NASA Contractor Report 3689

Fast Euler Solver for Steady, One-Dimensional Flows

Gino Moretti

CONTRACT NAS3-22772
JUNE 1983

25th Anniversary
1958-1983

NASA

NASA Contractor Report 3689

Fast Euler Solver for Steady, One-Dimensional Flows

Gino Moretti
G.M.A.F., Inc.
Freeport, New York

Prepared for
Lewis Research Center
under Contract NAS3-22772

National Aeronautics
and Space Administration

Scientific and Technical Information Branch

1983

TABLE OF CONTENTS

	Summary	1
1.	Introduction	1
2.	Outline of the technique	2
3.	Second-order accurate scheme	7
4.	First examples. Subsonic flows	8
5.	Technique for transonic flows	16
6.	Shock calculation	18
7.	Results for flows with shocks	23
8.	Conclusions	29

Summary

A numerical technique to solve the Euler equations for steady, one-dimensional flows is presented. The technique is essentially implicit, but is structured as a sequence of explicit solutions for each Riemann variable separately. Each solution is obtained by integrating in the direction prescribed by the propagation of the Riemann variables. The technique is second-order accurate. It requires very few steps for convergence, and each step requires a minimal number of operations. Therefore, it is three orders of magnitude more efficient than a standard time-dependent technique. The technique works well for transonic flows and provides shock fitting with errors as small as 0.001. Results are presented for subsonic and transonic problems. Errors are evaluated by comparison with exact solutions.

1. Introduction

Numerical solutions of the Euler equations, including accurate calculations of discontinuities, should provide the best description of compressible, inviscid flows. Nevertheless, when such flows are steady, a strong emphasis has been put, in recent years, on solvers of the potential equation, which apply relaxation methods. The assumption of a velocity potential inhibits variations of entropy. Consequently, positive jumps in pressure, commonly called "shocks", occur at constant entropy and do not satisfy the Rankine-Hugoniot conditions; legitimate doubts arise on the accuracy of the results. Strong arguments, however, have been set forth on defense of potential-relaxation techniques; above all, their speed of execution, and the possibility of exploiting multiple grid or multi-grid accelerating effects. Whatever accuracy may be obtained when vorticity and entropy effects are neglected, it is aimed at by using a very large number of computational nodes.

We claim that a good Euler solver can reach an equal (or higher) degree of accuracy with much fewer nodes. Recently, substantial progress has been made in the numerical integration of the Euler equations, by introducing robust and accurate integration schemes [1,2,3,4,5], methods for the correct handling of boundary conditions and means for the definition of suitable computational grids (these topics are reviewed in [6] and [7], respectively). Unfortunately, if a steady state solution has to be reached using a standard time-dependent Euler solver, convergence to the asymptotic solution is slow, and the number of operations to be performed per step is larger than in a potential-relaxation procedure. Consequently, the calculation turns out to be much more expensive, despite the use of a coarser grid.

We present here an extremely simple Euler solver for steady flows, which requires a minimal amount of operations per step and a small number of steps to achieve convergence. In principle, it amounts to solving two bidiagonal matrices per step.

For a better illustration of the method, we limit the present report to one-dimensional problems. We will examine our findings on two-dimensional problems in a successive paper.

The technique presented here is derived directly from the same ideas which produced the "lambda"-scheme in its more recent formulation [8]. Conscious and unconscious borrowing of ideas, however, is part of the creative labor of every new project and, whenever possible, should be acknowledged. The idea of sweeping (that is, of attempting to solve a bidiagonal matrix instead of a tridiagonal one) was inspired by MacCormack [9]. The modeling of boundary conditions is essentially due to M. Pandolfi, and L. Zannetti should be thanked here for his active faith in Riemann variables and an early suggestion of the relation between ΔR_1 and the shock Mach number. I wish also to acknowledge the collaboration of Miss Catherine Fahy in the preparation of the code and the organization and analysis of data, on a level above clerical routine.

2. Outline of the technique

To explain the origin of the technique, let us begin by writing the Euler equations in characteristic form:

$$R_{1t} - as_t + \lambda_1(R_{1x} - as_x) + d = 0$$
$$R_{2t} - as_t + \lambda_2(R_{2x} - as_x) + d = 0 \qquad (1)$$
$$s_t + u\, s_x = 0$$

where

$$R_1 = a/\delta - u, \quad R_2 = a/\delta + u \qquad (2)$$

$$\lambda_1 = u - a, \quad \lambda_2 = u + a \qquad (3)$$

$$d = a\, u\, A_x/A \qquad (4)$$

and a, u, s are the speed of sound, particle velocity and entropy, respectively; $\delta = (\gamma-1)/2$, and γ is the ratio of specific heats; finally, A(x) is the cross-sectional area of a quasi-one-dimensional duct. Written in the form (1), the Euler equations describe the propagation of pressure waves (here expressed in terms of Riemann variables, R_1 and R_2 and entropy) along the characteristics having slopes λ_1 and λ_2, respectively, and the convection of entropy along streamlines.

The interplay of pressure waves and entropy waves is an essential feature of unsteady flows. In a steady state, however, the third of (1) shows that the entropy must be constant wherever the flow is continuous and differentiable. Due to the presence of shocks in the flow, the entropy in a steady state is generally piecewise constant (note that contact surfaces cannot exist in a one-dimensional steady flow). If only the steady solution of a problem is of interest and we conceive of it as the asymptote of an unsteady process, the entropy terms can be dropped from (1), in order to find the steady solution in each region of continuous flow. This does not mean that the entropy does not influence the steady values of R_1 and R_2. In fact, its influence is felt through the boundary conditions, as we will see further on.

We will, consequently, simplify the Euler equations in two successive steps. In the first, we drop all entropy terms:

$$\begin{aligned} R_{1t} + \lambda_1 R_{1x} + d &= 0 \\ R_{2t} + \lambda_2 R_{2x} + d &= 0 \end{aligned} \tag{5}$$

The above system, which is correct only when the entropy is piecewise constant, will be used here solely for heuristic arguments, instead of the clumsier system (1). The following system, instead, is correct in all regions of continuous, steady flow, and it is the system whose numerical solution is sought:

$$\begin{aligned} \lambda_1 R_{1x} + d &= 0 \\ \lambda_2 R_{2x} + d &= 0 \end{aligned} \tag{6}$$

Let us assume that u is positive and consider subsonic flows only, for the time being. In subsonic flows, λ_1 is negative and λ_2 is positive. The way of coding unsteady Euler solvers, known as the λ-scheme [1], stems from the obvious fact, expressed by (5), that R_1-waves propagate in the direction of decreasing x, whereas R_2-waves propagate in the direction of increasing x. The two equations (5), thus, can be integrated separately at each computational node. Since the domains of dependence of R_1 and R_2 lie to the right and to the left of the point to be computed, respectively, the x-derivatives of R_1 and R_2 will have to be approximated by finite differences between the point in question and points on its right or its left, respectively.

Let $t=k\Delta t$ and $x=n\Delta x$. To a first order of accuracy, (5) can be approximated by:

$$R^{k+1}_{1,n} = R^k_{1,n} - \sigma\,[\lambda_1(R^k_{1,n+1} - R^k_{1,n}) + d\,\Delta x] \tag{7}$$

$$R^{k+1}_{2,n} = R^k_{2,n} - \sigma\,[\lambda_2(R^k_{2,n} - R^k_{2,n-1}) + d\,\Delta x] \tag{8}$$

Here, $\sigma = \Delta t/\Delta x$. In so doing, we have laid the foundation for an explicit code. Alternatively, we can approximate (5) by:

$$R^{k+1}_{1,n} = R^k_{1,n} - \sigma[\lambda_1(R^{k+1}_{1,n+1} - R^{k+1}_{1,n}) + d\,\Delta x] \tag{9}$$

$$R^{k+1}_{2,n} = R^k_{2,n} - \sigma[\lambda_2(R^{k+1}_{2,n} - R^{k+1}_{2,n-1}) + d\,\Delta x] \tag{10}$$

Then $R^{k+1}_{1,n}$ and $R^{k+1}_{2,n}$ are obtained as

$$R^{k+1}_{1,n} = \frac{R^{k}_{1,n} + \omega\sigma[|\lambda_1| R^{k+1}_{1,n+1} - d\,\Delta x]}{1 + \omega\sigma|\lambda_1|} \tag{11}$$

$$R^{k+1}_{2,n} = \frac{R^{k}_{2,n} + \omega\sigma[\lambda_2 R^{k+1}_{2,n-1} - d\,\Delta x]}{1 + \omega\sigma\lambda_2} \tag{12}$$

and an implicit code will be generated. In (11) and (12), a relaxation factor, ω has been introduced, in the spirit of relaxation techniques.

Let us discuss the explicit approach first. In applying the explicit λ-scheme to describe an unsteady flow, time must be advanced by the same Δt at all computational nodes. The CFL criterion for stability [10] requires σ to be the minimum of the values of $1/\lambda_2$ at all points. The advance in time is, thus, strongly limited. Conversely, in a computation which does not aim to provide a correct description of a time evolution, Δt can be let vary from point to point, provided that the discretized equations so obtained are compatible with the equations for steady flow, once the values of the physical parameters are locally stabilized.

Therefore, if we decide to integrate (7) and (8) separately over the entire region, and we begin by (8), we find that we can define σ locally, as equal to $1/\lambda_2$, so that (8) becomes:

$$R^{k+1}_{2,n} = R^{k}_{2,n-1} - d\,\Delta x/\lambda_2 \tag{13}$$

This formula can be used to integrate (8) without acting on (7), sweeping on the computational mesh from left to right. The procedure is in the spirit of relaxation techniques, but it can still be interpreted as a description of the propagation of a right-running wave if the appearance of (13) as a particular case of (8) is taken into account. In the standard, explicit integration of time-dependent equations, both right- and left-running waves are considered at every interval in order to update the flow correctly over all intervals but for a very short increment in time. Here, we attempt to describe the propagation of only one wave over the entire region without too much concern for the interaction of waves of the opposite family. Consequently, the value of R_2 at the right boundary point is, as far as the influence of the right-running waves is concerned, updated over a time of the order of the length of the duct, divided by an average value of $u+a$. As we proceed in this phase of the integration, we update u and a at every point, using the new value of R_2 and the old value of R_1, which is left unchanged.

The procedure defined by (13) can be interpreted as an implicit calculation, as in (12), with an infinitely large value of ω. The use of (13) is thus justified as optimal. In reaching the right boundary, we must apply the boundary condition. In problems of practical interest, the pressure is assigned at the exit of the duct. Insofar as there are no entropy sources, this is tantamount to assigning the speed of sound. Therefore, u can be evaluated from the second of (2) since R_2 has just been updated. Then, R_1 follows from the first of (2). It is now time to integrate R_1, sweeping from right to left. Here we may not start from (7) and simplify it as we did with (8) since λ_1 may get close to zero and the increment produced by d $\Delta x/\lambda_1$ may become too large and produce instabilities. We resort, instead, to (11), using a suitable value of ω. Larger values of ω speed up the convergence, but the theoretical gain rapidly tends to an asymptotic value. The advantage of the implicit formulation (11) over the explicit one stems from the fact that (11) is stable for large values of ω even if λ_1 tends to zero. Again, as in the preceding sweep, we will update u and a at every point, using the existing value of R_2 and the new value of R_1.

In reaching the left boundary, we apply the boundary condition by imposing the stagnation speed of sound, a_o. Since

$$a_o^2 = a^2 + \delta u^2 \tag{14}$$

and R_1 is accepted as computed, we obtain

$$u = \frac{-\delta R_1 + (\frac{\delta+1}{\delta} a_o^2 - \delta R_1^2)}{\delta + 1} \tag{15}$$

Then, a follows from (9) and R_2 from the second of (2). One integration step is thus completed, and the computation resumes with the right-running sweep.

