NASA TM-86773

NASA Technical Memorandum 86773

Simulation of Complex Three-Dimensional Flows

George S. Diewert, Herbert J. Rothmund, and Kazuhiro Nakahashi

July 1985

NASA

National Aeronautics and
Space Administration

NASA Technical Memorandum 86773

Simulation of Complex Three-Dimensional Flows

George S. Diewert, Ames Research Center, Moffett Field, California
Herbert J. Rothmund, ETA Systems, Inc., Saint Paul, Minnesota
Kazuhiro Nakahashi, Ames Research Center, Moffett Field, California

July 1985

NASA
National Aeronautics and
Space Administration

Ames Research Center
Moffett Field, California 94035

N85-32831 #

SIMULATION OF COMPLEX THREE-DIMENSIONAL FLOWS

GEORGE S. DEIWERT,[1] HERBERT J. ROTHMUND[2] AND KAZUHIRO NAKAHASHI[3]

ABSTRACT

The concept of splitting is used extensively to simulate complex three-dimensional flows on modern computer architectures. Used in all aspects, from initial grid generation to the determination of the final converged solution, splitting is used to enhance code vectorization, to permit solution-driven grid adaption and grid enrichment, to permit the use of concurrent processing, and to enhance data flow through hierarchal memory systems. Three examples are used to illustrate these concepts to complex three-dimensional flow fields: 1) interactive flow over a bump, 2) supersonic flow past a blunt-based conical afterbody at incidence to a free stream and containing a centered propulsive jet, and 3) supersonic flow past a sharp-leading-edge delta wing at incidence to the free stream.

INTRODUCTION

In this paper a general approach is described for constructing efficient flow codes for modern computer architectures which are capable of treating a wide variety of complex geometries and of describing a wide variety of flowfield features. Emphasis is placed on a data construct that permits the effective use of multivector processing and permits the treatment of a variety of flows without the need for major recoding. The concept of splitting, defined here to be the use of a sequence of one-dimensional operations to compute a multi-dimensional flow, is relied on to effect these goals.

The simulation of complex three-dimensional flows on digital computers involves two major processes. One is to define the appropriate equation set and solution algorithm so that the solution can be found in an efficient manner and the other is to adequately discretize the computational space so that pertinent flow features can be resolved efficiently. These two processes should be considered concurrently so that the best possible solution procedure is developed.

For more than 20 years it has been recognized that, for large data bases, directional splitting can be used to achieve computational efficiency. For example, Douglas and Gunn [1] in 1964 described an alternating direction implicit method for solving flow-field equations.; MacCormack [2] in 1969 used splitting in conjunction with his explicit method, and Yanenko [3] in 1971 published an excellent volume on "The Method of Fractional Steps." Beam and Warming [4] used approximate factorization to enhance their now widely used implicit method, and, in this meeting, Kovenya [5] again addresses the advantages of splitting.

During this same period there has been continued improvement both in the efficiency of numerical methods and in the speed of digital computers. These improvements permit us

[1]NASA Ames Research Center, Moffett Field, California, USA
[2]ETA Systems, Inc., Saint Paul, Minnesota, USA
[3]NRC Associate, NASA Ames Research Center, Moffett Field, California, USA

today to realistically simulate many complex flows of practical interest. The speed of digital computers has been realized both by improved hardware and by improved architecture. The architecture improvement that has been most dramatic is the vector processor concept. This concept permits orders of magnitude improvements in computer speeds. Modern algorithms should be designed, as much as possible, to take advantage of these vector processing architectures and, when this is done, the single most crucial item in the vector process is the design of the data base (Rizzi [6]).

Three-dimensional flowfields can be considered complex when either the geometry is complex (i.e., the computational space contains holes and corners) or the flowfield itself is complex (i.e., there are regions of strong viscous/inviscid interaction, separated shear layers, oblique shocks, etc.), or both. The problem of discretization (grid generation) is an important consideration in both cases, with the first case (geometry) emphasizing initial grid generation and the other emphasizing a solution-adaptive and/or grid-enrichment procedure. In both cases it is important to consider the solution algorithm, the computer architecture, and the design of the data base, simultaneously, in order to maintain an efficient solution method.

One procedure used to discretize complex three-dimensional flowfields is the zonal approach (e.g., Lee [7] and Holst et al. [8]). In these procedures the computational space is divided into separate and distinct zones, each of which, being simpler than the complete space, is discretized separately. The equation set and solution algorithm can be different for each zone. Inherent in these methods is a scheme to interface the solutions in the different zones to one another. Another procedure used to discretize complex flowfields is the embedded grid approach (e.g., Atta and Vadyak [9] and Benek et al. [10]). In these methods a grid of one type (say a high-resolution, body-oriented, near-surface grid) is completely embedded in a grid of another type (say a coarse grid for the external flow). Again, it is necessary to interface the solutions on the different grids to one another.

A third procedure,which takes advantage of the concept of splitting, is the block/pencil data base approach (e.g., Lomax and Pulliam [11] and Deiwert and Rothmund [12]). Here the topological space is subdivided into blocks which interface one another exactly. These methods are capable of treating complex geometries and flow fields and DO NOT INVOLVE INTERFACING. They are highly vectorizable and are readily amenable to concurrent processing, another architectual enhancement of modern computers.

It is the purpose of this paper to describe a general approach for efficiently simulating complex three-dimensional flows. Emphasis is placed on data structures compatible with modern multivector processors. The concept of splitting (or fractional steps) is used extensively in all aspects of the approach, including grid generation, grid adaption, the solution algorithm, and the data flow. The concepts are general and are relevant to a variety of numerical methods and schemes.

