AIAA 80-1348R

Solution of the Three-Dimensional Navier-Stokes Equations on a Vector Processor

Lawrence W. Spradley,* John F. Stalnaker,† and Alan W. Ratliff‡
Lockheed-Huntsville Research and Engineering Center, Huntsville, Ala.

The general interpolants method (GIM) was developed originally for the inviscid Euler equations and is extended in this work to the time-averaged Navier-Stokes and to solution algorithms for vector processors. The GIM algorithm is formulated for discretizing the unsteady equations of motion for arbitrary three-dimensional geometries and nonuniform grids. The finite-difference coefficients and transformation metrics are organized into an interleaved data base which allows vectorization with large vector lengths. The GIM algorithm is designed to take advantage of the pipelining features of the CDC STAR-100 machine. Results are presented for computation of two- and three-dimensional viscous flows. A two-dimensional nozzle-exhaust shear flow case is solved and compared to an inviscid slipline code. This nozzle-exhaust flow case is then extended to three dimensions and the steady solution obtained and displayed as pressure contours in the cross planes. The third problem consists of a simulated two-dimensional aircraft inlet flow which includes calculation of the "spillage" flow and a shock wave/boundary-layer interaction. Computer run times for a number of test problems have shown factors of six in speed improvement over a CDC 7600 scalar version of the code (factor of twelve on the new Cyber 203 version).

Nomenclature

$A_{ij}, B_{ij}, C_{ij}, D_{ij}$	= GIM nodal analog coefficient matrices
C	= mass fraction (concentration) of species 1 of binary gas
$\mathfrak{D}$	= binary diffusion coefficient
$\bar{E}_i$	= edge functions for hexahedral
E, F, G	= flux vectors in conservation law differential equation
$\mathcal{E}$	= total energy per unit volume
f	= any flow function
h	= static enthalpy
I	= general interpolation functions
i, j	= node points of an element
k	= thermal conductivity
M	= Mach number
m, n	= node points in assembled domain
$\bar{P}_i$	= corner points of hexahedral
P	= static pressure
q	= heat flux
R	= mass diffusion terms in species equation
$\bar{S}_i$	= surface functions for hexahedral
T	= static temperature
t	= time
u, v, w	= Cartesian coordinate velocity components
U	= vector of conservation variables
V	= volume of an element
W	= weight function
$\bar{X}$	= vector of Cartesian coordinates
x, y, z	= Cartesian coordinate components
γ	= ratio of specific heats
η	= local curvilinear element coordinates
ρ	= mass density
τ	= stress terms in Navier-Stokes equations
μ	= first coefficient of viscosity
λ	= second coefficient of viscosity ($-2/3\ \mu$)
ϕ	= conservation law differential equation
Φ	= discrete analog of ϕ

Presented as Paper 80-1348 at the AIAA 13th Fluid and Plasma Dynamics Conference, Snowmass, Colo., July 14-16, 1980; submitted Sept. 16, 1980; revision received March 5, 1981.

*Staff Engineer. Member AIAA.
†Scientist, Associate Research. Member AIAA.
‡Group Engineer.

Introduction

THE general interpolants method (GIM) code[1] was developed to analyze complex flowfields which defy solution by simple methods. The code uses numerical difference techniques to solve the full three-dimensional time-averaged Navier-Stokes equations in arbitrary geometric domains. The equations are cast in strong conservation law form and written in an orthogonal Cartesian coordinate system. Included are a continuity equation for global mass conservation, three components of momentum conservation, total energy conservation, and an equation for conservation of individual species of a binary gas. Pressure is related to the conservation variables through the ideal gas law for a binary mixture. A generalized geometry package is used to model the flow domain, generate the numerical grid of discrete points and to compute the local transformation metrics. The GIM approach essentially combines the finite element geometric point of departure with finite-difference explicit computation analogs. This provides a capability which takes advantage of the geometric flexibility of an element description and the computation speed of difference representations.

The numerical analogs of the differential equations are derived by representing each flow variable with general interpolation functions. The point of departure then requires that a weighted integral of interpolants be zero over the flow domain. By choosing the weight functions to be the interpolants themselves, the GIM formulation produces identically the classical implicit finite-element discrete equations. These forms are not used in the GIM code due to their fully implicit nature and inherent inefficiencies. Rather, the weight functions are chosen to be orthogonal to the interpolant functions which produces explicit finite-difference type discrete analogs. By appropriate choice of constants in the weight functions, the GIM becomes analogous to standard finite difference schemes such as centered, backward, forward, windward, and multistep predictor-corrector schemes. The GIM analogs, however, are automatically produced for arbitrary geometric flow domains and, hence, are a general point of departure and provide flexibility in the choice of differencing schemes.