Note that (13) and (11), regardless of the value of ω, are compatible with the partial differential equations for the steady state, that is, when $R^k_{1,n}$ coincides with $R^{k+1}_{1,n}$, to within the first order of accuracy.

3. Second-order accurate scheme

To achieve second-order accuracy in the asymptotic steady state solution, we approximate (6) by

$$\begin{aligned} (\lambda^k_{1,n+1} + \lambda^k_{1,n})(R^k_{1,n+1} - R^k_{1,n}) + (d_{n+1} + d_n)\,\Delta x = 0 \\ (\lambda^k_{2,n-1} + \lambda^k_{2,n})(R^k_{2,n} - R^k_{2,n-1}) + (d_{n-1} + d_n)\,\Delta x = 0 \end{aligned} \tag{16}$$

That such expressions are second-order accurate is easily seen. For example, the left-hand side of the first equation may be expanded, about point $x=n\Delta x$, in the form:

$$\begin{aligned} &[2\lambda + \lambda'\Delta x + \tfrac{1}{2}\lambda''\Delta x^2]\,[R'\Delta x + \tfrac{1}{2}R''\Delta x^2] + \\ &\quad + [2d + d'\Delta x + \tfrac{1}{2}d''\Delta x^2]\,\Delta x \end{aligned} \tag{17}$$

(all unessential indices having been dropped). By grouping terms with the same power of Δx and taking into account the first of (6), which we rewrite here:

$$\lambda R' + d = 0 \tag{18}$$

and its x-derivative:

$$\lambda' R' + \lambda R'' + d' = 0 \tag{19}$$

we see that (17) begins with terms of the third order in Δx.

Consequently, (13) and (11) are replaced by:

$$R^{k+1}_{2,n} = R^k_{2,n-1} - (d_{n-1}+d_n)\,\Delta x/(\lambda_{2,n} + \lambda_{2,n-1}) \tag{20}$$

$$R^{k+1}_{1,n} = \frac{R^{k}_{1,n} + \omega\sigma[\,|\lambda_{1,n}+\lambda_{1,n+1}|\, R^{k+1}_{1,n+1} - (d_n+d_{n+1})\Delta x]}{1+\omega\sigma\,|\lambda_{1,n}+\lambda_{1,n+1}|} \qquad (21)$$

A FORTRAN code for the procedure outlined above is shown in Fig. 1. Most of the notation is obvious and requires no clarification; NC is the total number of nodes, B is the logarithmic increment of the area of the duct, $A_x/A.\Delta x$), ANC is the prescribed speed of sound at the exit of the duct, Υ is the stagnation speed of sound, in the nondimensionalization adopted here, GD,GE,GG,GC and GDH mean δ, $1+\delta$, $1/\delta$, $(1+\delta)/\delta$ and $\delta/2$, respectively, and OM is $\omega\sigma$. One may note the extreme speed of the calculation. Each double sweep requires only 12(NC-1) multiplications for interior points, and 15 multiplications for the boundary conditions, a total of 12.NC+3 multiplications, as opposed to 38.NC+14 multiplications per step, as in a standard λ-scheme procedure, which is, per se, a fast-working code. Such an acceleration per step of almost a factor of 4 is compounded by a much higher speed of convergence, as we will show in what follows.

4. First examples. Subsonic flows

To illustrate the procedure, we present three calculations of subsonic flows in ducts. In all cases, the geometry of the duct is obtained by prescribing the steady-state Mach number distribution.

In the first case, we consider a convergent duct with a linear Mach number distribution between M=0.2 and M=0.8. We apply the procedure, as coded in Fig. 1, using as initial condition a state of rest throughout the duct with $a = \Upsilon^{1/2}$, except for the exit endpoint, where a is set equal to the value belonging to the steady state solution. First, we check the accuracy. We repeat the computation five times, using 4, 8, 16, 32 and 64 intervals respectively. The mean quadratic error in u, after convergence is reached, is shown in Table I. The products of the second column by the square of the first column, reported in the third column, clearly show that the procedure is accurate to the second order.

SUBROUTINE FOR SUBSONIC FLOW ONLY

```
  N=1
  DNA=A(1)*U(1)*B(1)
  AL2A=U(1)+A(1)
  DO 1 N=2,NC
  DN=A(N)*U(N)*B(N)
  AL2=U(N)+A(N)
  R(2,N)=R(2,N-1)-(DN+DNA)/(AL2+AL2A)
  U(N)=.5*(R(2,N)-R(1,N))
  A(N)=GDH*(R(2,N)+R(1,N))
  DNA=DN
1 AL2A=AL2
  U(NC)=U(NC)+(A(NC)-ANC)*GG
  A(NC)=ANC
  R(1,NC)=A(NC)*GG-U(NC)
  DNA=A(NC)*U(NC)*B(NC)
  AL1A=U(NC)-A(NC)
  DO 2 NN=2,NC
  N=NC+1-NN
  DN=A(N)*U(N)*B(N)
  AL1=U(N)-A(N)
  AL1B=ABS(AL1+AL1A)
  R(1,N)=(R(1,N)+OM*(AL1B*R(1,N+1)-DN-DNA)/(1.+OM*AL1B)
  U(N)=.5*(R(2,N)-R(1,N))
  A(N)=GDH*(R(2,N)+R(1,N))
  DNA=DN
2 AL1A=AL1
  U(1)=(-GD*R(1,1)+SQRT(GC*GAMMA-GD*R(2,1)**2))/GE
  A(1)=SQRT(GAMMA-GD*U(1)**2)
  R(1,1)=GG*A(1)-U(1)
  R(2,1)=GG*A(1)+U(1)
  RETURN
  END
```

Fig 1. Subroutine for subsonic flow only

N	E	N^2E
4	1.118E-3	0.0179
8	2.818E-4	0.0179
16	7.038E-5	0.0180
32	1.659E-5	0.0170
64	2.830E-6	0.0116

N=number of intervals
E=mean quadratic error in u

TABLE I

$\omega\sigma$	work	E
1	1190	6.845E-5
2	697	7.011E-5
4	442	7.051E-5
8	544	7.058E-5
16	1921	7.082E-5
32	935	7.034E-5
64	1836	7.085E-5

TABLE II

Next, we test the speed of convergence. To this effect, we consider as a unit of work the work needed to process one mesh point in the two sweeps of each computational step. Therefore, the total work for one computation is the product of the total number of points (NC) by the total number of steps. We recompute the flow using 16 intervals and increasing values of $\omega\sigma$ (1, 2, 4, 8, 16, 32 and 64). Table II shows the work needed to reach convergence. In the computer used in these tests (PDP 11/23), we accept that convergence is reached when the mean square difference in u between two successive steps is less than $\tau=5\times10^{-7}$. For values of $\omega\sigma > 4$, however, we had to increase τ to 10^{-6}. In the same Table, we also report the mean quadratic error in u at convergence, to evidence that the results are not affected by the choice of $\omega\sigma$. It may be noted that $\omega\sigma=4$ is an optimum value. Increasing it not only hampers the convergence but creates oscillations in the results from step to step; a disturbing phenomenon, even if the oscillations are only of the order of 10^{-7}.

At this stage, we can try to compare the efficiency of the present calculation and of a standard time-dependent λ-scheme technique. The comparison, however, offers some difficulties, since the converged accuracy of the standard technique which we used is worse than the converged accuracy of the present technique, when the same number of intervals is used. For example, for the same data and the same number of intervals, the standard technique converges in about 250 steps. Since the number of multiplications needed at every point, that is the work per point, is about 4 times as large, we need thus a total equivalent work of the order of 17000. The economical gain (a factor of 40) is remarkable, but it is grossly underestimated. In fact, the mean quadratic error in the standard calculation is about 7×10^{-4}; with the present technique it is about 7×10^{-5}. To reach a similar accuracy with the standard technique at its present stage, we would need more points and, consequently, a much larger amount of work. For example, the work needed to reach an accuracy of 1.73×10^{-4} with the standard technique is equal to 80000 (almost 200 times as large as with the new technique), and the accuracy is still a quarter of an order of magnitude lower. We can tentatively conclude that the new technique is more efficient than the standard

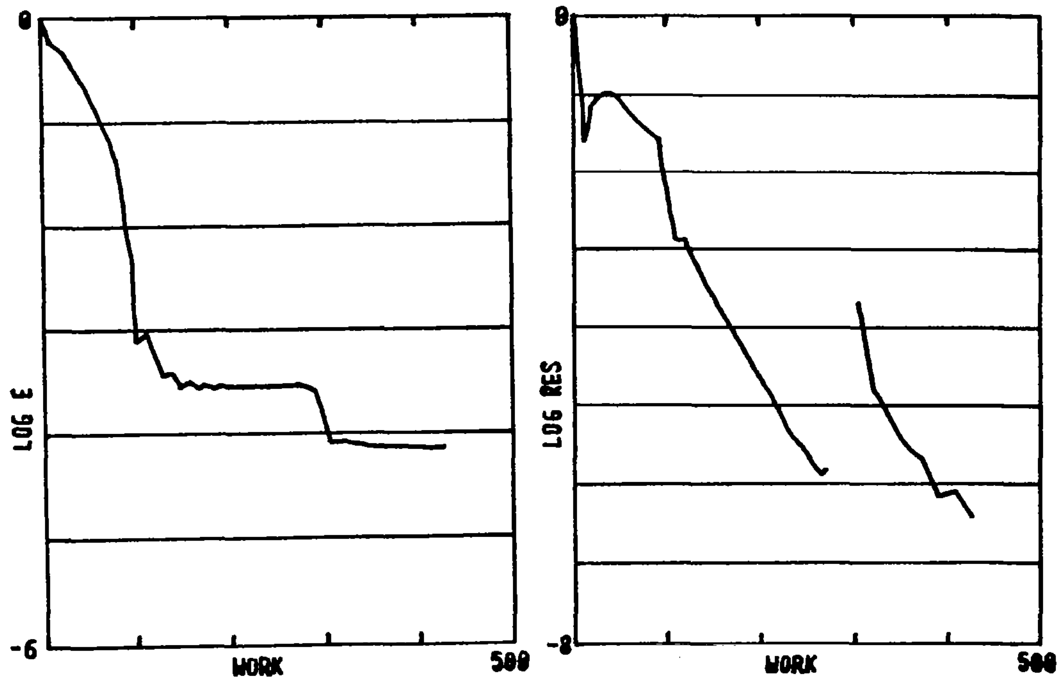


Fig 2. Mean square residuals and errors - Convergent duct

one by three orders of magnitude.

Finally, we can reduce the total work even more by working with a multiple grid. Since, in this case, we do not need more than 16 points to achieve a very high accuracy, we will use first a set of 8 intervals and then a set of 16, with an automatic switch when convergence is reached in the coarser mesh. In going from a coarser mesh to a finer one, we use second-order interpolations. The total work (in the case, $\omega\sigma=4$) equals 423. Mean square residuals and mean square errors are plotted in Fig. 2 for this case; work is on the abscissa. In Fig. 3 and 4 isobars are plotted for the present technique with x on the abscissa and work on the ordinate, and for the standard calculation with x on the abscissa and time on the ordinate. It is clear that the gain in convergence is achieved to the expense of an accurate description

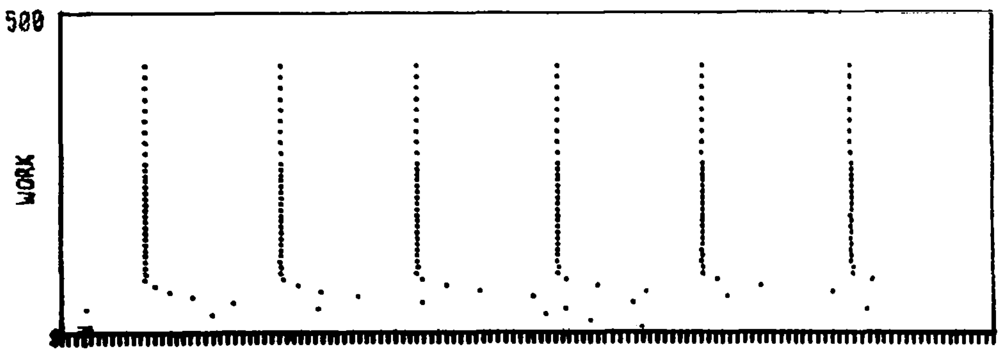


Fig 3. Isobars in the (x,work)-plane - Convergent duct
Accelerated technique

of the unsteady evolution.