GRID GENERATION

Recently an adaptive grid method was described [13,14] that is based on variational principles and is suitable for multidimensional steady and unsteady flows. The concept of splitting is used to make the method practical, efficient, and robust. In ref. 13 it was shown that by beginning with a uniform grid in a topological box (or system of topological boxes) functions describing the geometry of the body of interest can be used to stretch and cluster grid points and thus generate a suitable starting grid for complex, three-dimensional, flowfield computation. Subsequently this grid can be further refined, using the same scheme used in the initial generation, by adapting the grid points to the developing solution itself, and by enriching the grid in regions where even more detail is desired. The highlights of this scheme are outlined here.

Beginning with a uniform grid distribution in each of the three directions, the points are redistributed in one direction at a time, along grid lines, such that

$$\int_{s_i}^{s_{i+1}} w(s)\, ds = const \qquad (1)$$

for all i, where s_i is the arc length to the i^{th} gridpoint and $w(s)$ is a positive weighting function that defines the stretching and clustering. For distributions across wall-bounded shear layers, a geometric (exponential) function would be used. For distributions around curved surfaces, the local curvature would be used.

To maintain a uniform grid in ξ-space (i.e., computational space), Eq. (1) implies that

$$s_\xi w(\xi) = const \qquad (2)$$

where s_ξ is the metric coefficient and corresponds to the ratio of arc lengths in physical and computational space. Equation (2) is the Euler-Lagrange equation for the minimization of the integral

$$I_v = \int_0^1 w(\xi) s_\xi^2\, d\xi \qquad (3)$$

The minimization of this integral is analogous to minimizing the energy of a system of springs with constants $w(\xi)$ between each pair of grid points.

Additional contraints to control orthogonality (or skewness) and smoothness can be imposed similarly which results in minimizing the energy of a system of torsion springs, between grid lines, located at each node point. (For further details see ref. 14.) By considering these torsion forces from one side only (e.g., upwind) the concept of splitting is maintained, and simple marching procedures can be used in the secondary adapting directions (from one grid line to the next). In the principal adapting direction (along the grid line), the resulting equations are one-dimensionally elliptic and form a simple tridiagonal set.

The application of the scheme to three-dimensional grid generation is illustrated by considering the supersonic flow past a bump. Initially a rectangular grid (49 × 29 × 40) is generated with uniform spacing on each side of a rectangular parallelepiped. One surface of the rectangular parallelepiped is deformed to describe a bump by shortening the z-coordinate in a prescribed manner, fig. 1(a). The shape of the bump is defined by

$$z(x,y) = \begin{cases} .25\,[\sin(4\pi(x-.5) - .5\pi) + 1]\,[\cos(4\pi y) + 1] \\ \qquad \text{for } .5 \le x \le 1 \text{ and } 0 \le y \le .25 \\ 0, \qquad \text{for } x \le .5 \text{ or } 1 \le x \text{ or } .25 \le y \end{cases} \qquad (4)$$

This grid is then redistributed successively in each coordinate direction. Considering the i-direction first, the function, w, is given by an equation

$$w_i = z_{i-1,j,k} - 2z_{i,j,k} + z_{i+1,j,k} \qquad (5)$$

This expression for w clusters the grid points to regions of large curvature in the x-z plane. The end-point mesh spacings $\Delta x_{1,j,k} = x_{2,j,k} - x_{1,j,k}$ and $\Delta x_{imax-1,j,k} = x_{imax,j,k} - x_{imax-1,j,k}$ are specified. In the j-direction. w is specified by the similar expression to Eq. (5) in the y-z plane. Again. end spacings are specified. In the k-direction an exponential function is used to cluster points near the wall surface in order to resolve the wall-bounded shear layer.

The inclination of grid lines are controlled by using torsion springs to make the grid quasiorthogonal. This orthogonality is enhanced near the bump on the wall by increasing the torsion spring coefficients. As shown in fig. 1(b). the grid spacings and grid line inclinations are generically clustered in all three directions. It is generally easier to control the grid by using this scheme than by using elliptic methods. and the generality available is greater than with algebraic methods.

During the course of reaching a steady state. the resulting initial grid is subsequently adapted to the solution. The function w, used initially to generate the grid, is replaced by the computed density gradient. The solution is redistributed onto the newly distributed grid points by using simple one-dimensional interpolation schemes after each directional adaption. For unsteady flows, grid speeds can be determined and used in the conservation equations themselves (see ref. 14).

Figure 1(c) shows computed density contours before grid adaptation. Grid points are then twice adapted in both the j- and k-directions, fig. 1(d), before obtaining the final solution, which is shown in fig. 1(e). This final solution shows a crisper three-dimensional shock surface in front of the bump and separated shear layer behind the bump.

This consistent procedure of grid generation and flow-field computation, coupled with solution-adaptive rediscretization. considerably reduces the computational effort for the three-dimensional grid generation and flow-field computation and provides accurate solutions with a minimal number of points. This approach can be used with any of a wide variety of algorithms, one of which is described briefly in the next section.

ALGORITHM

The equations used to describe three-dimensional interacting flows are the Reynolds-averaged Navier-Stokes equations for compressible flow. To numerically solve these efficiently on present-day computers for systems described with large data bases requires the use of directional splitting. This splitting can be achieved quite naturally with explicit methods where communication between grid points is only with near neighbors (e.g., refs. 2 - 3). Implicit methods, however, require approximate factorization of the sequence of difference operators to realize directional splitting (e.g., ref. 4). This method was used for the computed examples presented herein and is also used here to illustrate the data flow for a split algorithm.

