 0$ on the interface concerned, or to give it whatever value you want it to have. And you specify a surface pressure by extrapolating from the interior according to $\partial p/\partial n = \rho q^2/R$, where n is the surface normal, and R is the surface curvature.

Finally, I would like to show you a few of the results, beginning with the non-lifting flow past a NACA 0012 aerofoil at Mach number 0.8. The inner part of my computing mesh for this problem is shown in Fig. 3. The development of the flow from impulsive starting conditions is shown in Fig. 4, where I have not used enthalpy damping, and in Fig. 5 where I have. You can see that in each case the plotted pressure distributions develop smoothly. However, if we measure the error as the r.m.s. value of $\Delta\rho/\Delta t$ within cells, then this quantity converges quite a lot faster using enthalpy damping. In Fig. 6 we have the corresponding information, for the rather harder case M, = 0.85, with the enthalpy damping.

References

[1] Rizzi, A. and Viviand, H. (1981). *Numerical methods for the computation of inviscid transonic flows with shock waves.* Notes on Numerical Fluid Mechanics, vol 3, Vieweg.

[2] Stetter, H. (1973). *Analysis of discretisation error for ordinary differential equations.* Tracts in Natural Philosophy, no 23 Springer.

[3] Gary, J. (1964). *On certain finite difference schemes for hyperbolic systems.* Math. Comp. 18 pp 1 - 18.

[4] Lawson, J.D. (1966). *An order 5 Runge-Kutta process with extended region of stability.* SIAM. J. Num. Anal. 3, no. 4, pp 593 - 606.

EDITOR'S NOTE

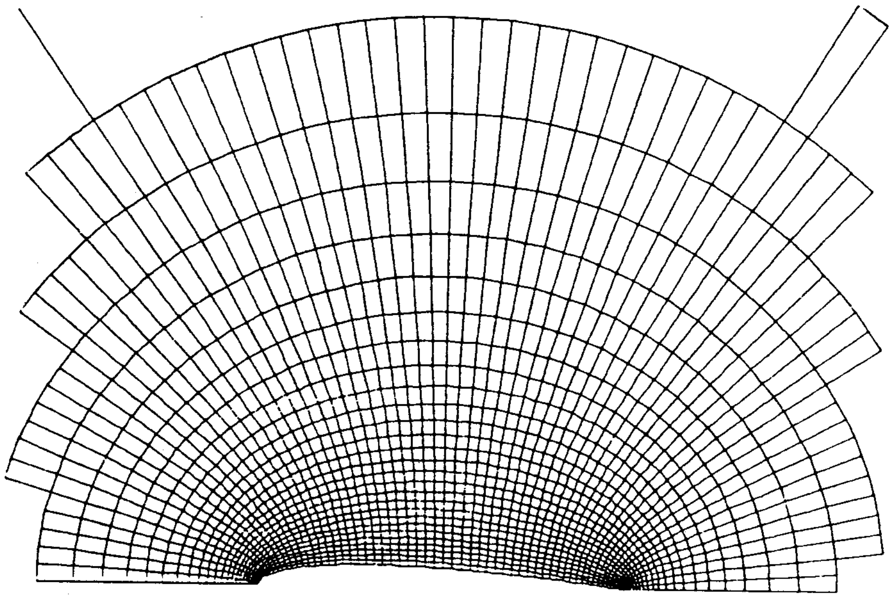

Figure 3: Part of the computing grid for a NACA 0012 Aerofoil. The total grid has 64 cells from front to rear and 32 cells radially

Since Professor Jameson gave the above lecture, he has gone on to develop his scheme into a three-dimensional method. Essentially no new mathematical problems arise. The analysis of the flux balance is done exactly as above, with each cell receiving increments of mass, momentum and energy due to fluxes across six faces, computed from average values of pressure, normal velocities etc., on those faces. A smoothing operation is carried out consisting of three one-dimensional operations, each identical to that given in the lecture, and then the solution can be advanced in time by the same four-stage Runge-Kutta scheme. The problems which do arise are concerned with programming. Many details of the coding have been altered to remove the pressure on storage.

Professor Jameson has very kindly made available some results which this method has produced. They relate to the ONERA M6 wing under test conditions $M_\infty = 0.84$, $\alpha = 3.06°$. The computational mesh consists of 128 cells around each aerofoil section, 32 cells along each spanwise line, and 16 cells going outward from the wing. The results of the Euler code are shown as solid lines and results from a potential method on the same grid as dashed lines. Fig. E1 (a)-(d) shows pressure distributions along those chordwise lines which lie at 12.5, 37.5, 62.5 and 87.5 per cent of the semi-span. Fig. E2 shows the Euler results only, at 20 chordwise sections. Fig. E2 (a) shows results for the upper surface of the wing, and Fig. E2 (b) shows results for the lower surface.

