AIAA 80-1364R

Transonic Potential Flow Computation about Three-Dimensional Inlets, Ducts, and Bodies

T. A. Reyhner*
Boeing Commercial Airplane Company, Seattle, Wash.

An analysis has been developed to predict transonic potential flow about three-dimensional objects including inlet, duct, and body geometries. Finite differences and line relaxation are used to solve the complete potential flow equation. The coordinate system is independent of body geometry and hence places no restrictions on body shape. Cylindrical coordinates have been used for the computer code. The analysis has been programmed for the Control Data Corporation CYBER 203 vector computer with this programming oriented to take advantage of the vector processing capabilities of the computer. Comparisons of computed results with experimental measurements are presented to verify the analysis.

Nomenclature

a	= speed of sound
L	= length of inlet, highlight to compressor face
M	= Mach number
M_{CF}	= Mach number at compressor face for uniform flow parallel to axis
$\boldsymbol{n}$	= unit normal to surface oriented into flowfield
n_x, n_r, n_θ	= components of $\boldsymbol{n}$
q	= velocity $(u^2+u_r^2+u_\theta^2)^{1/2}$
r	= radius
r_{FAN}	= radius of engine fan
r_{max}	= r at outer edge of computational cylinder
s	= arc length
u	= axial velocity component, $u=\phi_x$
u_r	= radial velocity component, $u_r=\phi_r$
u_θ	= circumferential velocity component, $u_\theta=(1/r)\phi_\theta$
v	= velocity component in y direction, $v=\phi_y$
w	= velocity component in z direction, $w=\phi_z$
x	= axial coordinate
y	= coordinate, $y=r\cos\theta$
z	= coordinate, $z=r\sin\theta$
$\beta, \beta_x, \beta_{Ix}\ldots$	= weighting parameters
γ	= ratio of specific heats
$\Delta x, \Delta r\ldots$	= step size in x, r, etc.
η	= nondimensional coordinate
θ	= circumferential coordinate
ϕ	= potential function
ϕ_n	= velocity normal to surface
ϕ_s	= $\partial\phi/\partial s$, velocity in direction of s
$\phi_{s_x}, \phi_{s_r}, \phi_{s_\theta}$	= velocity components for constant x, r, and θ cuts of surface, respectively
ω_x	= local x over-relaxation parameter
$\bar{\omega}_x$	= reference x over-relaxation parameter ($\bar{\omega}_x=1.85$)
ω_θ	= local θ under-relaxation parameter
$\bar{\omega}_\theta$	= reference θ under-relaxation parameter ($\bar{\omega}_\theta=0.9$)
$\phi_x \mid_S$	= ϕ_x at point S
$\mathfrak{C}$	= centerline (axis)

Superscript

$(\,)^+$	= value after updating

Subscripts

i	= index for x mesh values
j	= index for r mesh values
k	= index for θ mesh values
$\ell, \mathscr{U}, u$	= step-size subscripts
A,B,P,P'...	= points on grid or surface
S, S_1, S_2	= surface points
x,y,z,r,θ,s	= partial derivatives
$0,1,2$	= step-size subscripts
∞	= freestream

Received July 8, 1980; presented as Paper 80-1364 at the AIAA 13th Fluid and Plasma Dynamics Conference, Snowmass, Colo., July 14-16, 1980; revision received Feb. 9, 1981.

*Senior Specialist Engineer, Propulsion Research. Associate Fellow AIAA.

Introduction

DURING the past decade tremendous progress has been made in the field of transonic potential flow computation. The work started with Murman and Cole,[1] who used upwind differencing in supersonic flow regions in order to solve mixed elliptic and hyperbolic equations. Their work dealt with small disturbance theory with the flow principally aligned with one coordinate. Jameson[2] extended this approach to the full potential equation with arbitrary flow direction. Several classes of three-dimensional geometries have been solved using the full potential equation.[2-5] Most extensions to three dimensions have been based on some mapping from the physical coordinates to a more convenient computational coordinate system. If such mappings can be found, the computational problem, exclusive of the generation of the mapping, can be simplified greatly. The difficulty with this approach is that as the geometry of objects to be considered becomes more complex, it becomes more difficult to find convenient mappings that will provide reasonable coordinates for the computation. A further problem is that with a complex mapping it can be very difficult to have any physical understanding of the solution process.

A project has existed for several years to obtain solutions of the transonic potential flow equation in simple physical coordinates. This type of solution method does not require the finding of a new mapping for each new family of geometries. It also has the advantage that the entire solution procedure is in terms of physical coordinates and variables; thus any failures are much easier to interpret. The first work on this concept was for axisymmetric flow.[6] Later, the initial work was extended to axisymmetric geometries at angle of attack.[3] In this paper the extension to general three-dimensional geometries is presented for inlet, body, and duct geometries.

The analysis is discussed, and results of the analysis and comparisons with experiments are presented. Implementation of the analysis on the Control Data Corporation (CDC) CYBER 203 (STAR) computer is also presented. The CDC CYBER 203 is a very large fast virtual memory vector

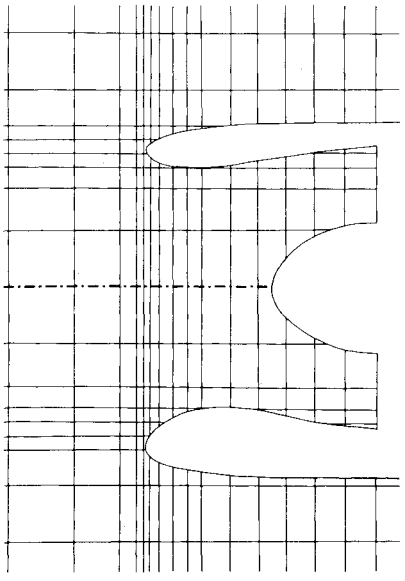


Fig. 1 Typical coarse computational mesh for a θ = const cut in the vicinity of the inlet.

processing computer. It features a million-word real memory plus virtual memory operation enabling large three-dimensional codes to be written without spending significant amounts of time on disk-memory management. The vector processing capability enables relatively short run times, so that the final program can be used often in production environment.

Approach

The full equation for compressible potential flow is solved in a cylindrical coordinate system using finite-difference techniques and line relaxation. The basic approach is extremely flexible with few fundamental limitations. The most significant limit on the method is the resolution capability of the meshes that can be used practically. The method is convergent with extremely fine meshes, but the convergence rate can be extremely slow and there are limits due to available computer storage. The method is not restricted to cylindrical coordinates; however, for inlet computations, cylindrical coordinates are quite efficient. The implementation of the basic method for this computer code is quite general for points interior to the computational flowfield boundaries, but there presently are limitations on what boundary conditions can be applied. A typical coarse computational mesh for an inlet geometry is shown in Fig. 1.

The method used for solution is very similar to that of the previous work.[3,6] Some changes have been made to improve the geometric capabilities of the code. Other changes have been made to increase the percentage of the code that can be vectorized.

Equation

The complete equation for compressible potential flow expressed in cylindrical coordinates (x,r,θ) is

$$(a^2-\phi_x^2)\phi_{xx}+(a^2-\phi_r^2)\phi_{rr}+\left(a^2-\frac{\phi_\theta^2}{r^2}\right)\frac{\phi_{\theta\theta}}{r^2}-2\phi_x\phi_r\phi_{xr}$$

$$-2\phi_r\phi_\theta\frac{\phi_{r\theta}}{r^2}-2\phi_\theta\phi_x\frac{\phi_{\theta x}}{r^2}+\left(a^2+\frac{\phi_\theta^2}{r^2}\right)\frac{\phi_r}{r}=0 \qquad (1)$$

where ϕ is the velocity potential and the local speed of sound a is given by

$$a^2=a_\infty^2-\frac{\gamma-1}{2}\left(\phi_x^2+\phi_r^2+\frac{\phi_\theta^2}{r^2}-q_\infty^2\right) \qquad (2)$$

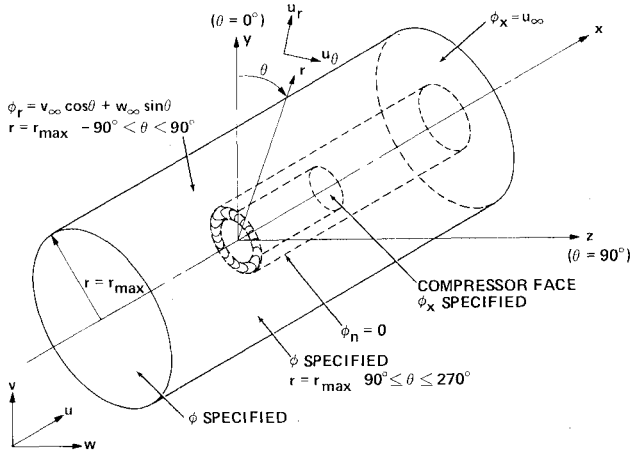


Fig. 2 Geometry and boundary conditions for inlet flowfield computation.

