General Disclaimer

One or more of the Following Statements may affect this Document

- This document has been reproduced from the best copy furnished by the organizational source. It is being released in the interest of making available as much information as possible.

- This document may contain data, which exceeds the sheet parameters. It was furnished in this condition by the organizational source and is the best copy available.

- This document may contain tone-on-tone or color graphs, charts and/or pictures, which have been reproduced in black and white.

- This document is paginated as submitted by the original source.

- Portions of this document are not fully legible due to the historical nature of some of the material. However, it is the best reproduction available from the original submission.

Produced by the NASA Center for Aerospace Information (CASI)
PRELIMINARY STUDY OF THE USE OF THE STAR-100 COMPUTER FOR TRANSONIC FLOW CALCULATIONS

JAMES D. KELLER AND ANTONY JAMESON

NOVEMBER 1977
PRELIMINARY STUDY OF THE USE OF THE
STAR-100 COMPUTER FOR TRANSONIC
FLOW CALCULATIONS

James D. Keller and Antony Jameson*
Langley Research Center

SUMMARY

A new explicit method for solving the transonic small-
disturbance potential equation is presented. This algorithm,
which is suitable for the new vector-processor computers such as
the CDC STAR-100, is compared to successive line over-relaxation
(SLOR) on a simple test problem. The convergence rate of the
explicit scheme is slower than that of SLOR. However, the
efficiency of the explicit scheme on the STAR-100 computer is
sufficient to overcome the slower convergence rate and allow an
overall speedup compared to SLOR on the CYBER 175 computer.

INTRODUCTION

The state-of-the-art of transonic flow calculations has
advanced to the point where two-dimensional flows, including the
effects of viscosity, can be computed in a relatively short time
on modern serial-type computers. For example, many people are
using a program developed at the Courant Institute of New York

*Courant Institute of Mathematical Sciences, New York University
University for the analysis of transonic flow past airfoils (ref. 1). This program gives accurate solutions to the full-potential equation, including the effects of boundary-layer displacement, in about 2 or 3 minutes on a CDC CYBER 175 computer. Three-dimensional, transonic, finite-difference calculations, however, are expensive on this type of computer. The three-dimensional program described in reference 1 takes about half an hour for an inviscid calculation on a fairly crude grid.

It is hoped that the use of the STAR-100 computer will allow accurate, three-dimensional, transonic flow calculations to be done economically. One way to achieve this goal is through the development of algorithms which can make full use of the unique architecture of the STAR-100. The STAR computer has a "pipeline" type of processor which is very efficient in doing arithmetic operations on long vectors (ref. 2). Unfortunately, the best available method for solving the transonic potential equation is successive line over-relaxation (SLOR), which is not amenable to vector arithmetic. The reason for this is the semi-implicit nature of the iterative method; that is, the calculations at a particular grid point require results from the current iteration at neighboring grid points and thus cannot be done in long vector operations.

This paper describes a new explicit algorithm which can be vectorized for use on the STAR-100. The new algorithm is applied to a simple test case and compared to SLOR on the CYBER 175
computer. A simple STAR program, including vector instructions, is given in the appendix.

TEST PROBLEM

The test problem chosen for this preliminary study is to solve the transonic, nonlinear, small-disturbance potential equation for a nonlifting parabolic-arc airfoil in a finite box with uniform grid, as shown in the sketch below.

Although this is a very simple physical situation, it still has some of the most difficult features of transonic flow fields as far as programming for the STAR is concerned. The governing partial differential equation is

\[
\left[1 - M_\infty^2 - (\gamma+1) M_\infty^2 \phi_x\right] \phi_{xx} + \phi_{yy} = 0
\]

The boundary conditions to be applied are
\[ \phi = 0 \] on the outer boundary

and

\[ \phi_y = \pi 4 \tau x \] on \( y = \pm 0, -0.5 \leq x \leq 0.5 \)

where \( \tau \) is the thickness of the parabolic-arc airfoil (which has a unit chord). For \( \tau = 0.1 \) and \( M_\infty = 0.9 \) the flow is supercritical. In regions where the coefficient of \( \phi_{xx} \) is positive, the flow is subsonic and the equation is elliptic type. In regions where the coefficient of \( \phi_{xx} \) is negative, the flow is supersonic and the equation is hyperbolic type. The general procedure for solving this equation is to replace the partial differential equation with a finite difference equation at each grid point. These finite difference equations are then solved iteratively.

