SECURITY CLASSIFICATION OF THIS PAGE

DTIC FILE COPY

REPORT DOCUMENTATION PAGE

(2)

AD-A199 982

1b. RESTRICTIVE MARKINGS

3. DISTRIBUTION/AVAILABILITY OF REPORT
Approved for public release; distribution unlimited.

4. PERFORMING ORGANIZATION REPORT NUMBER(S)

5. MONITORING ORGANIZATION REPORT NUMBER(S)
ARO 25162.1-EG-SBI

6a. NAME OF PERFORMING ORGANIZATION
AMTEC ENGINEERING, INC.

6b. OFFICE SYMBOL (If applicable)

7a. NAME OF MONITORING ORGANIZATION
U. S. Army Research Office

6c. ADDRESS (City, State, and ZIP Code)
11820 NORTHUP WAY, SUITE 200
BELLEVUE, WA 98005

7b. ADDRESS (City, State, and ZIP Code)
P. O. Box 12211
Research Triangle Park, NC 27709-2211

8a. NAME OF FUNDING/SPONSORING ORGANIZATION
U. S. Army Research Office

8b. OFFICE SYMBOL (If applicable)

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
DAAL03-87-C-0012

8c. ADDRESS (City, State, and ZIP Code)
P. O. Box 12211
Research Triangle Park, NC 27709-2211

10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
Three-Dimensional Viscous Flow Analysis for Moving Bodies Past Fixed Structures (Unclassified)

12. PERSONAL AUTHOR(S)
Kelton M. Peery and Scott T. Imlay

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final	FROM 7/1/87 TO 1/31/88	1988, MAY 13	43

16. SUPPLEMENTARY NOTATION
The view, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

17. COSATI CODES		
FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
3D, Navier-Stokes, Viscous Flow, Compressible Flow, Zonal, Unsteady

19. ABSTRACT (Continue on reverse if necessary and identify by block number)
A prototype computer program has been written to calculate three-dimensional viscous compressible flow past bodies in relative motion. The program solves the Navier-Stokes equations using mutliple zones of mesh. During the Phase I research and development effort, routines were written to allow the meshes to move and to couple the solutions in adjacent zones along sliding boundaries. The importance of a well planned software structure was demonstrated. The code was applied to a turbine rotor/stator interaction problem.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT
☐ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION
Unclassified

22a. NAME OF RESPONSIBLE INDIVIDUAL

22b. TELEPHONE (Include Area Code)

22c. OFFICE SYMBOL

DD FORM 1473, 84 MAR

83 APR edition may be used until exhausted.
All other editions are obsolete.

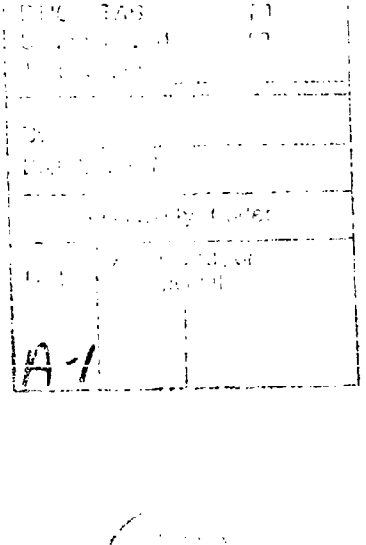
SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

ARO 25102.1-EG-SBI

Three-Dimensional Viscous Flow Analysis for Moving Bodies Past Fixed Structures

Final Report

Kelton M. Peery
and
Scott T. Imlay



U.S. Army Research Office

Contract No. DAAL03-87-C-0012

Amtec Engineering, Inc.
Bellevue, WA 98005

Summary

The overall objective of this research and development (phase I and the proposed phase II) contract is to develop a computer program for calculating the flow about aerodynamic bodies in relative motion. The technical objective of the Phase I effort was to develop and demonstrate a flow analysis procedure capable of simulating the unsteady three–dimensional compressible viscous flow associated with a turbine rotor/stator configuration. The technical question of how best to perform coupling between computational zones moving relative to each other was addressed, since it has the most significant influence upon the feasibility of the flow analysis procedure.