The GIM formulation is not a finite element method in the classical sense. Instead, finite-difference methods are used exclusively, but with the difference equations written in general nonorthogonal curvilinear coordinates. Transformations are used to transform the physical planes into

regions of unit cubes. The mesh is generated on this unit cube and the local metric coefficients generated. Each region of the flow domain is likewise transformed and then blended via the finite element formulation to form the full flow domain. In order to treat "completely-arbitrary" geometric domains, different transformation functions can be employed in different regions. The geometry part of the problem is thus treated much like a finite element technique while integration of the equations is done with finite-difference analogs.

The GIM code was developed originally for the inviscid equations of motion and operated on a CDC 7600 computer. Inclusion of viscous terms and computation of three-dimensional shear flow cases subsequently has been accomplished.[2] All three-dimensional Navier-Stokes solvers share two common difficulties—large storage and long run times. The reformulation of the GIM algorithm for the CDC STAR machine[3] is an attempt to alleviate these difficulties to some extent. The finite-difference equations are arranged such that vectorization[4] is accomplished with vector lengths equal to the number of grid points in the problem. This has, of course, not solved all the problem areas but has allowed calculation of three-dimensional viscous shear flows and two-dimensional shock/boundary-layer interactions.[5] For three-dimensional viscous flows presented in this paper, we compute the mixing layer with full viscous terms, but use inviscid free slip on solid boundaries. Thus, large numbers of grid points are not needed to resolve three-dimensional boundary layers. For two-dimensional viscous flows, the inclusion of no-slip walls, resolution of boundary layers and computation of shear flow regions have been accomplished.

This paper presents a summary of the GIM formulation, the vectorization procedure, and some example calculations. Details of each topic are in the references.[1-6]

Equations and Boundary Conditions

The partial differential equations solved here are the Navier-Stokes written in three-dimensional conservation law form for a Cartesian coordinate system.

$$\frac{\partial U}{\partial t}+\frac{\partial E}{\partial x}+\frac{\partial F}{\partial y}+\frac{\partial G}{\partial z}=0$$

$$U=\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ \rho w\\ \rho\mathcal{E}\\ \rho C\end{bmatrix}$$

$$E=\begin{bmatrix}\rho u\\ \rho u^2+P-\tau_{xx}\\ \rho uv-\tau_{xy}\\ \rho uw-\tau_{xz}\\ (\rho\mathcal{E}+P)u-u\tau_{xx}-v\tau_{xy}-w\tau_{xz}-q_x\\ \rho uC-R_x\end{bmatrix}$$

$$F=\begin{bmatrix}\rho v\\ \rho vu-\tau_{xy}\\ \rho v^2+P-\tau_{yy}\\ \rho vw-\tau_{yz}\\ (\rho\mathcal{E}+P)v-u\tau_{xy}-v\tau_{yy}-w\tau_{yz}-q_y\\ \rho vC-R_y\end{bmatrix}$$

$$G=\begin{bmatrix}\rho w\\ \rho wu-\tau_{xz}\\ \rho wv-\tau_{yz}\\ \rho w^2+P-\tau_{zz}\\ (\rho\mathcal{E}+P)w-u\tau_{xz}-v\tau_{yz}-w\tau_{zz}-q_z\\ \rho wC-R_z\end{bmatrix} \tag{1}$$

Throughout this writing these will be termed the Navier-Stokes equations although they include mass and energy conservation. The diffusion terms are defined by the following relations:

$$\tau_{xx}=2\mu\frac{\partial u}{\partial x}+\lambda\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}\right)$$

$$\tau_{yy}=2\mu\frac{\partial v}{\partial y}+\lambda\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}\right)$$

$$\tau_{zz}=2\mu\frac{\partial w}{\partial z}+\lambda\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}\right)$$

$$\tau_{xy}=\mu\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)\quad \tau_{xz}=\mu\left(\frac{\partial u}{\partial z}+\frac{\partial w}{\partial x}\right)\quad \tau_{yz}=\mu\left(\frac{\partial v}{\partial z}+\frac{\partial w}{\partial y}\right)$$

$$q_x=k\frac{\partial T}{\partial x}+\rho\mathcal{D}(h_1-h_2)\frac{\partial C}{\partial x}\quad q_y=k\frac{\partial T}{\partial y}+\rho\mathcal{D}(h_1-h_2)\frac{\partial C}{\partial y}$$

$$q_z=k\frac{\partial T}{\partial z}+\rho\mathcal{D}(h_1-h_2)\frac{\partial C}{\partial z}\quad R_x=\rho\mathcal{D}\frac{\partial C}{\partial x}$$

$$R_y=\rho\mathcal{D}\frac{\partial C}{\partial y}\quad R_z=\rho\mathcal{D}\frac{\partial C}{\partial z} \tag{2}$$

where the τ represents the viscous stress terms, q the heat conduction, and R the mass diffusion terms. The viscosity coefficients μ, λ, the thermal conductivity k, and the binary diffusion coefficient $\mathcal{D}$ are all assumed constant, laminar-type values for the viscous flows treated here. Sutherland's law and turbulent algebraic eddy viscosity models are available in the code.