The Navier-Stokes equations can be written in strong conservative form in generalized coordinates as

$$\partial_t Q + \partial_\xi (F \cdot \vec{g}^\xi) + \partial_\eta (F \cdot \vec{g}^\eta) + \partial_\zeta (F \cdot \vec{g}^\zeta) = 0 \tag{6}$$

where

$$Q = J^{-1}\begin{pmatrix}\rho\\ \rho u\\ \rho v\\ \rho w\\ e\end{pmatrix}, \quad F = J^{-1}\begin{pmatrix}\rho\bar{q}\\ \rho u\bar{q} + \tau\cdot\bar{e}_x\\ \rho v\bar{q} - \tau\cdot\bar{e}_y\\ \rho w\bar{q} - \tau\cdot\bar{e}_z\\ e\bar{q} - \tau\cdot\bar{q} - K_t\nabla T\end{pmatrix} \tag{7}$$

$\bar{e}_x$, $\bar{e}_y$, and $\bar{e}_z$ are the Cartesian unit vectors. $\bar{g}^{\xi}$, $\bar{g}^{\eta}$, and $\bar{g}^{\varsigma}$ are the contravariant base vectors, and J is the Jacobian of the transformation.

The stress tensor, τ, is written in terms of the transformed coordinates, ξ, η, and ς and, in keeping with the thin-layer approximation, only those terms are retained which will not result in cross derivatives. (I.e., $u_{\xi\xi}$, $v_{\xi\xi}$, etc., terms are retained but, $u_{\xi\eta}$, $v_{\xi\eta}$, etc., are not.) This permits the consideration of shear layers coaligned with each of the principal coordinate directions, is consistent with thin shear layer theory, and does not inhibit code vectorization.

The corresponding difference equation for Eq. 6, written in operator notation is

$$L_\eta L_\varsigma L_\xi \Delta_t Q = R_\xi + R_\eta + R_\varsigma \tag{8}$$

where the operators are defined by

$$L_\xi = (I + \Delta t\, \delta_\xi\, A^n - \epsilon_I\, J^{-1}\nabla_\xi\Delta_\xi J)$$

$$L_\eta = (I + \Delta t\, \delta_\eta\, B^n - \epsilon_I\, J^{-1}\nabla_\eta\Delta_\eta J)$$

$$L_\varsigma = (I + \Delta t\, \delta_\varsigma\, C^n - \epsilon_I\, J^{-1}\nabla_\varsigma\Delta_\varsigma J)$$

$$R_\xi = -\Delta t\, \delta_\xi\, (JF\cdot\bar{g}^{\xi})^n - \epsilon_E\, J^{-1}\,(\nabla_\xi\Delta_\xi)^2 JQ^n - \epsilon_P\, J^{-1}\,(\nabla_\xi\Delta_\xi p)(\nabla_\xi\Delta_\xi)JQ^n$$

$$R_\eta = -\Delta t\, \delta_\eta\, (JF\cdot\bar{g}^{\eta})^n - \epsilon_E\, J^{-1}\,(\nabla_\eta\Delta_\eta)^2 JQ^n - \epsilon_P\, J^{-1}\,(\nabla_\eta\Delta_\eta p)(\nabla_\eta\Delta_\eta)JQ^n$$

$$R_\varsigma = -\Delta t\, \delta_\varsigma\, (JF\cdot\bar{g}^{\varsigma})^n - \epsilon_E\, J^{-1}\,(\nabla_\varsigma\Delta_\varsigma)^2 JQ^n - \epsilon_P\, J^{-1}\,(\nabla_\varsigma\Delta_\varsigma p)(\nabla_\varsigma\Delta_\varsigma)JQ^n$$

and the δ_ξ, δ_η, and δ_ς are central-difference operators; Δ_ξ, Δ_η, and Δ_ς are forward-difference operators: and ∇_ξ, ∇_η, and ∇_ς are backward-difference operators in the ξ-, η-, and ς-directions, respectively. The Δ_t term is a forward-difference operator in time. For example,

$$\Delta_t Q = Q^{n+1} - Q^n$$

The Jacobian matrices are

$$A = \partial_Q(F_H\cdot\bar{g}^{\xi}) + J^{-1}\partial_Q(F_P\cdot\bar{g}^{\xi})J$$

$$B = \partial_Q(F_H\cdot\bar{g}^{\eta}) + J^{-1}\partial_Q(F_P\cdot\bar{g}^{\eta})J$$

$$C = \partial_Q(F_H\cdot\bar{g}^{\varsigma}) + J^{-1}\partial_Q(F_P\cdot\bar{g}^{\varsigma})J$$

where F_H contains only the convective-like terms of the flux vector F, and F_P contains only the gradient diffusive terms of F. A combination of fourth-order (ϵ_E) and second-order (ϵ_P) explicit smoothing terms and second-order (ϵ_I) implicit smoothing terms have been added to control nonliner instabilities. Variable time stepping. based on local Mach number and on the local Jacobian. can be used to increase the convergence rate to steady state.

Equation 8 is solved in successive sweeps of the data base. with each sweep inverting one of the operators on the left-hand side. The solution is advanced in time by adding $\Delta_t Q$ to Q after the third sweep.

In general the data are operated on four times for each time step advance. First the right-hand side of Eq. 8 is formed by passing through the data base for each direction. one at a time. and then the left-hand-side operators are inverted one by one. For two of the directions (first and third in the operator inversion sequence). both the right-side operator and left-side inversion can be determined in the same step. Typically the data base will be organized for efficient vector operation in one direction only. Depending on the computer architecture. either vector operations employing wide strides or gather/scatter data inversions must be used to realize efficient vector operations in the other two directions. If neither of these constructs are available. then arithemetic inversions. while more costly. should be used

Most modern vector processors are currently available with sufficient high-speed memory that hierarchal memory storage is not a serious problem in the development of efficient codes for most problems. Earlier processors having less. say. than two million words of high-speed memory. however, posed serious constraints on algorithm development for three-dimensional flows. Treatment of some of these problems is discussed in some length in refs. 11 and 12.

DATA STRUCTURE

Geometries for realistic three-dimensional flow problems are not simple. Yet we must have a way to solve a variety of flows without having to recode for every problem. Furthermore. application of boundary conditions should not have a significant effect on the efficiency of the vectorization or on the use of concurrent processing. And finally we must be able to redistribute grid points to achieve adequate resolution at minimal cost.