The Phase I work plan consisted of four main tasks. The first was to develop a general interzone boundary condition: a procedure to couple adjacent zones of mesh in relative motion. This was done using trilinear interpolation across the "sliding" boundary dividing the moving zones. The second task was to incorporate moving mesh terms and the new interzone boundary conditions into an existing three–dimensional multi–zone Navier–Stokes code. This was done using a "layered" software structure designed to minimize the amount of problem specific programming required. The third task was to demonstrate the analysis on a turbine rotor/stator flow field and the last task was to document the results.

The results of the Phase I Research and Development effort are encouraging. They have shown the benefits of developing a layered software structure, as well as demonstrated the feasibility of the coupling procedure for interzone boundary conditions between zones in relative motion. The technical feasibility of completing the proposed flow analysis code is high. The computer program resulting from the follow on funding will have potential applications throughout the aerospace industry for analysis of aerodynamic bodies in relative motion.

Contents

1 Nomenclature 3

2 Introduction 4

3 Solution Procedure 6
- 3.1 Mathematical Model . . . 6
- 3.2 Solution Procedure . . . 8
  - 3.2.1 Inviscid Flux . . . 10
  - 3.2.2 Moving Mesh Fluxes . . . 11
  - 3.2.3 Viscous Flux . . . 11
- 3.3 Boundary Conditions . . . 12
  - 3.3.1 Subsonic Inflow . . . 12
  - 3.3.2 Subsonic Outflow . . . 15
  - 3.3.3 Noslip Wall . . . 16
  - 3.3.4 Periodic Boundary Conditions . . . 16

4 Zonal Boundary Conditions 19
- 4.1 Simply Connected Butt Joint Boundary . . . 20
- 4.2 General Sliding Boundary . . . 20

5 Turbine Rotor/Stator Calculation 26
- 5.1 Approach . . . 26
- 5.2 Results of Calculations . . . 32

6 Conclusions 39

7 Recommendations 40

List of Figures

1 Finite–Volume Mesh . . . 8
2 Finite-Volume Cell . . . 9
3 Cascade Flow . . . 18
4 Application of Periodic Boundary Conditions for cascade flow . . . 18
5 Midspan Mesh for Turbine Rotor/Stator calculation . . . 22
6 Fictitious Boundary Cells of Zone 1 Overlap Sliding Boundary . . . 22
7 Two-Step Interzone Data Transfer Process . . . 23
8 Two possible divisions of a quadralateral cell into triangles . . . 24
9 Subtriangles with P within (left) and outside of (right) the triangle . 24
10 Zonal Configuration for Turbine Rotor/Stator Calculation . . . 26

11 Layered Code Structure of Pilot Code 28
12 Interzone Data Transfer Between Non-aligned Zones 30
13 Periodic Boundary Conditions for Stator Zone 31
14 Mesh used for stationary rotor calculation 33
15 Pressure contours from stationary rotor calculation 34
16 Velocity vectors from stationary rotor calculation 34
17 Streamlines from stationary rotor calculation 35
18 Turbine Rotor/Stator calculation — coarse midspan meshes at three times during the calculation 36
19 Turbine Rotor/Stator calculation — midspan velocity vectors at three times during the calculation 37
20 Turbine Rotor/Stator calculation — midspan streamlines at two times during the calculation 38