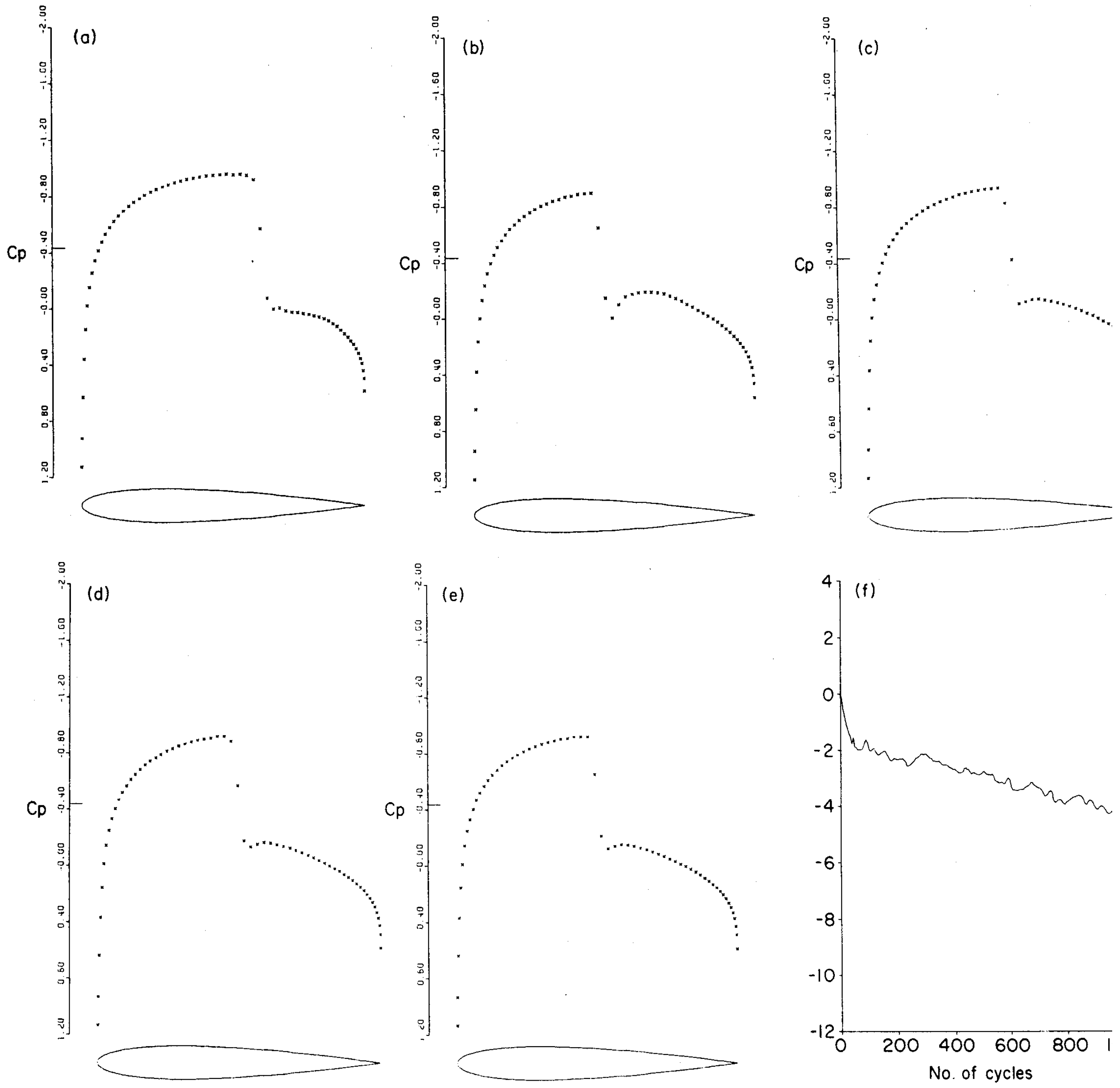

Figure 4: Solutions of the flow past NACA 0012 aerofoil section at $M_\infty = 0.80, \alpha = 0°$; (a)-(e) Show pressure distributions after 200, 400, 600, 800 and 1000 cycles of Runge-Kutta process; (f) shows the logarithm of the r.m.s. error. This calculation does **not** use enthalpy damping