The velocity components $(u,\ u_r,\ u_\theta)$ for the cylindrical coordinate system are

$$u=\phi_x, \qquad u_r=\phi_r, \qquad u_\theta=\phi_\theta/r \qquad (3)$$

which are related to the Cartesian mesh $(x,\ y,\ z)$ velocity components $(u,\ v,\ w)$ by

$$u_r=v\cos\theta+w\sin\theta, \qquad u_\theta=-v\sin\theta+w\cos\theta \qquad (4)$$

Boundary Conditions

The boundary condition at solid surfaces is that the velocity normal to the surface ϕ_n equals zero. The boundary condition at the exit of a duct or at the compressor face of an inlet is that the axial velocity is fixed at the uniform value that gives a specified mass flow. At the left boundary of the computational field, the potential function ϕ is specified. For an inlet flowfield computation (Fig. 2) at the left boundary,

$$\phi=u_\infty x+v_\infty r\cos\theta+w_\infty r\sin\theta+\text{const.} \qquad (5)$$

Also for an isolated inlet in a freestream the following additional boundary conditions are used. Equation (5) is used to specify ϕ for 90 deg $\le\theta\le$ 270 deg and $r=r_{max}$. The outflow velocity ϕ_n for the computational cylinder is specified by

$$\phi_n\Big|_{r=r_{max}}=\phi_r\Big|_{r=r_{max}}=v_\infty\cos\theta+w_\infty\sin\theta \qquad (6)$$

for -90 deg $<\theta<$ 90 deg and $r=r_{max}$. ϕ_x is specified as equal to u_∞ at the right boundary of the computational field outside of the inlet.

The boundary conditions on the far field for an isolated nacelle or body are only approximate. The correct boundary condition is that the velocities approach the freestream values far from the body. This boundary condition cannot be satisfied directly for a potential-flow solution in a finite-computational volume. As written, the boundary conditions specify either the normal or tangential component of velocity at any surface. In most cases the normal component of velocity is specified for outflow surfaces and the tangential component for inflow surfaces. The exceptions are nonzero yaw angle or negative angle of attack. For these cases, the boundary conditions on some of the surfaces could be changed; for example, the analysis is convergent for a negative angle of attack which essentially reverses the boundary conditions on the top and bottom of the computational cylinder. Changes to the far-field boundary conditions would probably effect accuracy and convergence rate of the solution; however, no extensive studies of these effects have been made.

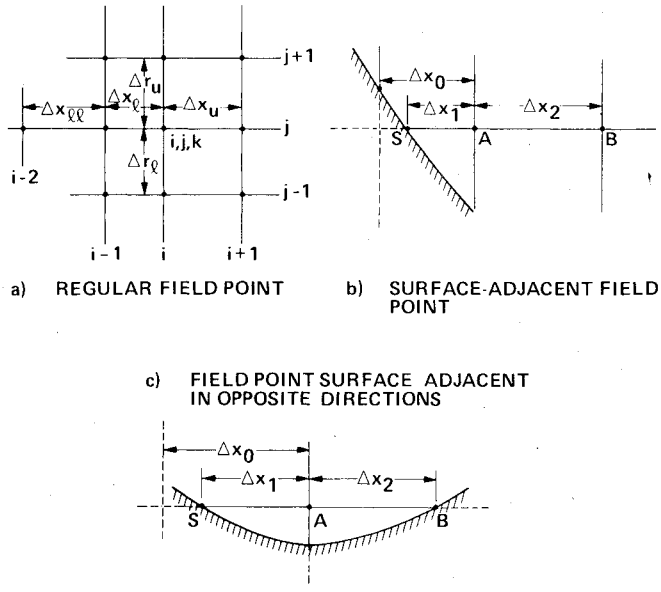


Fig. 3 Typical configurations for field-point differencing.

The present far-field boundary conditions are satisfactory provided the boundaries are located far enough from the body. Locating the far-field boundaries five to ten nacelle diameters away from the body is usually sufficient to insure minor effects of the boundary conditions on the computed flowfield near the body. If the far-field boundary is so placed, the effects of the boundary conditions are small and are of the order of or smaller than other approximations such as treating a nacelle as an isolated nacelle.

Finite-Difference Equations

The potential flow equation is used to generate finite-difference equations for all mesh points inside the flowfield. This is done by replacing all the partial derivatives with difference quotients. This creates a large system of coupled nonlinear equations. These equations are then solved by the successive line over-relaxation (SLOR) technique.[7] Line relaxation was chosen since it is more efficient than a point by point or an explicit scheme, yet not significantly more difficult to implement.

A penalty involved in using a line-relaxation scheme on a vector computer is that the tridiagonal solver used to solve the simultaneous equations for a line does not vectorize efficiently. However, the ratio of computing time spent solving the equations with the tridiagonal solver vs the time spent calculating the equation coefficients is small. Also the SLOR scheme takes many less relaxation sweeps to reach a given level of convergence than an explicit technique. A sweep is a complete solving in sequential order of all the difference equations for the flowfield.

The SLOR scheme consists of first linearizing the equations by using old ϕ values to compute the coefficients of the second derivatives or ϕ_r for the last term in Eq. (1). An old ϕ value is either the value from the previous relaxation sweep or, for the first sweep, the initial field which is taken to be uniform flow with the freestream velocity. The new values of ϕ, noted as ϕ^+, are the values that are calculated from the finite-difference equations. The finite-difference equations are linearized such that the difference equations are solved simultaneously only along radial lines. The linearization is such that the solution process is in terms of successive axial planes. Since the difference equations have been linearized, the process is iterated until the difference between the old and new ϕ values satisfies a tolerance.

Difference equations are also generated for boundary points using the appropriate boundary condition. Difference equations for the surface points are solved simultaneously with the difference equations for the field points to which they are adjacent.

The difference equations are generated using a nonconservative approach. As a consequence there is a failure to conserve mass across shocks. The errors due to failure to exactly conserve mass appears to be relatively small.

Weighting and Alternate Formulas

A possible problem with this approach to solving the potential flow equation is that the step size between a surface point and an adjacent field point can be arbitrarily small. Derivatives are calculated by dividing by the step size, and thus any error in the potential function ϕ can be magnified by the reciprocal of the step size which can become arbitrarily large. This can cause accuracy and stability problems if not considered. A solution to this difficulty is to use alternate formulas that do not use the field point when the step size to a surface point is very small. The field point is calculated by interpolating between the surface point or points and other field points. To obtain a smooth solution process, a weighted combination of alternate formulas is used if the surface points are closer to the field points than one-half the local field-point spacing. This procedure, which is called weighting in this paper, has been used for many difference quotients and for generating the field-point equations. Because of space limitations only typical formulas are shown.

Difference Quotients

Three-point difference quotients are used for first- and second-derivative calculations. The first-derivative quotients are second-order accurate. Because a variable grid is used, the second-derivative quotients, with the exception of the central-differenced cross derivatives, are of first order. Upwind differencing is used where the flow is supersonic. The differencing used is similar to the preceding work[3] except near surfaces, where significant changes have been made to improve accuracy and to eliminate problems with certain mesh/surface configurations.