The second case is similar to the first but only apparently as simple. The duct is divergent, with a linear Mach number distribution between 0.8 and 0.2. We use the same code and the same type of initial conditions as in the preceding case. Convergence is much harder to reach now, regardless of the computational technique. For conciseness, we present only results obtained with a multiple grid (8 and 16 intervals), and increasing values of $\omega\sigma$. Our results are summarized in Table III. Mean square residuals and mean square errors in u are plotted in Fig. 5 for the case $\omega\sigma=64$; work is on the abscissa. We have also tried a run with three levels of mesh refinement (8, 16 and 32 intervals, successively), and one with four levels, up to 64 intervals. The value of τ in all these cases was 5×10^{-7}. In the three-level run, a total work of 5497 was needed to reach $E=1.06\times10^{-4}$. In the four-level run, the computation stopped after a work of 6797,

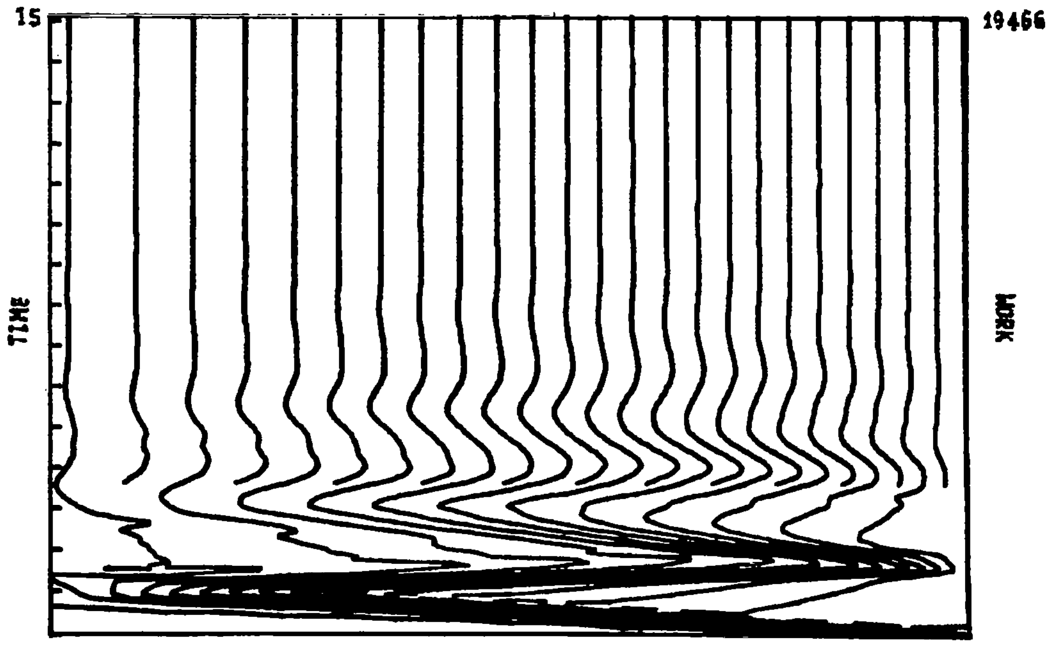


Fig. 4. Isobars in the (x,t)-plane - Convergent duct Standard technique

with $E=5.88\times10^{-5}$. The theoretical accuracy of 2.65×10^{-5} for a second-order accurate technique was not reached, but for this one needs a computer working on more digits and a lower value of τ. As far as the standard technique is concerned, we have run a case with the same number of intervals and reached convergence after a work equal to 160000 but with an accuracy of only 6×10^{-3}. Using 32 intervals, we reached an accuracy of 1.8×10^{-3} with a total work equal to 528000.

$\omega\sigma$	work	E
1	14850	5.48E-4
2	8107	4.96E-4
4	5049	4.96E-4
8	4265	4.49E-4
16	3857	4.49E-4
32	3311	4.49E-4
64	3154	4.49e-4

TABLE III

Again, the efficiency of the present technique is at least three orders of magnitude better than the efficiency of the standard technique. For the sake of comparing the acceleration in convergence during the initial phase of the calculation, in Fig. 6 and 7 isobars are plotted for the present technique with x on the abscissa and work on the ordinate, and for a standard calculation with x on the abscissa and equivalent work (that is, number of

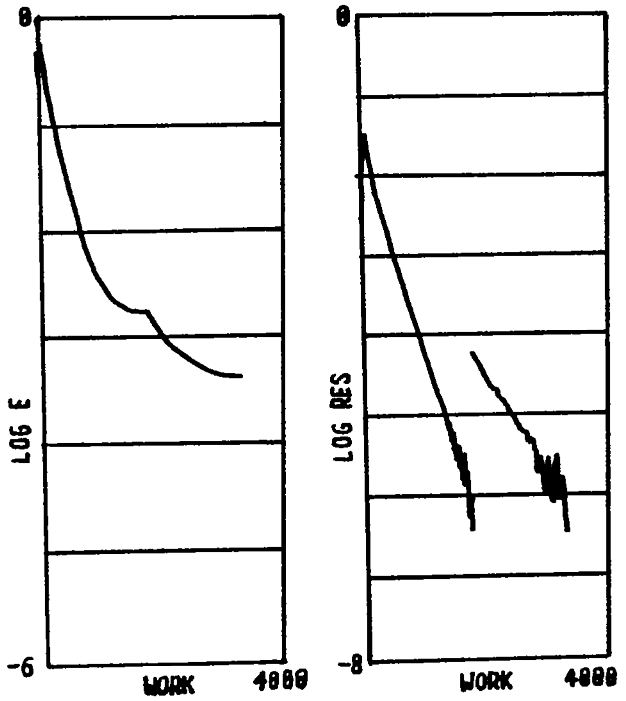


Fig. 5. Mean square residuals and errors - Divergent duct

points times number of steps times 4) on the ordinate.

Finally, we consider a third case, in which a symmetric, parabolic Mach number distribution is assigned, with initial and final values of 0.2 and a maximum value of 0.9. The corresponding duct is now a symmetric nozzle with a throat. In this case, it took a work equal to 15406 to reach an accuracy of 3.44×10^{-3}, using a mesh of 16 intervals initially and then, for very few steps, a mesh of 32 intervals. The value of $\omega\sigma$ in this case was 2. In Fig. 8 the exact distribution of the Mach number is shown by a solid line, and the computed values are denoted by O's.

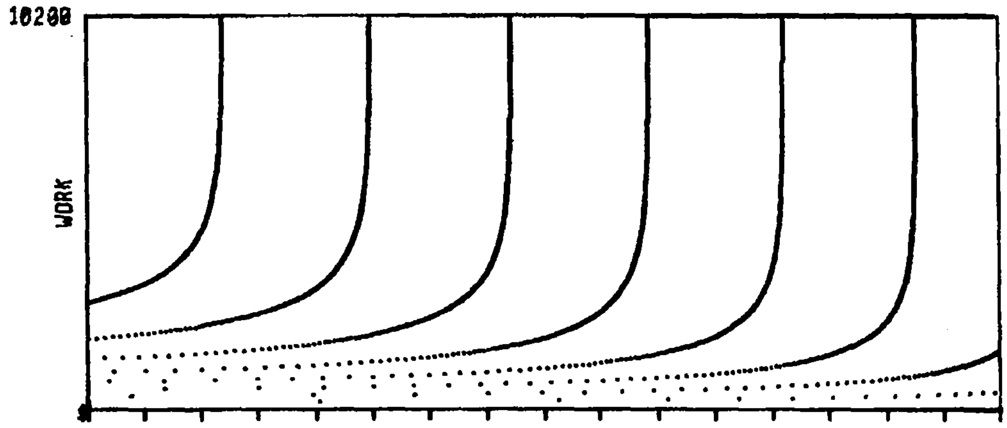


Fig. 6. Isobars in the (x,work)-plane - Divergent duct
Accelerated technique

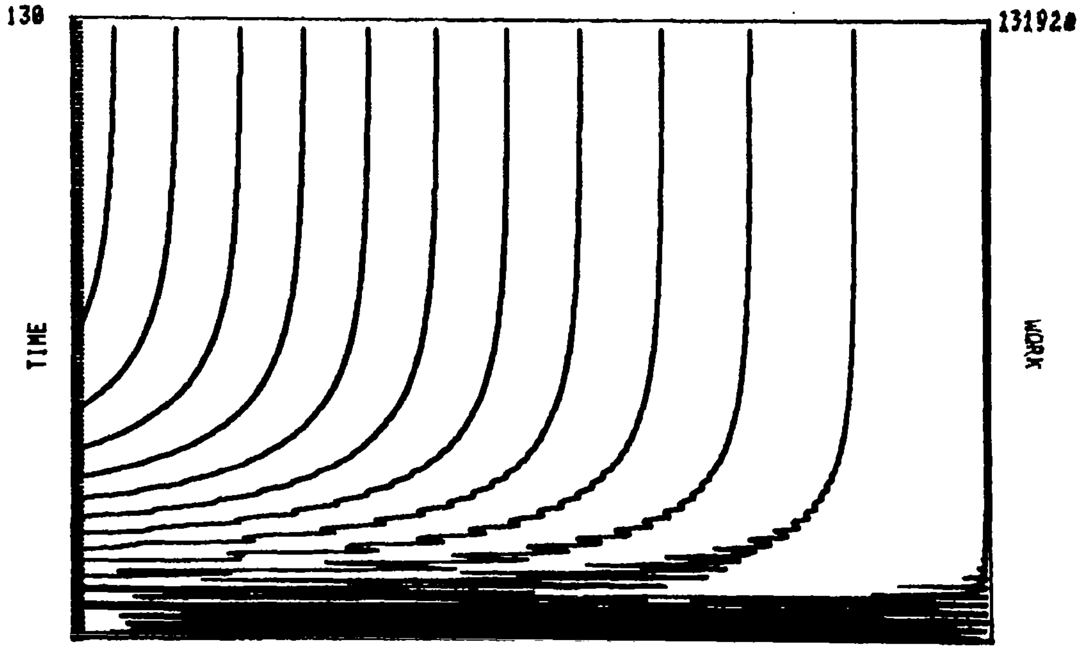


Fig. 7. Isobars in the (x,t)-plane - Divergent duct
Standard technique

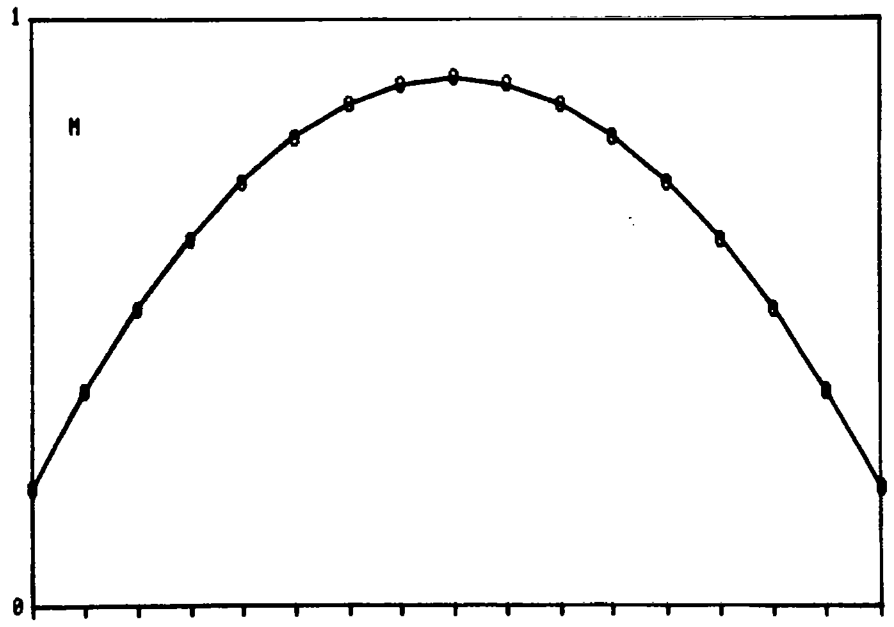


Fig. 8. Exact and computed $M(x)$ for a symmetric nozzle with a throat

5. Technique for transonic flows

In Section 2, it was explained how the sweeping integration concept originated from the physical evidence of right-running R_2-waves and left-running R_1-waves. Consistency of numerics and physics in the handling both of interior points and of boundary conditions is, in our opinion, responsible for the high efficiency of the code, both in speed and accuracy. In this respect, it may be interesting to note that the technique described above is a hybrid between explicit and implicit techniques. It seems that the wave character of the unsteady flow has to be preserved in order to maintain a steady pattern which, per se, does not show an explicit propagation of waves; minute perturbations are easily carried away to the boundaries and properly adjusted.