Boundary conditions for specified nodal points in the mesh are treated via finite-difference techniques.[1] A set of input flags are used to define a point as either known or fixed inflow/outflow, symmetry condition, corner flow condition, no slip wall, wall tangency (slip), supersonic downstream (one-side difference), or zero gradient wall difference. Details of the boundary condition treatment are found in the references.[1-6]

The calculation can start at time = 0 with initial conditions as input, or it can be restarted from the output solution of previous iterations. The solution proceeds for a specified number of time integration steps or can be run until a steady state is reached.

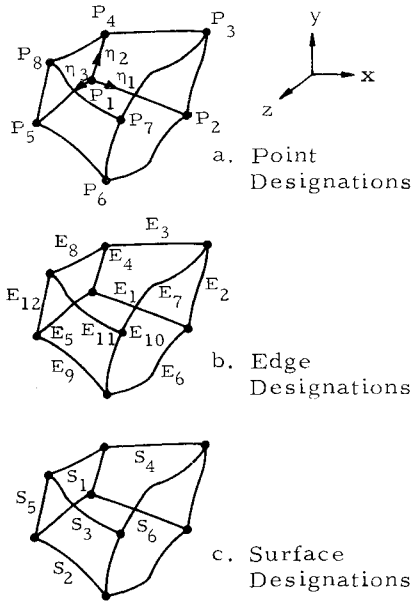


Fig. 1 General hexahedral showing local intrinsic coordinates, point, edge, and surface designations.

Geometric Treatment

An important concept of the theory of finite elements is the "disconnectness" property. This means that a domain can be divided into a finite number of pieces called elements and the approximation to the functional distribution over each element can be studied independently. Thus the approximating functions for each element completely define the behavior of the function profile within the element without consideration of its ultimate location in the full model. After each element is defined, the complete discrete model of the body is obtained by "assembling the system." The assembly is performed by means of mathematical blending of each subdomain while maintaining continuity at the junctions.

The general interpolants method uses these concepts, borrowed from finite element theory, to obtain discrete finite-difference models of the Navier-Stokes equations in arbitrary geometric domains. The development is done in local curvilinear intrinsic coordinates. Analytical regions such as rectangles, spheres, cylinders, hexahedrals, etc., have these intrinsic or natural coordinates. Complex regions can be subdivided into a number of smaller regions that can be described by analytic functions. The degenerate case is to subdivide small enough to use very small straight-line segments. Intrinsic curvilinear coordinate systems can be produced by a univalent mapping of a unit cube onto the simply connected bounded domain.[7] Thus, if a transformation can be found that will map a unit cube univalently onto a general analytic domain, then any complex region can be piecewise transformed and blended using general interpolants.

Consider the general hexahedral configuration shown in Fig. 1. The local intrinsic coordinates are η_1,η_2,η_3 with origin at point P_1. The shape of the geometry is defined by 8 corner points, $\bar{P}_i$ as shown in Fig. 1a; 12 edge functions, $\bar{E}_i$ (Fig. 1b); and 6 surface functions, $\bar{S}_i$ (Fig. 1c). This shape is then fully described if the edges and surfaces can be specified as continuous analytic vector functions $\bar{S}_i(x,y,z)$, $\bar{E}_i(x,y,z)$. Based on the work of Gordon and Hall[7] we have developed a general relationship between physical Cartesian space and local curvilinear intrinsic coordinates. This relationship is given by the following general trilinear interpolant

$$\begin{aligned}\bar{X}(\eta_1,\eta_2,\eta_3) &= (1-\eta_1)\bar{S}_5+\eta_1\bar{S}_6+(1-\eta_2)\bar{S}_2+\eta_2\bar{S}_4\\ &+(1-\eta_3)\bar{S}_1+\eta_3\bar{S}_3-(1-\eta_1)(1-\eta_2)\bar{E}_5-(1-\eta_1)\eta_2\bar{E}_8\\ &-\eta_1(1-\eta_2)\bar{E}_6-\eta_1\eta_2\bar{E}_7-(1-\eta_1)(1-\eta_3)\bar{E}_4\\ &-(1-\eta_1)\eta_3\bar{E}_{12}-\eta_1(1-\eta_3)\bar{E}_2-\eta_1\eta_3\bar{E}_{10}\\ &-(1-\eta_2)(1-\eta_3)\bar{E}_1-(1-\eta_2)\eta_3\bar{E}_9-\eta_2(1-\eta_3)\bar{E}_3\\ &-\eta_2\eta_3\bar{E}_{11}+(1-\eta_1)(1-\eta_2)(1-\eta_3)\bar{P}_1\\ &+(1-\eta_1)(1-\eta_2)\eta_3\bar{P}_5+(1-\eta_1)\eta_2(1-\eta_3)\bar{P}_4\\ &+(1-\eta_1)\eta_2\eta_3\bar{P}_8+\eta_1(1-\eta_2)(1-\eta_3)\bar{P}_2\\ &+\eta_1(1-\eta_2)\eta_3\bar{P}_6+\eta_1\eta_2(1-\eta_3)\bar{P}_3+\eta_1\eta_2\eta_3\bar{P}_7\end{aligned} \tag{3}$$