First-Derivative Difference Quotients

The first-derivative difference quotients are used to calculate the coefficients of the partial-differential equation, in the central-difference formulas for the cross derivatives, and for the last term of Eq. (1). They are calculated using all old values for ϕ, except for the differencing for ϕ_r in the last term of Eq. (1) which uses all new values. If a surface is involved in the differencing, special weighting may be used to obtain better numerical behavior. Refer to Fig. 3 for the nomenclature used in the following formulas.

For a regular field point (i, j, k), a typical formula is

$$\phi_x = \frac{\Delta x_\ell^2 \phi_{i+1,j,k} + (\Delta x_u^2 - \Delta x_\ell^2)\phi_{i,j,k} - \Delta x_u^2 \phi_{i-1,j,k}}{\Delta x_\ell \Delta x_u (\Delta x_\ell + \Delta x_u)} \tag{7}$$

At a surface adjacent point A, the formulas for ϕ_x at A are

$$\phi_x^{(1)} = \frac{\Delta x_1^2 \phi_B + (\Delta x_2^2 - \Delta x_1^2)\phi_A - \Delta x_2^2 \phi_S}{\Delta x_1 \Delta x_2 (\Delta x_1 + \Delta x_2)} \tag{8}$$

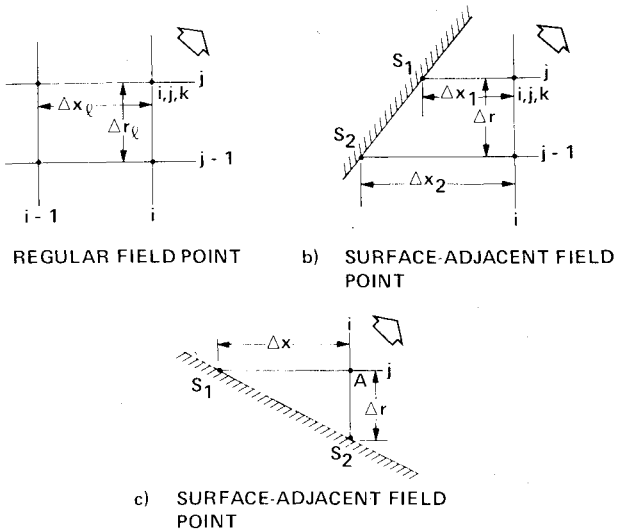
$$\phi_x^{(2)} = \frac{1}{\Delta x_1 + \Delta x_2}\left[(\Delta x_2 - \Delta x_1)\,\phi_x\Big|_S + 2\Delta x_1 \frac{\phi_B - \phi_S}{\Delta x_1 + \Delta x_2}\right] \tag{9}$$

$$\phi_x = \beta \phi_x^{(1)} + (1-\beta)\phi_x^{(2)} \tag{10}$$

where

$$\beta = 1 \qquad 2\Delta x_1 \geq \Delta x_0$$

$$= 2\Delta x_1 / \Delta x_0 \qquad 2\Delta x_1 < \Delta x_0$$



a) REGULAR FIELD POINT b) SURFACE-ADJACENT FIELD POINT

Fig. 4 Nomenclature for upwind differencing for ϕ_{xr}.

If the point B is also a surface point, the following formula may also be weighted into the calculation:

$$\phi_x^{(3)} = \frac{\Delta x_2 \phi_x \Big|_S + \Delta x_1 \phi_x \Big|_B}{\Delta x_1 + \Delta x_2} \tag{11}$$

Difference Switching

A stable computational scheme is obtained by using central differencing where the flow is subsonic and upwind differencing in supersonic regions. In the transonic regime a smooth stable switch between difference formulas is required. This analysis uses a straightforward procedure that works well and is independent of flow direction. The second derivatives, ϕ_{xx}, ϕ_{rr}, and $\phi_{\theta\theta}$, switch from central to upwind differencing as their respective coefficients, $(a^2-\phi_x^2)$, $(a^2-\phi_r^2)$, and $[a^2-(1/r^2)\phi_\theta^2]$, go from positive to negative. The cross derivatives are central differenced where the flow is subsonic and upwind differenced when either velocity component is supersonic. In between, the following weighting combining central and upwind differencing is used:

$$\phi_{xr} = \beta \phi_{xr} \Big|_{\text{central difference}} + (1-\beta)\phi_{xr} \Big|_{\text{upwind difference}}$$

$$\beta = \frac{(a^2-u^2)(a^2-u_r^2)}{(q^2-u^2)(q^2-u_r^2)} \tag{12}$$

As can be seen, $\beta=1$ when $q=a$ and $\beta=0$ when $|u|=a$ or $|u_r|=a$. The formulas for the other cross derivatives are similar.

Second Derivatives, ϕ_{xx}, ϕ_{rr}, and $\phi_{\theta\theta}$

For the central-difference formulas, ϕ_{rr} is taken in the line direction for the line relaxation, and thus uses all new values. ϕ_{xx} must, by necessity, use some old values; an over-relaxation factor ω_x is used to increase the convergence rate. $\phi_{\theta\theta}$ uses old values at $k-1$ and $k+1$ in order that an entire plane of coefficients may be calculated at one time. This yields a gain in computational efficiency by increasing vector lengths. ω_θ is less than one, and is an under-relaxation parameter required for stability. Surface points may be substituted for field points in the following formulas with appropriate changes in the step sizes.

$$\phi_{xx} = \left\{ \Delta x_\ell \phi_{i+1,j,k} - (\Delta x_\ell + \Delta x_u)\left[\frac{1}{\omega_x}\phi_{i,j,k}{}^{+} + \left(1-\frac{1}{\omega_x}\right)\phi_{i,j,k} \right] \right.$$

$$\left. + \Delta x_u \phi_{i-1,j,k}{}^{+} \right\} \Big/ \left[\tfrac{1}{2}\Delta x_\ell \Delta x_u (\Delta x_\ell + \Delta x_u) \right] \tag{13}$$

$$\omega_x = 1 + \left(\frac{\Delta x_\ell + \Delta x_u}{\max(\Delta x_\ell, \Delta x_u)} - 1 \right)(\bar{\omega}_x - 1) \tag{14}$$

$$\phi_{rr} = \frac{\Delta r_\ell \phi_{i,j+1,k}{}^{+} - (\Delta r_\ell + \Delta r_u)\phi_{i,j,k}{}^{+} + \Delta r_u \phi_{i,j-1,k}{}^{+}}{\tfrac{1}{2}\Delta r_\ell \Delta r_u (\Delta r_\ell + \Delta r_u)} \tag{15}$$

$$\phi_{\theta\theta} = \left\{ \Delta\theta_\ell \phi_{i,j,k+1} - (\Delta\theta_\ell + \Delta\theta_u)\left[\frac{1}{\omega_\theta}\phi_{i,j,k}{}^{+} + \left(1-\frac{1}{\omega_\theta}\right)\phi_{i,j,k} \right] \right.$$

$$\left. + \Delta\theta_u \phi_{i,j,k-1} \right\} \Big/ \left[\tfrac{1}{2}\Delta\theta_\ell \Delta\theta_u (\Delta\theta_\ell + \Delta\theta_u) \right] \tag{16}$$

$$\omega_\theta = 1 + \left(\frac{\Delta\theta_\ell + \Delta\theta_u}{\max(\Delta\theta_\ell, \Delta\theta_u)} - 1 \right)(\bar{\omega}_\theta - 1) \tag{17}$$

The following is a typical upwind difference formula ($\phi_x > a$). $\phi_{i-2,j,k}$ may be replaced with a surface value and $\Delta x_{\ell\ell}$ with the distance to the surface point.

$$\phi_{xx} = \frac{(\Delta x_\ell + \Delta x_{\ell\ell})(\phi_{i,j,k}{}^{+} - \phi_{i-1,j,k}{}^{+}) - \Delta x_\ell (\phi_{i,j,k} - \phi_{i-2,j,k})}{\tfrac{1}{2}\Delta x_\ell \Delta x_{\ell\ell}(\Delta x_\ell + \Delta x_{\ell\ell})} \tag{18}$$

For the configuration of Fig. 3b the upwind formula for ϕ_{xx} at A is

$$\phi_{xx} = \left(\frac{\phi_A^{+} - \phi_S^{+}}{\Delta x_1} - \phi_x \Big|_S \right) \Big/ \frac{1}{2}\Delta x_1 \tag{19}$$

Weighting may be used if field and surface points that are extremely close together are used in the formulas.

