Block-Structured Solution of Transonic Flows

A. ECER*

Abstract. A computational scheme for the block-structured solution of transonic flows was developed. A complex, three-dimensional flow problem is divided into a number of smaller sub-regions (blocks). The computational grid is generated for each of the blocks and assembled into a single grid. The flow equations are again solved individually for each block and iteratively combined to provide a global solution. Potential, Euler and Navier-Stokes equations can be solved for each of the blocks. The paper summarizes the procedure and discusses its applications on different computers.

Introduction. A computational scheme for the block-structured solution of transonic flows was developed based on the assumption that the solution of complex flow problems requires treatment of the entire problem in terms of a number of smaller sub-regions (blocks). The solution of such a problem includes grid generation, numerical solution and display of the results. It also usually involves an iterative process where the grid is modified several times and the solution scheme is performed through several steps where the results from an initial set of runs are utilized to start the subsequent set of runs. The block-structured solution scheme was developed to provide a computational environment where all of these tasks can be performed efficiently for complex flows. The user controls the solution procedure by describing a particular flow regions and later modifying them locally while these changes are automatically propogated to the remaining flow regions.

In the present paper, the basic steps of the block-structured solution generation scheme are reviewed. The basic consideration in designing such a scheme are listed. The implementation of the scheme on existing computer systems are reviewed.

*Purdue University, School of Engineering and Technology, Indianapolis, Indiana, 46223, U.S.A.

Block-Structured Solution Scheme. The block-structured solution of three-dimensional transonic flows include the following steps:

. Design of a Block-Structure,
. Generation of a Computational Grid,
. Solution of the Equations,
. Interpretation of Results,
. Modification of the Computational Grid,
. Solution of the Revised Grid,
. Interpretation of Results and Comparisons.

The above scheme provides a realistic summary of the steps involved in solving three-dimensional problems. In most cases, the revisions of the grid are performed more than once. It becomes very important for the user to be able to inspect the results in detail and implement the necessary modifications with sufficient control and ease. In the following, the description of these steps and related considerations are summarized.

Design of the Block-Structure. In terms of the solution of a three-dimensional problem, this step is the fundamental one. The user decides to divide the problem into a series of blocks. In most general terms, the blocks are chosen to identify flow structures in a complex flow field. For a simple example, such as the two-dimensional transonic flow around an airfoil, one may decide to identify the flow structures around the leading and trailing edges of the airfoil as well as around the shocks. In some cases, these blocks may be further sub-divided providing the user the capability to control the generation and modifications of the computational grid in detail.

Block-Structured Grid Generation Scheme. The finite element grid generation schemes have been traditionally based on the definition of a series of blocks. Many commercial finite element grid generation packages [1,2] employ the definition of a series of blocks as a basis of finite element grid generation process. Several years ago we have further developed some of these concepts in a form which is more suitable for analyzing complex, three-dimensional flow problems. A block-structured grid generation scheme was developed where the user specifies a block-structure as a first step before the grid generation process [3]. The convectivity of the blocks are defined as a first step. The developed procedure can model any irregular block-structures. One can define blocks which have four, five or six surfaces. The neighboring blocks can be attached, separated or coupled. Also, void blocks can be placed to model the irregularities in the block structure as necessary.

Once the block-structure is defined, the rest of the operations are performed on a block-by-block basis. A computational grid is assigned for each block. The grid can be irregular inside each block. For each flow region, an appropriate grid is designed. For example, around the leading edge of an airfoil a radial grid will provide most efficient and accurate results. The grid inside each block is defined in terms of the number of grid points on each surface and along the edges defining boundary surfaces. As one refines the grid inside each block, the number of grid points on each surface and edge increases. The scheme automatically keeps a record of these values and provide the user with the details of these adjustments. If, for example, the user would like to reduce the level of the grid refinement away from the airfoil, special blocks are employed with irregular grid structures. Our strategy has been to utilize irregular (unstructured) grids away from critical regions and try to adopt a structured grid at critical flow regions. In generating structured grids inside each block, one may have to use four-noded to eight-noded, three-dimensional finite elements as necessary. Figure 1 illustrates the typical steps involved in the block structured grid generation scheme.

The use of a block-structured grid generation scheme enables the user to design appropriate computational grids without restrictions of structural grids. Once the grid is generated the solution scheme is again defined on a block-by-block basis.

Block-Structured Solution of Transonic Flows. A block-structured solution scheme for transonic flows was developed based on the following assumptions:

. A relaxation process is defined for each block,
. For each block, either potential, Euler or Navier-Stokes equations can be solved,
. Nodes between the neighboring blocks are matched exactly,
. The variables between the neighboring blocks are matched by using Dirichlet boundary conditions.

In this paper, only a short summary of the solution scheme is presented [4,5].

The Navier-Stokes equations for steady, viscous, compressible flows can be written by using a Clebsch transformation of the velocity vector,

$$\underline{u} = \underline{\nabla}\phi + S\underline{\nabla}\eta + \underline{\tau} \qquad (1)$$

into the following form:

$$\underline{\nabla}.(\rho\underline{u}) = 0 \qquad (2)$$

$$\underline{\nabla}.(\rho\underline{u}S) = \Phi \qquad (3)$$

$$\rho\underline{u}.\underline{\nabla}\eta = - p/R \qquad (4)$$

where $\underline{u}$ is the velocity vector, S is the entropy, ρ is the density, p is the pressure, R is the gas constant, Φ is the viscous dissipation function and ϕ and η are the Lagrangian multiplier associated with conservation of mass and entropy equations respectively [6]. $\underline{\tau}$ is a generalized shear stress tensor which can be written in terms of the Lagrange multiplier η and the rate of strain tensor e_{ij} as follows:

$$\tau = 4\mu\eta[e_{ij} - 2\mu_i] \qquad (5)$$

Equations (3-4) are cast into a second-order form by premultiplying with the convection operator. The relaxation scheme can then be summarized as follows:

. assume a velocity field
. solve equations (3) and (4) to obtain η and S distribution,
. calculate the shear stresses, $\underline{\tau}$,
. solve the conservation of mass to calculate ϕ,

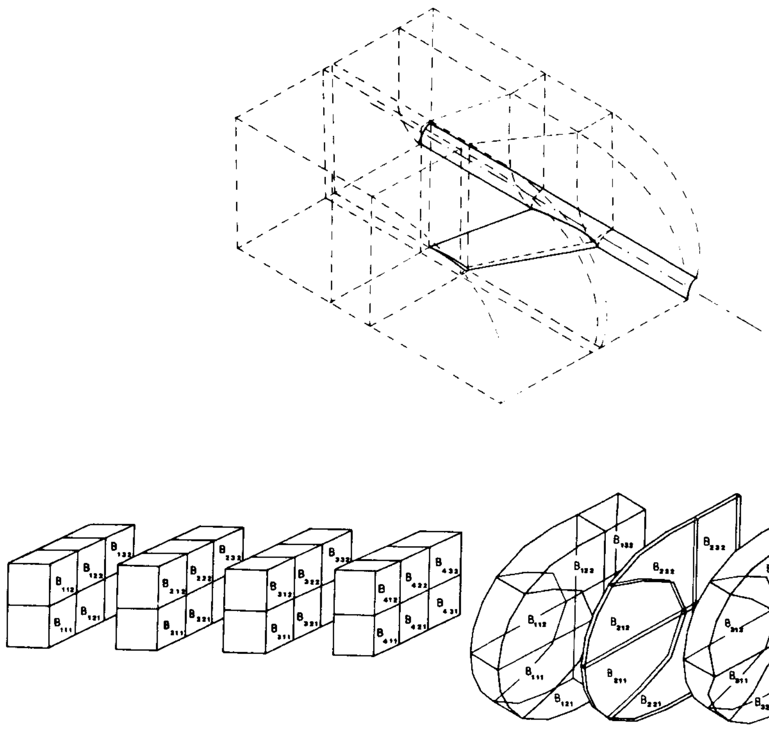

FIG. 1a. Block-Structure for the Wing-Body Problem.

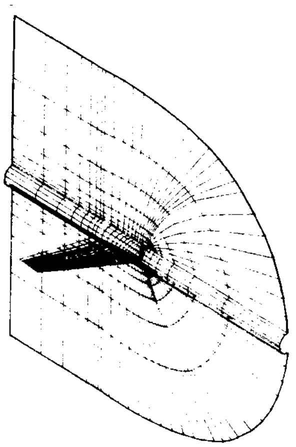

FIG. 1b. Grid Distribution over the Wing-Body.

. calculate a new velocity field and repeat the procedure.

In terms of a block-structured solution scheme one has to perform the above generations for each block individually. The scheme then becomes a block relaxation scheme. A block relaxation scheme was defined where the operations are performed on a block basis rather than in terms of inter-block surfaces or edges. Each block is solved individually during each step of the iterations. The resulting block-surface solutions are compared along the neighboring block interfaces. Corrections are provided from each block surface which are employed to solve the individual block one more time. Before designing the details of the block-structured relaxation scheme, it is important to mention that one does not have to solve the same equations for each block. As stated before, each block represents a particular flow region. If one decides to solve Navier-Stokes equations for a particular block, all three equations (Eqs. 2-5) are solved for the primary variables ϕ, S and η. If Euler equations are required then the same set of equations are solved by ignoring the viscous dissipation term in equation (3) and the shear stresses in equation (1). No slip boundary condition is not required in this case since it only enters the formulation during the formulation of shear stresses in equation (5). For potential flows, only the velocity potential is required from the solution of conservation of mass equation in equation (2). Between neighboring blocks, ϕ, S and η values have to be matched. For blocks with potential flows S and η are known and need not to be calculated. Similarly, for blocks with Euler equations, the shear stresses are assumed to be zero and are not calculated. Thus, the formulation is general for all three types of flows.

Block-Structured Relaxation Scheme. The block-structured relaxation scheme is applied to two sets of equations: conservation of mass equation and the transport equation for S and η. The basic procedure for this scheme is summarized below [5,6].

Conservation of Mass Equation. For the case of the conservation of mass equation where:

$$\underline{\nabla}.(\rho\underline{\nabla}\phi) = \underline{g}(S,\eta) \tag{6}$$

one considers the solution of the equation for each block. The relaxation scheme for each block can be written as follows:

$$\underline{A}^{o}\Delta\Phi^{n} = \underline{g} - \underline{A}^{n}\underline{\Phi} \tag{7}$$

$$\underline{\phi}^{n+1} = \underline{\phi}^{n} + \omega\Delta\underline{\phi}^{n} \tag{8}$$

where ω is the relaxation parameter and $\underline{A}^{o}$ represents the Laplace operator for each block calculated for incompressible flow conditions. This symmetric matrix is decomposed and stored to be used at each iteration step.

The problem is solved at two steps:

. A distribution of normal velocities is assumed for all block interfaces.

. Each block is individually solved by using the relationship in equation (6) with Neumann type boundary conditions.

. The values of Φ at interblock surfaces are compared and aver-averaged for each surface.

. Each block is re-analyzed using this time Dirichlet type boundary conditions.

. The boundary fluxes on each surface are calculated and aver-averaged.

. A new velocity field is calculated and the iteration cycle is repeated.

The above solution scheme is robust and quite efficient for transonic flows [4]. Generally, the block in which the shock is located determines the rate of convergence. In most cases, the number of iterations required for convergence is similar to the ones obtained by solving the same problem as a single block.

Transport Equations for S and η. The solution of the two transport equations include the second order form of two equations again with symmetric operations. One can summarize the problem as follows:

Consider a transport problem with:

$$\frac{\partial}{\partial s}(gS) = \Phi \tag{9}$$

where S is the streamline direction and g is the flow speed. In this case, one has to define the entropy at the inlet as a reference value since only the changes in entropy rather than the absolute value of entropy will change the flow field.

We transfer the operator into a second order form by premultiplying with the convection operator as follows:

$$\frac{\partial}{\partial s}\left[g \frac{\partial}{\partial s}(gS)\right] = \frac{\partial}{\partial s}[g\Phi] \tag{10}$$

The original differential equation (9) becomes the additional boundary condition to be imposed at the exit boundary for this second-order differential equation. The new differential equation requires either the Dirichlet or Neumann type of boundary conditions depending whether it is a boundary point at the upstream or downstream of a streamline.

The block-structured solution of this differential equation is rather different than the first one. Dirichlet type boundary conditions are specified on the surfaces where the flow enters the block. At other surfaces of the elements for which the flow leaves the block, only the Neumann type of boundary conditions are required. In this case, no averaging between the neighboring blocks is required. Instead, the information regarding the entropy S and η is convected in the flow direction between the neighboring blocks. These equations are solved only once during each iteration step. Figure 2 illustrates the block-structured solution of Euler equations around an airfoil.

Parallel Processing of Blocks. The use of block-structured solution technique allows the parallel processing of several blocks at the same time. By performing the computations on several blocks in parallel, one

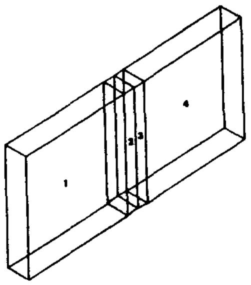

FIG. 2a. Four-Block Structure.

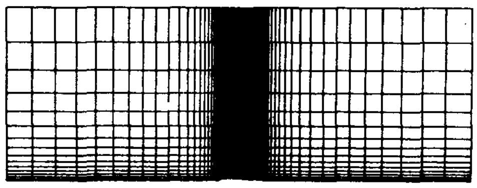

FIG. 2b. Finite Element Grid.

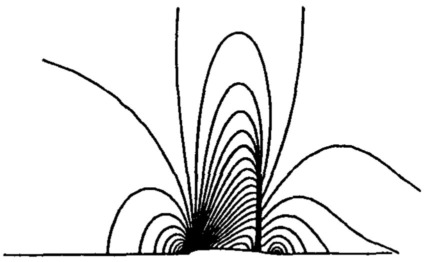

FIG. 2c. Mach Contours

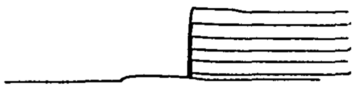

FIG. 2d. Contours of Vorticity/Pressure.

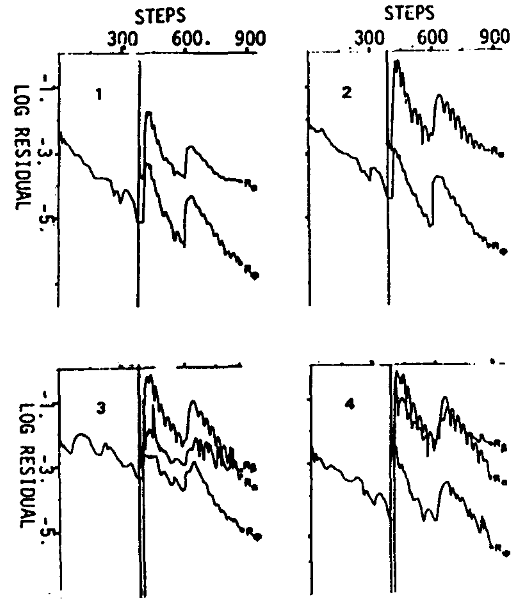

FIG. 2e. Residual History for the Euler Solution around a NACA0012 Airfoil at $M_\infty = 0.85$ for Different Blocks.

aims at utilizing all of the available computer resources. Important characteristics of the developed scheme, in terms of parallel processing, can be summarized as follows:

. By performing a block relaxation scheme on each of the individual blocks one decomposes two sets of banded coefficient matrices: representing the Laplace operator and the convection operator in the second-order form. Both of these operators are symmetric and are stored considering the banded form.

. When bandwidths are small for each block (≤ 80) and only a forward elimination-backward substitution is required for each block, the computation of the residual vector becomes the critical factor. In this case, all of the element information related to the geometry, such as the shape functions and their derivatives, Jacobians, etc., are calculated and stored once. The cost of computing the residuals and solving the equations then become comparable when the solution part is vectorized.

. The main computational concern then becomes the reduction of I/O cost such that computations still remain on the critical path i.e. no I/O wait is encountered. The scheme was implemented on two computers CRAY-XMP and IBM 3090. On both machines, the memory is divided into parts where two or three blocks can be stored at the same time. In this case, while one block is being processed, another block can be read-in or written-out. Based on the fact that the information related to a single block can be read-in by a single read command, asynchronous I/O operations do not slow down the processors. The critical requirement for a given computer becomes the size of the memory which is available to store two or three blocks at the same time, as well as the number and speed of I/O devices. The use of high speed I/O devices improves the speed of the present process considerably. The main memory of the computer limits the maximum size of the block which can be processed. For blocks with specified wavelengths, the computational cost is linearly proportional to the number of nodes and the number of blocks.

. Current studies with Intel IPSC computer provides an experiment when several processors with many I/O channels and large memories are available. In this case, rather than storing all of the blocks on the disk, one keeps each block in the memory of an individual processor. Each block is coupled to several neighboring processors, each containing the neighboring block. In this case, all geometry data and coefficient matrices are calculated and stored in real memory. The only I/O operation involves the transfer of solution vectors relating the inter-block surfaces. These are balanced between neighboring blocks and the iterations continue for each block on a single processor. This type of approach requires more loosely coupled processors. For a given block-structure of different size blocks, one may like to arrange processors with different size memories to fit the topology of the block-structure.

Refinement of Grids. As indicated at the beginning of this paper, the analysis of a complex flow problem can be considered as an iterative process rather than a one step solution. The block-structured solution scheme provides a tool for performing such a task. We expect the user:

. to identify the block-structure,

. to generate a grid,

. to identify potential, Euler or Navier-Stokes blocks, and
. to solve the problem approximately.

Once this initial step is completed, one may:

. modify the grid in one block,
. decide to change from one of the above three formulations (potential, Euler, Navier-Stokes) to another,
. define turbulence models for viscous flow regions,
. provide special modeling of critical regions such as shocks, trailing edges, thin boundary layers, etc.

Once these modifications are made for each block, they are automatically propogated throughout the flow field. The flow field obtained from the first step is utilized as an initial step for the second stage of the computations. It is also expected that information obtained for the analysis of particular flow structures can be employed to design better blocks for the subsequent analyses. Adaptive grid techniques again seem much more practical when considered at a block level. One can provide error estimates for a specific flow structure and provide a specific strategy for adopting the grid.

Conclusions. In this paper, the advantages of a block-structured solution technique for solving large aerodynamics problems was discussed. As the complexity of the problems increase for three-dimensional problems, the cost of preparing good computational grids increases together with the time spent for actually solving the problem. One has to utilize a block-structured grid generation and solution scheme for solving such problems, as discussed in this paper. The computer hardware available for solving large problems are also limited at the present time. The developed scheme allows the use of secondary storage to eliminate the restrictions of main memory, while, masking the cost of performing related I/O operations. The use of a block-structured solution scheme allows the use of existing computer hardware as well as parallel processors which are under development for solving large problems.

REFERENCES

(1) SDRC Inc., SUPERTAB User's Manual, 1983.

(2) PDA Inc., PATRAN-G User's Manual, 1984.

(3) A. ECER, J.T. SPYROPOULOS and J. MAUL, A Block-Structured Finite Element Grid Generation Scheme for the Analysis of Three-Dimensional Transonic Flows, 23 (1985), pp. 1483-1490.

(4) A. ECER and J.T. SPYROPOULOS, Block-Structured Solution Scheme for Analyzing Three-Dimensional Transonic Potential Flows, AIAA-86-0510, (1986).

(5) A. ECER , J.T. SPYROPOULOS and V. RUBEK, Block-Structured Solution for Euler Equations for Transonic Flows, AIAA-86-1080, (1986).

(6) A. ECER, J.T. SPYROPOULOS and O. ATAKAR, Block-Structured Solution of Euler Equations for Transonic Flows, AIAA-87-0351, (1987).