where the $\bar{X}$ vector is the Cartesian coordinates

$$\bar{X}(\eta_1,\eta_2,\eta_3)=\begin{bmatrix}x(\eta_1,\eta_2,\eta_3)\\ y(\eta_1,\eta_2,\eta_3)\\ z(\eta_1,\eta_2,\eta_3)\end{bmatrix} \tag{4}$$

and $\bar{S}_i$, $\bar{E}_i$ are the vector functions defining the surfaces and edges, respectively, and $\bar{P}_i$ are the (x,y,z) coordinates of the corner points. At the corners of the region, the edge and surface functions degenerate to the points themselves (since they obey cardinality), hence $\bar{X}$ becomes the coordinates of the points $\bar{P}_i$.

Edge functions that currently are included in the GIM code are linear segments, circular arcs, conics (elliptical, parabolic, hyperbolic), helical arcs, and sinusoidal segments. Surface functions included are flat plates, cylindrical surfaces, and edges of revolution. This library of available functions simply is called upon piecewise via input to the computer code. With this transformation, any point in local coordinates η_1,η_2,η_3 can be related to global Cartesian coordinates x,y,z by Eq. (3).

The finite difference grid of discrete points is generated using this concept of multivariate blending function interpolation. The general interpolant, Eq. (3), can accommodate virtually any stretching function to concentrate points near walls, large gradient regions, etc. In addition, the edges of the local hexahedrals can be segmented allowing another means of grid spacing control. The user specifies the number of points desired in each direction and the type and degree of stretching. Note that there is no requirement for orthogonality of the grid lines since the GIM formulation includes the full metric tensor.

Finite Difference Discretization

The method of general interpolants thus has been applied to a treatment of arbitrary three-dimensional geometric regions. We now apply the GIM approach to a discretization of the equations of motion.

Equation (3) relates any flow gradient in η_1,η_2,η_3 space back to physical space. We then need a method of approximating any flowfield function f in local coordinates. This is done with a set of general interpolation functions $I(\eta_1,\eta_2,\eta_3)$ such that

$$f(\eta_1,\eta_2,\eta_3)=\sum I_j(\eta_1,\eta_2,\eta_3)f_j \tag{5}$$

where f_j are the flow variables at the corner points of the element. The simplest form for the I_j are the trivariable Lagrange interpolants. The theory itself does not restrict the I_j to be linear, but in the present code we currently have only the trilinear interpolants. Any flow gradient can then be computed as

$$\frac{\partial f}{\partial \eta_i}=\sum\frac{\partial I_j}{\partial \eta_i}f_j \tag{6}$$

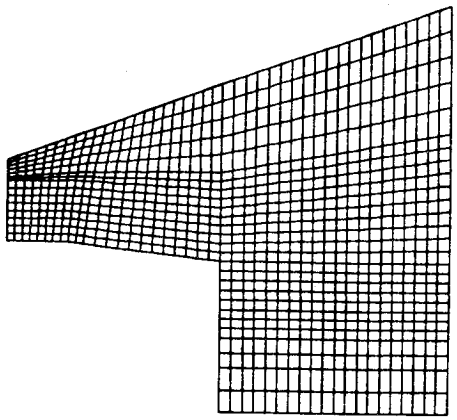


Fig. 3 Grid for two-dimensional shear flow.

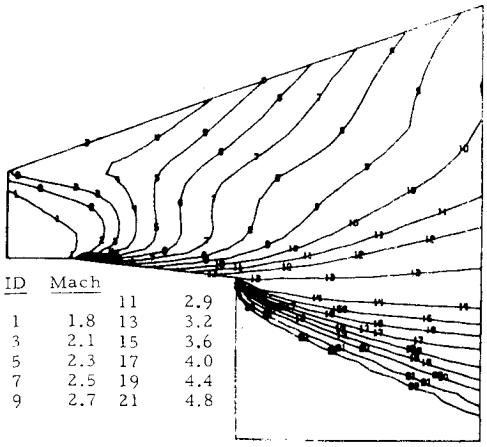


Fig. 4 Mach contours for two-dimensional shear flow.