Such considerations should be kept in mind when attempting an extension of the technique to transonic flows. Here we describe a procedure for a case where a supersonic region is comprised between two subsonic regions. The interest of the case resides not only in its appearance in Laval nozzles with shocks, but in its close relation with patterns about supercritical airfoils.

A few considerations are in order, before outlining a technique.

1) We assume that the supersonic region is bounded by a shock downstream. This is, indeed, the only case of practical interest.

2) In a steady configuration, the entropy has only two values; one to the left of the shock (which can be assumed as zero) and one to the right of the shock.

3) The R_2-waves always propagate from left to right.

4) The R_1-waves propagate from right to left in subsonic regions and from left to right in the supersonic region.

5) R_2 is discontinuous across the shock. Its jump, however, is very small. Therefore, the value of R_2 behind the shock may be computed applying (20), but using only local information for d and λ_2, and then incrementing it by its shock jump, as given below.

6) The shock makes the set of values of R_1 on its left and the set of values of R_1 on its right totally independent of one another, a fact which is automatically reflected by the code, when it is written to satisfy property 4) above.

7) The jump in R_1 across the shock, ΔR_1, divided by the speed of sound in front of the shock, a_{s1}, is a single-value function of the relative Mach number of the shock, defined by

$$M = \frac{u_{s1} - W}{a_{s1}} \qquad (22)$$

where u_{s1} is the velocity in front of the shock and W is the shock velocity. Therefore, if a_{s1} and ΔR_1 are known, M is also known.

8) The sign of λ_1 changes across the sonic point. Consequently, the updating of R_1 in the subsonic region at the left should start at the last subsonic point, without using information from the next supersonic point, and its updating in the supersonic region should start at the first supersonic point, without using information from the subsonic region (the information is transmitted by R_2).

The code for the transonic problem, shown in Fig. 9, is more complicated than the subsonic code shown in Fig. 1. The total number of operations, however, is not increased. A few comments will facilitate the reading of Fig. 9. The symbols, NSON and NS, denote the value of N at the first supersonic point and at the first point following the shock, respectively; DR2 is the jump of R_2 across the shock. Loop 21 determines the location of the sonic point at every step. The calculation of R_1 starts at N=NC and proceeds backwards until the point denoted by NS has been computed; then it jumps to the statement labeled 4, which resumes the calculation at the last subsonic point in the first region. The calculation proceeds backwards again until point N=1 is evaluated; then it jumps to the statement labeled 6, which resumes the calculation at the first supersonic point and proceeds forward until the last supersonic point has been computed. Finally, the shock is evaluated, the point behind it is corrected accordingly (in a separate subroutine) and the boundary condition at N=1 is applied as in the subsonic code. The exit boundary condition is imposed here by giving the logarithm of the exit pressure, PE (the pressure of the gas at rest being 1) and using the exit entropy, S(NC) as equal to the entropy produced by the shock.

6. Shock calculation

Let us assume, for argument's sake, that we know in which interval the shock is located. Then, all points to the left of it are computed properly by our technique; indeed, R_2 depends only on the left boundary condition and R_1 is not influenced by the right boundary condition, due to the presence of the shock (recall property 6) above). To evaluate properly the points to the right of the shock, however, two elements are needed: the

SUBROUTINE FOR TRANSONIC FLOWS

```
      DNA=A(1)*U(1)*B(1)
      AL2A=U(1)+A(1)
      DO 1 N=2,NC
      DN=A(N)*U(N)*B(N)
      DR2A=0.
      IF(N.NE.NS)GO TO 15
      DNA=DN
      AL2A=U(N)+A(N)
      DR2A=DR2
   15 AL2=U(N)+A(N)
      DNB=DN+DNA
      AL2B=AL2+AL2A
      DNA=DN
      R(2,N)=R(2,N-1)-DNB/AL2B+DR2A
      U(N)=.5*(R(2,N)-R(1,N))
      A(N)=GDH*(R(2,N)+R(1,N))
    1 AL2A=AL2
      NSON1=2
      NSON2=NC
      IF(NSON.LE.NSON1+3)GO TO 22
      NSON1=NSON-3
      NSON2=NSON+1
   22 DO 21 N=NSON1,NSON2
   21 IF(U(N).GE.A(N).AND.U(N-1)
     1.LT.A(N-1)) NSON=N
      ANC=SQRT(GAMMA*EXP(PE/GA+2.
     1*GD*S(NC)))
      U(NC)=U(NC)+(A(NC)-ANC)*GG
      A(NC)=ANC
      R(1,NC)=A(NC)*GG-U(NC)
      DNA=A(NC)*U(NC)*B(NC)
      AL1A=U(NC)-A(NC)
      N=NC
      LITE=-1
    3 N=N-1
      IF(N.EQ.NS-1) GO TO 4
      IF(N.LE.0) GO TO 6
   55 N1=N+1
    5 AL1=U(N)-A(N)
      DN=A(N)*U(N)*B(N)
      DNB=DNA+DN
      AL1B=AL1+AL1A
      DNA=DN
      AL1B=ABS(AL1B)
      R(1,N)=(R(1,N)-OM*(-AL1B*
     1R(1,N1)+DNB))/(1.+OM*AL1B)
      U(N)=.5*(R(2,N)-R(1,N))
      A(N)=GDH*(R(2,N)+R(1,N))
      AL1A=AL1
      IF(LITE.EQ.1)GO TO 7
      GO TO 3
    4 N=NSON-1
      DNA=A(N)*U(N)*B(N)
      AL1A=U(N)-A(N)
      GO TO 55
    6 IF(NS.EQ.-10)GO TO 13
      N=NSON-1
      DNA=A(NSON)*U(NSON)*B(NSON)
      AL1A=U(NSON)-A(NSON)
      LITE=1
    7 N=N+1
      IF(N.EQ.NS) GO TO 8
      N1=N-1
      GO TO 5
    8 CALL SHOCK
   13 U(1)=(-GD*R(1,1)+SQRT(GC*GAMMA
     1-GD*R(1,1)**2))/GE
      A(1)=SQRT(GAMMA-GD*U(1)**2)
      R(2,1)=GG*A(1)+U(1)
      R(1,1)=GG*A(1)-U(1)
      RETURN
      END
```

Fig. 9. Subroutine for transonic flows

value of R_2 at the point called NS, from which the values of R_2 at all the following points depend, and the value of S which, combined with the exit pressure, defines the exit speed of sound and, consequently, the proper exit boundary condition and R_1 from the exit point to the point behind the shock.

For a correct solution of the problem, thus, the shock must be fitted. Assuming, again, that the abscissa, x_s, of the shock is known, R_1 and R_2 can be extrapolated from upstream to the shock point in order to find the values of u and a at point A immediately in front of the shock (called u_A and a_A, respectively), and R_1 can be extrapolated from downstream to find its value, R_{1s}, immediately behind the shock. Let M be the relative Mach number of the shock, as defined in (22), with u_A and a_A in lieu of u_{s1} and a_{s1}, respectively; and let

$$\Sigma = \Delta R_1/a_A \tag{23}$$

Then, M and Σ are related by the equation:

$$\Sigma = \frac{[(\gamma M^2-\delta)(1+\delta M^2)]^{1/2} + \delta(M^2-1)}{\delta(1+\delta)M} - \frac{1}{\delta} \tag{24}$$

which can be approximated, for $1 \leq M \leq 4$, by the linear function:

$$M = 1 + g_1 \Sigma \tag{25}$$

where g_1 is defined by computing (25) at M=4:

$$g_1 = \frac{4\delta(1+\delta)}{11\delta-4+[(16\gamma-\delta)(1+16\delta)]^{1/2}}$$

Therefore, M can be found with four correct digits by first using (25), then by computing a correction to Σ, using a secant method:

$$\Sigma_1 = 2\,\Sigma - \Sigma'$$

where Σ' is obtained from (24), and finally by using (25) again:

$$M = 1 + g_1 \Sigma_1$$

Having so found the shock Mach number, three important quantities can be evaluated: the entropy behind the shock,

$$S = \frac{1}{2\gamma\delta}\left[\ln\frac{\gamma M^2-\delta}{1+\delta} - \gamma\ \ln\frac{(1+\delta)M^2}{1+\delta M^2}\right] \tag{26}$$

the jump in R_2 across the shock:

$$\Delta R_2 = a_A\left[\Sigma + \frac{2\,(1-M^2)}{(1+\delta)M}\right] \tag{27}$$

and the shock velocity, W, given by (22). The new value of R_2 at N=NS is thus found by first getting R_2 at point B immediately behind the shock,

$$R_{2B} = R_{2A} + \Delta R_2 \tag{28}$$

and then by interpolation:

$$R_{2,NS} = \frac{R_{2B} + \eta R_{2,NS+1}}{1 + \eta} \tag{29}$$

where

$$\eta = (x_{NS} - x_s)/\Delta x \tag{30}$$

Having now both R_1 and R_2 at point N=NS, u and a can be recomputed at the same point.

The shock velocity has no direct meaning, in a computation which aims to describe a steady state, where W is obviously zero. Nevertheless, it plays a very important role in moving the shock before convergence is reached. It is important to move the shock in order to locate it in the proper interval. To this effect, x_s is updated at every step by adding $W\,\Delta t$ to it, multiplied by a factor, α. The effect of such a factor is simply to accelerate the motion of the shock (a value of 3 seems to be convenient); it has obviously no effect on the steady state location of the

SHOCK SUBROUTINE

```
      EPS=(XS-X(NS-1))*DDX
      ETA=1.-EPS
      R1S=R(1,NS)+ETA*(R(1,NS)-R(1,NS+1))
      AS1=A(NS-1)+EPS*(A(NS-1)-A(NS-2))
      US1=U(NS-1)+EPS*(U(NS-1)-U(NS-2))
      R2S1=GG*AS2+US2
      DR1A=(R1S-R2S1+US1+US1)/AS1
      EM=1.+GS1*DR1A
      SMIN=EM**2
      DR1B=2.*DR1A-((SQRT((GAMMA*SMIN-GD)*(1.+GD*SMIN))+GD*
     1(SMIN-1.))/(GE*EM)-1.)*GG
      EM=1.+GS1*DR1B
      SMIN=EM**2
      DUM=GE*EM
      DIM=GD*SMIN
      DEM=GAMMA*SMIN-GD
      W=US1-AS1*EM
      XS=XS+W*DX*ALPHA
      SS2=(ALOG(DEM/GE)-GAMMA*ALOG(GE*SMIN/(1.+DIM)))/(2.*GD*GAMMA)
      DR2=AS1*(DR1A+2./(GE*EM)*(1.-SMIN))
      R2S2=R2S1+DR2
      R(2,NS)=(R2S2+ETA*R(2,NS+1))/(2.+ETA)
      U(NS)=.5*(R(2,NS)-R(1,NS))
      A(NS)=GDH*(R(2,NS)+R(1,NS))
      IDUM=XS*NA
      NS=IDUM+2
      DO 1 N=NS,NC
    1 S(N)=SS2
      IF(NS.EQ.NSO)RETURN
      IF(NS.GT.NSO)GO TO 14
      R(1,NS)=2.*R(1,NSO)-R(1,NSO+1)
      R(2,NS)=2.*R(2,NSO)-R(2,NSO+1)
      GO TO 3
   14 R(1,NSO)=2.*R(1,NSO-1)-R(1,NSO-2)
      R(2,NSO)=2.*R(2,NSO-1)-R(2,NSO-2)
    3 NSO=NS
      U(NS)=.5*(R(2,NS)-R(1,NS))
      A(NS)=GDH*(R(2,NS)+R(1,NS))
      RETURN
      END
```

Fig. 10. Shock calculation

shock, since W vanishes.