1 Nomenclature

c_p	Specific heat at constant pressure
c_v	Specific heat at constant volume
e	Internal energy per unit mass ($e = c_v T$)
E	Total Energy per unit Volume $E = \rho[e + 1/2(u^2 + v^2 + w^2)]$
F_i	Fluxes in the i-component direction
k_l	Laminar coefficient of heat conduction
k	Coefficient of heat conduction (molecular and turbulent)
$\vec{n}$	Unit outward normal vector to a closed surface, S, in equation 1
p	Static Pressure
$\vec{P}$	Combined flux vector,$\vec{P} = F_i\vec{i} + F_2\vec{j} + F_3\vec{k}$, in equation 1
Pr_l	Laminar Prandtl number, $Pr_l = \frac{\mu_l c_p}{k_l} = 0.72$
Pr_t	Turbulent Prandtl number, $Pr_t = \frac{\mu_t c_p}{k_t} = 0.9$
$\vec{P}\cdot\vec{S}_{i+1/2}$	Fluxes through surface separating cells i, j, k and $i+1, j, k$
q_i	Combined molecular and turbulent heat fluxes
R	Gas Constant
S	A closed surface surrounding volume, V, in equation 1
U	Conservative Variables, $U = [\rho, \rho u, \rho v, \rho w, E]^T$
W	Primative Variables, $W = [u, v, w, p, T]^T$
t	Time
T	Temperature
u_i	i^{th} component of velocity
x_i	i^{th} component of position
δ_{ij}	Kronecker delta ($\delta_{ij} = 1$ when $i = j$ and $\delta_{ij} = 0$ otherwise)
ρ	Density
γ	Ratio of specific heats
τ_{ij}	Combined viscous and turbulent stress tensor
μ	Combined molecular and turbulent viscosity coefficient
μ_l	Laminar viscosity coefficient
ϕ	Parameter in interpolation to surface ($\phi = 0$; First order,$\phi = 1$, Second order)
κ	Parameter in interpolation to surface ($\kappa = -1$; Fully upwind differencing, $\kappa = 1$; Central differencing)

2 Introduction

Computer codes that numerically simulate the flow of fluids are increasingly being used in the design and analysis of aircraft components. Although full three-dimensional viscous calculations can be expensive, they also can provide information that is extremely beneficial Use of flow analysis codes can reduce costs of parametric testing and/or increase the variety and extent of configurations that are considered during a design process. Many times a component must be designed for conditions outside any existing experimental data base or at conditions that make parametric testing impractical. For such situations numerical simulation may be the principal method used in the design process.

The design of aircraft components has evolved to a sophisticated level. Many critical components such as the flow passages in turbomachinery, nozzles, and inlets operate at very high efficiencies (at least at their design points). Further performance improvements will be difficult and expensive to obtain using 'cut and try' testing methods or even two–dimensional flow analysis procedures. It appears that three–dimensional flow analysis capable of handling complex configurations will eventually provide the only economical method for designing aircraft components that have significantly greater performance.

One area of particular interest to the U.S. Army is that of unsteady flows in which one or more aerodynamic bodies move relative to a fixed structure. Examples of this flow situation are: turbomachinery flow passages, helicopter blade/fuselage interaction, propeller/wing/nacelle interactions, dispensing of submunitions from missiles, missile stage separation transient, and release/launch of munitions from aircraft. All of these flow situations have complex geometries, are inherently unsteady, and have significant viscous effects. The experimental data of Dring, et al[8] show that the temporally varying pressure measured at the leading edge of a turbine rotor can be as much as 72% of the exit dynamic pressure when the spacing between moving rotor and fixed stator blades is small. Obviously, the unsteady effects in turbomachinery can be substantial. There is, therefore, a need for an engineering analysis tool for simulating unsteady viscous flows past three– dimensional aerodynamic bodies in relative motion.

Numerous calculations of cascade flows have been described in the literature (for example[9,10,5,7]). Except for the work of Rai[5,7] all of the known work has been steady flows or has used equation sets simpler than the Navier-Stokes equations. The work by Rai in[7] suggests that the flow analysis technology is available and is practical for full three-dimensional unsteady turbomachinery flow calcula-

tions. However, all of the existing computer codes for cascade flows are 'research codes'--codes written to develop algorithms. They are not engineering tools. An engineering tool needs to be easy to use. It must be reliable and have a minimum number of input variables that must be 'tuned' (e.g. smoothing terms) to achieve a satisfactory solution. It must include a mesh generator for at least one specific class of configurations. It must have comprehensive and readable documentation of its use, limits of applicability, and theory. It must be thoroughly validated. In addition, for it to be user friendly, it must check the user input data for errors and then give informative messages back to the user. The real benefit of flow analysis in the design of aircraft components will only be realized after engineering tools of the sort described above become available to design engineers.