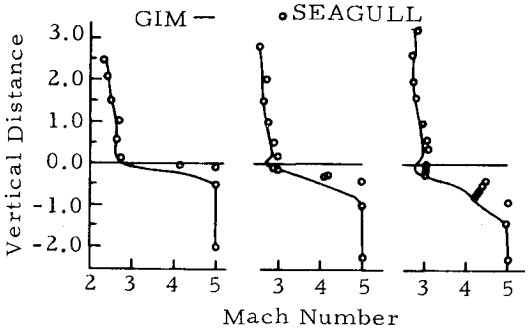


Fig. 5 Mach number vs vertical distance for comparison of GIM and SEAGULL (two-dimensional shear flow).

exhaust gas are assumed to have the same properties in this case for comparison to SEAGULL. The grid consists of 940 points as shown in Fig. 3. The GIM steady-state solution is displayed as contours of Mach number in Fig. 4. The tabular inserts give the contour value for each line. The comparison of the two codes is given in Fig. 5. Plotted here is Mach number vs vertical (y) coordinate at three axial (x) stations downstream of the nozzle exit. The value $y=-2$ is the undisturbed freestream and $y=2.5$, etc., are the upper nozzle wall. As seen from this comparison, the codes agree very well for the inviscid flow and show the expected trend in the mixing region. The shock-fitting code shows a discrete discontinuity in Mach number while GIM shows a smeared region as expected. Note that inviscid, free-slip wall boundaries were used in this case.

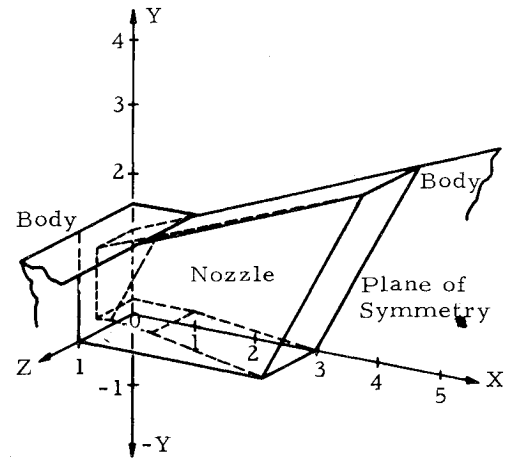


Fig. 6 Three-dimensional nozzle-exhaust flow configuration.

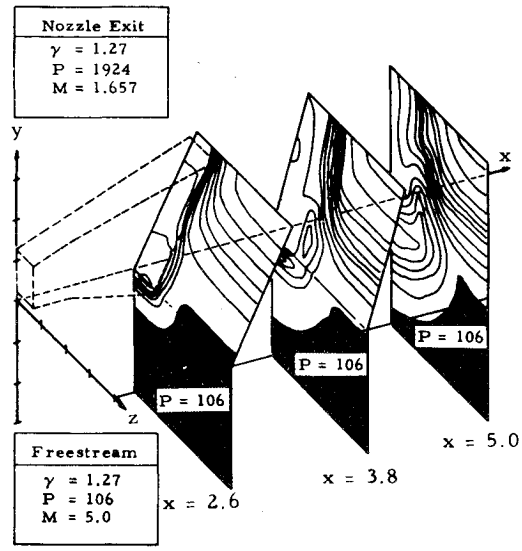


Fig. 7 Steady-state pressure contours for three-dimensional shear flow case.

The second problem is an extension of this two-dimensional example and consists of a three-dimensional viscous shear layer, resulting from the interaction of a nozzle exhaust stream with the freestream, both beneath and beside the nozzle. The configuration, shown in Fig. 6, consists of a rectangular nozzle suspended below a body or wing. The nozzle has a sharp 20-deg upward turn at $x=0$, a sharp 6-deg turn downward at $x=1$, and a sharp 6-deg turn outward along the dashed line located on the sidewall. The rectangular nozzle exit plane is inclined 30 deg from vertical. The body has a sharp 20-deg turn upward at $x=1$. The problem was analyzed with free-slip walls, laminar viscosity in the mixing region, and 9254 grid points.

Figure 7 shows steady-state pressure contours at three axial stations. An outline of the nozzle has been superimposed for reference. The first plane shows the expanded nozzle flow just downstream of the exit plane and the freestream pressure gradient due to the expansion. The next two planes show how the external flow rolls upward and outward when not constrained by the nozzle walls. The pressure gradient across the shock wave (due to the overexpansion) is also illustrated. No data or previous solutions are available for this complex configuration, but the results do show correct physical trends.