Cross Derivatives, ϕ_{xr}, $\phi_{r\theta}$, and $\phi_{\theta x}$

The subsonic difference formulas for cross derivatives use Eq. (7) or (8) with previously computed values of ϕ_x or ϕ_r substituted for ϕ.

$$\phi_{xr} = \frac{\partial(\phi_x)}{\partial r}, \qquad \phi_{r\theta} = \frac{\partial(\phi_r)}{\partial\theta}, \qquad \phi_{\theta x} = \frac{\partial(\theta_x)}{\partial\theta} \tag{20}$$

No weighting is used.

Typical examples of upwind differencing are shown next. Refer to Figs. 4a, b, c, respectively, for notation

$$\phi_{xr} = \frac{\phi_{i,j,k}{}^{+} - \phi_{i-1,j,k}{}^{+} - \phi_{i,j-1,k}{}^{+} + \phi_{i-1,j-1,k}{}^{+}}{\Delta x_\ell \Delta r_\ell} \tag{21}$$

$$\phi_{xr} = \left(\frac{\phi_{i,j,k}{}^{+} - \phi_{S_1}{}^{+}}{\Delta x_1} - \frac{\phi_{i,j-1,k} - \phi_{S_2}}{\Delta x_2} \right) \Big/ \Delta r \tag{22}$$

$$\phi_{xr} = \left(\frac{\phi_A{}^{+} - \phi_{S_1}{}^{+}}{\Delta x} - \phi_x \Big|_{S_2} \right) \Big/ \Delta r \tag{23}$$

Special θ-Difference Quotients

Special difference quotients for ϕ_θ and $\phi_{\theta\theta}$ are derived in Ref. 3. These formulas are equivalent to the usual formulas to the order of $\Delta\theta^2$. They are considerably more accurate than the standard formulas if $\Delta\theta$ is large ($\Delta\theta=\pi/4$ or $\pi/2$) and ϕ is of the form

$$\phi(x,r,\theta) = \phi_1(x,r) + \phi_2(x,r)\sin\theta + \phi_3(x,r)\cos\theta \tag{24}$$

ϕ has approximately the preceding form in the far field for inlet computations, and in the near field if the inlet is nearly axisymmetric and centered on the axis. The advantage of the special formulas is improved accuracy for very coarse meshes.

Such very coarse meshes can be desirable if a sequence of computational meshes is used for convergence acceleration. The formulas are

$$\phi_\theta = [\,(1-\cos\Delta\theta_\ell)\,(\phi_{i,j,k+1}-\phi_{i,j,k}) + (1-\cos\Delta\theta_u) \times (\phi_{i,j,k}-\phi_{i,j,k-1})\,]\,/\,[\sin\Delta\theta_\ell\,(1-\cos\Delta\theta_u) + \sin\Delta\theta_u\,(1-\cos\Delta\theta_\ell)\,] \tag{25}$$

$$\phi_{\theta\theta} = \left\{\sin\Delta\theta_\ell\phi_{i,j,k+1} - (\sin\Delta\theta_\ell+\sin\Delta\theta_u)\left[\frac{1}{\omega_\theta}\phi_{i,j,k} + \left(1-\frac{1}{\omega_\theta}\right)\phi_{i,j,k}\right] + \sin\Delta\theta_u\phi_{i,j,k-1}\right\} \div [\sin\Delta\theta_\ell(1-\cos\Delta\theta_u) + \sin\Delta\theta_u(1-\cos\Delta\theta_\ell)] \tag{26}$$

Axis

The points on the axis ($r=0$) are calculated by using the Cartesian form of the potential equation and require 0, 90, 180, and 270 deg to be included in the mesh. The Cartesian form of the potential equation is

$$(a^2-\phi_x^2)\phi_{xx} + (a^2-\phi_y^2)\phi_{yy} + (a^2-\phi_z^2)\phi_{zz} - 2\phi_x\phi_y\phi_{xy} - 2\phi_y\phi_z\phi_{yz} - 2\phi_z\phi_x\phi_{zx} = 0 \tag{27}$$

The difference formulas are straightforward.

Surface Derivatives

Before the field is swept, the velocity is calculated at all surface points using all old ϕ values. First, the velocity components ϕ_{s_x}, ϕ_{s_r}, and ϕ_{s_θ} are computed. These are the components of velocity along constant x, r, and θ cuts of surfaces, and except at end points of the cuts are computed using central-difference formulas similar to Eq. (7). One-sided differences are used at end points. Weighting is used to eliminate oscillations due to very closely spaced points. For any given surface point, only two of these components can be computed in this manner. As an example, ϕ_{s_x} cannot be computed directly for an x-intersection surface point, that is, a point created by a mesh line parallel to the axis intersecting the surface.

At any point the remaining surface derivative can be computed from

$$n_x\sqrt{n_r^2+n_\theta^2}\,\phi_{s_x} + n_r\sqrt{n_\theta^2+n_x^2}\,\phi_{s_r} + n_\theta\sqrt{n_x^2+n_r^2}\,\phi_{s_\theta} = 0 \tag{28}$$

if the magnitude of the direction cosine (n_x with ϕ_{s_x} etc.) is not very small, or by interpolating along a cut from values at adjacent points. After all three surface velocities are available at each point, the velocity components ϕ_x, ϕ_r, and ϕ_θ for surface points are computed from

$$\phi_x = n_x\phi_n - n_r\sqrt{n_x^2+n_r^2}\,\phi_{s_\theta} + n_\theta\sqrt{n_x^2+n_\theta^2}\,\phi_{s_r} \tag{29a}$$

$$\phi_r = n_r\phi_n - n_\theta\sqrt{n_r^2+n_\theta^2}\,\phi_{s_x} + n_x\sqrt{n_r^2+n_x^2}\,\phi_{s_\theta} \tag{29b}$$

$$(\phi_\theta/r) = n_\theta\phi_n - n_x\sqrt{n_\theta^2+n_x^2}\,\phi_{s_r} + n_r\sqrt{n_\theta^2+n_r^2}\,\phi_{s_x} \tag{29c}$$

where $\phi_n = 0$.

These velocity components are used in the calculation of the cross derivatives, ϕ_{xr}, $\phi_{r\theta}$, and $\phi_{\theta x}$, for subsonic flow, in the surface-point boundary-condition Eq. (34) under certain circumstances, and for program output.

Equation Weighting for Near Surface Points

The finite-difference equation for a field point next to one or more surface points can cause numerical problems if the spacing to adjacent surface points is small relative to the local mesh spacing. If a field point is close to a surface point, it is desirable that the ϕ values of the field and surface points be consistent. This analysis resolves these problems by using the flow equation for the field point when it is not close to surface points, and interpolates for the field point using the surface point value(s) when the surface and the field point are very close. This is accomplished by the following formulas which weight the flow equation and interpolation formulas together according to the step size. The weights β_x, β_r, and β_θ for the x, r, and θ interpolation formulas, respectively, are computed by using the following relationships:

$$\begin{aligned} \beta_{1x} &= 1-2\Delta x_1/\Delta x_0 && 2\Delta x_1 \le \Delta x_0 \\ &= 0 && 2\Delta x_1 > \Delta x_0 \\ &= 0 && \text{field point is not adjacent in } x \text{ to a surface point} \end{aligned} \tag{30}$$

The formulas for β_{1r} and $\beta_{1\theta}$ are similar.

$$\beta_x = \beta_{1x} - \tfrac{1}{2}\beta_{1x}(\beta_{1r}+\beta_{1\theta}) + \tfrac{1}{3}\beta_{1x}\beta_{1r}\beta_{1\theta} \tag{31}$$

The formulas for β_r and β_θ are similar.

The interpolation formula (see Fig. 3b) for x interpolation is

$$\phi_A^+ = (1-\eta^2)\phi_S^+ + \eta^2\phi_B^* + (\Delta x_1+\Delta x_2)\phi_x|_S\,\eta(1-\eta) \tag{32}$$

where

$$\eta = \Delta x_1/(\Delta x_1+\Delta x_2)$$

and ϕ_B^* may be ϕ_B or ϕ_B^+ if available.