This contract (Phase I and the proposed Phase II) would lead to an engineering tool for the numerical simulation of three– dimensional flows around aerodynamic bodies moving past fixed structures. This flow analysis procedure would divide the computational domain into one or more zones in which a single body-fitted mesh can easily be generated. Mesh zones would also be allowed to move in time–each with a unique velocity and acceleration to provide capability for moving bodies. The Phase I effort, being reported here, was intended to determine the feasibility of the engineering tool discussed above. The Phase I work concentrated a single problem—the unsteady rotor/stator interaction problem solved previously by Rai[5,7]. The work plan consisted of the following four tasks

1. Develop and Incorporate Interzone Boundary Conditions into an Existing Multi-zone 3D Navier-Stokes Code

2. Incorporate Moving Mesh Terms into the Navier-Stokes Code

3. Demonstrate the Analysis on a Turbine Rotor/Stator Flow Field

4. Document Results

The work on Task 1 is described in detail in Section 3.2.2, and Task 3 is described in Section 5.

3 Solution Procedure

The pilot code solves the compressible Navier-Stokes equations using a time-marching solution procedure and a multiple zone grid. The solution procedure is discussed in this section and the interzone boundary conditions are discussed in Section 3. The discussion of the solution procedure is broken into three parts: Section 2.1 describes the mathematical model, Section 2.2 describes the solution procedures used, and Section 2.3 describes the boundary conditions used. Innovations developed during Phase I include the moving mesh terms described in Section 2.2 and the periodic boundary conditions described in Section 2.3. The principle innovation is the general sliding boundary described in Section 3.2.

3.1 Mathematical Model

The governing equations that are solved by the pilot code are the time-dependent three-dimensional compressible Navier-Stokes Equations. These equations are given below in integral[1] form[1].

$$\frac{\partial}{\partial t}\iiint_V U\, dV + \iint_S \vec{P}\cdot\vec{n}\, dS = 0 \tag{1}$$

where

$$\vec{P} = F_1\vec{i} + F_2\vec{j} + F_3\vec{k}$$

and

$$U = \begin{pmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho u_3 \\ E \end{pmatrix}$$

$$F_i = \begin{pmatrix} \rho u_i \\ \rho u_i u_1 + p\delta_{1i} - \tau_{i1} \\ \rho u_i u_2 + p\delta_{2i} - \tau_{i2} \\ \rho u_i u_3 + p\delta_{3i} - \tau_{i3} \\ (E+p)u_i - u_j\tau_{ij} + q_i \end{pmatrix}$$

where t is the time, V is an arbitrary volume, S is the surface surrounding V. In the above equations the standard summation convention (sum over repeated indices) is used.

[1]The integral form of the equations is strongly conservative.

The following auxillary equations are needed to close the system.

$$\begin{aligned} E &= \rho\left[e + \frac{1}{2}u_j u_j\right] \\ \tau_{ij} &= \mu\left[\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) - \frac{2}{3}\delta_{ij}\frac{\partial u_k}{\partial x_k}\right] \\ q_i &= -k\frac{\partial T}{\partial x_i} \\ p &= \rho(\gamma - 1)\, e = \rho RT \\ e &= c_v T \end{aligned}$$

Here μ and k are the sums of the molecular and turbulent values of the viscosity and heat conduction coefficients.

The molecular viscosity is obtained from Sutherlands law for air.

$$\mu_l = 1.4519(10)^{-6}\frac{T^{3/2}}{T + 110.0}$$

The coefficient of heat conduction is then obtained by assuming a constant Prandtl number, $Pr_l = 0.72$.

$$k_l = \frac{\mu_l c_p}{Pr_l}$$

For turbulent flows the algebraic model of Baldwin and Lomax[2] is used to calculate μ_t. The turbulent heat conduction coefficient is then calculated assuming a turbulent Prandtl number of 0.9.