The third problem consists of a two-dimensional model of an aircraft inlet. The calculation involves the complete

The code uses these concepts to produce a discrete analog of the Navier-Stokes equations for a single analytical region. The point of departure is the requirement that

$$\int_V W\phi \mathrm{d}V = 0 \tag{7}$$

where W is an arbitrary weight function and Φ represents the differential Eq. (1). The general interpolants I are used to approximate each of the flow variables U,E,F,G and are substituted into Eq. (7) to obtain a discrete analog

$$\Phi \approx [A_{ij}]\dot{U}_j + [B_{ij}]E_j + [C_{ij}]F_j + [D_{ij}]G_j \tag{8}$$

where $\dot{U}_j,E_j,F_j,G_j$ are the flow variables at the node points of an element and the coefficient matrices, geometrically dependent, are given by the following integrals.

$$A_{ij} = \int_V W_i I_j \mathrm{d}V \qquad C_{ij} = \int_V W_i \frac{\partial I_j}{\partial y}\mathrm{d}V$$

$$B_{ij} = \int_V W_i \frac{\partial I_j}{\partial x}\mathrm{d}V \qquad D_{ij} = \int_V W_i \frac{\partial I_j}{\partial z}\mathrm{d}V \tag{9}$$

The transformations for each element, Eq. (4), are substituted into Eq. (9) to obtain quadratures in local element coordinates η_j.[6]

The procedure is then to obtain individual element matrices by numerical Gaussian quadrature evaluation of Eq. (9). The B,C,D matrices may be viewed as "derivative takers" or finite-difference coefficients that contain both the spatial difference scheme and the transformation metrics. The classic treatment of metric data usually employs finite-difference calculations to relate the physical and computational domains. This procedure introduces "noise" into the metrics such that the flowfield itself often feels the disturbance. The GIM treatment is virtually free of metric noise because the coefficients are obtained by fifth-order Gauss quadrature evaluation of Eq. (9). Interpolation functions and high-order quadrature both have a smoothing effect. The metric data, and their derivatives, have shown amazing smoothness when plotted vs the physical coordinates. Another good test of metric noise is to run a uniform flowfield, introduce a perturbation, then relax the calculation to steady state. A GIM code solution returned to a uniform field in 200 steps with the largest error of the order of 10^{-13} in velocity. This general interpolants formulation appears to have added advantage of smooth and accurate metric coefficients.

To model a complex three-dimensional domain, it is first subdivided into regions and Eq. (8) obtained for each region. After all regions are so processed, the coefficients simply are added at all node points that are connected to each other. This "assembly" is characteristic of finite element procedures.

Upon assembly then the ordinary differential equations for the full domain have the form

$$[A_{mn}]\dot{U}_n + [B_{mn}]E_n + [C_{mn}]F_n + [D_{mn}]G_n = 0 \tag{10}$$

At this point, the weight functions W are arbitrary. Thus, in general, Eq. (10) is implicit in the unsteady derivative. If we choose the weight functions W to be the general interpolants I, i.e., $W=I$, then Eq. (10) is a classical Galerkin finite element model. We do not, however, use this description in the GIM code. Rather, we choose the weight functions to be orthogonal to the general interpolants. Inspection of Eq. (9) shows immediately that the A matrix becomes diagonal. Likewise then Eq. (10) becomes explicit in the time derivative. The GIM formulation thus allows either explicit or implicit time-derivative coupling with the finite-difference coefficients containing all of the geometric transformations and mesh spacings. The GIM/STAR code contains only the explicit finite-difference analogs. A fully implicit scheme for three-dimensional problems requires the simultaneous solution of a large system of algebraic equations. These operations are not as readily vectorizable on a machine such as STAR as are the fully explicit schemes.[4]

The ordinary differential-equation system [Eq. (10)] is solved by multistep predictor-corrector schemes,[1] which are analogous to the explicit MacCormack method.[8] After the unsteady derivatives are completely determined (unconstrained), the boundary conditions are applied.[1] The updated state vector is then given by an Euler predictor. Repeating the sequence using provisional values results in a two-step predictor-corrector scheme. With alternating forward-backward spatial differencing a MacCormack scheme in general curvilinear coordinates is obtained.

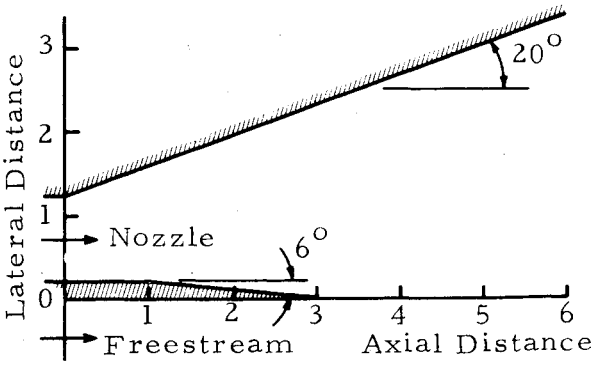
The GIM/STAR code is divided into four modules: mesh generation, nodal analog assembly, unsteady integration, and data display. Most of the CP time for this solution is taken in the integration module. Recall that the form of the algebraic expression coded in GIM is given by Eq. (10) where summation is implied on n over all node points having an influnce on point m. The coefficient arrays B,C,D are constants, dependent only on geometry and grid spacing. Note then that the largest number of operations are of the form

$$\sum_{n=}^{N} B_{mn}E_n \quad m=1,2,\ldots K \tag{11}$$

Where N is at most 27 for three dimensions, but is typically 8 for one-sided differences. The value of K, however, is quite large, i.e., total number of nodes in the problem. This easily can be several thousand for two dimensions and tens of thousands for three dimensions. The vectorization procedure[4] for the STAR system is to use K as the vector length and reverse the order of summation and multiplication. One part of a sum is computed for each node m, followed by a second part, and continued until all components are computed. The scalar method is to perform the matrix product in the normal manner. Our vectorized scheme proceeds down diagonals and then across rows.[6] The result is that long vector lengths are used in the pipeline multiply process.