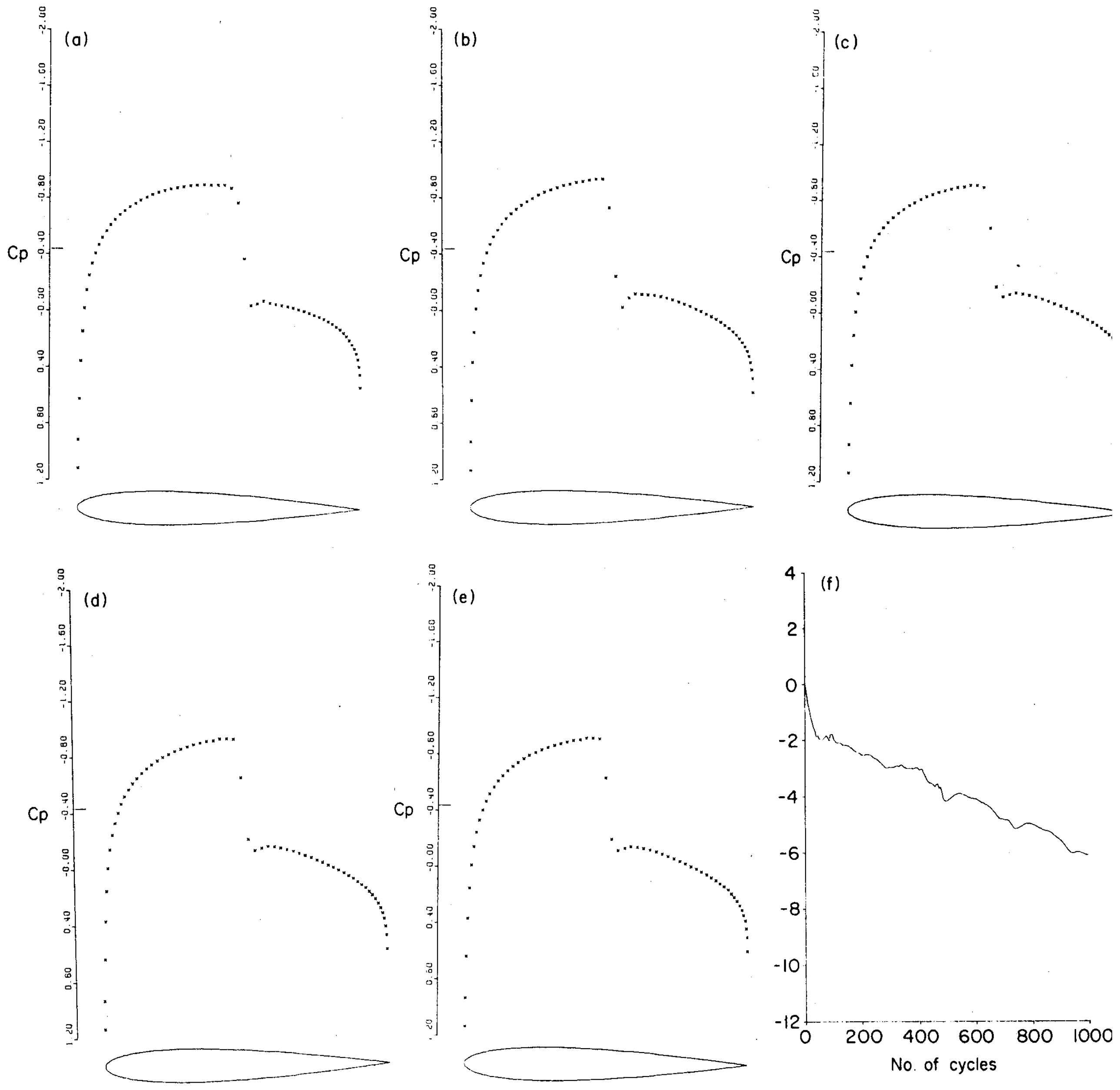

Figure 5: Solutions of the flow past NACA 0012 aerofoil section at $M_\infty = 0.80, \alpha = 0°$; (a)-(e) Show pressure distributions after 200, 400, 600, 800 and 1000 cycles of Runge-Kutta process; (f) shows the logarithm of the r.m.s. error. This calculation **does** use enthalpy damping