A code for the shock calculation is shown in Fig. 10. There, NSO is the value of NS at the preceding step, DDX=$1/\Delta x$ and GS1=g_1. The last part of the subroutine provides a smooth transition when the shock point travels from one interval to the next, in either direction.

We conclude by showing how the presence of a shock is detected for the first time. So long as the flow is shockless, every interval is monitored by computing the pertinent value of Σ and the corresponding M, defined by (25). A shock is said to exist in the first interval for which $M \geq 1.1$. Its abscissa is then defined as the middle point of the interval and the shock is tracked, from that moment on, as described above.

7. Results for flows with shocks

To test the code, we chose a nozzle so defined that, in the absence of shock, the Mach number would be linearly distributed between 0.5 and 2.5. Then we assume that a shock is present at x=0.63 (the total length of the nozzle being 1). In such a way, the subsonic flow behind the shock can be evaluated, and the exit pressure to generate it is made known. The logarithm of the latter is PE, mentioned in Section 5. In Table IV, we summarize results of runs made using 8, 16, 32 and 64 intervals. The total work, the shock location (x_s), the shock Mach number (M), and the mean quadratic error in u (E) are given.

N	work	x_s	M	E
8	900	0.6244	1.770	3.09E-1
16	3400	0.6256	1.752	5.32E-3
32	9180	0.6281	1.756	3.11e-4
64	16445	0.6295	1.757	1.36E-4

TABLE IV

Note that the exact location of the shock, as said above, is 0.6300, and the exact shock Mach number is 1.7600. Another run was made, taking 30 steps with 8 intervals, 50 steps with 16 intervals and then letting the number of steps for 32 and 64 intervals be controlled by $\tau=5\times10^{-7}$ (for fewer intervals, due to sizeable oscillations of the shock, the mean quadratic residual never drops below 10^{-4}). After a work equal to 16445, the computation stops with the shock located at 0.6285, and a shock Mach number equal to 1.7571. The mean quadratic error in u is now 1.37×10^{-4}. In Figs. 11 and 12 we show the mean quadratic error and the mean quadratic residual for the latter case, as functions of the work. In Figs. 13 through 16 we show the exact and computed Mach number distribution, entropy distribution, mass flow and stagnation speed of sound squared (as before, exact values are shown by solid lines and computed values by O's). The only graph showing a flaw is the one for the mass flow. Errors generated at the shock are equivalent to a mass source. It must be noted, however, that the error in the mass flow is only 2/10 of 1%. Finally, Fig. 17 shows how fast the correct shock location is reached (x_s is on the ordinate; work is on the abscissa).

Once more, we wish to compare the speed of convergence of the present technique with the speed of convergence of a standard technique. The results are presented in Figs. 18 and 19; Fig. 18 shows the isobars for the present run and Fig. 19 shows the isobars for a really time-dependent run. The work spent in reaching the top of Fig. 19 is equal to 156000 (one order of magnitude larger than with the present technique); but the mean quadratic error in u is 7.74×10^{-3} (almost two orders of magnitude larger).

It should be noted that minor changes in the coding, which do not produce different truncation errors but merely different round-off errors, may affect the location of the shock and its Mach number, or the mass flow and the total temperature <u>downstream of the shock</u> by about 1/10 of 1%. For example, if (27) is substituted by the equivalent expression:

$$\Delta R_2 = \frac{a_{s1}}{\delta(1+\delta)M}\left(\left[(\gamma M^2-\delta)(1+\delta M^2)\right]^{1/2}-(1+\delta)M+\delta-\delta M^2\right) \quad (31)$$

and the last run is repeated, the converged distributions of mass flow and total temperature are as shown in Figs. 20 and 21. It is, indeed, hard to draw a conclusion from experiments of this kind. It is clear that, upstream of the shock, the accuracy can

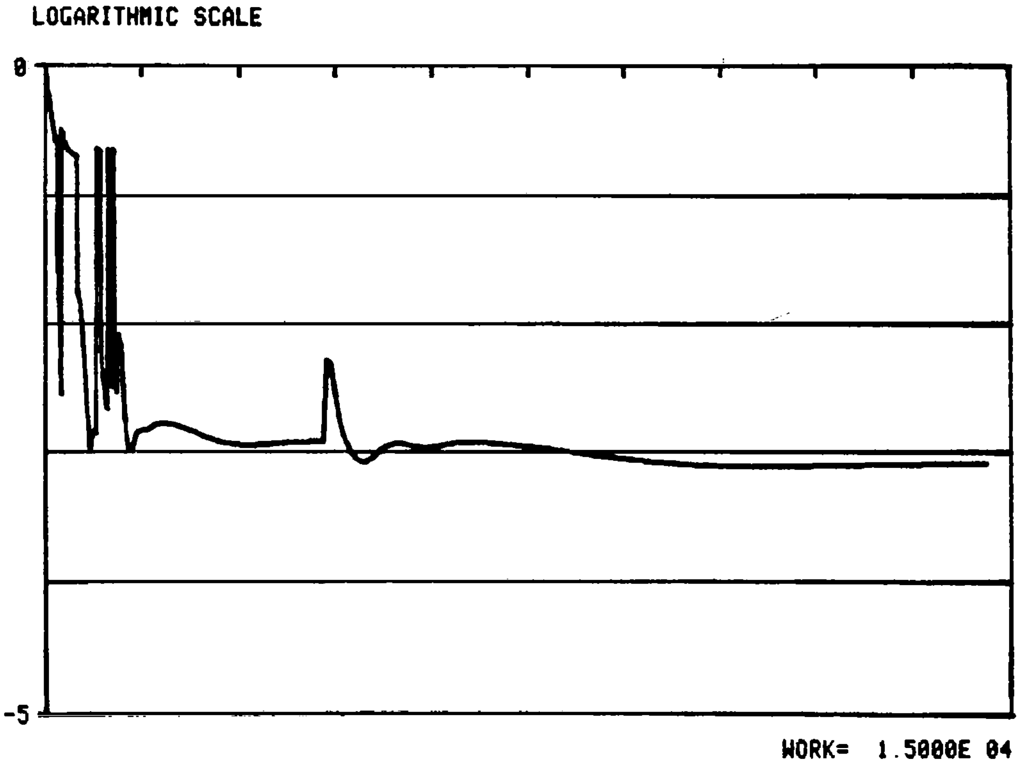


Fig. 11. Mean square error for flow with shock - Multiple grid

be controlled as simply and efficiently as in the subsonic cases. One can see from the isobar plots (Fig. 18, for example) that minor variations at the shock upset the entire flow downstream of it. Any attempt to fix the problem, however, produces results which are exposed to the same degree of randomness as the calculations mentioned above, because this is in the nature of things. In practice, we may conclude that, so long as an error less than 1/10 of 1% in the shock or behind it is irrelevant, all our results are identical, regardless of minor changes in the coding, changes in factors such as $\omega\sigma$, α or β, and changes in the initial conditions.

We have also conducted tests using different Mach number distributions (that is, different geometries) and different shock