Physically Equation 1 represents a very simple idea: the time rate of change of mass, momentum, and energy within an arbitrarily chosen volume, V, is equal to the apparent flux of these quantities inward through the surface, S, surrounding the volume. The finite volume method consists of breaking the flow field up into a large number of nearly hexahedral finite volume cells, as shown in Figure 1, and applying the integral equations directly to each volume. The finite volume method is the natural form for a conservative solution procedure and it avoids certain metric singularities which require special treatment in a finite difference method.

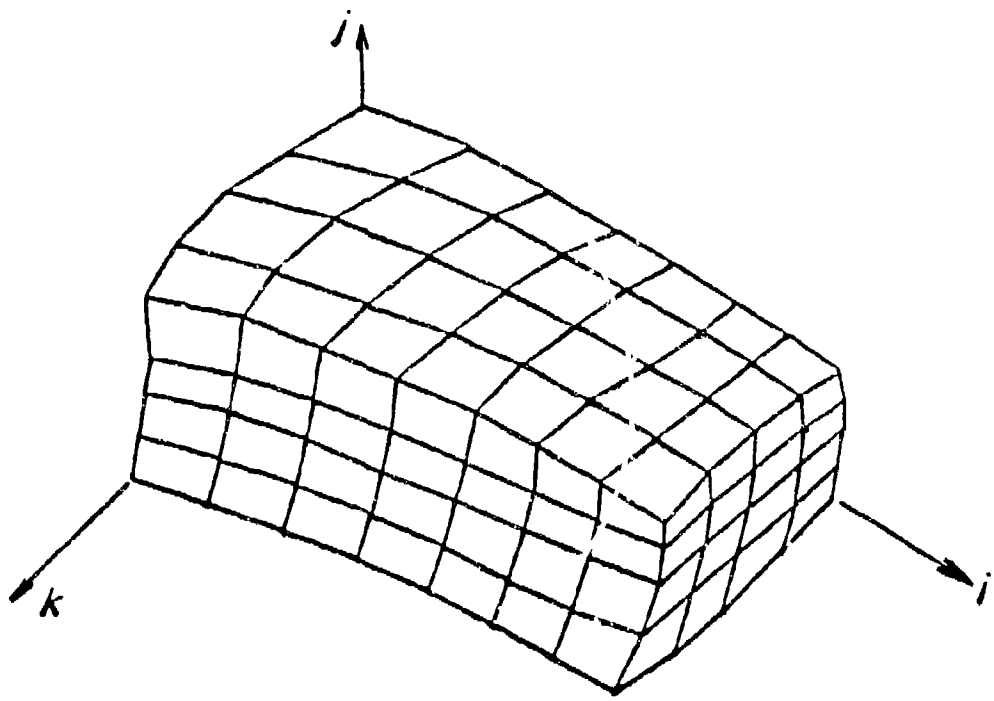


Figure 1: Finite–Volume Mesh

3.2 Solution Procedure

The pilot code is a zonal code using body-fitted nonorthogonal grids for each zone. Each zonal grid is structured such that is has a logical i, j, k ordering. Using a large number of zones patched together a body-fitted grid may be generated for very complex geometries. The zonal grids are generated separately and read into the pilot code from a file. The user specifies the zonal connectively.

The Navier-Stokes equations are solved using a time-marching finite-volume method. The method is derived by applying the integral equation, Equation 1, to a finite-volume given by indicies i, j, k (see Figure 2).

$$\frac{d}{dt}\left(U_{i,j,k} Vol_{i,j,k}\right) = -\left(D_i \vec{P}\cdot\vec{S} + D_j \vec{P}\cdot\vec{S} + D_k \vec{P}\cdot\vec{S}\right)_{i,j,k}$$

where $U_{i,j,k}$ is the mean value of U in cell i, j, k and $D_i \vec{P}\cdot\vec{S}$ for example, represents the difference of the fluxes through opposing faces of the cell in the i-index direction.