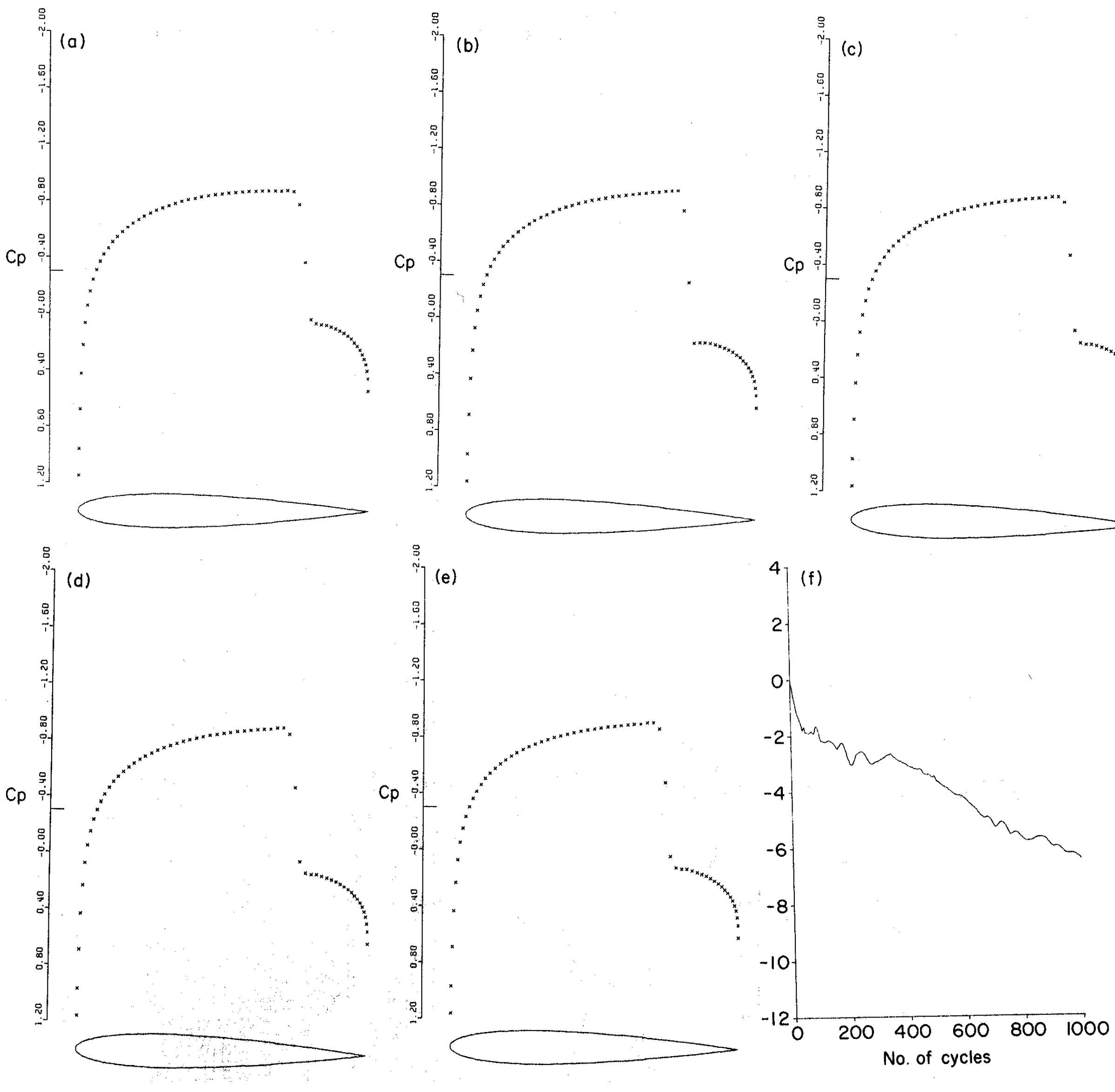

Figure 6: Solutions of the flow past NACA 0012 aerofoil section at $M_\infty = 0.85, \alpha = 0°$; (a)-(e) Show pressure distributions after 200, 400, 600, 800 and 1000 cycles of Runge-Kutta process; (f) shows the logarithm of the r.m.s. error. This calculation **does** use enthalpy damping