Results of Computation

The code described in the preceding sections has been applied to a variety of viscous flows in two and three dimensions. The three problems shown here are examples of this computation. The first example is a two-dimensional simulation of a scramjet exhaust with a moving freestream. The case is thus a supersonic free mixing shear flow and the solution can be compared to a forward marching integrator. The code chosen for comparison is SEAGULL,[9] which is an inviscid, supersonic marcher with a slipline treatment of the flow interaction region. The configuration and flow conditions are shown in Fig. 2. The freestream gas and the



	γ	M	P
Nozzle	1.27	1.657	1924
Freestream	1.27	5.0	106

Fig. 2 Two-dimensional nozzle-exhaust configuration.

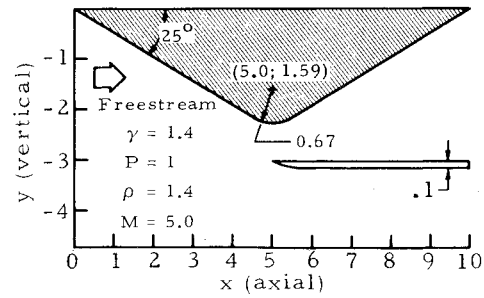


Fig. 8 Configuration for two-dimensional model inlet flow.

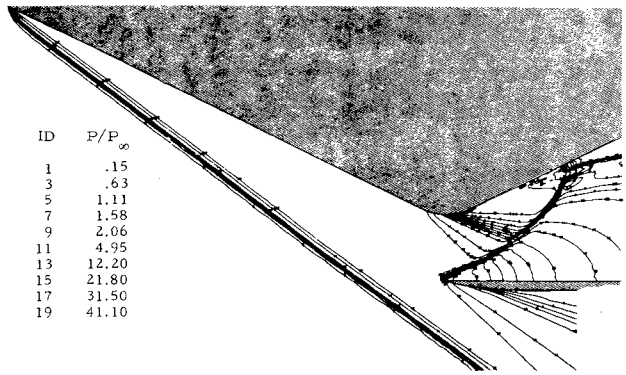


Fig. 9 Pressure contours for the two-dimensional inlet.

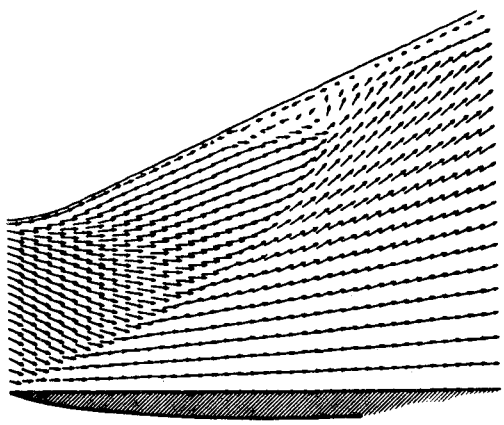


Fig. 10 Profiles of velocity in diffuser section.

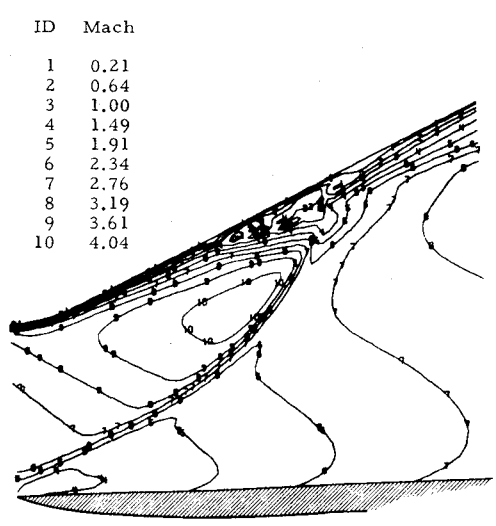


Fig. 11 Profiles of Mach number in diffuser section.