This test problem represents a simple physical situation which is of little practical interest. A more useful program should allow for lifting flows and should extend the outer boundary farther away from the airfoil. This could be done either by using a stretched grid or by some type of grid nesting using additional coarse grids around the small region considered here. The test case does, however, include the major difficulties to be overcome in using the STAR-100 computer for transonic flows. For example, it has supercritical flow which requires a change from one type of difference equation at subsonic (elliptic) points to another type of difference equation at supersonic (hyperbolic) points. It also requires the use of an explicit iterative scheme to solve the
difference equations if the arithmetic operations are to be done using long vector instructions.

**SEMI-IMPLICIT SOLUTION METHOD**

The most common method used to solve the finite difference equations is successive line over-relaxation (SLOR). This iterative scheme is implemented as follows:

Compute $U \equiv 1 - M_\infty^2 - (\gamma+1) M_\infty^2 \phi_x$

using

$$\phi_x = \frac{\phi_{i+1,j} - \phi_{i-1,j}}{2\Delta x}$$

If $U > 0$ (subsonic points), central differences are used together with over-relaxation ($\omega > 1$) to give:

$$U \frac{\phi_{i+1,j} - 2 \frac{\phi_{i,j}^{n+1} - 2 \left( 1 - \frac{1}{\omega} \right) \phi_{x,j}^{n} + \phi_{i-1,j}^{n+1}}{\Delta x^2}}{+ \frac{\phi_{i,j+1}^{n+1} - 2 \phi_{i,j}^{n+1} + \phi_{i,j-1}^{n+1}}{\Delta y^2}} = 0$$

where the superscript indicates the iteration number.

If $U < 0$ (supersonic points), an upwind difference is used for $\phi_{xx}$ to give:
\[
\frac{\phi_{i,j}^{n+1} - 2\phi_{i-1,j}^{n+1} + \phi_{i-2,j}^{n+1}}{\Delta x^2} + \frac{\phi_{i,j+1}^{n+1} - 2\phi_{i,j}^{n+1} + \phi_{i,j-1}^{n+1}}{\Delta y^2} = 0
\]

At the airfoil boundary the \( \phi_{yy} \) term is replaced by

\[
\phi_{yy} = \frac{\phi_{i,j+1} - 2\phi_{i,j} + \phi_{i,j-1}}{\Delta y^2} - \frac{2\phi_{y} | y=0}{\Delta y}
\]

which takes into account the fact that the nonlifting flow is symmetric (i.e., \( \phi_{i,j-1} = \phi_{i,j+1} \) at \( y = 0 \)). Each iteration is generated one column at a time going from left to right, by solving a tridiagonal system of equations at each column. This nonconservative scheme is similar to that originally proposed by Murman and Cole (ref. 3). See references 4 and 5 for a discussion of related conservative schemes.

**EXPLICIT SOLUTION METHOD**

The explicit solution method uses values of the potential function from the two previous iterations in order to update the potential function for the present iteration (thus termed a three-level scheme). The method is implemented as follows:

Compute \( U = 1 - M_{\infty}^2 - (\gamma+1) M_{\infty}^2 \phi_x \) using

\[
\phi_x = \frac{\phi_{i+1,j}^{n} - \phi_{i-1,j}^{n}}{2\Delta x_{i,j}}
\]
If \( U > 0 \) (subsonic points), central differences are used at iteration, or "time," level \( n \):

\[
\phi_{i,j}^{n+1} - 2\phi_{i,j}^n + \phi_{i,j}^{n-1} + (2 - P_1) \left( \phi_{i,j}^n - \phi_{i,j}^{n-1} \right) = D_1 R_1
\]

where

\[
R_1 = U \left( \phi_{i+1,j}^n - 2\phi_{i,j}^n + \phi_{i-1,j}^n \right) + \frac{\Delta x^2}{\Delta y^2} \left( \phi_{i,j+1}^n - 2\phi_{i,j}^n + \phi_{i,j-1}^n \right)
\]

and

\[
D_1 = \frac{P_1 P_2^2}{2 \left[ U + \left( \frac{\Delta x}{\Delta y} \right)^2 \right]}
\]

If \( U < 0 \) (supersonic points), the upwind \( \phi_{xx} \) difference is formed at "time" level \( n - 1 \), while \( \phi_{yy} \) is centrally-differenced and averaged in time level \( n \), to give:

\[
\phi_{i,j}^{n+1} - 2\phi_{i,j}^n + \phi_{i,j}^{n-1} + 2\phi \left( \phi_{i,j}^n - \phi_{i,j}^{n-1} \right) \phi_{i-1,j}^n + \phi_{i-1,j}^{n-1} = D_2 R_2
\]

where

\[
\phi = \frac{\Delta x}{\Delta y}
\]
\[ R_2 = \bar{U} \left( \phi_{i,j}^{n-1} - 2\phi_{i-1,j}^n + \phi_{i-2,j}^n \right) \]

\[ + \left( \frac{\Delta x}{\Delta y} \right)^2 (1-\sigma) \left( \phi_{i,j+1}^n - 2\phi_{i,j}^n + \phi_{i,j-1}^n \right) \]

\[ + \sigma \left( \phi_{i-1,j+1}^n - 2\phi_{i-1,j}^n + \phi_{i-1,j-1}^n \right) \]

\[ \bar{U} = (1-\sigma) U + \sigma \bar{U} \]

\[ \bar{U} = 1 - M_\infty^2 - (\gamma+1) M_\infty^2 \left( \frac{\phi_{i,j}^{n-1} - \phi_{i-2,j}^{n-1}}{2\Delta x} \right) \]

\[ \sigma = \frac{2}{\min \left( 1, \sqrt{|U|} \frac{\Delta y}{\Delta x} \right)} \]

and

\[ D_2 = \frac{\left( \frac{\Delta y}{\Delta x} \right)^2}{\max \left[ 1, |U| \left( \frac{\Delta y}{\Delta x} \right)^2 \right]} \]

A von Neumann stability analysis of this last scheme with the \( \bar{U} \) replaced by \( U \) shows why the \( \phi_{xx} \) derivative is evaluated.
at the \( n-1 \) iteration and why the \( \phi_{yy} \) derivative is a weighted average between \( \phi_{yy} \) at the \( i \) column and \( \phi_{yy} \) at the \( i-1 \) column. To do the von Neumann analysis, let \( \phi \) at the \( k \)th iteration be

\[
\phi(k) = g e^{ikx} e^{iny}
\]

Also let \( m\Delta x = \xi \) and \( n\Delta y = \eta \). Putting these definitions into equation (1) gives

\[
g^2 - 2g + 1 + 2\sigma (g - 1 - ge^{-i\xi} + e^{-i\xi})
\]

\[
= D_2 \left\{ U (1 - 2e^{-i\xi} + e^{-2i\xi})
\right.
\]

\[
+ g \left( \frac{\Delta x}{\Delta y} \right)^2 \left[ (1-\sigma)(e^{i\eta} - 2 + e^{-i\eta})
\right.
\]

\[
+ \sigma (e^{-i\xi} e^{i\eta} - 2e^{-i\xi} e^{-i\eta}) \right\}
\]

(2)

Note that

\[ D_2 U = -\sigma^2 \]