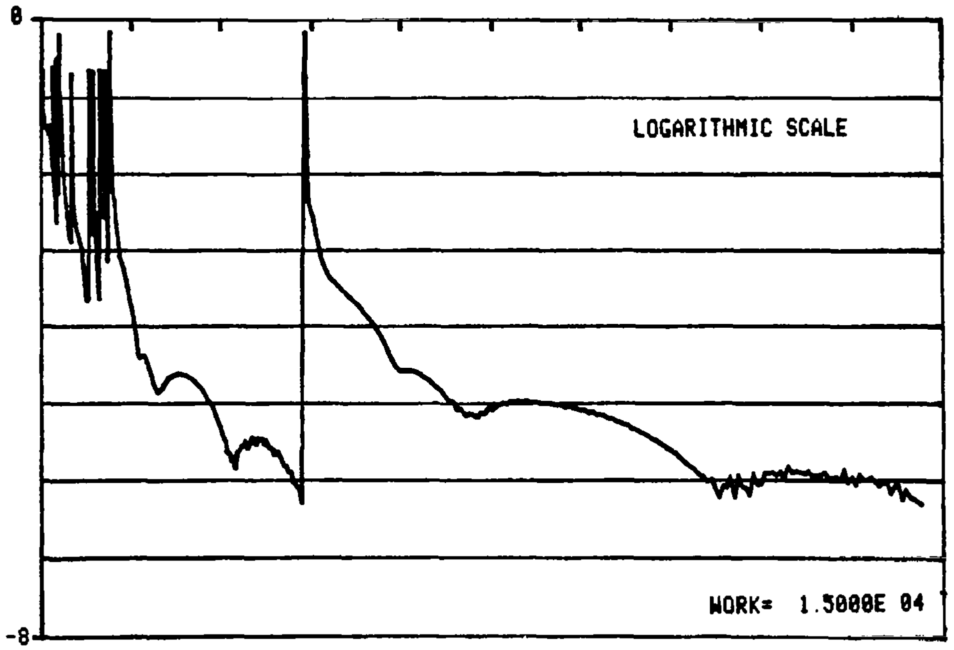


Fig. 12. Mean square residual for flow with shock - Multiple grid

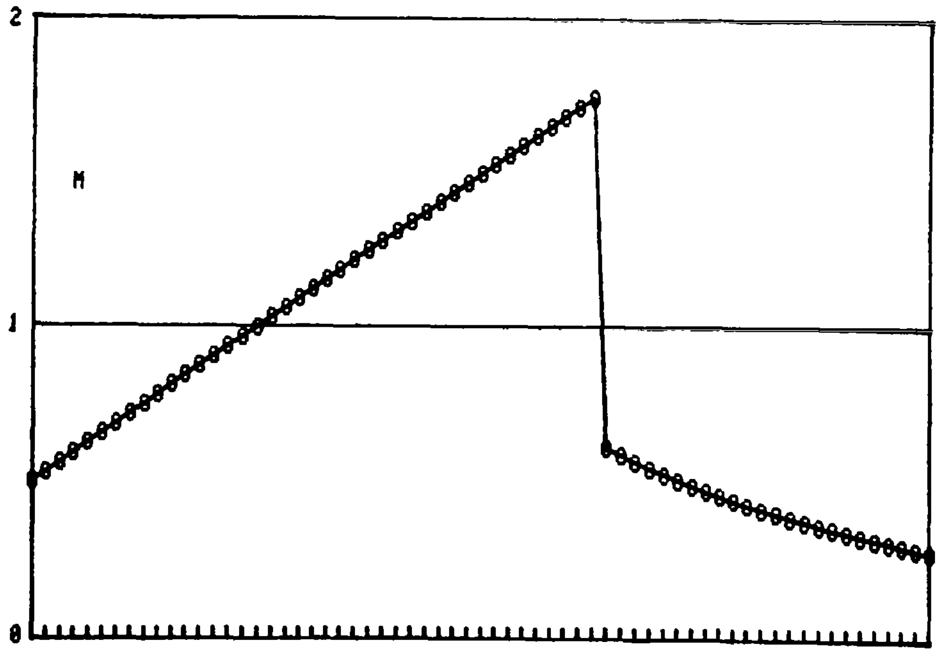


Fig. 13. Exact and computed M(x) for flow with shock

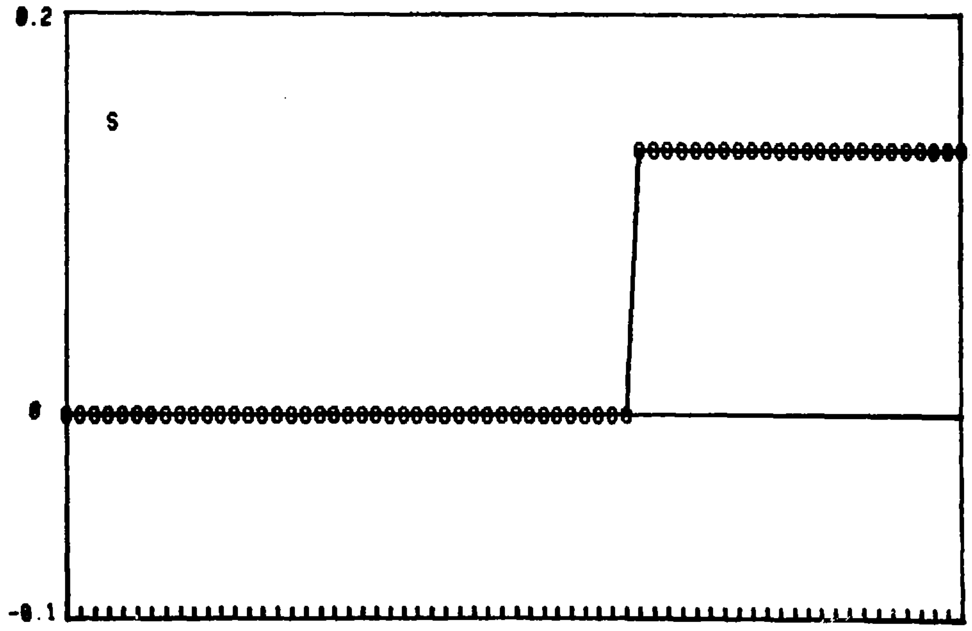


Fig. 14. Exact and computed S(x) for flow with shock

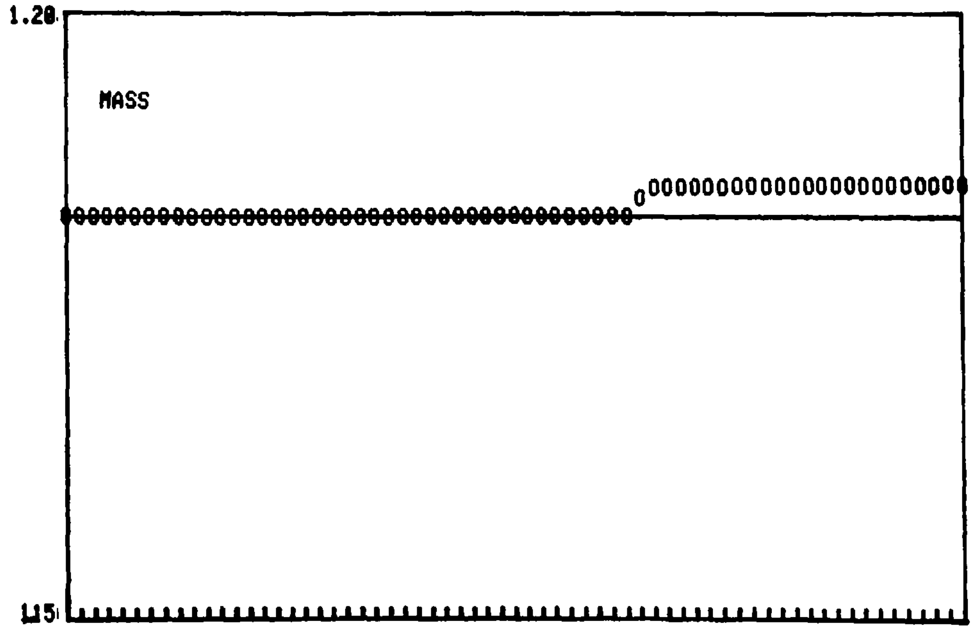


Fig. 15. Exact and computed mass flow for flow with shock

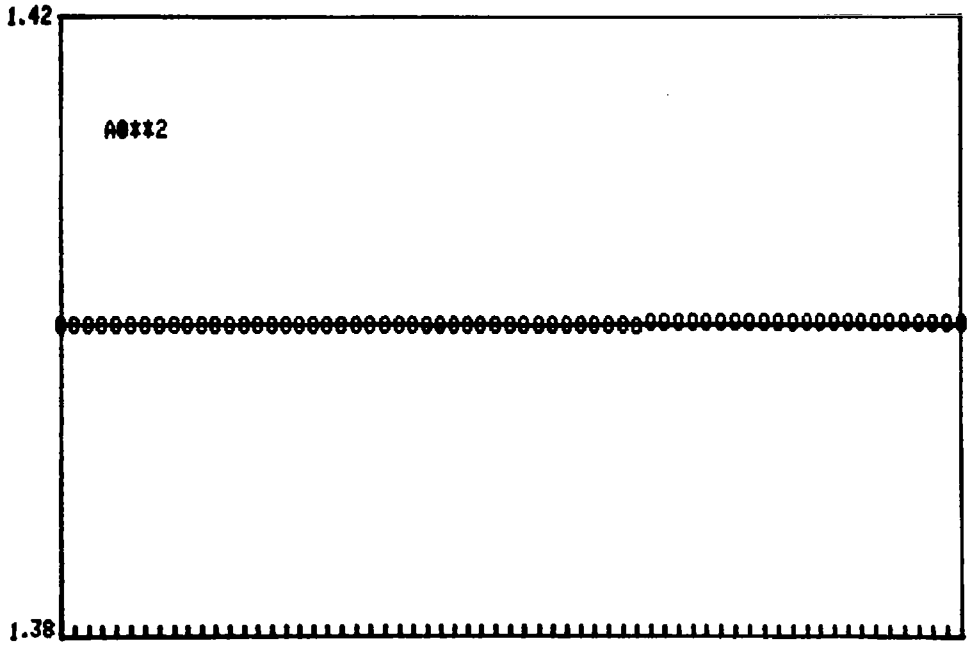


Fig. 16. Exact and computed $a_o^2(x)$ for flow with shock

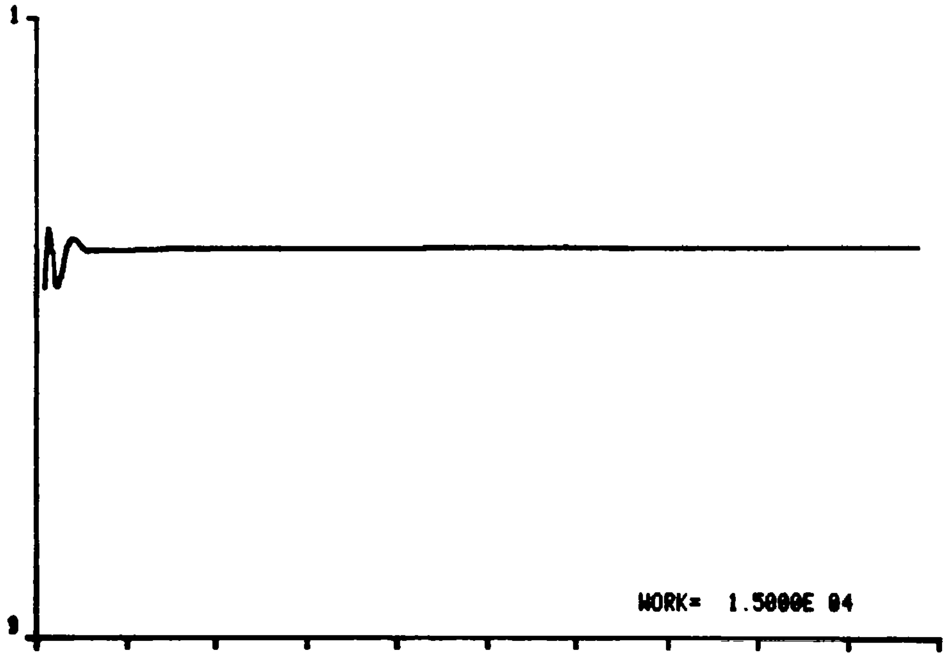


Fig. 17. Shock location as a function of work

locations, and we have found similar results. For example, with the same geometry as above, we obtained the results summarized in Table V.

shock location		shock Mach number	
theor.	comp.	theor.	comp.
0.6000	0.5999	1.700	1.700
0.6100	0.6095	1.7200	1.712
0.6200	0.6218	1.7400	1.741
0.6250	0.6258	1.7500	1.748

TABLE V

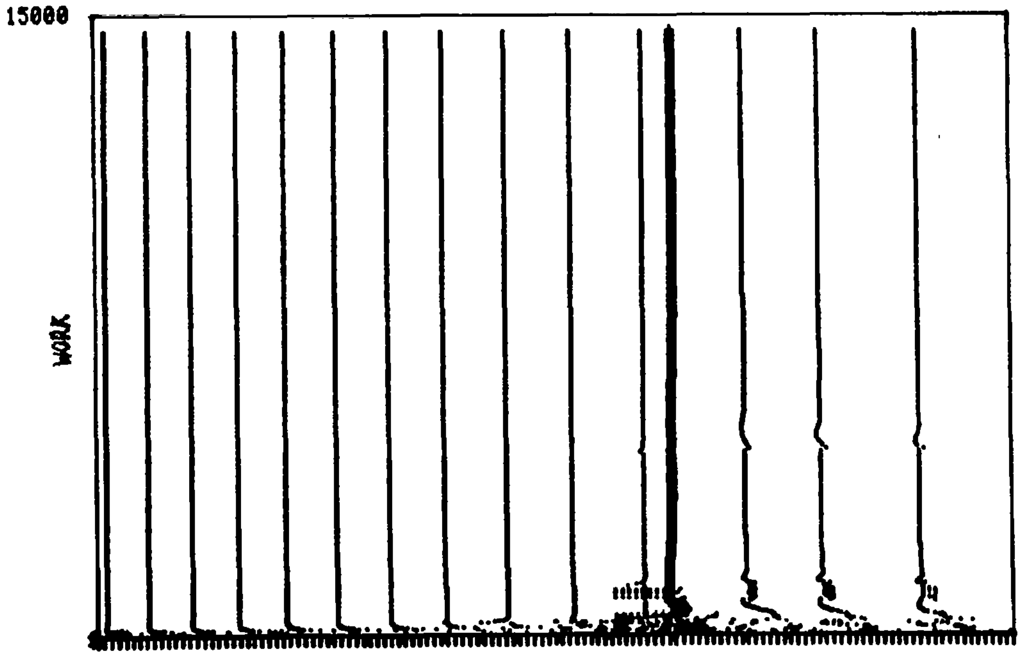


Fig. 18. Isobars in the (x,work)-plane - Nozzle with shock
Accelerated technique

8. Conclusions

We have presented a technique for the numerical treatment of one-dimensional, steady, inviscid flows. The technique is extremely fast and accurate. The total labor involved in the cal-