One way to effect these goals is to map the physical space containing our flow field into a topological box (or group of boxes) in computational space. This permits the use of a general-solution algorithm for a variety of physical geometries. By using body-oriented coordinates in physical space. all boundary conditions can be mapped to planar surfaces in computational space. For algorithms employing the splitting concept, the boundary conditions can be imposed one-dimensionally at the ends of the topological box. and highly efficient vector operations can be used to operate in between. The data can be subdivided into blocks that interface their neighbors in a one-to-one manner. These blocks can be combined to form "pencils" of data in each coordinate direction. Concurrent processing can be used to operate on more than one pencil in each direction simultaneously. There need be no special interfacing between these blocks when the splitting concept is employed. The redistribution of grid points during the course of the solution process can be realized in a straightforward manner by remapping between the physical and computational space. using the variational principles discussed in a previous section. with the concept of splitting used to maintain efficiency. Two examples are chosen to illustrate these concepts: 1) flow past a blunt-based conical afterbody at incidence to a free stream and containing a centered propulsive jet. and 2) flow past a sharp-leading-edge delta wing at incidence to the free stream.

Consider first the afterbody geometry. Shown in fig. 2a is a cone-cylinder forbody with a conical afterbody containing a centered conical nozzle. A physical space control volume containing this body and the flow field about it is a hemispherical (upstream) cylindrical (downstream) shape. In computational space the control volume can be described by a topological box with a notch cut out to fit the body geometry. This is a convenient way to treat geometries with corners and holes. This topological box can be subdivided into blocks which can be matched to specific regions of the physical space and can be stacked together in pencils for efficient vector operations. Shown in fig. 2d is an example of a computational space for the forebody/afterbody geometry. Here the computational space has been subdivided into 14 blocks, which are numbered in the figure. Block 1 has one side coincident with the jet exit plane and blunt-base boundary, and one side coincident with the downstream control volume centerline. Block 2 extends further downstream and has one side coincident with the downstream control volume centerline and another side coincident with the downstream exit plane. These two blocks are stacked together to form a pencil in the ξ-direction, with boundary conditions imposed on the jet exit plane and blunt base and on the downstream exit plane. Block 3 has one side coincident with the upstream centerline and one coincident with the cone/cylinder forebody. Block 4 lies on the cylindrical forebody and on the conical afterbody. These are followed in the ξ-direction by blocks 5 and 6, where block 6 has one side coincident with the downstream exit plane. Blocks 3 through 6 are joined to form a pencil in the ξ-direction with boundary conditions on the upstream centerline and on the downstream exit plane. Blocks 7 through 10 are coaligned with blocks 3 through 6 as are blocks 11 through 14. Blocks 7 and 11 have one side coincident with the upstream centerline, blocks 10 and 12 have one side coincident with the downstream exit plane, and blocks 11 through 14 have one side coincident with the far-field lateral boundary. Each of the 14 blocks has opposing sides coincident with the leeward and windward planes of bilateral symmetry. All remaining sides are adjacent to a neighbor block with a one-to-one correspondence, and no special interfacings are necessary. In the η-direction the blocks are stacked together to form pencils such that blocks 3-7-11 form one pencil with boundary conditions on the forbody surface and at the far field, blocks 4-8-12 form a second pencil with boundary conditions on the cylinder and afterbody surface and at the far field, and blocks 1-5-9-13 and 2-6-10-14 form pencils with boundary conditions on the downstream centerline and far field. In the ς-direction the pencil lengths are just one block long. The blocks can, however, be grouped together to form broader based pencils such as blocks 1 and 2, blocks 3-4-7-8-11-12, and blocks 5-6-9-10-13-14. It is the pencil base dimensions that define the vector length used for vectorized operations. In the present example the block dimensions are each 40 in the ξ-direction, 40 in the η-direction for boxes 1 and 2, 20 for boxes 3 through 14, and 20 in the ς-direction for all blocks. The block boundaries are shown in physical space in fig. 2b and 2c. With the blocks in computational space remaining fixed, the grid in physical space is solution adapted to density and pressure gradients. The computed density and pressure contours and final adapted grid are shown in fig 3 in the vicinity of the afterbody only.

The computed results shown in fig. 3 are for a Mach 2 flow past a blunt-based, 8°, half-angle, conical afterbody containing a Mach 2.5 centered propulsive jet emanating from a 20° half-angle, conical nozzle. The jet-to-free stream static pressure ratio is 2:1, and the body is at a 6° incidence to the oncoming free stream. Plotted in fig. 3a are computed density contours in the bilateral plane of symmetry with the windward in the lower portion and the leeward in the upper. Figure 3b shows computed pressure contours. Flow-field features obvious in these displays include: 1) the boundary layer on the afterbody surface; 2) a lower-density gas on the leeward compared with the windward; 3) a rapid expansion around the nozzle lip; 4) an oblique shock, weaker on the leeward than the windward, that radiates at an angle about 30° to the body axis and emanating just off the annular base; 5) the plume boundary (slip surface) which extends radially downstream of the base just outside the high-density jet flow; 6) a barrel shock that extends downstream of the base inside the plume boundary; and 7) a rapid expansion in the central part of the jet from the high-density, high-pressure exit plane to a low-density, low-pressure core some 1.5 calibers downstream.