The time derivative is approximated using forward in time differencing:

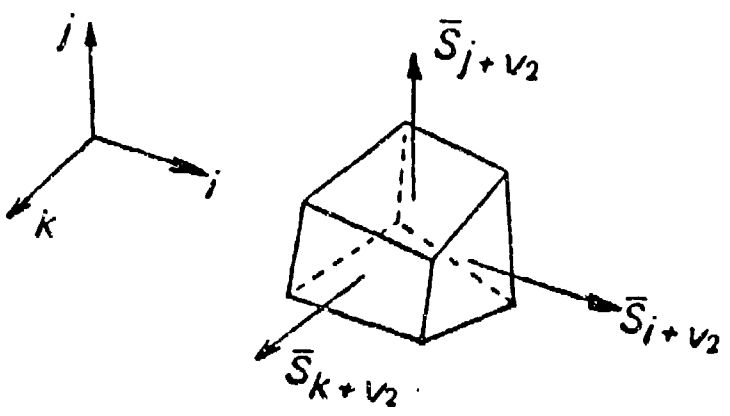


Figure 2: Finite-Volume Cell

$$\frac{Vol_{i,j,k}}{\Delta t_{i,j,k}} \delta U_{i,j,k} = - \left(D_i \vec{P} \cdot \vec{S} + D_j \vec{P} \cdot \vec{S} + D_k \vec{P} \cdot \vec{S} \right)^n_{i,j,k} \tag{2}$$

where

$$\delta U_{i,j,k} = U^{n+1}_{i,j,k} - U^n_{i,j,k}$$

The fluxes must now be approximated in terms of the $U_{i,j,k}$. For conciseness consider only the $i + 1/2$ surface.

For the approximation of the flux through a surface the inviscid and diffusion terms of the flux vector are considered separately.

$$\vec{P} \cdot \vec{S} = \vec{P} \cdot \vec{S}^{inv} + \vec{P} \cdot \vec{S}^{diff}$$

These terms are then evaluated in a manner consistent with the predominant nature of the equations in the limit as $Re \to \infty$ (hyperbolic) and $Re \to 0$ (parabolic); i.e., upwind differencing for the inviscid terms and central differencing for the diffusion (viscous stress and heat flux) terms.

3.2.1 Inviscid Flux

The evaluation of the inviscid terms is based on the flux vector splitting method developed by Steger and Warming.[3] This method splits the flux vector $\vec{P}\cdot\vec{S}$ into vectors containing the information propogating in the direction of $\vec{S}, f^+$, and in the direction opposite to $\vec{S}, f^-$. The flux vector for the i+1/2 surface is then written

$$\vec{P}\cdot\vec{S}^{(inv)}_{i+1/2} = \left[f^+\left(U^-\right) + f^-\left(U^+\right)\right]_{i+1/2}$$

The notation $f^+(U^-)$ denotes f^+ evaluated at U^-. U^- and U^+ represent upwind interpolations to the cell interfaces

$$U^{\mp}_{i\pm1/2} = U_{i,j,k} \pm \frac{\phi}{4}\left[(1\mp\kappa)\,\nabla_i + (1\pm\kappa)\,\Delta_i\right]U_{i,j,k} \tag{3}$$

where $\nabla_i U_{i,j,k} = U_{i,j,k} - U_{i-1,j,k}$, is a backward difference operator in the i-direction and $\Delta_i U_{i,j,k} = U_{i+1,j,k} - U_{i,j,k}$ is a forward difference operator in the i-direction. The parameter, κ, determines the nature and accuracy of the spacial differencing: $\kappa = -1$ corresponds to a second–order fully-upwind scheme, $\kappa = +1$ to a second–order central difference scheme, and $\kappa = 1/3$ to a third–order upwind biased scheme. For first–order accuracy ϕ is set to zero and for higher order accuracy ϕ is set to one.

In general, some form of flux limiting is required to avoid oscillations at shock waves. The limiting can be implemented by locally varying the difference quotients in the interpolation, Equation 3, in relation with local gradients of the solution. If properly implemented a limiter will yield sharp monotonic shock waves. In this investigation a simple form of limiting, based on the second difference of pressure, was used. In particular

$$\phi_{i+1/2} = MAX\left(1 - C_{lim}\,DDP,\ 0.0\right)$$

where DDP is the magnitude of the second difference of pressure normal to the surface (in this case, the i-direction). The differencing parameter, κ, remains unchanged. This limiting reduces the form of differencing toward first–order upwind differencing in regions of strong pressure gradients (such as shock waves). The spacial order of accuracy remains second–order everywhere unless ϕ is actually zero.