The final finite-difference equation for the field point A is

$$\begin{aligned} &(1-\beta_x-\beta_r-\beta_\theta)(\text{finite-difference flow equation}) \\ &\quad + C\beta_x\,(x \text{ interpolation equation}) \\ &\quad + C\beta_r\,(r \text{ interpolation equation}) \\ &\quad + C\beta_\theta\,(\theta \text{ interpolation equation}) = 0 \end{aligned} \tag{33}$$

where C is the coefficient of ϕ_A^+ in the finite-difference flow equation for point A.

Surface Boundary Condition

The boundary condition at solid surfaces is no flow through the surface, which requires that the velocity normal to the surface ϕ_n be zero. The velocity ϕ_n can be expressed in terms of its components and the direction cosines (n_x, n_r, n_θ) of the unit surface normal $\boldsymbol{n}$,

$$\phi_n = n_x\phi_x + n_r\phi_r + n_\theta(\phi_\theta/r) \tag{34}$$

The method of differencing the surface boundary condition has been substantially changed from the preceding work[3] to eliminate problems where some of the field points used in the previous difference formulas were not in the flowfield. This can occur, for example, where a concave region exists.

Referring to Fig. 5 for notation, the difference formula for ϕ_n for a r-intersect surface point created by a radial mesh line

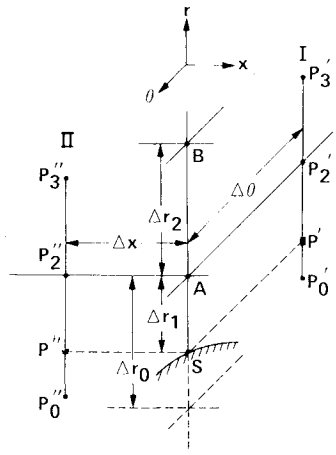


Fig. 5 Nomenclature for typical surface-point boundary-condition equation.

intersecting the surface is

$$n_x\left[2\frac{\phi_S^+-\phi_{P''}}{\Delta x}-\phi_x\Big|_{P''}\right]+n_r\left\{\beta\left[2\frac{\phi_A^+-\phi_S^+}{\Delta r_1}-\phi_r\Big|_A\right]\right.$$
$$\left.+(1-\beta)\left[2\frac{\phi_B^+-\phi_S^+}{\Delta r_1+\Delta r_2}-\phi_r\Big|_B\right]\right\}$$
$$+\frac{n_\theta}{r}\left[2\frac{\phi_S^+-\phi_{P'}}{\Delta\theta}-\phi_\theta\Big|_{P'}\right]-\phi_n=0 \qquad (35)$$

where

$$\beta=(2\Delta r_1/\Delta r_0)^2 \qquad 2\Delta r_1\le\Delta r_0$$
$$=1 \qquad 2\Delta r_1>\Delta r_0$$

The values of ϕ at the points P′ and P″ are obtained by a second-order interpolation along the radial lines I and II. The points P_0 and P_3 may be field or surface points. If the points P′ and P″ are not in the flowfield but are in the body, Eq. (35) is modified by using the value for ϕ_θ or ϕ_x (if P′ and P″ are not in the field, respectively) computed using central differencing along the surface and Eqs. (29) instead of differencing using $\phi_{P'}$ or $\phi_{P''}$.

The formulas for θ- or x-surface intersection points are similar. If the surface point is also a mesh point, P′ and P″ are field points, point B is not used, and $\beta=1.0$. If the surface point is on the axis or adjacent to the axis, similar formulas are used.

Code Organization

One of the major problems with three-dimensional codes is storage in the computer. Storing a three-dimensional field when using a typical mesh requires a large number of words. As an example, an inlet mesh with x, r, and θ dimensions of $81\times41\times16$, respectively, requires 53,136 locations for the potential function alone. If a copy of the potential function at a previous step is required for extrapolation or other convergence acceleration procedures, another 53,136 locations are required. Any geometric information requires additional storage. As these numbers are large compared to what is available, even for the new vector "super" computers, it is essential that the scheme used for storing and accessing geometrical information be designed to minimize storage requirements. Disk storage may be used to provide more memory, but large time penalties are involved making it desirable to minimize such use. Another criterion for developing codes is computational efficiency. Code efficiency on a vector machine is improved by ensuring that most computation is done in a vector mode. For the CYBER 200 family of computers, efficiency increases with vector length to a maximum vector length of 65,535.

Storing a flag or index for each field point so that the code can determine if the field point is next to the surface, and if so which surface point, requires at least one storage location for each field point for geometric information. This type of scheme is conceptually simple, but inefficient. To minimize storage and maximize the percentage of vector code, an inverse scheme is used in this code. Surface-affected field points are processed by sweeping through the surface points. A bit array, rather than an array of full words (64 bits), is used to signal whether grid points are in the flowfield or not. A bit array is used for each of the three coordinates to mark whether field points are surface adjacent in either or both directions in that coordinate. Total storage for the four bit arrays is 4/64 of 53,136 or 3321 words. The bit arrays are used to inhibit storage of contributions to coefficients from regular field-point formulas when the field point is irregular (surface affected). The correct contributions to coefficients of the finite-difference equations are then generated and stored by sweeping through the surface points. The geometric information stored for each surface point includes the index of the adjacent field point.

Surface points are grouped by type for each axial plane (plane perpendicular to axis), adjacent in x, adjacent in r, adjacent in θ, or both a surface point and a field point. Quantities needed for coefficient calculation are gathered for all surface points of the same type, the coefficients are calculated as a group using vector arithmetic, and the contributions to the coefficients are then scattered to the correct field-point equations. This maximizes the number of vector calculations and minimizes the storage of geometric information. The primary penalty is the inefficiency of the gathering and scattering operations on the CYBER 203 computer. The gather/scatter operations are efficient on other computers and projected versions of the CYBER 203.

Solution Process

The first step for a relaxation sweep of the field is to calculate the three components of velocity for all surface points. The field is then swept one axial plane at a time. The first derivatives at all field points on an axial plane are calculated one plane ahead of the plane being solved and stored for that plane, the plane being swept, and the previous plane. After the derivatives are calculated, the coefficients for all the difference equations for the plane are calculated. The first radial line is solved using a tridiagonal solver, the point on the axis is updated, and the remaining radial lines are then solved using the new ϕ value on the axis.

Convergence Acceleration

The basic method described is stable and convergent, but the rate of convergence for dense meshes can be very slow and computer times are significant. There is substantial benefit from any procedure that will speed convergence and decrease the cost of a solution. Analysis and code development have been structured to facilitate both current and possible future techniques for convergence acceleration. One technique currently used is extrapolation of solutions as described in Ref. 6. The theory for this technique can be obtained by approximating the solution process as a linear one and looking at eigenvalues of a matrix. The number of eigenvalues is of the order of the number of field points. This approach behaves best when the eigenvalues are discrete, which is more likely when there are few of them. As a consequence, this technique works best for very coarse meshes and much less satisfactorily for fine meshes.

Another technique for convergence acceleration is initial convergence on a coarse mesh to obtain an approximate solution, continuation of the convergence on a medium density mesh to obtain a more accurate approximation to the solution, and then final convergence on a fine mesh. The number of mesh in each direction is doubled for each successive mesh. There is approximately a factor of 16 change in

convergence rate per sweep in terms of computer central-processor time between successive meshes. This arises due to a factor of eight change in number of mesh points and a factor of two in the rate of ϕ change per sweep. Once the solution nears convergence on one of the coarse meshes, there is no further gain in continuing relaxation on the mesh because of the large truncation errors. For the meshes used in the calculations of this paper, there was about one-third decrease in run time using this approach. The reason for such a small improvement is the very slow convergence on the final mesh. It takes half as many sweeps to improve the solution from coarse meshes on the final mesh as it would to do the entire calculation using the finest mesh. Very good convergence on a coarse mesh takes less than 100 sweeps; good convergence using the fine mesh exclusively takes about 400 sweeps. Use of a still finer mesh starts to become very impractical due to the number of sweeps required and the time per sweep.