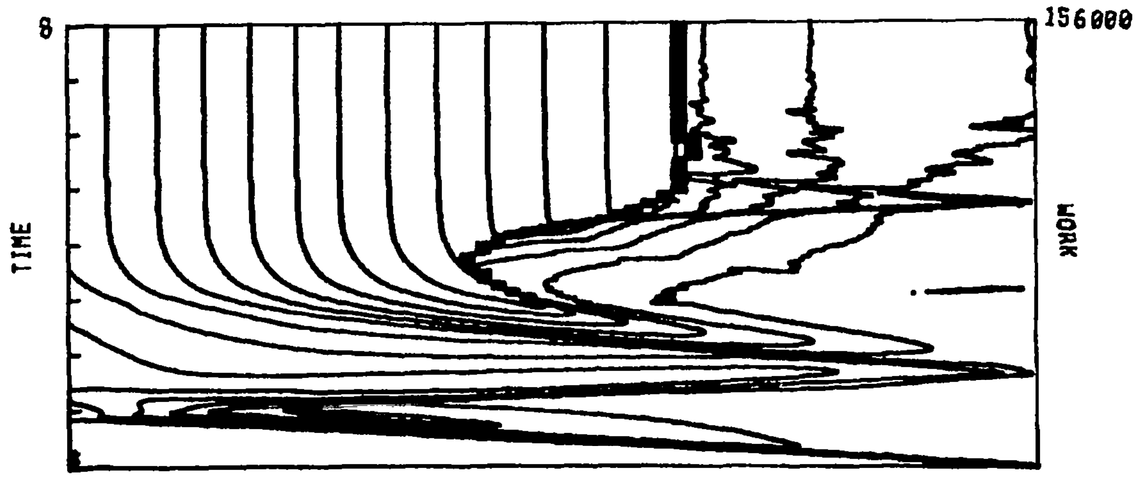


Fig. 19. Isobars in the (x,t)-plane - Nozzle with shock
Standard technique

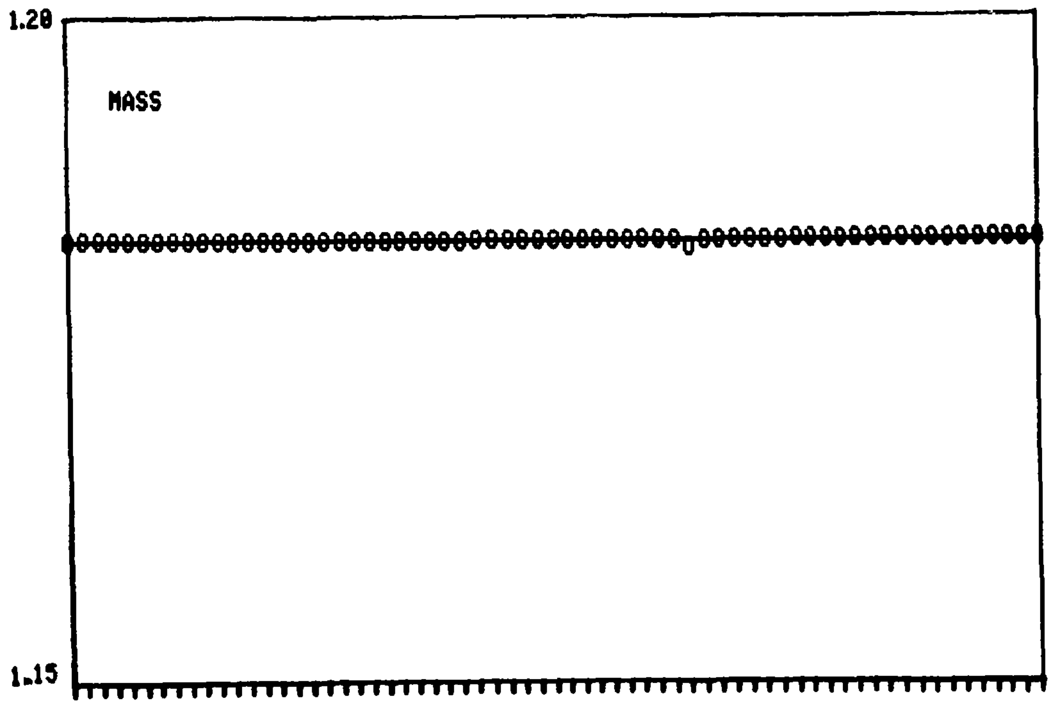


Fig. 20. Exact and computed mass flow for flow with shock

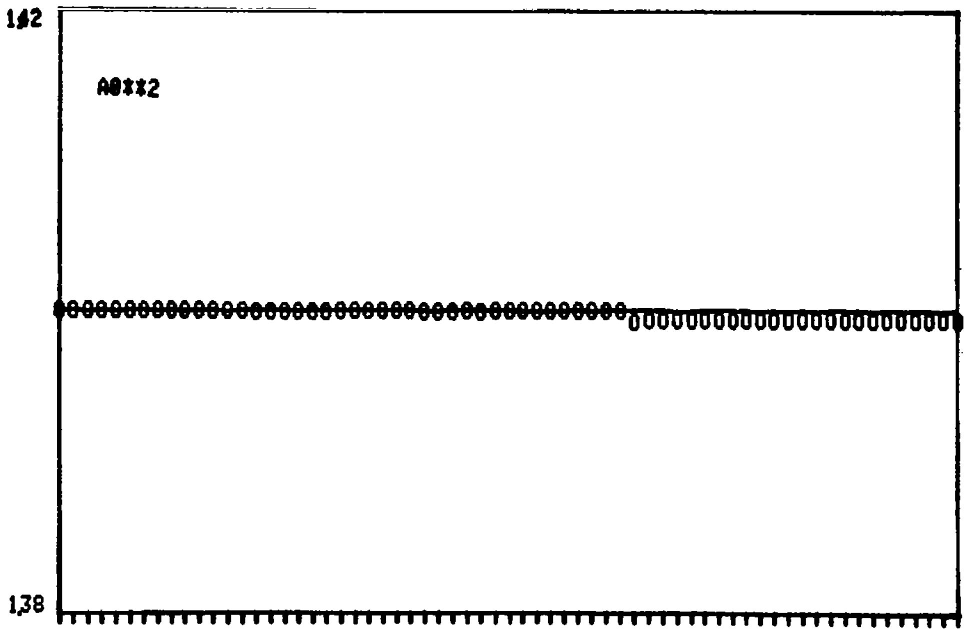
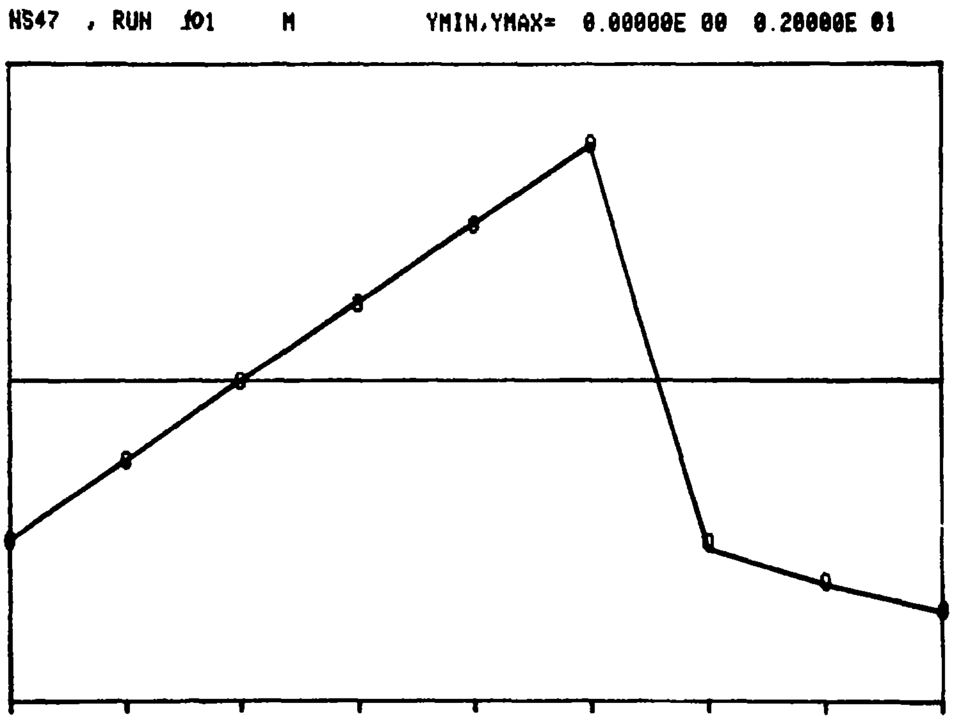


Fig. 21. Exact and computed $a_o^2(x)$ for flow with shock

culation (and, consequently, the total running time) is at least three orders of magnitude smaller than for a standard, time-dependent technique with the same degree of accuracy. Shocks are fitted with great accuracy also. It is important to note that our estimates of accuracy are not made by comparing numerical results of runs made with different number of intervals, but by comparing the computed results to exact solutions. Therefore, our "errors" are real errors, and should not be confused with what is call "error", at times, by other authors. The latter is instead what we called "residual", that is, the mean square difference between values at one step and values at the preceding step.

One final word should be spent on the subject of resolution. We have seen, indeed, that good results in flows with shocks are obtained when a relatively large number of nodes is used (in our examples above, we have used a maximum of 64 intervals), but reasonable results are also obtained with fewer intervals, as Table IV shows. Some deterioration on local values of sensitive parameters, such as mass flow and total temperature, may be ex-



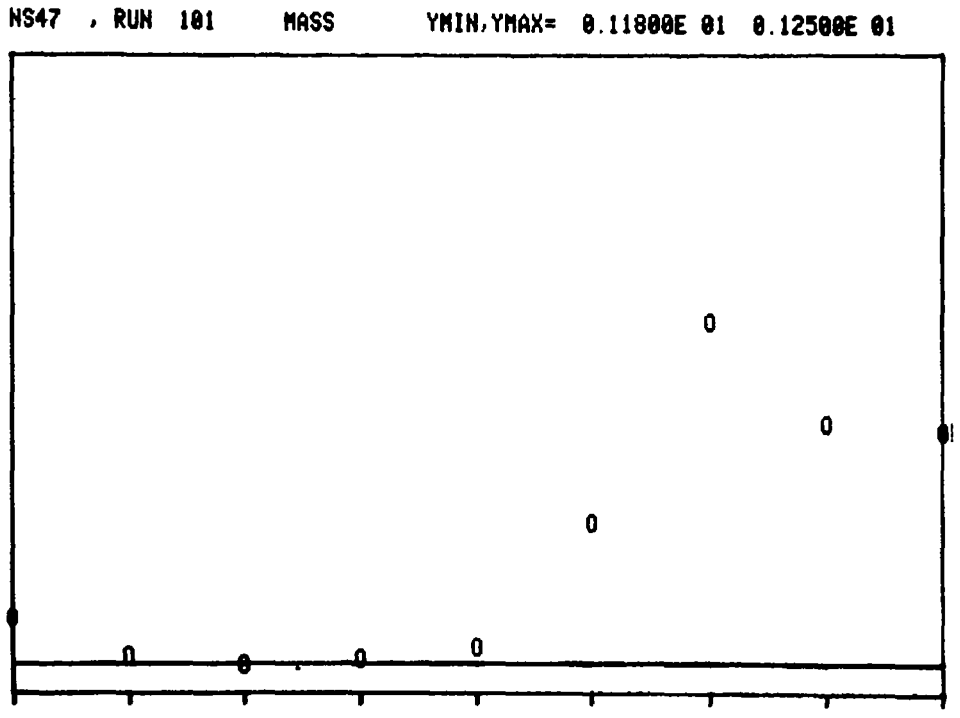
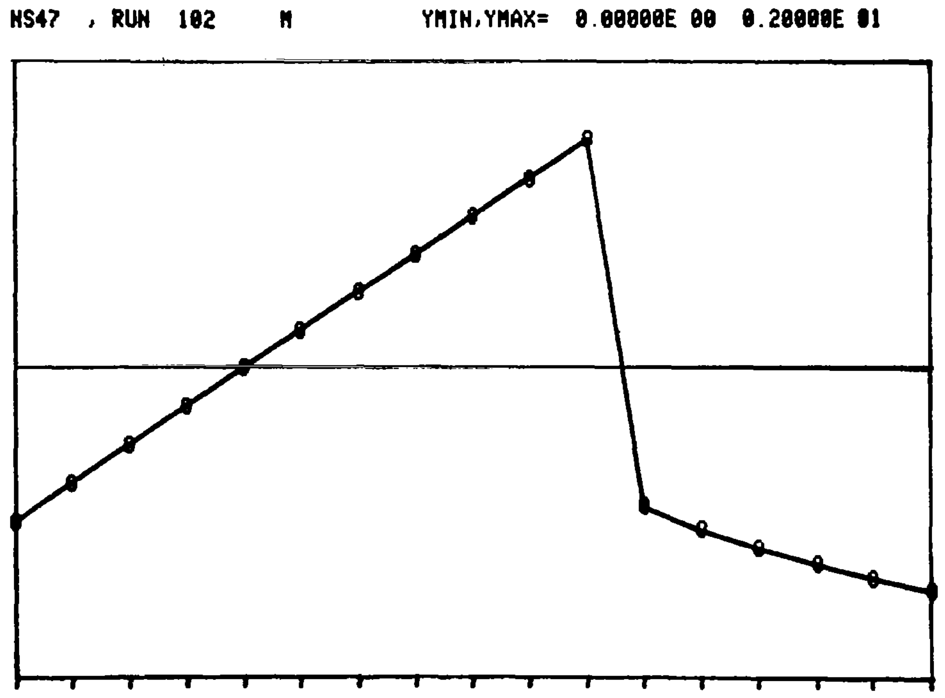