Consider next the flow past a sharp-leading-edge delta wing at incidence to the free stream. Shown in fig. 4 is an exploded view of a suggested block structure in both physical and computational space. Blocks 1 through 3 correspond to the windward with the top surface of block 1 coinciding with the lower surface of the wing. Blocks 2 and 3 correspond to the regions downstream of the trailing edge and away from the leading edge, respectively. Blocks 4 through 6 correspond to the leeward, with blocks 5 and 6 corresponding to the downstream and lateral regions, respectively. The bottom surface of block 4 corresponds to the lee surface of the wing and is coincident with the top surface of block 1. This topology is mapped to describe the Dillner delta wing configuration[15] which has a 6% biconvex circular arc profile. Two ξ-pencils are formed by blocks 1-2-3 and 4-5-6, respectively, similarly for two ς-pencils. Four η-pencils are formed by block 1, block 4, blocks 2 and 5, and blocks 4 and 6, respectively. Boundary conditions for the ξ-pencils are supersonic inflow at the upstream boundary and outflow at the downstream. For the ς-pencils symmetry is imposed on one plane and free-stream conditions on the other. For the η-pencils free stream is imposed on the bottom and top surfaces. For the two pencils formed by blocks 1 and 4, though, the wing surface boundary conditions are used on the planes that are coincident with the wing. Initially the block dimensions in the ξ,η,ς-directions are (18x30x18), (12x30x18), (30,30,12). As the solution develops in time the grid is enriched such that these dimensions become (36x35x36), (24x35x36), (30,35,24), (36x55x36), (24x55x36), and (30,55,24), respectively. This grid is also solution-adapted to computed density gradients. Shown in fig. 5a are static surface pressure contours which show a low-pressure region just underneath the leading-edge vortex. In fig. 5b are computed density contours for selected streamwise planes. These contours indicate the low-density fluid in the leading-edge vortex. In fig. 5c are computed particle paths showing the leading edge vortex and surface streamlines.

CONCLUDING REMARKS

The concept of directional splitting in conjunction with a block/pencil data structure has been described for efficiently computing complex three-dimensional flow fields, requiring large data bases, on modern multivector processors. The block data structure permits discretization of complex geometries (i.e., topologies containing holes and corners) while at the same time permitting concurrent vector processing. The concept of splitting is used in grid generation and grid adaption to effect optimal discretizations with a minimal number of points. Splitting is also used to simplify the application of boundary conditions for vectorized algorithms and to eliminate any need for special interfacing of data blocks. Two illustrative examples have been given to show how these concepts are applied. With a little imagination a wide variety of flowfields can be effectivly treated in a similar manner, without any recoding of the general solution procedure.

REFERENCES

[1]Douglas. J.. and Gunn. J. E.. "A General Formulation of Alternating Direction Methods." Numer. Math.. Vol. 6, pp. 428-453. 1964

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

[3]Yanenko. N. N.. "The Method of Fractional Steps." Springer-Verlag. New York. 1971

[4]Beam. R. and Warming. R. F.. "An Implicit Finite Difference Algorithm for Hyperbolic Systems in Conservation-Law-Form." for Complex Fluid Flow Problems." J. Comp. Phys.. Vol. 22. Sept. 1976. pp. 87-110

[5]Kovenya. V. M.. Tarnavsky. G. A. and Cherny. S. G., "Numerical Modeling of Three-Dimensional Flows on the Basis of Splitting Up Method." International Symposium on Computational Fluid Dynamics - Tokyo. Sept. 1985.

[6]Rizzi. A.. "Vector Coding the Finite-Volume Procedure for the Cyber 205." Lecture Series Notes 1983-04, von Karman Inst. for Fluid Dynamics, Brussels, 1983.

[7]Lee. K. D.. "3-D Transonic Flow Computations Using Grid Systems with Block Structure." AIAA Paper 81-0998. 1981.

[8]Holst. T. L.. Thomas. S. D.. Kaynak. U.. Gundy. K. L.. Flores. J.. and Chaderjian. N. M.. "Computational Aspects of Zonal Algorithms for Solving the Compressible Navier-Stokes Equations in Three Dimensions." International Symposium on Computational Fluid Dynamics - Tokyo. Sept. 1985.

[9]Atta. D. H. and Vadyak. T.. "A Grid Interfacing Zonal Algorithm for Three Dimensional Transonic Flows about Aircraft Configurations." AIAA Paper 82-1017. 1982

[10]Benek. J. A.. Buning. P. G. and Steger. J. L.. "A 3-D Chimera Grid Embedding Technique." AIAA Paper 85-1523-CP. 1985.

[11]Lomax. H. and Pulliam. T. H.. "A Fully Implicit Finite-Difference Factored Code for Computing Three-Dimensional Flows on the ILLIAC IV.." Parallel Computations. G. Rodrigue. Ed.. Academic Press. New York. 1982, pp. 217-250

[12]Deiwert. G. S. and Rothmund, H.. "Three-Dimensional Flow Over a Conical Afterbody Containing a Centered Propulsive Jet: A Numerical Simulation," AIAA Paper 83-1709. 1983.

[13]Nakahashi. K. and Deiwert, G. S.. "A Three-Dimensional Adaptive Grid Method," AIAA Paper 85-0486. 1985.

[14]Nakahashi. K. and Deiwert. G. S.. "A Self-Adaptive-Grid Method with Application to Airfoil Flow." AIAA Paper 85-1525-CP, 1985.

[15]Drougge. G. and Larson. P. O., "Pressure Measurements and Flow Investigation on Delta Wings at Supersonic Speed." FFA Report No. 57, 1956.

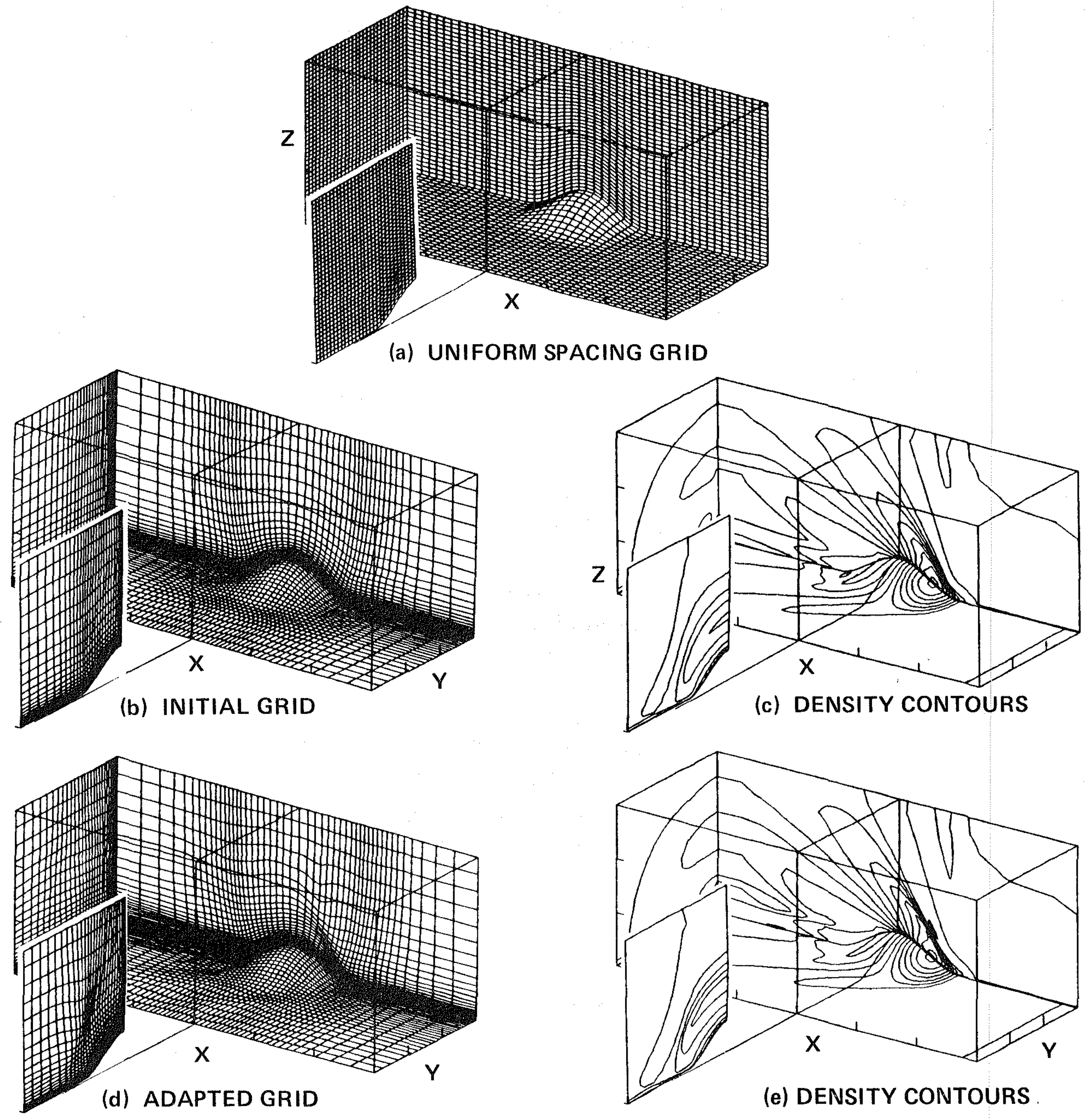

Fig. 1. Solution sequence for three-dimensional flowfields - Supersonic flow past a bump.

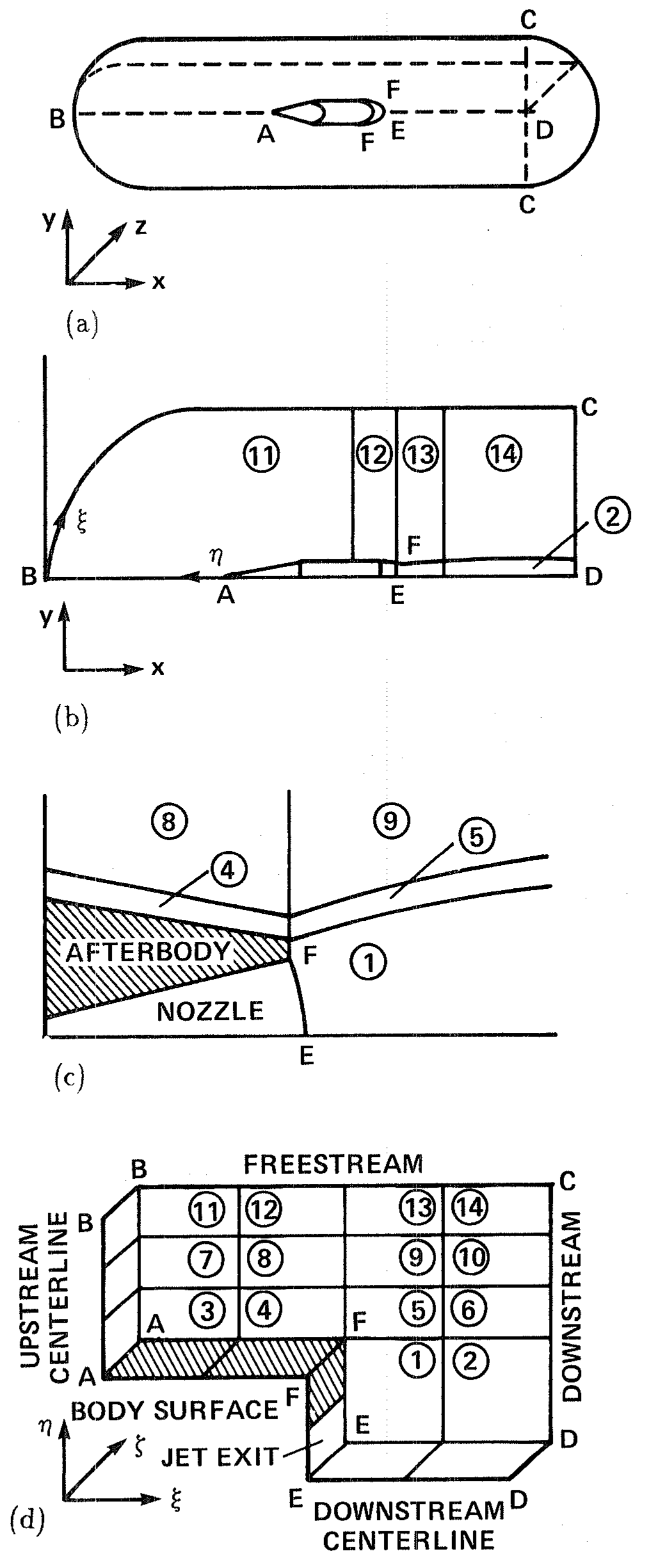

Fig. 2. Afterbody Block Structure: (a) Physical domain, (b) Block structure in physical space. (c) Base region detail. (d) Block structure in computational space.

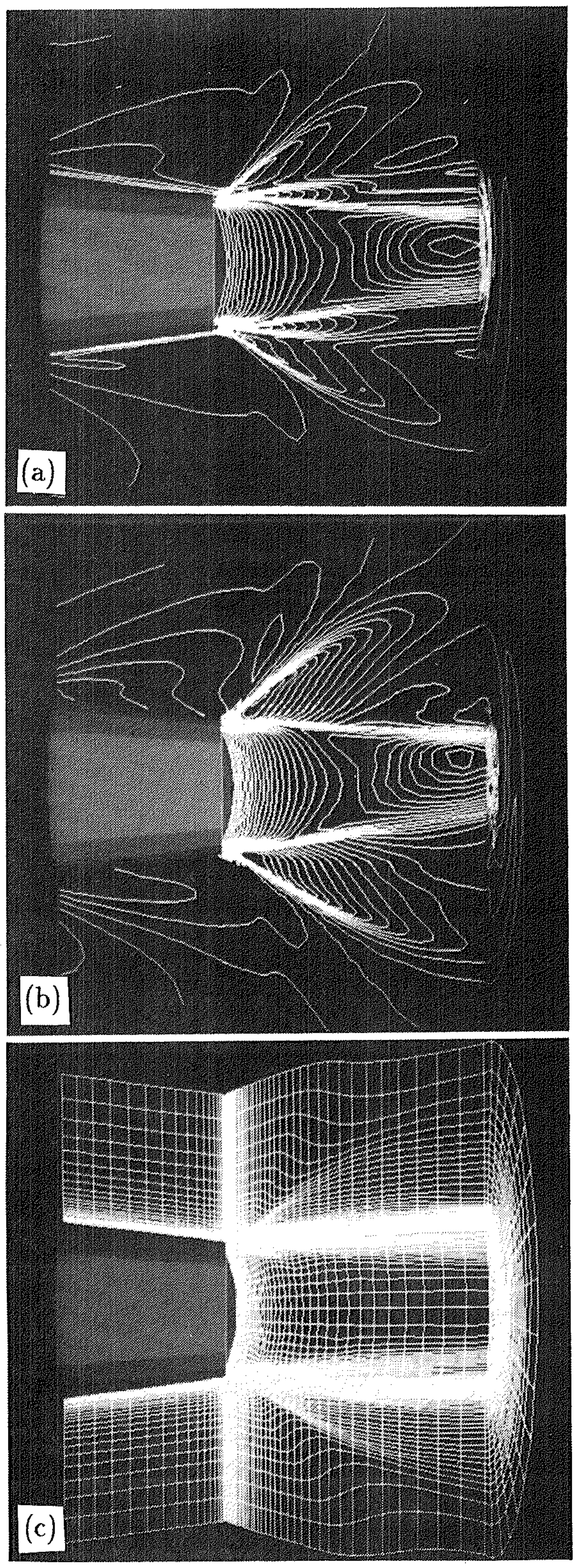

Fig. 3. Computed afterbody flowfield: (a) Isopycnics, (b) Isobars, (c) Solution adapted grid. $M_\infty = 2.0$, $M_j = 2.5$, $p_j/p_\infty = 2.0$, $\alpha = 6^\circ$.

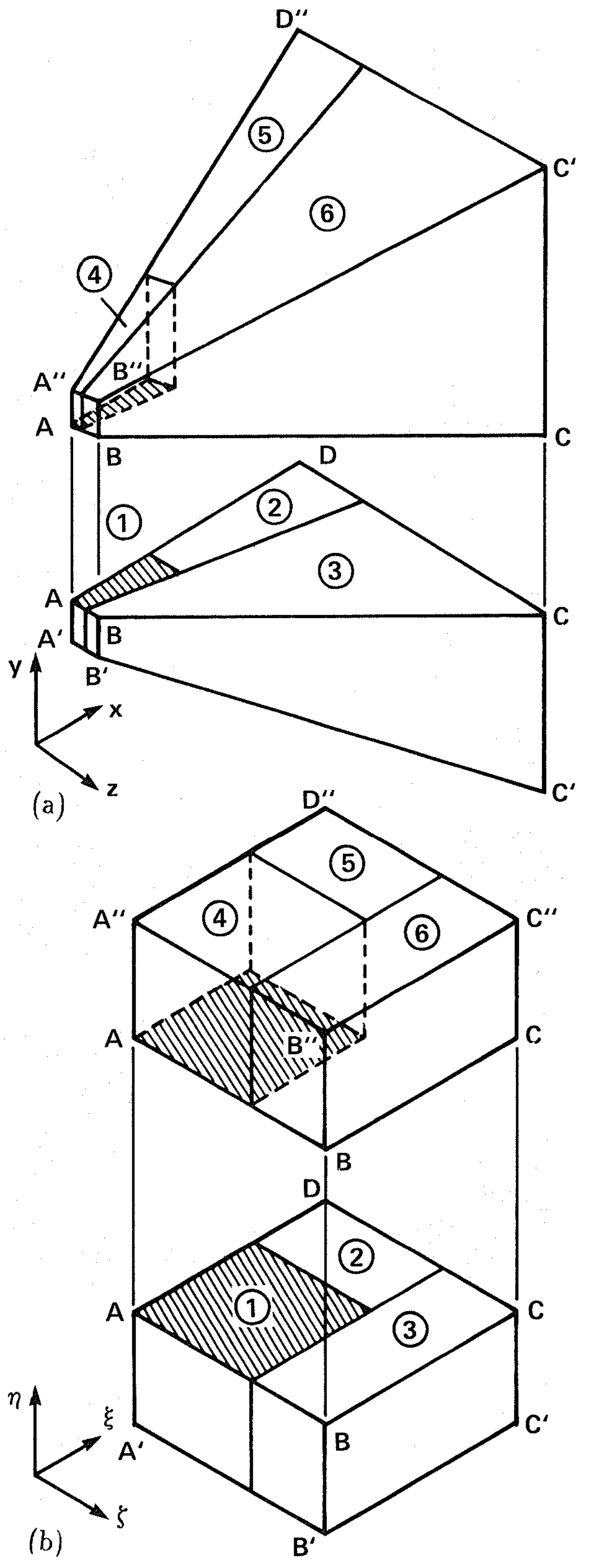

Fig. 4. Delta wing block structure: (a) Physical space, (b) Computational space.

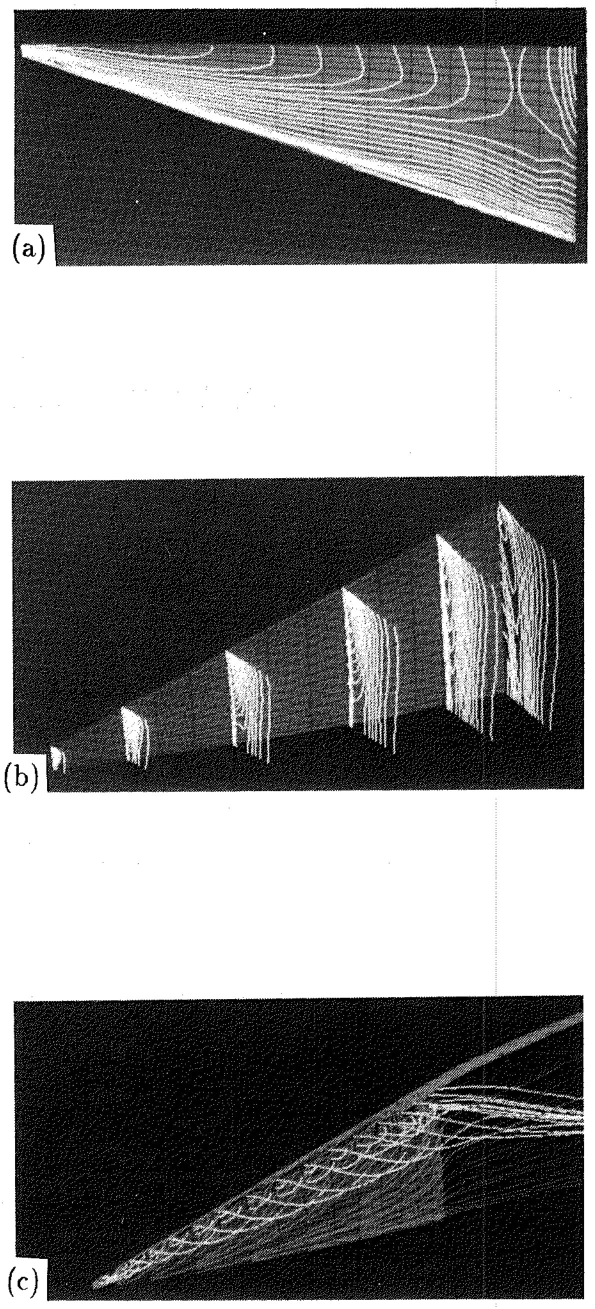

Fig. 5. Computed delta wing flowfield: (a) Lee surface isobars, (b) Leeward isopycnics, (c) Leading edge vortex and surface stream-lines. $M_\infty = 1.5$, $\alpha = 15^\circ$.

1. Report No. NASA TM-86773	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle SIMULATION OF COMPLEX THREE-DIMENSIONAL FLOWS		5. Report Date July 1985
		6. Performing Organization Code RFT
7. Author(s) George S. Deiwert, Herbert J. Rothmund (ETA Systems, Inc., St. Paul, Minn.), and Kazuhiro Nakahashi		8. Performing Organization Report No. 85338
9. Performing Organization Name and Address Ames Research Center Moffett Field, CA 94035		10. Work Unit No. T6465
		11. Contract or Grant No.
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, DC 20546		13. Type of Report and Period Covered Technical Memorandum
		14. Sponsoring Agency Code 505-3101

15. Supplementary Notes
Point of Contact: George S. Deiwert, Ames Research Center, M/S 202A-1, Moffett Field, CA 94035, (415) 694-5894 or FTS 464-5894.

16. Abstract

The concept of splitting is used extensively to simulate complex three-dimensional flows on modern computer architectures. Used in all aspects, from initial grid generation to the determination of the final converged solution, splitting is used to enhance code vectorization, to permit solution-driven grid adaption and grid enrichment, to permit the use of concurrent processing, and to enhance data flow through hierarchal memory systems. Three examples are used to illustrate these concepts to complex three-dimensional flow fields: 1) interactive flow over a bump, 2) supersonic flow past a blunt-based conical afterbody at incidence to a free stream and containing a centered propulsive jet, and 3) supersonic flow past a sharp-leading-edge delta wing at incidence to the free stream.

17. Key Words (Suggested by Author(s)) Vector processors Data structures Grid adaptation Fluid flow		18. Distribution Statement Unlimited Subject Category - 64	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 14	22. Price* A02

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

End of Document