A further refinement on the mesh changing is the multilevel technique described by Brandt.[8] This procedure uses coarse meshes to resolve the errors determined on finer meshes. The number of sweeps required on the finest mesh should be small (on the order of 10-30) and independent of the fineness of the mesh. The majority of sweeping would be on the coarsest mesh. As can be seen from the discussion of the preceding paragraph this technique becomes better and better, relative to the existing approach, as the meshes become more dense. The analysis has been established with the multilevel procedure in mind, and it will be attempted in the future.

Geometry Specification

Specifying a complex three-dimensional geometry is a difficult problem. The computer code for this analysis requires that all intersections of the mesh with the geometry, plus components of the surface normals, be input. The flow code then orders the points and makes some checks for completeness and consistency. The actual values could come from any source; no particular order is assumed by the flow code. The three-dimensional geometries analyzed in this paper were described by a geometry system that breaks the surface into subsections or patches. The coordinates of the surface on the patch are specified by parametric bicubics. This leads to an explicit specification of the surface. A separate program exists to intersect the surface so described with a computational mesh and generate the coordinates and surface normals at the mesh intersections.

Geometric Capability

The code, in its present form, can handle inlets with or without centerbodies, three-dimensional bodies, ducts, and bodies or inlets in a duct (wind tunnel, for example).

Computational Parameters

The analysis is programmed in CDC CYBER 200 FORTRAN language 1.4[9] for the CYBER 203 (formerly called the STAR 100A) computer. The code makes extensive use of the vector extensions of this language and is explicitly vectorized. Use is made of most of the million-word memory of the CYBER 203 at the NASA Langley Research Center.

A typical fine mesh for an inlet calculation is 69 axial mesh by 37 radial mesh by 16 circumferential mesh, or 40,848 points. These are the maximum dimensions for using a solution process involving three successive meshes at the current time. The array dimensions will eventually be changed to allow more grid points. The calculation of a solution using just the fine mesh to reasonable convergence takes approximately 400 sweeps and 7-8 min of central processor time. Essentially the same solution can be obtained using a sequence of three grids in approximately 5½ min. A reasonable convergence pattern is 100 sweeps on the coarsest mesh ($18\times10\times16$), 150 sweeps on the middle mesh ($35\times19\times16$), and 200 sweeps on the fine mesh. Typical times per sweep are 0.18 s/sweep on the coarse mesh, 0.42 s/sweep on the middle mesh, and 1.1 s/sweep on the fine mesh. Some additional time is required for setting up geometry parameters, mesh changing, and printing the solution. Solutions have been calculated using a sequence of 4, 8, and 16 circumferential meshes, but the coarsest mesh has had a $\Delta\theta$ of 90 deg and the special θ differencing discussed earlier has not been fully incorporated. Consequently, better convergence is presently obtained by keeping the number of θ mesh constant. Run time is slightly dependent on the number of supersonic field points that have to be calculated.

The code has been shown to be extensively vectorized. The CYBER 203 has a new scalar unit and memory that result in a factor of five improvement over the CYBER 200 in handling scalar (nonvector) code. The vector processing capability is essentially unchanged. The run times for this code were reduced only 25% between execution on the CYBER 203 and CYBER 200 indicating scalar code is only a small contribution to the total computer time. There is evidence that gather/scatter operations are a significantly time-consuming part of the code. A recent change in the FORTRAN compiler provides a factor of four to six improvement of efficiency of gather/scatter operations. This change decreased total run times 15%.

Results

Comparisons between the analysis and experiment have been made for two asymmetric geometries and one axisymmetric geometry. In addition, a mixer lobe geometry has been run to verify the program's general three-dimensional capabilities, although no experimental results are available for comparison. The first comparison between theory and experiment is for an axisymmetric 1.26% contraction ratio, 51-cm-diam inlet, tested[10] in the NASA Lewis Research Center 9×15-ft V/STOL propulsion wind tunnel. The contraction ratio of an inlet is the square of the ratio of the radius of the highlight to the radius of the throat; the highlight being the most forward point on the inlet. The contraction ratio is a measure of the thickness of the inlet lip. A high-contraction ratio tends to indicate better cross wind and low speed, high angle-of-attack performance, but greater cruise drag.

The 1.26 contraction-ratio inlet was chosen for a test case because it was easy to handle the geometry early in code development. It is a severe test case because of the high Mach numbers encountered on the inlet lip. The comparison between theory and experiment for the inviscid surface Mach

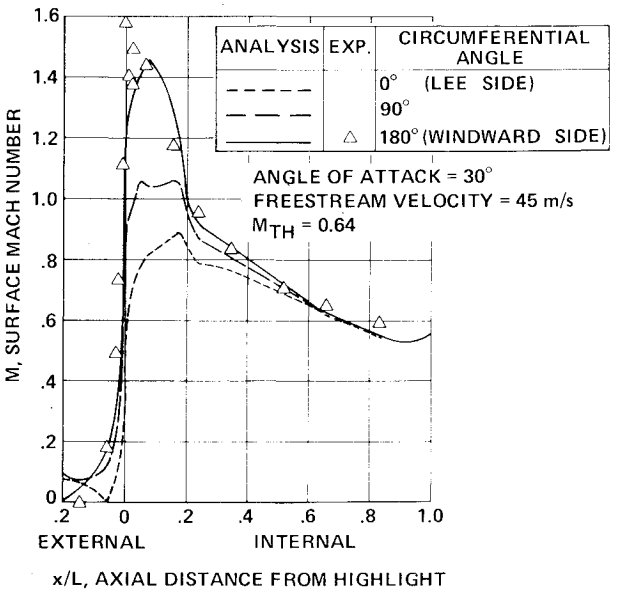


Fig. 6 Cowl surface Mach number distribution for 1.26 contraction ratio inlet.

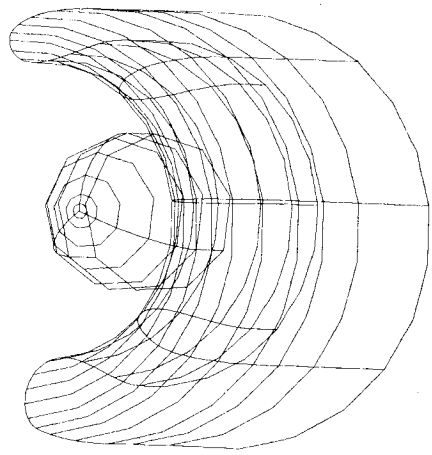


Fig. 7 Graphical display of V/STOL airplane inlet geometry.

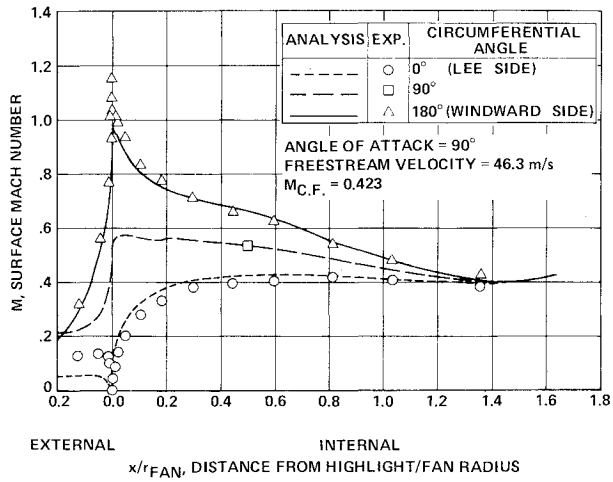


Fig. 10 Cowl surface Mach number distribution for an asymmetric V/STOL airplane inlet.

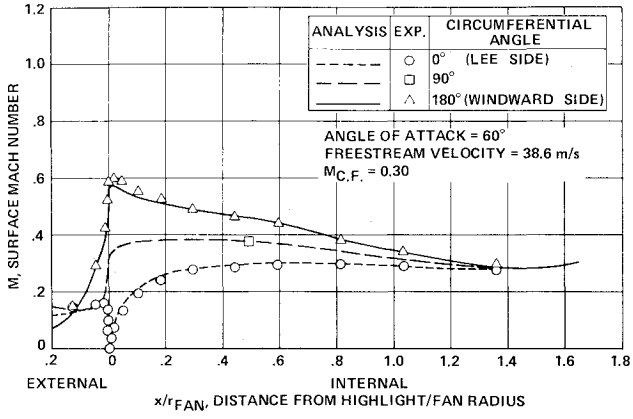


Fig. 8 Cowl surface Mach number distribution for an asymmetric V/STOL airplane inlet.