Also define

\[
\rho \equiv 1 - 2\sigma^2 \frac{1}{|U|} \frac{1}{(\Delta x/\Delta y)^2} \sin^2 \frac{n}{2}
\]
so equation (2) becomes

\[
g^2 - 2p g \left[ 1 - \sigma (1 - e^{-iF}) \right] + \left[ 1 - \sigma (1 - e^{iF}) \right]^2 = 0
\]

or

\[
g = (p \pm i \sqrt{1 - \rho^2}) \left[ 1 - \sigma (1 - e^{-iF}) \right]
\]

Thus, in order to have \(|g| \leq 1\), it is necessary and sufficient to have both \(|\rho| \leq 1\) and \(\sigma \leq 1\). The inequality \(|\rho| \leq 1\) implies that \(\sigma \leq \sqrt{|U| A u \Delta x} \).

This convenient factoring of the expression for the amplification factor was made possible by choosing this particular representation for the \(\phi_{xx}\) and \(\phi_{yy}\) derivatives. In practice the coefficient of \(\phi_{xx}\) is \(\bar{U}\) rather than \(U\); this averaging makes the scheme approach second-order accuracy as \(\sigma\) approaches one.

It should be noted that this explicit scheme is not only a different iterative algorithm from the semi-implicit scheme, but it also has a different steady-state solution. This difference occurs because of the weighted averaging done on both the \(\phi_{yy}\) term and on the coefficient of the \(\phi_{xx}\) term for supersonic points.

CONVERGENCE AND TIMING COMPARISONS

Short computer programs have been written to solve the sample problem using the two methods described. The semi-implicit SLOR
method was coded in standard FORTRAN IV and run on a CDC CYBER 175 computer (which is about 2 \( \frac{3}{2} \) times as fast as a CDC 6600 for this type of problem). The explicit method was coded in STAR FORTRAN (ref. 6) and run on a CDC STAR-100 computer. The STAR code includes the use of vector instructions in the iteration loop. It also includes the use of bit control vectors to distinguish between subsonic and supersonic points. The bit control vectors provide the capability of performing the complicated supersonic calculations only at supersonic points, which are collected into a vector through the "compress" and "expand" type of instructions available on the STAR. The STAR code for this problem is listed in the appendix.

The computations were done on three different grids. Each calculation was terminated when the value of the largest residual in the flow field was less than \( \frac{1}{2} (\Delta x^2 + \Delta y^2) \). Each calculation was run with experimentally-determined optimum values of the parameters for that algorithm so that convergence was attained in a minimum number of cycles. The results in the table below show that the new three-level explicit scheme has a slower convergence.

<table>
<thead>
<tr>
<th>Grid size</th>
<th>SLOR ON CYBER 175</th>
<th>EXPLICIT METHOD ON STAR-100</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cycles to converge</td>
<td>Time to converge</td>
</tr>
<tr>
<td>40X40</td>
<td>42</td>
<td>1.242</td>
</tr>
<tr>
<td>80X80</td>
<td>98</td>
<td>12.118</td>
</tr>
<tr>
<td>160X160</td>
<td>244</td>
<td>116.095</td>
</tr>
</tbody>
</table>
rate than SLOR. However, the efficiency of this new scheme on the STAR computer is enough to make up for the slower convergence rate and still allow an overall reduction in computing time. Not surprisingly, the speedup is greater for the cases with finer grids than for the 40X40 case.

CONCLUDING REMARKS

This preliminary study has shown that a new explicit method for solving the transonic small-disturbance potential equation on the STAR-100 computer can almost halve the computer time required for this type of computation when compared to successive line over-relaxation on the CDC CYBER 175 computer. These results are limited to a relatively simple problem with a uniform cartesian grid. Although the speedup is not as great as desired, it is enough to justify further study of this method. The effects of lift and of grid stretching on the convergence rates of the schemes should be investigated. Also, the new explicit scheme should be applied to the full potential equation.