Fig. 22. Exact and computed M(x) and mass flow for flow with shock 8 intervals



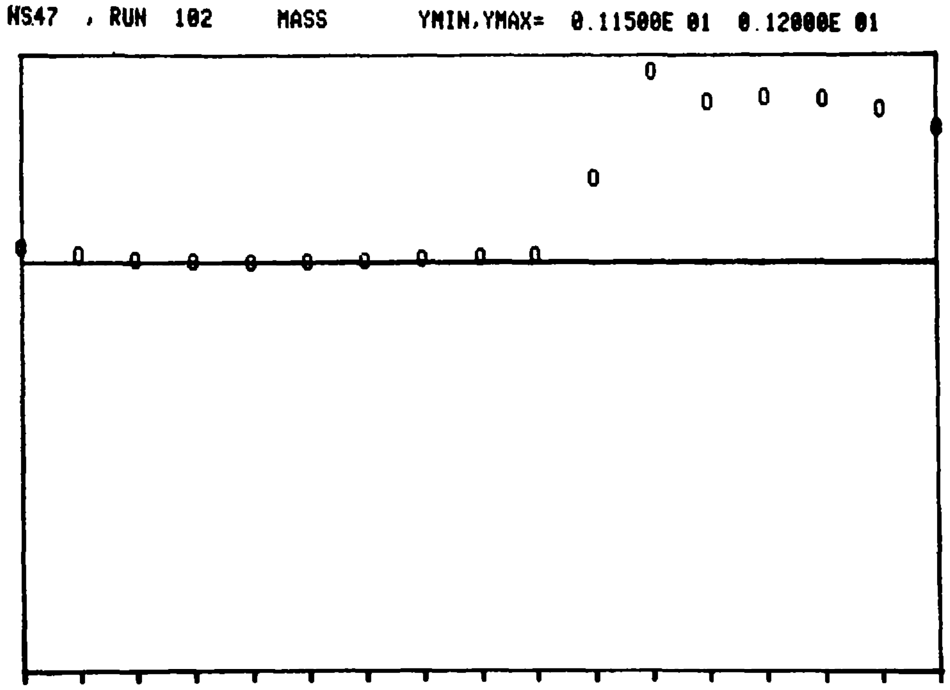


Fig. 23. Exact and computed M(x) and mass flow for flow with shock
16 intervals

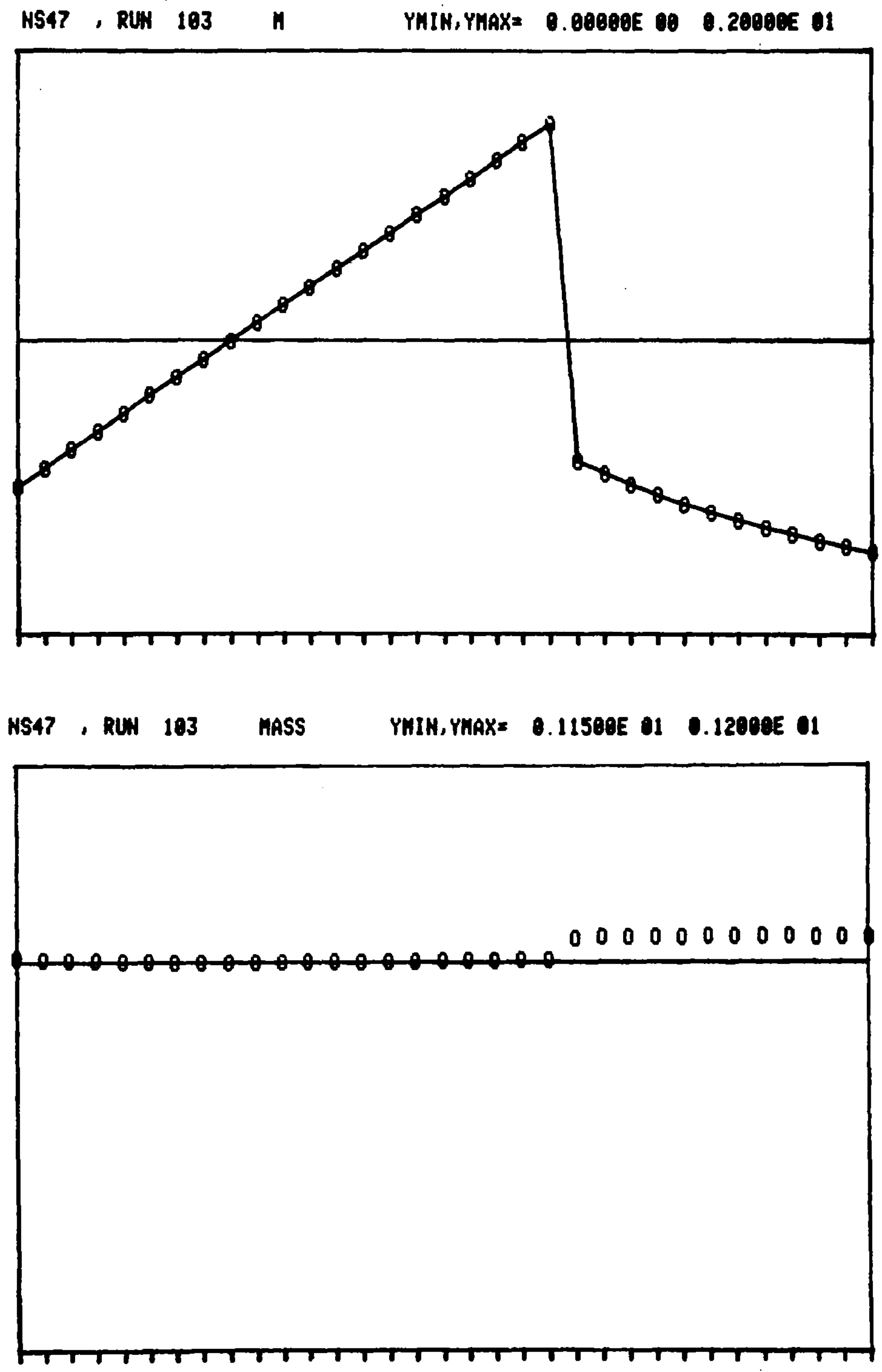


Fig. 24. Exact and computed M(x) and mass flow for flow with shock
32 intervals

pected if the number of computational nodes is very small. It is important to develop a feeling for what is more likely to deteriorate, in view of extensions to two-dimensional problems, where the number of mesh points is often much smaller than in one-dimensional cases. To this effect, we present plots of M and ρuA as functions of x, in Figs. 22 through 24, having used 8, 16 and 32 intervals respectively. The Mach number is perfect in all cases; the mass flow behind the shock, instead, deteriorates when few interval are used.

References

1. Moretti, G., The λ-scheme, Comp. and Fluids 7, 191-205, 1979.
2. Steger, J.L. and Warming, R.F., Flux vector splitting of the inviscid gasdynamic equations, J. Comp. Phys., 40, 263-293, 1981.
3. Roe, P.L., Approximate Riemann solvers, parameter vectors and difference schemes, J. Comp. Phys, 43, 357-372, 1981.
4. Osher, S., and Solomon, F., Upwind difference schemes for hyperbolic systems of conservation laws, to appear in Math. of Comp.
5. van Leer, B., Towards the ultimate conservative difference scheme, IV. A new approach to numerical convection, J. Comp. Phys., 23, 276-299, 1977.
6. Moretti, G., A physical approach to the numerical treatment of boundaries in gas dynamics, NASA CP 2201 ("Numerical boundary condition procedures"), pp. 73-96, 1981.
7. Moretti, G., Grid generation using classical techniques, NASA CP 2166 ("Numerical grid generation techniques"), pp. 1-36, 1980.
8. Moretti, G. and Zannetti, L., A new, improved computational technique for two-dimensional unsteady compressible flows, AIAA Paper 82-0168, 1982.
9. MacCormack, R.W., A numerical method for solving the equations of compressible viscous flow, Paper AIAA-81-0110, AIAA 19th Aerosp. Science Meeting, St. Louis, Missouri, 1981.
10. Courant, R., Friedrichs, O. and Lewy, H., Ueber die partiellen Differenzengleichungen der mathematischen Physik, Mathem. Ann., 100, 32-74, 1928.

1. Report No. NASA CR-3689	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle FAST EULER SOLVER FOR STEADY, ONE-DIMENSIONAL FLOWS		5. Report Date June 1983
		6. Performing Organization Code
7. Author(s) Gino Moretti		8. Performing Organization Report No. None
		10. Work Unit No.
9. Performing Organization Name and Address G.M.A.F., Inc. P.O. Box 184 Freeport, N.Y. 11520		11. Contract or Grant No. NAS3-22772
		13. Type of Report and Period Covered Contractor Report
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D.C. 20546		14. Sponsoring Agency Code 505-31-01 (E-1607)

15. Supplementary Notes

Final report. Project Manager, Robert M. Stubbs, Fluid Mechanics and Instrumentation Division, NASA Lewis Research Center, Cleveland, Ohio 44135.

16. Abstract

A numerical technique to solve the Euler equations for steady, one-dimensional flows is presented. The technique is essentially implicit, but is structured as a sequence of explicit solutions for each Riemann variable separately. Each solution is obtained by integrating in the direction prescribed by the propagation of the Riemann variables. The technique is second-order accurate. It requires very few steps for convergence, and each step requires a minimal number of operations. Therfore, it is three orders of magnitude more efficient than a standard time-dependent technique. The technique works very well for transonic flows and provides shock fitting with errors as small as 0.001. Results are presented for subsonic and transonic problems. Errors are evaluated by comparison with exact solutions.

17. Key Words (Suggested by Author(s)) Numerical gas dynamics; One-dimensional flows; Flows in nozzles; Transonic flows	18. Distribution Statement Unclassified - unlimited STAR Category 02

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of pages	22. Price*
Unclassified	Unclassified	38	A03

*For sale by the National Technical Information Service, Springfield, Virginia 22161

NASA-Langley, 1983