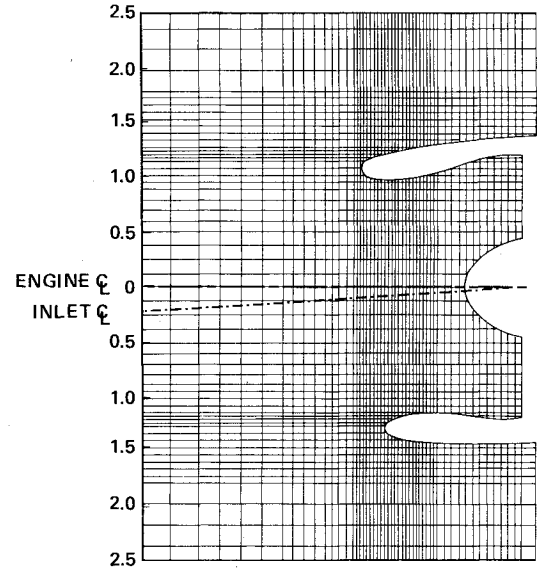


Fig. 11 Computational flowfield and mesh in the vicinity of a commercial-transport turbofan-engine-type inlet.

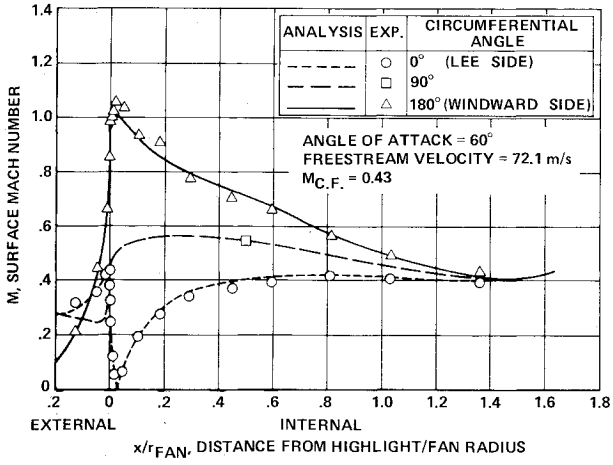


Fig. 9 Cowl surface Mach number distribution for an asymmetric V/STOL airplane inlet.

number is shown in Fig. 6. The results shown are for a high-throat Mach number so there is a large region of supersonic flow and a strong shock. The pressure jump due to the shock wave is smeared over several mesh intervals by the analysis. Agreement between the analysis and experiment is excellent except at the highlight (forward-most point of cowl), where the analysis only qualitatively predicts the spike in the Mach number distribution. This spike is a very local phenomenon caused by a curvature discontinuity on the cowl at the highlight, and the maximum Mach number is probably affected by the boundary layer.

The second comparison between theory and experiment is for an asymmetric tilt-nacelle V/STOL airplane inlet tested in the NASA Ames Research Center 40×80-ft wind tunnel. The model had a fan-face diameter of 1.397 m and was used with a high-bypass turbofan engine. Figure 7 presents a graphical display of the inlet and spinner geometry. The geometry description used in the analysis is smooth and the entire inlet was specified. Only half the inlet is shown in Fig. 7, and curves are represented using linked straight-line segments due to limitations of the plotting equipment. The test and results of the test are described in Ref. 11. Tabular data from the test is found in Ref. 12.

Predictions of the analysis have been compared with experimental measurements for three test cases (Figs. 8-10). There are two comparisons for an angle of attack of 60 deg. The second comparison is for a greater airflow and freestream Mach number than the first. The last comparison is a

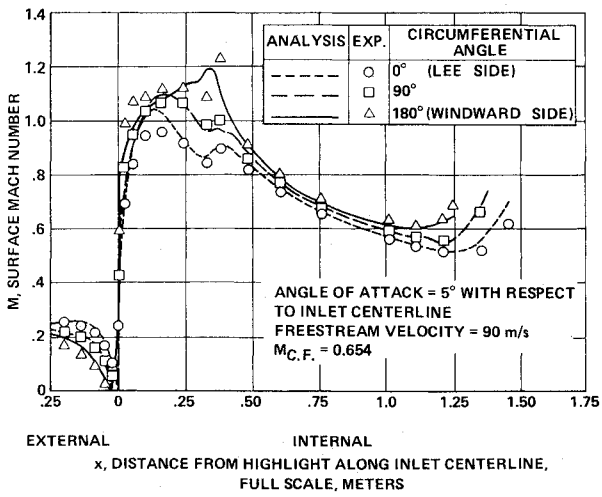


Fig. 12 Cowl surface Mach number distribution for an asymmetric commercial-transport turbofan-engine-type inlet.

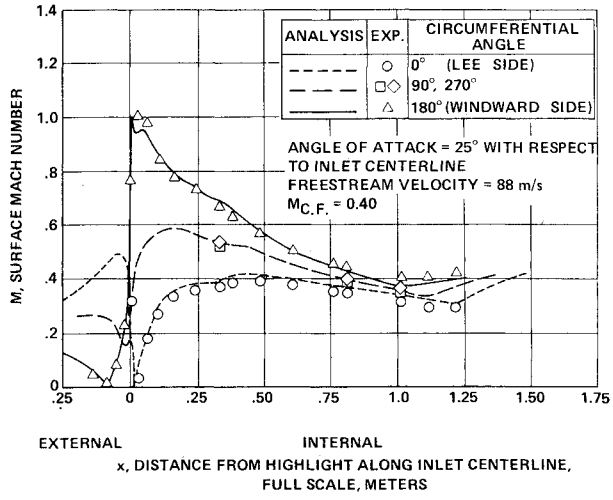


Fig. 14 Cowl surface Mach number distribution for an asymmetric commercial-transport turbofan-engine-type inlet.

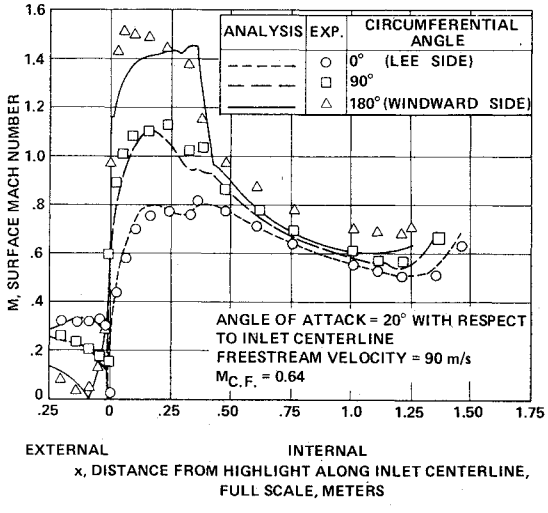


Fig. 13 Cowl surface Mach number distribution for an asymmetric commercial-transport turbofan-engine-type inlet.

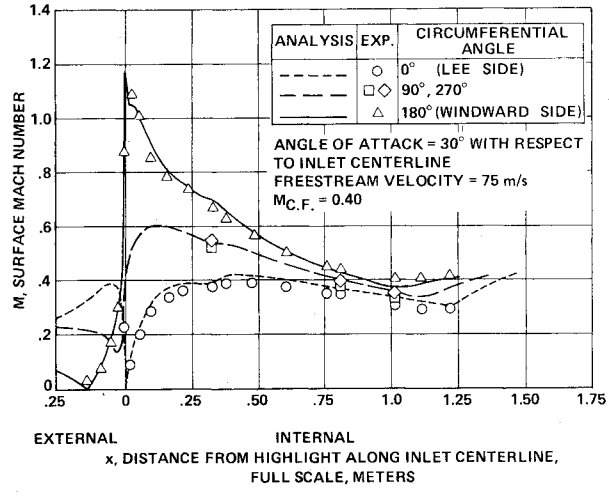


Fig. 15 Cowl surface Mach number distribution for an asymmetric commercial-transport turbofan-engine-type inlet.

crosswind situation, angle of attack equal to 90 deg. The comparisons between analysis and experiment are excellent. There is a slight, but consistent, underprediction of the peak Mach number. This is possibly due to grid density and is discussed later. There is a poor prediction of the external lee side for 90-deg angle of attack (Fig. 10). This probably is an interference or boundary-layer effect in the experiment.