There are several possibilities for obtaining further reductions in computer time. One is through the development of more efficient algorithms. Improvements might be made in the convergence rate of explicit algorithms, or other vectorizable algorithms might be developed. Another possibility is through programing techniques to get successive line over-relaxation to run as efficiently as possible on the STAR-100 computer. Although
the method is semi-implicit, there are portions of it which can be written in vector instructions of short length.

The program listing in the appendix should serve as an introduction to some of the programing techniques available on the STAR-100 which are useful for transonic flow calculations.
APPENDIX

PROGRAM LISTING

PROGRAM QT(INPUT, OUTPUT, TAPE5=INPUT, TAPE6=OUTPUT)
COMMON P(41,41), FP(41), IMAX, JMAX, JH, COVR, MIT, IMAXM1, JMAXM1
* , DX, DY, DX2R, P1, P2, XMINF, RSH, DY2R
CALL R3CLCKS(CPH, ALL)
WRITE(6,9902) CPU, ALL
XMINF=.9
P1=1.83
P2=.99
RSH=1.4
IMAX=41
JMAX=41
MIT=1000
CF=1.
P2=4.,3.
PH=4.,3.
IMAXM1=IMAX-1
JMAXM1=JMAX-1
DX=PH/IMAXM1
DY=PH/JMAXM1
JH=JMAXM1/2+1
DX2R=1./DX**2
DY2R=1./DY**2
COVR=CF*.5*(DX**2+DY**2)
DO 10 I=1,IMAX
FP(I)=0.
10 CONTINUE
XLE=-.5
XTE=.5
EPS=1.0E-06
IH=IMAXM1/2+1
ILE=IFIX((XLE-EPS)/DX)+IH
ITE=IFIX((XTE+EPS)/DX)+IH
DO 11 J=ILE,ITE
X=(I-IH)*DX
FP(I)=.4*X
11 CONTINUE
NT=IMAXM1*JMAX
P(1,1,NT)=0.
CALL PICTURE
CALL PRESS
STOP
9902 FORMAT(1X,A10,1X,A10)
END

ORIGINAL PAGE IS
OF POOR QUALITY
SUBROUTINE PICTURE

C THIS SUBROUTINE DOES THE RELAXATION

COMMON P(41,41), PP(41,41), IMAX, JMAX, JW, COVR, HIT, IMAXH, JMAXH

* DX, DY, DX2, P1, P2, XHINF, RSH, DY2H

C DIMENSION SOME ARRAYS FOR TEMPORARY STORAGE

DIMENSION PP(41,41), PM(41,41)
DIMENSION T1(41,41), T2(41,41), T3(41,41)

C HIT MU(41,41), ER(41,41), IR(41,41), MUN, E50, IRD, B1D

C DESCRIPTOR T1D, T2D, T3D, PD, PPD, PHD

C DESCRIPTOR MUN, ERO, IRD, B1D

C DESCRIPTOR T1ND, T2ND, T3ND, TAND, T5ND, T6ND, T7ND, T8ND

CALL Q3CLOCKS(CPU, ALL)

WRITE(6, 9900) CPU, ALL

C ASSIGN THE DESCRIPTIONS TO SPECIFIC ARRAYS

NP=IMAX*JMAX=2*JMAX=2
ASSIGN T1D, T1(2, 2WP)
ASSIGN T2D, T2(2, 2WP)
ASSIGN T3D, T3(2, 2WP)
ASSIGN PD, P(2, 2WP)
ASSIGN PPD, PP(2, 2WP)
ASSIGN PHD, PH(2, 2WP)
ASSIGN MUN, MU(2, 2WP)
ASSIGN ERO, ER(2, 2WP)
ASSIGN IRD, IR(2, 2WP)

C SET UP A FIT ARRAY THAT HAS ONES AT INTERIOR POINTS

C AND ZEROS AT THE EDGE POINTS

IRD=8*0
DO 1 I=2, IMAXM1
DO 1 J=2, JMAXM1
1 IR(I, J)=B*1

IRD=NOT, IRD

XX=1, XHINF**2
AC=(RSH+1.)*XHINF**2/(2.*DX)

D1=0.5*P1*P2*P2
AR2=(DX/DY)**2

AR2R=1./AR2

U2=P2*P2/AR2

TDY=2.*DY

S=1.1

C INITIALIZE THE TEMPORARY ARRAYS

T1D=0.
T2D=0.
T3D=0.
P1D=0.
P2D=0.