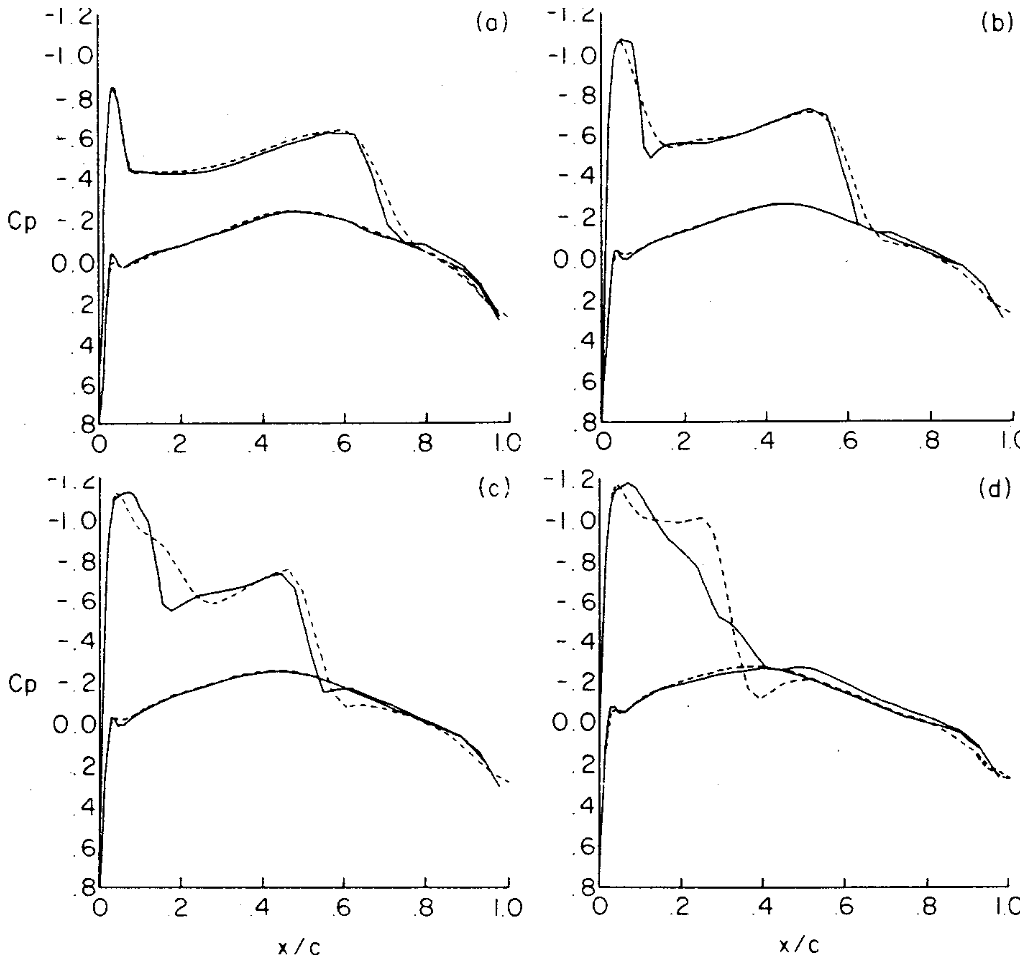

Figure E1:

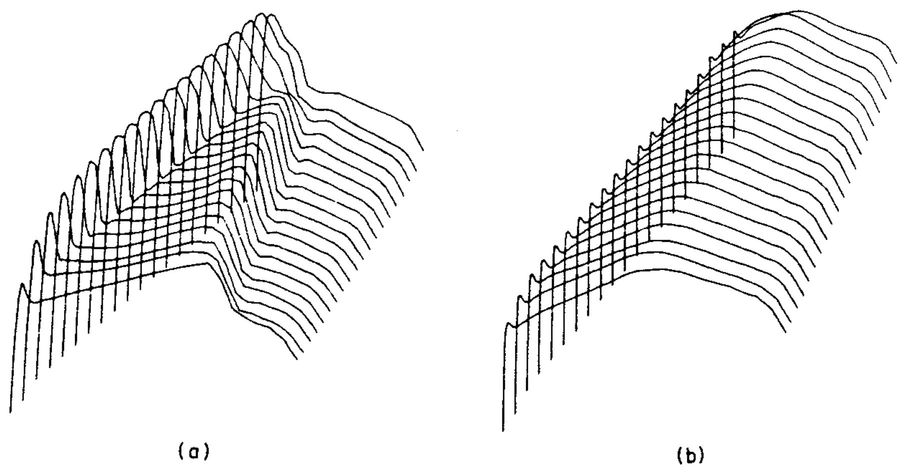

Figure E2: