# Block-Structured Solution Scheme for Analyzing Three-Dimensional Transonic Potential Flows

Akin Ecer* and John T. Spyropoulos†
*Purdue University at Indianapolis, Indianapolis, Indiana*

**A block-structured solution scheme is developed for the analysis of three-dimensional transonic flows. The scheme is based on the solution of potential flow equations for individual blocks representing subvolumes of the flowfield. Based on a block-structured grid generation scheme, appropriate computational grids are generated for each of the blocks depending on the complexity of the local flowfield. The iterative scheme is designed to provide a solution of a large problem in terms of an assembly of smaller problems for each block. Numerical results illustrate the applicability of the method for analyzing flow around a NACA-0012 profile. Different block-structures are analyzed to demonstrate the robustness and the accuracy of the developed method. Finally, a three-dimensional wing-body configuration is analyzed and the results are compared with those obtained through experiments.**

## I. Introduction

THE solution of three-dimensional, transonic flows around complex aircraft configurations requires considerable computational effort. The capability for solving such large problems relies heavily on the availability of larger and faster computers. Over the last 20 years, considerable progress has been made in terms of hardware available for solving computational fluid dynamics problems.[1,2] While the use of vector processing has become much more popular during the last decade, parallel processing is also becoming more available to researchers. For the next 10 years, the major emphasis in hardware development is expected to be in terms of availability of multiprocessing capabilities and larger memories, as well as some increase in computational speeds.

In developing computational procedures for solving large problems, evaluating the efficiency of a scheme is a difficult process since it is very closely related to the employed computer hardware. It also strongly depends on the computational grid structure employed in the analysis. For example, it is possible to define a computational fluid dynamics problem on a "box"-type grid structure with $M \times N \times K$ grid points in each direction and choose appropriate hardware which can solve the particular problem in central memory. The determination of the efficiency of a numerical scheme under such idealized conditions is rather straightforward. However, when the grid structure becomes irregular and when the size of the problem becomes large (in comparison with what is available with current computers) cost and efficiency comparisons become more complicated.

If a single block with a regular grid structure, as mentioned above, could be fitted around an aircraft geometry, the number of grid points required to model the complete flowfield can grow very rapidly. Attempts are being made to improve this situation by embedding blocks with several levels of grid refinement inside a regular but coarser grid structure.[3] The strategy then involves 1) the utilization of computational schemes which are fast on regular grids, and 2) the modeling of irregular geometries using several blocks with regular "box"-type grids. Once such a system is in place, a second problem to be considered is the number of grid points in a single block which can be fitted in a particular vector machine and the data transfer between the neighboring blocks.

Received Jan 24, 1986; revision submitted Oct. 6, 1987. 

The procedure presented in this paper is based on the finite element method. This method provides an alternate approach than the one described above for the solution of the same problem. It enables the use of irregular grids which can be fitted to simulate irregular flowfields around complex geometries.[4] It also provides the flexibility for modifying an existing grid to provide a better approximation basis locally for the employed numerical scheme. The applications of irregular grids require development of numerical schemes where no regularity of the nodal connectivities is assumed. By using such numerical schemes, one can exploit the advantages of irregular, yet more efficient grids.[5] Also, for irregular grids, rather than employing point or line relaxation schemes, one has to employ block-relaxation schemes, since no regularity of the grid is assumed. Such schemes produce desirable convergence rates and are not affected by the occurrence of irregular grid structures. Two basic constraints for solving large problems then become, 1) the size of the block-relaxation operator as it grows proportional to the size of the block, and 2) the geometric definition of an irregular grid which requires much more data than a regular grid.

Based on the above considerations, a block-by-block solution scheme is developed. The main objective of this approach is to divide a large problem into smaller components in terms of a series of individual blocks in a general manner. Rather than attempting to solve a large problem most efficiently on a single processor, one can divide the problem into smaller ones and try to develop an "intelligent" strategy which is more suitable for parallel processing. Parallel processing for large systems is a popular subject addressed by many researchers today.[6] The main objective here is to exploit the physical characteristics of the problem, both for grid generation and numerical solution, where each block corresponds to a subvolume in flow space. A "substructuring" scheme is developed where each of the "substructures" corresponds to a particular flow region. One can then design grids, use efficient solution schemes for each of the blocks, depending on the characteristics of the flowfield, and allocate computer resources in a parallel processing environment. The computational strategy can be summarized along the following lines:

1) Development of a block-structured grid generation scheme.[4]

2) Development of a unified formulation for solving potential, Euler, and Navier-Stokes equations.[7]

3) Development of a block-structured solution scheme for solving the steady-state flow equations through a relaxation scheme.

In this paper, the basic features of the block-structured solution scheme and its application to only three-dimensional transonic flows are presented. The computational strategy is designed to provide the capability for generating grids with varying degrees of refinement for each block and solving potential, Euler or Navier-Stokes equations for each block, depending on the characteristics of the flowfield. An accurate, efficient, yet general implementation of the boundary conditions between neighboring blocks under such conditions is extremely important in a block-structured solution scheme. In this paper, the boundary conditions for the conservation of mass equations are discussed in detail for the developed scheme.

## II. A Block-Structured Solution Scheme for Potential Flows

### A. Potential Flow Problem

The analysis of compressible, irrotational flows requires only the solution of the conservation of mass equation,

$$\nabla(\rho \boldsymbol{u}) = 0 \qquad \text{in } \Omega \tag{1}$$

where $\boldsymbol{u}$ is the velocity vector, $\rho$ is the density, and $\Omega$ is the flow region to be analyzed. The density is a function of the local velocity which can be written as follows:

$$\rho = C(K^2 - \boldsymbol{u}\cdot\boldsymbol{u})^{1/\gamma-1} \tag{2}$$

In the above equation, $C$, $K$, and $\gamma$ are known constants. The boundary conditions require the specification of the normal mass flux on the boundary surface $\Gamma$.

$$\rho \boldsymbol{u}\cdot\boldsymbol{n} = f \qquad \text{on } \Gamma \tag{3}$$

where $f$ is a known function specified on the boundary and $\boldsymbol{n}$ is the unit normal vector. By using the condition for irrotationality, one can substitute the velocity potential to eliminate the velocity vector as follows:

$$\boldsymbol{u} = \nabla\phi \tag{4}$$

The conservation of mass equation then becomes second-order,

$$\nabla\cdot(\rho\nabla\Phi) = 0 \qquad \text{in } \Omega \tag{5}$$

where the velocity potential has to be assigned an arbitrary value at one point to remove the singularity due to the introduction of Eq. (4). Since the conservation of mass equation has to be satisfied for the entire flow domain over a closed boundary, i.e.,

$$\oint_\Gamma \rho \boldsymbol{u}\cdot\boldsymbol{n}\, d\Gamma = 0 \tag{6}$$

the specified value of $\Phi$ can be arbitrary.

### B. Transonic Potential Flows

In the case of transonic potential flows the above equations are modified by upwinding the density as follows:

$$\tilde{\rho} = \rho + \alpha\frac{\partial\rho}{\partial s} \qquad \alpha = \mu\left(1 - \frac{1}{M^2}\right) \tag{7}$$

where $\partial\rho/\partial s$ is the gradient of the density in the flow direction and $\alpha$ is a coefficient. Iterative solution of the transonic potential equation in the following form:

$$\nabla\cdot(\tilde{\rho}\nabla\phi) = 0 \qquad \text{in } \Omega \tag{8}$$

provides stable solutions to the transonic potential equation (8). In this paper, the same procedure is applied to the block-structured solution of the equations, where upwinding is performed on an element basis for each block individually. Otherwise, the procedure described below is the same for subsonic and transonic flows.

### C. Block-Structured Formulation of the Problem

For developing a block-structured solution scheme, the problem is defined over a series of flow regions which will be called blocks in the following form:

$$\nabla\cdot(\rho\nabla\Phi_i) = 0 \qquad \text{in } \Omega_i$$
$$\rho\boldsymbol{n}\cdot\nabla\Phi_i = f_i \qquad \text{on } \Gamma_i \tag{9}$$

where $\Omega_i$ and $\Gamma_i$ are the block volume and block surface for each block, where $i$ indicates the block number. The block boundaries can be further classified in two parts: between the neighboring blocks and around the global boundary of the entire flow region. The corresponding boundary conditions can be distinguished as follows:

$$\rho\boldsymbol{n}\cdot\nabla\Phi_i = f_i \qquad \text{on } \Gamma_i^0 \tag{10}$$

$$\rho\boldsymbol{n}\cdot\nabla\Phi_i = g_i \qquad \text{on } \Gamma_i^c \tag{11}$$

where $\Gamma_i^0$ is the global boundary and $\Gamma_i^c$ is the interblock boundary for the $i$th block. Of course, some blocks may have no connection to global boundaries.

The problem can then be defined as the solution of conservation of mass equation (7) for each block together with the determination of unknown interblock boundary fluxes $g_i$. This can be achieved through an additional constraint which specifies that the velocity potentials are continuous across the interblock boundaries. We can define the corresponding variational problem in the following form:

$$\delta\Pi = \sum_i \int_{V_i} \rho\nabla\Phi_i\cdot\nabla\delta\Phi_i\, dV + \int_{\Gamma_i^0} f_i\delta\Phi_i\, d\Gamma + \int_{\Gamma_i^c} g_i\delta\Phi_i\, d\Gamma = 0 \tag{12}$$

By taking variations with respect to $\Phi$, we can derive the differential equation in Eq. (9) and the boundary conditions in Eqs. (10) and (11). Also, since we assumed that $g_i$ is an unknown, we can take a variation with respect to $g_i$. This produces the additional constraint equation in terms of the compatibility of velocity potentials across neighboring block boundaries as follows:

$$\sum_i \int_{\Gamma_i^c} \Phi_i\delta g_i\, d\Gamma = 0 \tag{13}$$

Since the same $g_i$ is employed for neighboring blocks, this constraint will involve the velocity potentials which correspond to the grid points on such block pairs as follows:

$$\Phi_{i,j} - \Phi_{n,j} = 0 \tag{14}$$

where $i$ and $n$ are the two neighboring blocks and $j$ is the number for the boundary surface between these two blocks.

At this point, one can introduce the finite-element formulation, where the flow region is divided into first a series of blocks and then a group of finite elements. The distribution of velocity potential over each element for block number $i$ is approximated by:

$$\Phi_i(x, y, z) = N_k(x, y, z)\Phi_{ik} \tag{15}$$

where $N_k$ is called a shape function (a simple polynomial) and $\Phi_{ik}$ is the nodal value of the velocity potential at node $k$. One

can also approximate the unknown boundary fluxes along the block interfaces in terms of nodal flux values.

After substituting the finite-element approximations and taking variation with respect to nodal with respect to nodal values of velocity potentials, one can write the following set of discrete equations:

$$A_i \Phi_i + C^t_{i,j} \lambda_j = F_i \tag{16}$$

where $A$, $F$, $C$, $\Phi$, and $\lambda$ are global matrices for each block $i$ defined by assembling element matrices for each element $e$

$$A_{i,e} = \int_{V_{i,e}} \rho \left( N_{,x} N^t_{,x} + N_{,y} N^t_{,y} + N_{,z} N^t_{,z} \right) dV \tag{17}$$

$$F_{i,e} = \int_{\Gamma^0_{i,e}} \rho_e f_e N \, d\Gamma \tag{18}$$

$$C_{i,j,e} = -\int_{\Gamma^c_{i,e}} \rho_e \tilde{N} N^t \, d\Gamma \tag{19}$$

$$\lambda_j = [\lambda_k]_j \tag{20}$$

Here $i$ indicates the number of the block, $j$ the number of the surface, $k$ the node number, and $e$ the element number. For $\rho > 0$, the coefficient matrix in Eq. (17) is symmetric and positive-definite.

The variation with respect to nodal values of boundary fluxes ($\lambda_k$) produces a set of constraint equations defining the compatibility of velocity potentials between the blocks:

$$\sum_i C_{i,j} \Phi_i = 0 \tag{21}$$

As can be seen from Eq. (21), matrix $C_{i,j}$ picks up all the nodes on surface $j$ located on the block $i$ and assembles the constraint conditions. In coupled form, Eqs. (16) and (21) can be written as follows:

$$\begin{bmatrix} [A_i] & [C^t_{i,j}] \\ [C_{i,j}] & [0] \end{bmatrix} \begin{bmatrix} [\Phi_i] \\ [\lambda_j] \end{bmatrix} = \begin{bmatrix} [F_i] \\ [0] \end{bmatrix} \tag{22}$$

For a block-structure shown in Fig. 1, Eq. (22) can be written in the form

$$\begin{bmatrix} A_1 & 0 & 0 & C^t_{1,1} & 0 \\ 0 & A_2 & 0 & C^t_{2,1} & C^t_{2,2} \\ 0 & 0 & A_3 & 0 & C^t_{3,2} \\ C_{1,1} & C_{2,1} & 0 & 0 & 0 \\ 0 & C_{2,2} & C_{3,2} & 0 & 0 \end{bmatrix} \begin{bmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \\ \lambda_1 \\ \lambda_2 \end{bmatrix} = \begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ 0 \\ 0 \end{bmatrix} \tag{23}$$

Here, there are three blocks and two interconnecting surfaces. In the above equation, the coupling matrices for each row of $\lambda$ equations represent the contributions to a surface from the nodes located on the two neighboring blocks. As indicated above, unknowns are the nodal velocity potentials $\Phi_k$ and the nodal boundary fluxes $\lambda_k$. The potential vectors for each block are ($\Phi_i$, $i = 1,3$) and the vectors for each surface are ($\lambda_j$, $j = 1,2$).

### D. Basic Considerations for the Development of the Iterative Solution Scheme

One can, of course, attempt to solve the coupled equations directly in the form of Eq. (22). Matrices $A_i$ correspond to individual coefficient matrices for each block. Most of the computational effort in solving such a system will involve the treatment of the coupling matrices between the blocks. In the developed procedure, coupled equations are solved by using an iterative procedure. For the description of this iterative procedure, first consider a simpler system consisting of two blocks and a single coupling surface. In this case, the equations can be written as

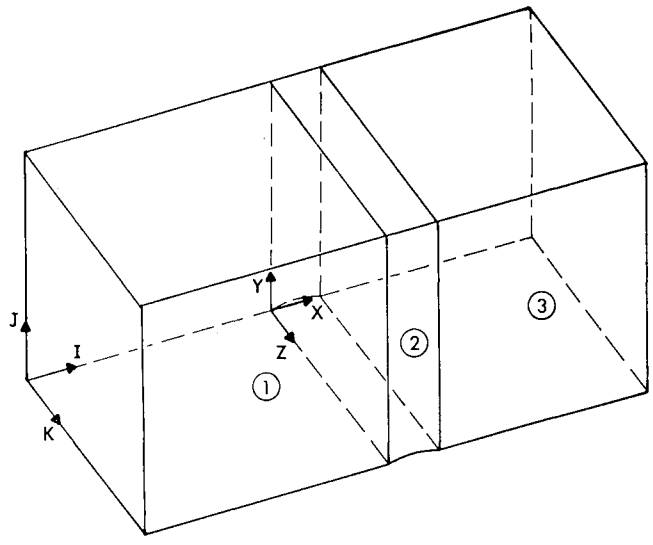


**Fig. 1 Block-structure for an airfoil problem with three blocks.**

$$\begin{bmatrix} A_1 & 0 & C^t_{1,1} \\ 0 & A_2 & C^t_{2,1} \\ C_{1,1} & C_{2,1} & 0 \end{bmatrix} \begin{bmatrix} \Phi_1 \\ \Phi_2 \\ \lambda_1 \end{bmatrix} = \begin{bmatrix} F_1 \\ F_2 \\ 0 \end{bmatrix} \tag{24}$$

Here, $A_1$ and $A_2$ represent the Laplace operator for two blocks. $C_{1,1}$ and $C_{2,1}$ are matrices for defining the coupling between the two blocks. The problem described by Eq. (24) can be restated as a discrete variational problem in the following form:

$$\tilde{\Pi} = \tfrac{1}{2} \Phi^t_1 A_1 \Phi_1 + \tfrac{1}{2} \Phi^t_2 A_2 \Phi_2 + \lambda_1 \left( C^t_{1,1} \Phi_1 + C^t_{2,1} \Phi_2 \right) - \Phi^t_1 F_1 - \Phi^t_2 F_2 \tag{25}$$

It can be shown that the variation of the above functional with respect to $\Phi_1$, $\Phi_2$, and $\lambda$ produces the above set of equations.

At this point, one can define a new variational problem as follows: Instead of the unknown flux vector $\lambda$, by assuming a known flux vector $\lambda^*$, the following variational problem is created:

$$\tilde{\Pi}^* = \tfrac{1}{2} \Phi^{*t}_1 A_1 \Phi^*_1 + \tfrac{1}{2} \Phi^{*t}_2 A_2 \Phi^*_2 + \lambda^{*t}_1 \left( C_{1,1} \Phi^*_1 + C_{2,1} \Phi^*_2 \right) - \Phi^{*t}_1 F_1 - \Phi^{*t}_2 F_2 \tag{26}$$

Taking variation with respect to $\Phi^*_1$ and $\Phi^*_2$, one can write:

$$A_1 \Phi^*_1 + C^t_{1,1} \lambda^*_1 = F_1 \qquad A_2 \Phi^*_2 + C^t_{2,1} \lambda^*_1 = F_2 \tag{27}$$

We then prove a certain theorem, which relates the solution of the variational problem in Eq. (26). (The proof of these theorems are presented in Ref. 8.)

*Theorem:* If $\lambda^*$ is any arbitrary flux distribution on a boundary surface, the solution of Eq. (27) produces upper and lower bounds of the vector $\Phi$ on the same boundary surface, to the solution of the original problem of Eq. (24).

A similar statement can be made for the inverse problem as follows:

*Theorem:* If $\Phi^*$ is any arbitrary velocity potential distribution on a boundary surface, the solution of Eq. (27) produces upper and lower bounds of the vector $\lambda$ on the same boundary surface, to the solution of the original problem of Eq. (24).

Based on the forementioned theorems, an iterative procedure was developed using the following principles:

1) Calculate $\Phi_1^*$ and $\Phi_2^*$ from Eq. (27) which provides bounds for the solution at the boundary ($C_{1,1}^l\Phi$ and $C_{2,1}^l\Phi$).

2) Calculate an estimate of $\Phi$ at the boundaries bases on $\Phi_1^*$ and $\Phi_2^*$.

3) Solve Eq. (24) by assuming the velocity potentials at the boundaries are specified.

4) Using Eq. (27), calculate $\lambda_{1,1}^*$ and $\lambda_{2,1}^*$ which provide bounds for the next estimate of $\lambda$.

5) Repeat the iteration procedure.

The preceding scheme (1–5) only demonstrates that at each iteration, a new set of upper and lower bounds can be obtained for boundary fluxes and boundary velocity potentials. The next step is to show that this iterative scheme is stable. Also it is necessary to understand the important factors effecting the rate of convergence.

Computationally, the efficiency of this scheme is based on the assumption that a large system can be divided into a series of blocks which can be individually stored in the main storage area of a computer. Blocks can be processed individually in this scheme while the calculation of the surface fluxes is done through a relaxation scheme for each surface without solving a large system of equations. These computational considerations will later be discussed in detail.

#### E. The Convergence Characteristics of the Iterative Scheme

To understand the convergence characteristics of the iterative scheme, again consider the simple two-block model.

1) Start the solution by assuming an initial flux vector at the boundary: $\lambda_1^0$. Calculate the potential vectors from Eq. (27).

2) Calculate the potential vectors on both sides of the boundary surface ($\Phi_{1,1}^*$, $\Phi_{2,1}^*$) and average these vectors on the boundary by using an averaging parameter $\alpha$.

$$\Phi_{B,1}^* = \alpha\Phi_{1,1}^* + (1-\alpha)\Phi_{2,1}^* \tag{28}$$

3) Calculate the boundary fluxes at the same boundary for neighboring blocks ($\lambda_{1,1}^*$, $\lambda_{2,1}^*$) by specifying potentials on the boundary surface as constraints.

4) Average the boundary fluxes at the boundaries with a second averaging parameter $\beta$.

$$\lambda_1^1 = \beta\lambda_{1,1}^* + (1-\beta)\lambda_{2,1}^* \tag{29}$$

By combining Eqs. (27)–(29), one can write the following recurrence relationship:

$$\lambda^{n+1} = \boldsymbol{C} + \boldsymbol{D}\lambda^n \tag{30}$$

where the coefficient matrix $\boldsymbol{D}$ can be written as

$$\boldsymbol{D} = \left[\beta\tilde{\boldsymbol{A}}_1 - (1-\beta)\tilde{\boldsymbol{A}}_2\right]\left[\alpha\tilde{\boldsymbol{A}}_1^{-1} - (1-\alpha)\tilde{\boldsymbol{A}}_2^{-1}\right] \tag{31}$$

Here, $\tilde{\boldsymbol{A}}_{1,1}$ and $\tilde{\boldsymbol{A}}_{2,1}$ are reduced coefficient matrices related to the boundary surface in the following form:

$$\tilde{\boldsymbol{A}}_1 = \boldsymbol{C}_{1,1}\boldsymbol{A}_1\boldsymbol{C}_{1,1}^l \qquad \tilde{\boldsymbol{A}}_2 = \boldsymbol{C}_{2,1}\boldsymbol{A}_2\boldsymbol{C}_{2,1}^l \tag{32}$$

The rate of convergence depends on the eigenvalues of the coefficient matrix $\boldsymbol{D}$, which can be simplified as

$$\boldsymbol{D} = (1-\alpha-\beta+2\alpha\beta)\boldsymbol{I} - \beta(1-\alpha)\tilde{\boldsymbol{A}}_1\tilde{\boldsymbol{A}}_2^{-1} - \alpha(1-\beta)\tilde{\boldsymbol{A}}_1\tilde{\boldsymbol{A}}_1^{-1} \tag{33}$$

where $\boldsymbol{I}$ is the identity matrix. In the case of two similar blocks, with $\boldsymbol{A}_1 = \boldsymbol{A}_2$, and $\alpha = \beta = 0.5$, an exact solution is obtained after one step.

It was shown in Ref. 8 that the stability of the iterative scheme depends on the relative size of the eigenvalues of the coefficient matrices of neighboring blocks. It was demonstrated that if the neighboring blocks are not similar in terms of shape, size, and grid refinement, one cannot expect convergence from the iterative scheme. Of course, in a general system, a series of eigenvalues are involved, which may be widely spread.

In order to improve the ratio of convergence, a relaxation parameter was introduced into the scheme in the following manner:

1) Once the estimates for velocity potentials are calculated and an average is computed in Eq. (28), the velocity potentials for each surface are relaxed in the following manner:

$$\begin{aligned}\tilde{\Phi}_{1,1} &= \omega\tilde{\Phi}_{B,1} + (1-\omega)\Phi_{1,1}^* \\ \tilde{\Phi}_{2,1} &= \omega\tilde{\Phi}_{B,2} + (1-\omega)\Phi_{2,1}^*\end{aligned} \tag{34}$$

2) An average for $\lambda$ vector is again calculated from Eq. (29). However, in this case, the $\lambda$ vector is relaxed again with the same relaxation factor to calculate the fluxes for each surface as follows:

$$\lambda_1^1 = \omega\lambda_1^1 + (1-\omega)\lambda_1^0 \tag{35}$$

In this scheme, $\omega$ becomes the controlling factor for providing stability. In the actual computations, as it will be discussed later, both $\alpha$ and $\beta$ were fixed at 0.5 and the stability was controlled by only changing $\omega$.

#### F. Convergence Characteristics of Transonic Flows

As discussed, the convergence characteristics of the equations were investigated by assuming a constant density for all the elements. In the case of transonic flows, the upwinding of the density provides a mechanism for convecting the errors in the flow direction in the supersonic pocket, until they reach the shock.[7] This behavior was not considered in the above discussions. However, it is assumed that by upwinding the density, the same behavior will be obtained for the elements inside each of the blocks. By updating the boundary fluxes at each time step, these errors will be convected between the blocks. By using a set of numerical examples, it will be demonstrated that even when the supersonic pocket is divided into two or three blocks, there are no computational difficulties in convecting the errors in the supersonic pocket.

## III. Computational Procedure

In the previous sections, the fundamental principles and objectives in designing the present computational scheme were summarized. In this section, the numerical procedure is described in detail, emphasizing its computational aspects. A flowchart summarizing the iterative scheme is shown in Fig. 2. In this flowchart, block surface is defined as a single surface attached to two neighboring blocks.

As can be seen from this flowchart, the first step is to initialize the flow speed and density in each element in all blocks and assign a boundary mass flux distribution for all global boundaries and interblock boundaries as defined in Eqs. (8) and (9). During the computations, the problem is initialized by assuming the flow to be incompressible everywhere and by assuming the flux boundary conditions at the upstream to be valid for all block boundaries.

Then, the first set of operations is performed by considering each of the blocks one at a time. There is no restriction to the sequence in which the blocks can be operated on. Given a group of processors, the block operations can be distributed among these processors. For each block, operation A is performed, which is described in subsection A. This operation involved the solution of the conservation of mass equation for each block individually under the set of given boundary fluxes as specified in Eqs. (8) and (9).

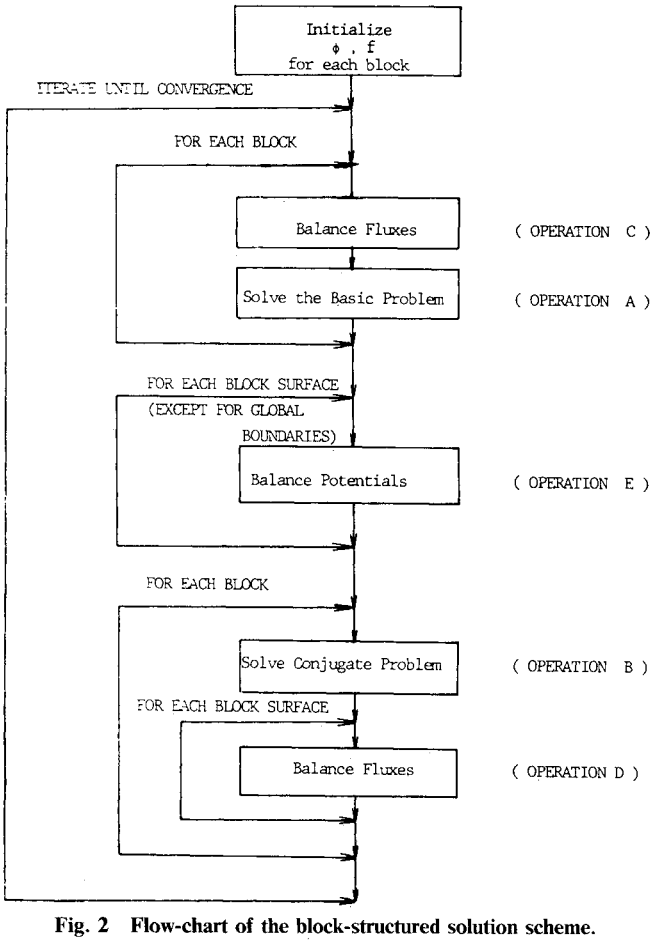


**Fig. 2 Flow-chart of the block-structured solution scheme.**

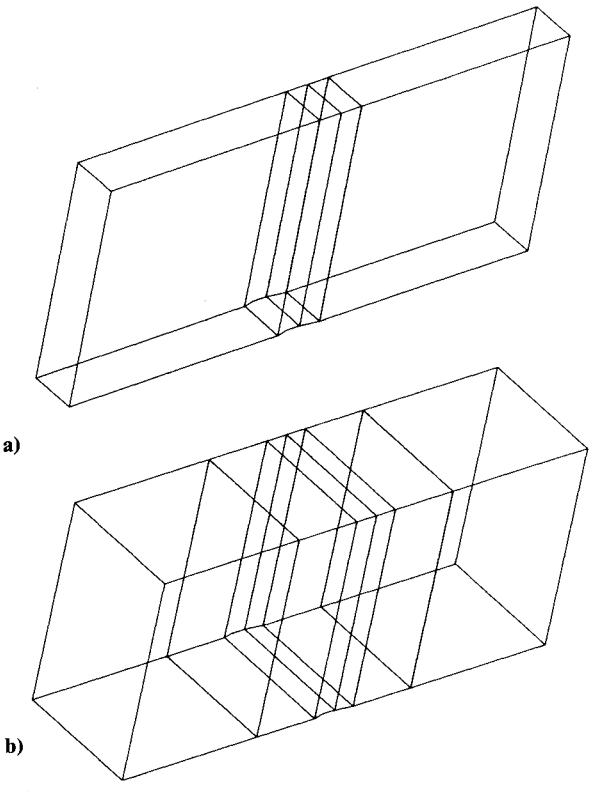


**Fig. 3 Block-structures for the analysis of transonic flow around a NACA-0012 profile. a) Grid with four blocks, b) grid with six blocks.**

The second set of equations include the treatment of each of the surfaces. This is called operation E in the flowchart. In this case, for each surface, velocity potentials calculated from individual blocks are picked, averaged by using Eq. (28) and relaxed as defined in Eq. (34). Again each surface is addressed and processed individually.

The last loop is again a block operation which is called operation B. For each block, the values of surface potentials are picked, averaged, and relaxed. These values are then imposed as boundary conditions in Eq. (11) to be used in the solution of the conservation of mass equation once more but with this new set of boundary conditions. The outcome of this operation is the boundary mass fluxes for each block. This information is written on a file of surfaces again to be averaged and relaxed. Thus, each time a block is processed, its contribution to the mass flux balancing (operation D), is simultaneously performed as indicated in the flowchart. The details of these operations are listed next.

### A. Operation A

Operation A involves the solution of the potential flow equation for each block as defined in Eqs. (7)–(9). At each iteration, Eq. (16) has to be solved to determine the nodal values of the velocity potentials. Rather than solving Eq. (16) directly, the following iterative procedure is employed[7]:

$$A_i^0 \Phi_i^{n+1} = F_i - C_{i,j}^t \lambda_j^n - A_i^n \Phi_i^n \tag{36}$$

Here, $n$ indicates the iteration step, where $\Phi$ and $\lambda$ vectors are calculated from the previous iteration. Coefficient matrix $A_i^n$ is determined by using the densities obtained from the previous iteration step. $A_i^0$ is calculated at the first step by employing a fixed density distribution.

The left-hand-side coefficient matrix is a symmetric, positive-definite matrix which can be decomposed only once and stored. In this case, each time a residual vector is determined for a vector, a forward-backward substitution operation is performed on this coefficient matrix.

During the computations, considerable time was spent in terms of formulating the residual vector. This computational time can be reduced considerably if the matrices in Eqs. (17)–(20) can be stored and read-in at every iteration step. Only the density has to be recalculated at each iteration. In this case, the core requirements increase by approximately 84 × number of elements for each block.

### B. Operation B

Operation B involves the solution of conservation of mass equation for a slightly different problem. In this case, the velocity potentials at the interblock boundaries are specified and the normal mass fluxes at the same boundaries have to be determined.

### C. Operation C

Consider a single block with a series of boundary fluxes on each surface. For a rectangular block this would mean six individual surfaces. In the present scheme, all of these fluxes are calculated individually through computations involving the neighboring blocks. It is known that one can solve the conservation of mass equation only if it is satisfied globally for the entire block. This process ensures that the conservation of mass is always satisfied before operation A is performed.

### D. Operation D

Operation D involves the balancing of fluxes as calculated from neighboring blocks. At the end of operation B, a set of boundary fluxes are calculated on each surface. In the case of blocks with matching grid points on neighboring surfaces, the above operation is conducted directly on the nodal values of the $\lambda$ vector. It should be mentioned that the above operation is in fact a block-based operation. For each element of the

block, $\lambda$ vector is calculated for each surface and stored on a surface-based data file.

#### E. Operation E

Operation E is very similar to operation D but somewhat more direct. Velocity potentials are calculated from each surface, averaged, and relaxed. Again a block-based information file is generated which involves the velocity potentials for each of the grid points for a block. At each step, surface potentials are updated once.

#### F. File Structure of the Operation

Two types of basic information are necessary for the operation: velocity potentials and boundary fluxes. This information is stored for both operations C and D in the following manner.

1) Block file for operation C—This file includes the following information for each block: 1) velocity potentials for each node, 2) geometry of all the elements, 3) shape functions for all of the elements, 4) element coefficient matrices in Eqs. (17)–(20), and 5) decomposed coefficient matrices for each block.

This file is read in for one block at a time. All blocks can be processed sequentially or in any order.

2) Surface file for operation D—This file includes the following information for each surface: 1) surface fluxes for elements on both sides of the surface, 2) geometry of the elements on both sides of the surface, 3) shape functions for these elements, and 4) element matrices for these elements.

If one reviews the flowchart, the following I/O operations are observed: operations A and B require access to the block file while operations C and D require access to the surface file.

## IV. Discussion of Results

To illustrate the applicability of the developed numerical procedure for solving transonic flow problems, a series of test cases were analyzed. As discussed in the previous sections, the convergence rate and bounds of convergence for this scheme is expected to depend on the size of the blocks and the grid refinement in each block. One can expect variations in the size of different blocks and the necessary grid refinement for each block in the case of complex three-dimensional flows. Also the location of the supersonic pocket in terms of the block-structure may be important for transonic flows. To test the robustness of the scheme in terms of both accuracy and stability, a series of block-structures was considered. In some cases, these were chosen to provide the least favorable conditions for convergence of the scheme. The test cases include two-dimensional and three-dimensional cases described as: 1) Transonic flow around a NACA-0012 airfoil ($M_\infty = 0.83$, 0.85), and 2) Transonic flow around a wing-body configuration ($M_\infty = 0.90$, $\alpha = 3$ deg).

#### A. Analysis of Transonic Flows Around a NACA-0012 Airfoil

A four-block and a six-block structure were employed for analyzing this problem as shown in Fig. 3, as well as a single-block structure for comparison purposes. In all cases, the supersonic pocket was located inside two neighboring blocks. There was a block interface located at the leading and trailing edges of the airfoil. Such choices were made to test the robustness of the scheme in terms of modeling transonic flows with multiple blocks. The number of grid points employed for each of the blocks in each case are listed as follows:

Four-Block Structure: $19 \times 26 \times 2$, $19 \times 26 \times 2$, $19 \times 26 \times 2$, $19 \times 26 \times 2$.

Six-Block Structure: $8 \times 26 \times 2$, $12 \times 26 \times 2$, $19 \times 26 \times 2$, $19 \times 26 \times 2$, $12 \times 26 \times 2$, $8 \times 26 \times 2$.

As can be seen from these lists, only one element was placed in the third dimension. A cross section of the grid is shown in Fig. 4 which included $26 \times 73 \times 2 = 3796$ grid points with 37 grid points distributed over the airfoil in the flow direction. The obtained numerical results were compared with

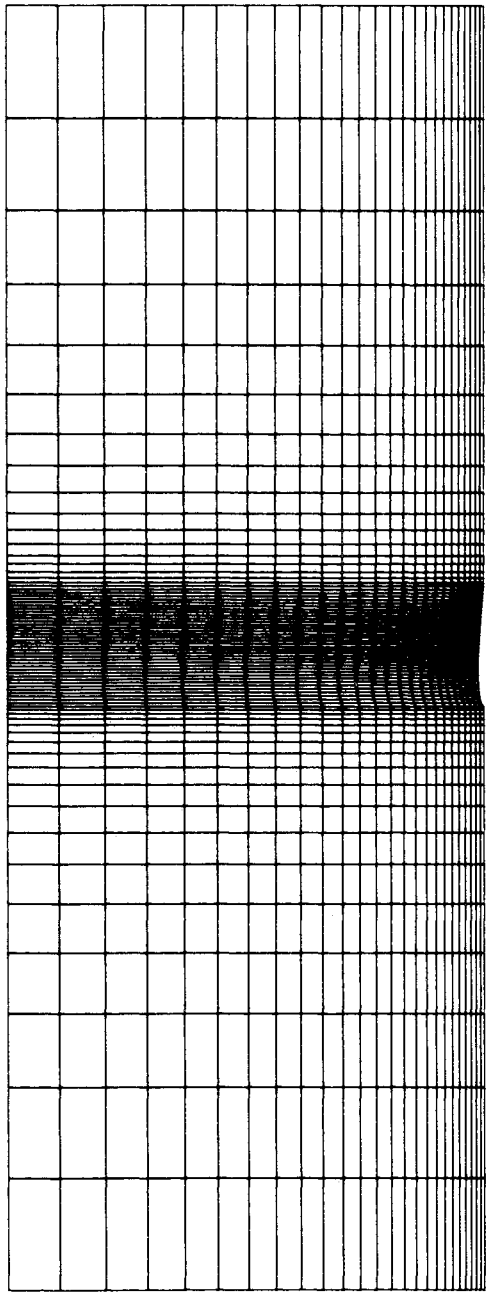

**Fig. 4 Computational grid for the analysis of transonic flow around the NACA-0012 airfoil.**

results obtained by others,[9] as shown in Fig. 5. For all block-structures including the single-block solution, the identical numerical results were obtained for the given grid.

The main objective of the developed procedure was to improve the efficiency of the relaxation schemes previously employed for the single-block solutions. A comparison of the efficiency was made here in terms of solving the same computational problem by using different numbers of blocks. This comparison was made in terms of the rate of convergence of different block-structures. Figure 6 shows the convergence rates for one-block, four-block, and six-block solutions. In these figures, the solution history is shown in terms of artificial density parameter $\mu$, as defined in Eq. (7), and the relaxation parameter $\omega$.[10] As can be seen from these solutions, the rate of convergence of the single-block solution is similar to the rate of convergence of the particular block where the shock is

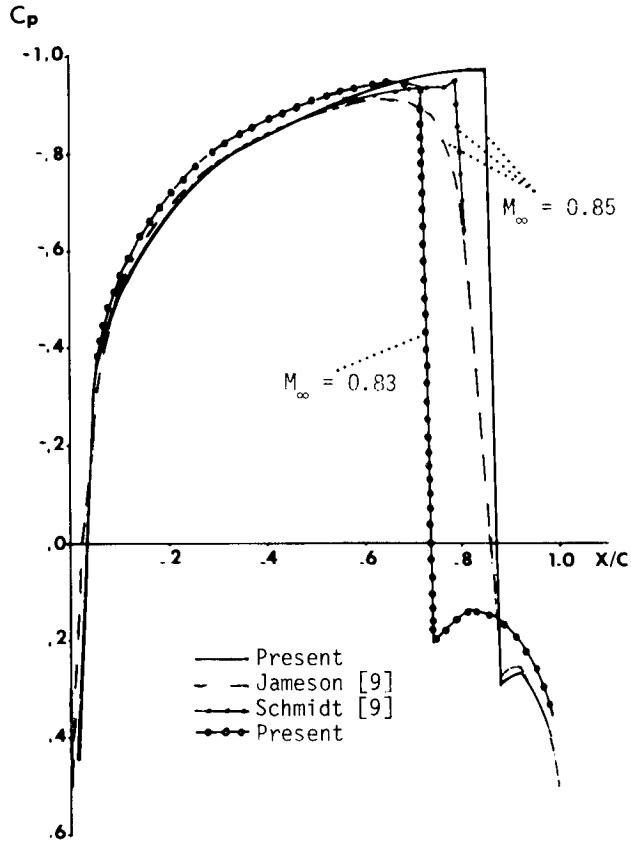


Fig. 5 Variation of pressure coefficient around a NACA-0012 airfoil for transonic flows.

located. In fact, the residuals of the other blocks follow the same pattern after a delay proportional to the location of the particular block with respect to the block with the shock.

It is difficult to make a comparison at this stage in terms of computational cost of performing such computations, since it strongly depends on the computational hardware. When all of the information describing the geometry of all of the blocks can be stored in the main memory, the computational cost is directly proportional to the decomposition and forward-backward processing of the Laplace operator as a whole versus in block-structured form. However, in the case of the block-structured solution scheme, one has to solve two sets of equations by specifying fluxes and velocity potentials, instead of one for the single-block solution. For larger problems, the cost of bringing the geometry and the related information to the main memory becomes a major factor. A careful comparison of such effects requires studying available computer systems in terms of their applicability of the developed procedure in terms of I/O and CPU costs. At this point, however, in terms of evaluating the efficiency of the present scheme, one can say that the block-structured solution scheme requires the same number of relaxation sweeps to obtain the same solution as required of a single-block relaxation scheme.

### C. Transonic Flows around a Wing-Body Configuration

A 24-block grid structure was chosen for analyzing the transonic flow problem around a wing-body configuration as shown in Fig. 7.[4,5] The details of the generated grid are shown in Fig. 8. The total number of grid points was 8207 and the maximum number of grid points in one block was 672. The design of this particular grid structure was discussed in detail in Ref. 4. Comparison of obtained results with the experimental results is shown in Fig. 9. The discrepancy between the two tests of results can be attributed to the coarseness of the grid around the leading edge of the wing profile which was studied

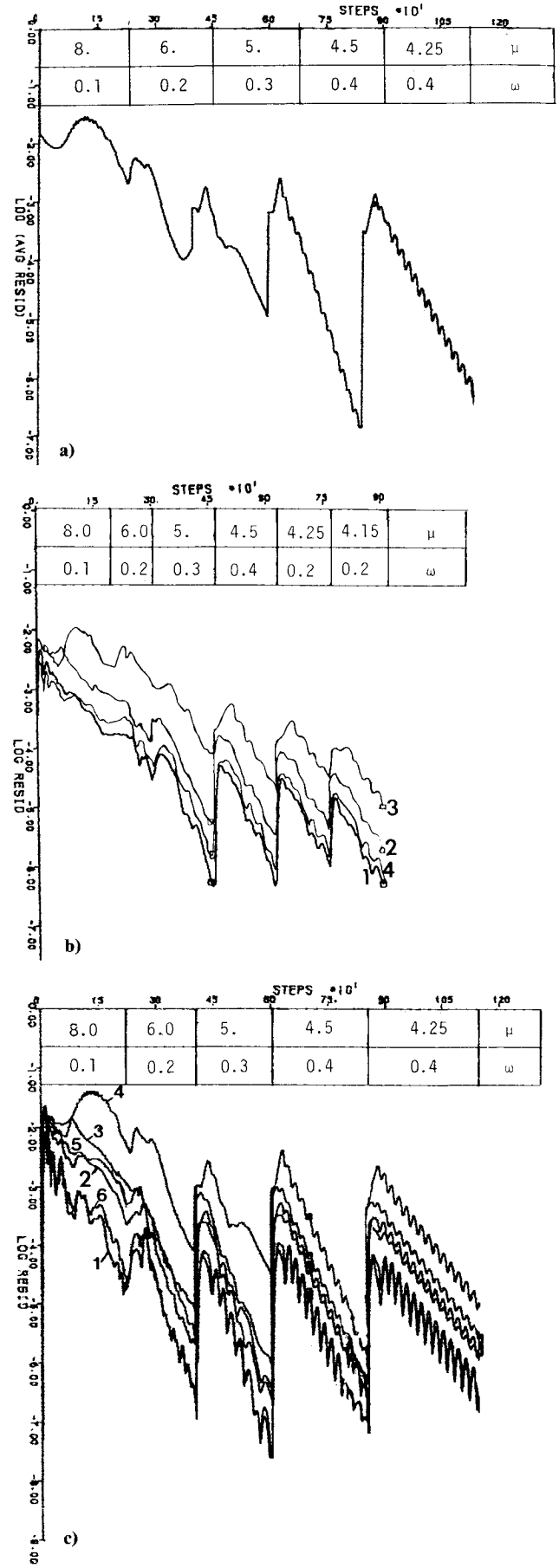


Fig. 6 Residual history for the transonic potential flow around the NACA-0012 profile. a) Single-block solution, b) four-block solution, c) six-block solution.

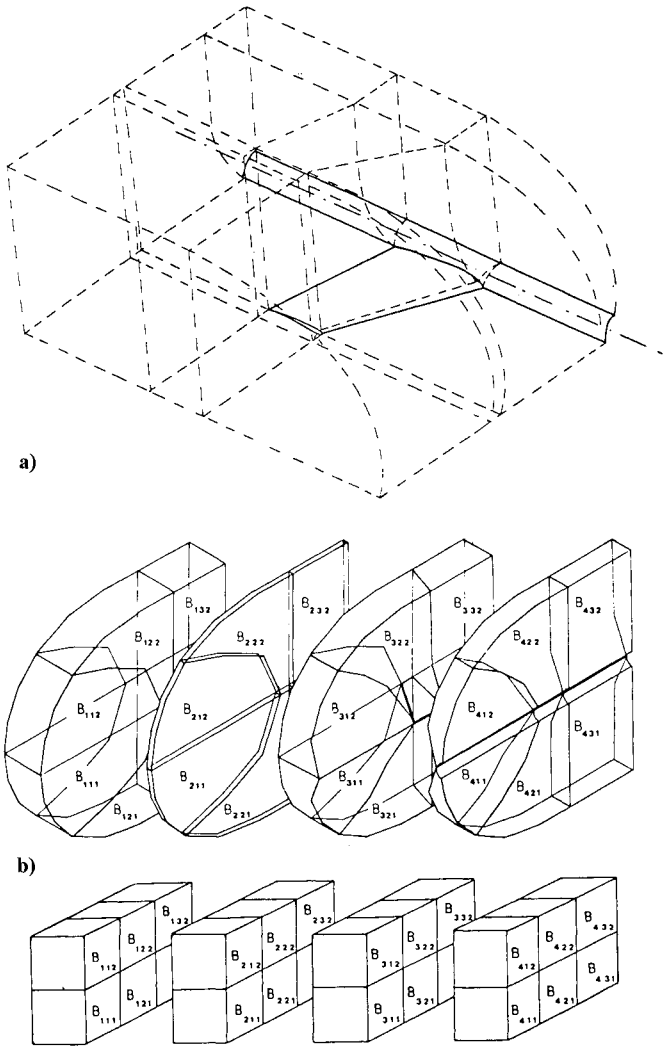


Fig. 7 Block-structure for analyzing the transonic flow around the wing-body configuration. a) Distribution of blocks in the upper level of the wing-body configuration, b) block representation of the flow domain in the real and computational spaces.

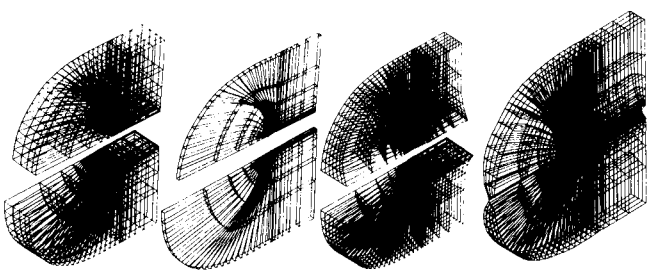

Fig. 8 Computational grids for the blocks for the wing-body configuration (3 blocks are joined together in each column).

in detail in Ref. 5. The number of iterations required for convergence in this case was 750 and involved a variable relaxation factor $\omega$ in the range of 0.10–0.45.

## V. Conclusions

In this paper, a general procedure for the solution of transonic potential flow equations using a block-structured approach was presented. The basic objective of this work was to develop a methodology for analyzing large flow problems involving complex geometries, including:

1) the generation of a computational grid around a complex geometry based on a block-structured grid generation scheme, and

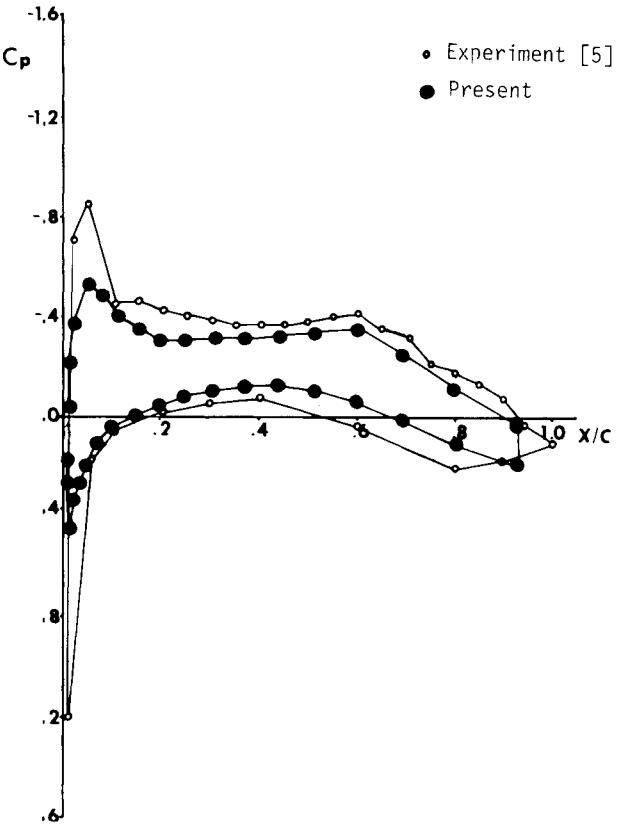


Fig. 9 Variation of pressure coefficients along the wing root for the wing-body problem.

2) the analysis of such a system by using a block-structured scheme.

Using the present scheme, one can develop computational grids for each flow region and solve the equations on a block-by-block basis. Although the block-structured scheme presented in this paper was only for potential flows, the procedure can be extended to the solutions of Euler and Navier-Stokes equations. Such an approach provides the capability of solving different types of equations approximating the flowfield for different blocks or flow regions. The method was being applied to the solution of Euler equations including the capability to solve coupled potential and Euler solutions at different flow regions.

In this paper, it was demonstrated that one can solve a transonic potential flow problem in block-structured form by using the same number of relaxation steps employed for a single-block solution. The accuracy of both types of solutions is the same. The computations require two sweeps per relaxation step over each block in the block-structured form compared to one per relaxation step in the single-block approach. However, at any time during the computation, only the data describing a single-block geometry and flow conditions are required. The procedure lends itself to parallel processing and efficient usage of computer resources in terms of I/O and CPU capabilities.

## Acknowledgments

This research was sponsored by the Air Force Office of Scientific Research under Contract F49620-83-K-0034. Support provided by the IUPUI and Indiana University Computer Centers is gratefully acknowledged.

## References

[1]Rakich, J.V., "Introduction to the Proceedings of the Symposium on Computational Fluid Dynamics," AIAA Computational Fluid Dynamics Conference, Palm Springs, CA, July 1973.

[2]Proceedings of the AIAA Computational Fluid Dynamics Conference, Danvers, MA, July 1983.

[3]Ruppert, P.E. and Lee, K.D., Patched Coordinate Systems, *Numerical Grid Generations*, Ed. by J.F. Thompson, North-Holland, Amsterdam, 1982.

[4]Ecer, A., Spyropoulos, J., and Maul, J., "A Block-Structured Finite Element Grid Generation Scheme for the Analysis of Three-Dimensional Transonic Flows," *AIAA Journal*, Vol. 23, Oct. 1985, pp. 1483–1490.

[5]Ecer, A., Citipitioglu, E., and Bhutta, B., "Design of Finite Element Grids for the Computation of Three-Dimensional Transonic Flow Around a Wing," AIAA Paper 82-1019, June 1982.

[6]Hockney, R.W. and Jesshope, C.R., *Parallel Computers*, Adam Hilger Ltd., Bristol, U.K., 1981.

[7]Ecer, A. and Akay, H.U., "A Finite Element Formulation of Euler Equations for the Steady Transonic Flows," *AIAA Journal*, Vol. 21, March 1983, pp. 343–350.

[8]Ecer, A. and Spyropoulos, J.T., "Block-Structured Solution Scheme for Analyzing Three-Dimensional Transonic Potential Flows," AIAA Paper 86-0510, Jan. 1986.

[9]Rizzi, A. and Viviand, H. (Eds.), *Numerical Methods for the Computation of Inviscid Transonic Flows with Shock Waves*, Friedr. Vieveg & Sohn, Braunschweig/Wiesbaden, 1981.

[10]Akay, H.U. and Ecer, A., "Finite Element Analysis of Transonic Flows in Highly Staggered Cascades," *AIAA Journal*, Vol. 19, Sept. 1981, pp. 1174–1182.