WRITE(6, 9901)

POINTS=(IMAX=2)*(JMAX=2)

RMAX0=10,

RAVGO=10,

RASUHO=10.
RASUPO=10.
M=1
50 CONTINUE
C COMPUTE U
T1D=XX-AC*(P(3,2;NP)-P(1,2;NP))
C SET THE MUD(I,J) BIT TO ONE AT SUPERSONIC POINTS
MUD=T1D,LT,0.
C SET MUD TO ZERO AT THE EDGE POINTS
MUD=MUD,AND,1BD
C COMPUTE A CENTRAL PHI=XX AT I,J
PPD=P(1,2;NP)-PD-PF+P(3,2;NP)
C COMPUTE A CENTRAL PHI=YY AT I,J
T2D=P(2,1;NP)-PD-PD+P(2,3;NP)
C COMPUTE THE RESIDUAL
PPD=T1D*PPD+AR2*T2D
C CHANGE THE RESIDUAL AT THE AIRFOIL SURFACE
PP(2,J);I'MAX-2)=PP(2,J);I'MAX-2)+1HDY*FP(2;I'MAX-2)
T3D=0.
C PUT ZEROS IN THE RESIDUAL ARRAY AT THE EDGE POINTS
PPD=88VCNT(T3D;END;PPD)
C PUT ZEROS IN THE RESIDUAL ARRAY AT SUPERSONIC POINTS
PPD=88VCNT(T3D;MUD;PPD)
T3D=73DSPPD*T3D)
RMAXSUB=88SMAX(T3D)*DX2R
RMAXSUB=88MAX(T3D)*DX2R
C COMPUTE THE NEW PHI VALUES
PPD=1/(T1D+AR2)*PPD+PD*(1-P1)*(PD-PMD)
C NUMBER OF SUPERSONIC POINTS
NS=88SCNT(MUD)
RMAXSUP=0.
RMAXSUP=0.
IF(NS,EQ,0)GO TO 300
C SUPERSONIC CALCULATIONS
C ASSIGN TEMPORARY STORAGE FOR SUPERSONIC CALCULATIONS
ASSIGN T1ND,,DYN,NS
ASSIGN T2ND,,DYN,NS
ASSIGN T3ND,,DYN,NS
ASSIGN T4ND,,DYN,NS
ASSIGN T5ND,,DYN,NS
ASSIGN T6ND,,DYN,NS
ASSIGN T7ND,,DYN,NS
ASSIGN T8ND,,DYN,NS
ASSIGN R1D,,DYN,NS
C MAKE A VECTOR OUT OF THE VALUES OF U AT SUPERSONIC POINTS
T1ND=88VCMPRS(T1D;MUD;T1ND)
C COMPUTE SIGMA
T3D=VABS(T1ND;T3D)*AR2R
T4ND=VSOKT(T3ND;T4ND)
B1D=T4ND,GT,1.
T5ND=1.
MAKE A VECTOR OUT OF THE VALUES OF THE CENTRAL PHI-YY AT SUPersonic POINTS