A series of comparisons have been made for a typical commercial-transport turbofan-engine inlet design. It has an asymmetric lip section varying both in section shape and contraction ratio. The contraction ratio on the lee side is 1.246 and on the windward side 1.28. The inlet was designed with an axisymmetric diffuser. The inlet centerline is then tilted down 5 deg with respect to the engine centerline, and a short transition region exists between the inlet and the compressor (fan) face. The tilt of the inlet centerline is for the purpose of reducing cruise drag by aligning the inlet with the local flowfield while still placing the engine and hence its thrust vector in the desired orientation. The inlet flowfield was computed using engine centerline oriented coordinates, and as a consequence, the inlet appears very asymmetric. A cross section of the inlet along with a computational mesh is shown in Fig. 11. Dimensions are in meters for the full-scale inlet.

A 0.16 scale model of the inlet has been tested in the Boeing 9×9-ft low-speed propulsion wind tunnel. The same inlet configuration in a 0.47 scale was tested in the NASA Ames Research Center 40×80-ft wind tunnel. The NASA Ames test model was modified slightly in the vicinity of the fan face to attach to a different engine than that for which the inlet was originally designed. The Boeing 9×9-ft tunnel test used a suction source instead of an engine at the fan face. The 9×9-ft test has results at greater airflows than could be obtained in the Ames test.

The comparisons between analysis and experiment for the 0.16 scale model are shown in Figs. 12 and 13. Figures 14 and 15 show results for the 0.47 scale model. The comparisons are excellent. The most significant discrepancy is for the highest peak Mach number (Fig. 13). Some of the discrepancy is due to blockage effects of a thick boundary layer in the diffuser on the windward side. This condition is near where the inlet separates. The analysis curves are drawn to the compressor station. As the inlet is tilted relative to the engine, each curve ends at a different inlet centerline station.

A single lobe of a mixer has been analyzed with this code. The lobe is shown in Fig. 16. This is a shaded graphic representation[13] of the upper surface of the mixer. The inner

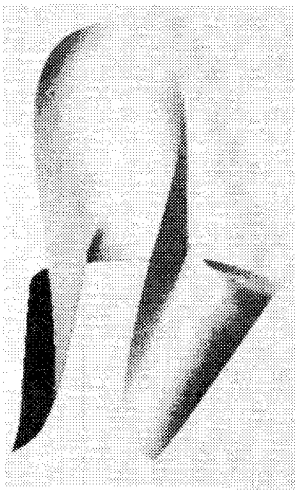


Fig. 16 View of upper surface of mixer lobe.

surface was a constant-diameter cylinder. The lobe extended from 0-45 deg and represents one lobe of an eight-lobe jet-engine mixer nozzle. Half a lobe could have been analyzed by taking advantage of the center plane of symmetry. The mixer shape was developed as a computer-program test case and not an actual mixer. It is shown to illustrate the geometric capabilities of the code. As there is no experiment with which to compare, the computed results are not shown.

Accuracy

The comparisons between analysis and experiment for the V/STOL airplane inlet (Figs. 8-10) show almost perfect agreement except at the point of peak Mach number on the windward side of the inlet. The progression of peak Mach number values for the case of Fig. 9 is 0.743, 0.893, and 1.007 for the three mesh levels used, clearly indicating a trend of an increase in peak Mach number for finer meshes. It is believed that the fine mesh used is the coarsest that will yield adequate accuracy for inlet design and analysis. The mesh used is an excellent tradeoff between cost and accuracy, since the solution for any significantly denser mesh would be extremely expensive to compute. It does appear that use of a finer mesh would result in higher predicted peak Mach numbers and slightly better agreement with experiment. Such calculations will be made when the multilevel technique of Brandt[8] is implemented.

There is an additional check on solution accuracy built into the program. The computed velocity and density profiles in the inlet are integrated at each axial station to compute mass flow and the error relative to the mass flow enforced as a boundary condition is determined. Typical maximum errors are 0.5-1.5%. The magnitude of the error is a function of the complexity of the geometry and the degree of convergence.

Overall, the program appears to very accurate for predicting inviscid flowfields. At flow conditions where Mach number gradients are large and boundary layers are thicker, or shock-wave/boundary-layer interactions are present, agreement is not as good. It is expected that use of more dense meshes and more severe convergence tolerances would improve agreement slightly with experiment at the cost of significantly increased computer-time requirements.

Conclusions

This approach has been shown to work and be extremely powerful. In its present form the code has very flexible geometric capabilities. A local modification to geometry, such as a bump, has no effect on the solution procedure.

The next major area of effort will be an attempt to speed the relaxation process in order to allow finer meshes with resultant greater accuracy and/or more complex geometries. The code has been structured to enable use of the multilevel technique.

Acknowledgments

Use of the CYBER 203 computer is through a cooperative agreement with the NASA Langley Research Center, Hampton, Va. The author wishes to acknowledge the assistance of R. G. Jorstad of Boeing Computer Services, Inc., who aided with the development of the computer code.

References

[1] Murman, E. M. and Cole, J. D., "Calculation of Plane Steady Transonic Flows," *AIAA Journal,* Vol. 9, Jan. 1971, pp. 114-121.

[2] Jameson, A., "Iterative Solution of Transonic Flows Over Airfoils and Wings, Including Flows at Mach 1," *Communications in Pure and Applied Mathematics,* Vol. 27, May 1974, pp. 283-309.

[3] Reyhner, T. A., "Transonic Potential Flow Around Axisymmetric Inlets and Bodies at Angle of Attack," *AIAA Journal,* Vol. 15, Sept. 1977, pp. 1299-1306.

[4] Caughey, D. A. and Jameson, A., "Numerical Calculation of Transonic Potential Flow About Wing-Body Combinations," *AIAA Journal,* Vol. 17, Feb. 1979, pp. 175-181.

[5] Chen, L.-t. and Caughey, D. A., "Higher-Order, Finite-Difference Scheme for Three-Dimensional Transonic Flowfields about Axisymmetric Bodies," *Journal of Aircraft,* Vol. 17, Sept. 1980, pp. 668-676.

[6] Reyhner, T. A., "Cartesian Mesh Solution for Axisymmetric Transonic Potential Flow Around Inlets," *AIAA Journal,* Vol. 15, May 1977, pp. 624-631.

[7] Lomax, H. and Steger, J. L., "Relaxation Methods in Fluid Mechanics," *Annual Review of Fluid Mechanics,* Vol. 7, Annual Reviews Inc., Palo Alto, Calif., 1975, pp. 63-88.

[8] Brandt, A., "Multilevel Adaptive Computations in Fluid Dynamics," *AIAA Journal,* Vol. 18, Oct. 1980, pp. 1165-1172.

[9] CDC Cyber 200 FORTRAN Language 1.4, Reference Manual, Control Data Corporation, Sunnyvale, Calif., No. 60457040, 1979.

[10] Albers, J. A., "Comparison of Predicted and Measured Low-Speed Performance of Two 51-Centimeter-Diameter Inlets at Incidence Angle," NASA TM X-2937, 1973.

[11] Syberg, J. and Koncsek, J. L., "Low Speed Tests of a Fixed Geometry Inlet for a Tilt-Nacelle V/STOL Airplane," NASA CR-151922, 1977.

[12] Shain, W. M., "Test Data Report, Low-Speed Wind Tunnel Tests of a Full Scale Lift/Cruise-Fan Inlet, With Engine, At High Angles of Attack," NASA CR-152055, Jan. 1978.

[13] Lane, J. M., Carpenter, L. C., Whitted, T., and Blinn, J. F., "Scan Line Methods for Displaying Parametrically Defined Surfaces," *Communications of the ACM,* Vol. 23, Jan. 1980, pp. 23-24.