flowfield in the configuration shown in Fig. 8. The compression/expansion surface is attached to the body or wing of the flight vehicle. The compression surface makes a sharp 25-deg turn at $x=0$. It turns 50 deg through a circular arc centered about $x=5$ into the 25-deg expansion surface. The expansion surface and the lower cowl form the diffuser. The freestream flight conditions are also shown in the figure. All flow variables are made dimensionless with the freestream quantities. For inlets with fixed geometry it is important to know the amount of flow captured by the inlet and the amount that spills into the freestream. The model inlet was designed inviscidly to capture two-thirds of the incident flow. The calculation was started upstream of the compression surface which turns at 25 deg to the horizontal. The $M=5$ freestream flow generates a bow shock off the leading edge of the ramp. The calculation involved determining the amount of flow captured and computing the diffuser flowfield. The problem was run in two parts with the GIM code on the STAR machine. The spillage flow was determined first and found to be 66% of the incoming stream (inviscidly). This agrees well with the numbers for which the simulated inlet was designed. The bow shock was captured as a part of the solution.

The flowfield distribution at the diffuser entrance was then used to drive the internal flow and allow the performance parameters to be determined on the propulsion surface. The flow angularity produces a shock wave off the cowl lip which propagates into the diffuser. Since the incoming flow was computed inviscidly, it was necessary to impose a laminar boundary layer on the upper wall of the nozzle to obtain the solution. The boundary-layer profile at the throat was estimated by a quadratic laminar profile with a Reynolds number of 10^4.

The steady-state profiles for the "front end" and diffuser portions of the flowfield are shown together in Fig. 9. The plot is pressure contours for the full configuration. The capture of the bow wave is seen to be very steep and is well resolved by the GIM grid. The interesting features of this case lie in the diffuser flowfield where the cowl shock, the expanding flow and the laminar boundary layer all interact. For clarity, an enlargement of this flowfield is given in Figs. 10 and 11 for velocity and Mach number. All nodes are not plotted in the velocity map because the 3000 node grid shows up almost black with arrows. These figures show clearly the curvature of the shock wave as it interacts with the expanding nozzle flow. Also evident is the shock wave/boundary-layer interaction and ultimate separation of the boundary layer.

Conclusions

A three-dimensional Navier-Stokes solver for arbitrary geometries was developed and coded in FORTRAN on the CDC STAR 100 vector processor and has shown factors of 6 improvement in speed over a scalar 7600 version. Using the latest version of the STAR, termed Cyber 203, an additional factor of 2 improvement has been achieved. These run times for the GIM code render it economically feasible to compute full Navier-Stokes flowfields in two dimensions and some viscous flows in three dimensions. The run time (s) per grid point per time step for GIM on the Cyber 203 is 2.2×10^{-5} for two dimensions and 3.3×10^{-5} for three dimensions.

Acknowledgments

This work was supported by NASA Langley Contracts NAS1-15783 and NAS1-15795. The authors are grateful for the interest and continued support of this program by the contract monitors J. L. Hunt, Hypersonic Aerodynamics Branch, and J. P. Drummond, Hypersonic Propulsion Branch. Also acknowledged for computer programming support is J. Lambiotte and L. Howser of the Analysis and Computation Division at NASA Langley. The authors also thank members of the Lockheed-Huntsville Computational

Fluid Dynamics Section and, in particular, P. G. Anderson who developed the GIM geometry module and executed the CDC 7600 data cases.

References

[1] Prozan, R. J., Spradley, L. W., Anderson, P. G., and Pearson, M. L., "The General Interpolants Method," AIAA Paper 77-642, June 1977.

[2] Spradley, L. W., Anderson, P. G., and Pearson, M. L., "Computation of Three-Dimensional Nozzle-Exhaust Flows with the GIM Code," NASA CR-3042, Aug. 1978.

[3] Spradley, L. W. and Pearson, M. L., "GIM Code User's Manual for the STAR-100 Computer," NASA CR-3157, Nov. 1979.

[4] Lambiotte, J. J. and Howser, L. M., "Vectorization on the STAR Computer of Several Numerical Methods for a Fluid Flow Problem," NASA-TN D-7545, July 1974.

[5] Spradley, L. W., Stalnaker, J. F., and Ratliff, A. W., "Hyperbolic/Parabolic Development for the GIM/STAR Code," NASA CR-3369, Dec. 1980.

[6] Spradley, L. W., Stalnaker, J. F., and Ratliff, A. W., "Computation of Three-Dimensional Viscous Flows with the Navier-Stokes Equations," AIAA Paper 80-1348, July 1980.

[7] Gordon, W. J. and Hall, C. A., "Construction of Curvilinear Coordinate Systems and Applications to Mesh Generation," *Journal of Numerical Mathematics*, Vol. I, 1973, pp. 461-477.

[8] MacCormack, R. W., "The Effect of Viscosity in Hypervelocity Impact Cratering," AIAA Paper 69-354, May 1969.

[9] Salas, M. D., "Shock Fitting Method for Complicated Two-Dimensional Supersonic Flows," *AIAA Journal*, Vol. 14, May 1976, pp. 583-588.