MAKE A VECTOR OUT OF THE VALUES OF THE CENTRAL PHI-YY ONE POINT UPSTREAM OF EACH SUPersonic POINT

COMPUTE PHI-YY TO USE AT SUPersonic POINTS

MAKE A VECTOR OUT OF OLD VALUES OF PHI AT SUPersonic POINTS AND ONE AND TWO POINTS UPSTREAM OF EACH SUPersonic POINT

COMPUTE U-Tilde

COMPUTE U-Bar

COMPUTE THE RESIDUAL

EXPAND THE RESIDUALS AT SUPersonic POINTS BACK TO A TEMPORARY FULL ARRAY

CHANGE THE RESIDUALS AT THE AIRFOIL SURFACE

RECOMPRESS THE RESIDUALS AT SUPersonic POINTS

COMPUTE D2

COMPUTE THE NEW VALUES OF PHI AT SUPersonic POINTS

EXPAND THE NEW VALUES OF PHI AT SUPersonic POINTS TO A FULL ARRAY AND PUT THEM IN THE PHI-PLUS ARRAY

FREE

CONTINUE

MOVE THE PHI ARRAYS TO NEW ITERATION LEVEL

1.7
NTMP=NS
IF(NS,EG,0)NTP=P*1
FNS=FLOAT(NTMP)
RASUP=RSUM/RNS
IF(NS,EG,0)RASUP=10.
RASUB=(RSUM-RSUSUP)/(POINTS-FNS)
RAVG=RNS/POINTS
SRA=RAVG/RAVGO
SRAU=RASUP/RASUP
SRAU=RASUP/RASUP
RMAX=RAVG
FADVGRAVG
RASUH=RASUB
RASUP=RASUP
WRITE(6,9902)N,RMAX,SRM,RAVG,SRA,RSUR,SPASUR,RSASUP,NS
IF(RMAX,LE,COVRP,N,GE,M1)GO TO 500
IF(N,EG,1)RMAXI=RMAX
IF(N,EG,1)RAVI=RAVG
N=N+1
GO TO 50
500 CONTINUE
SRAU=(RMAX/RMAXI)**(1./FLOAT(N))
SRAU=(RAVG/RAVGI)**(1./FLOAT(N))
WRITE(6,9903)SRM,SRA
CALL D3CLOCKS(CPU,ALL)
WRITE(6,9905)CPU,ALL
RETURN
9901 FORMAT(3H0,N4X4HRMAX6X3HSP5X4HRASUP6X3HRSR5X4HRSUP3X2HNS)
9902 FORMAT(IX,13,E11.5,F7.4,E11.5,F7.4,E11.5,F7.4,E11.5,F7.4,E14)
9903 FORMAT("OVERALL SPECTRAL RADIUS IS"F7.4," BASED ON RMAX AND"
*F7.4," BASED ON RAVG.")
9904 FORMAT("ENTERING PICTURE AFTER"F10.5," SECONDS CPU TIME AND"
*F10.5," SECONDS ELAPSED TIME")
9905 FORMAT("ITERATIONS TOOK"F10.5," SECONDS CPU TIME AND"F10.5,
*SECONDS ELAPSED TIME")
END
SUBROUTINE PRESS
COMMON P(41),FP(41),IMAX,JMAX,J,YSEQ,IT,IMAXM1,JMAXM1
* ,DY,DY,DX2P,R1,P2,XMINF,RSH,DY2R
DIMENSION CP(41)
J=JW
DO 1 I=2,IMAXM1
CP(I)=(P(I+1,J)-P(I-1,J))/DX
1 CONTINUE
WRITE(6,9902)(I,CP(I),I=2,IMAXM1)
RETURN
9902 FORMAT("1 I CP"/(10,F10.6))
END
REFERENCES


A new explicit method for solving the transonic small-disturbance potential equation is presented. This algorithm, which is suitable for the new vector-processor computers such as the CDC STAR-100, is compared to successive line over-relaxation (SLOR) on a simple test problem. The convergence rate of the explicit scheme is slower than that of SLOR. However, the efficiency of the explicit scheme on the STAR-100 computer is sufficient to overcome the slower convergence rate and allow an overall speedup compared to SLOR on the CYBER 175 computer.