3.2.2 Moving Mesh Fluxes

The intregal equation, Equation 1, may be applied to a finite volume mesh that is moving. The limits of the integrals are then functions of of time and their evaluation is more complicated. In the case where the volumes of the cells remain constant–rigid body motion of a zone–the descrete conservation equation, Equation 3.2, retains the same form and the only affect of moving the mesh is to modify the fluxes, $\vec{P}\cdot\vec{S}$. The modified fluxes are

$$(\vec{P}\cdot\vec{S})_m - v_n\bar{U}|\vec{S}|$$

where $(\vec{P}\cdot\vec{S})_m$ is the flux function without moving mesh, v_n is the velocity of the surface normal to itself, and $\bar{U}$ is some mean value of the conservative variables. During Phase I, $\bar{U}$ was taken as the average of the conservative variables in the cells on either side of the surface.

3.2.3 Viscous Flux

The contribution to the fluxes from the diffusion terms (viscous stresses and heat conduction) are approximated using standard central differences. Following the development in Reference[11] these fluxes can be written in terms of second differences of the primative variables in the three generalized coordinate directions

$$\vec{P}\cdot\vec{S}^{(diff.)}_{i+1/2} = F_d\left(W_\xi, W_\eta, W_\zeta\right) \tag{4}$$

where ξ, η, and ζ correspond to the directions of increasing i, j, and k indices, respectively. These derivatives are approximated as follows

$$\begin{aligned} W_\xi &= W_{i+1,j,k} - W_{i,j,k} \\ W_\eta &= 0.25\left(W_{i+1,j+1,k} - W_{i+1,j-1,k} + W_{i,j+1,k} - W_{i,j-1,k}\right) \\ W_\zeta &= 0.25\left(W_{i+1,j,k+1} - W_{i+1,j,k-1} + W_{i,j,k+1} - W_{i,j,k-1}\right) \end{aligned} \tag{5}$$

Using equations 5 and the relationship between the conservative variables and the primative variables the discrete approximation to the diffusion fluxes, Equation 4, is

written in terms of the conservative variables within the 10 cells nearest the i+1/2 face. The thin layer form of the equations are obtained by neglecting W_η and W_ζ in Equations 4 and 5.

3.3 Boundary Conditions

Five types of boundary conditions are needed for the rotor-stator interaction problem: subsonic inflow, subsonic outflow, noslip wall, periodic, and interzone. The first four boundary conditions will be discussed here while discussion of the last will be deferred to Section 4.

The implementation of all boundary conditions is simplified by the use of boundary cells. Those are extra cells added outside the solution domain solely for the purpose of satisfying boundary conditions. When boundary cells are used the solution process within the interior of the solution domain may proceed without any special differencing near the boundaries. The current solution procedure reuquires two layers of boundary cells.

3.3.1 Subsonic Inflow

The treatment of the inflow boundary conditions is guided by the theory of characteristics. A locally one-dimensional flow has five characteristic equations with slopes $u', u', u', u'+c$, and $u'-c$. If the flow field is supersonic then all five characteristic equations are propogating information in the positive x-direction. In this case all data must be specified at the inflow boundary. If the flow is subsonic at the inflow boundary, then one of the characteristics, the $u'-c$ characteristic, has a negative slope and it propogates information from the interior upstream to the inflow boundary. In this case only four items may be specified at the inflow boundary and the fifth item must be allowed to vary as the solution progresses.

For the case of subsonic inflow, the stagnation pressure, stagnation temperature, and flow angles are specified. These quantities are related to the static pressure and static temperature by the following equations:

$$\frac{p}{p_t} = \left[1 - \frac{\gamma-1}{\gamma+1}\left(\frac{V_{Tot}}{a_*}\right)^2\right]^{\frac{\gamma}{\gamma-1}}$$

$$
\begin{aligned}
\frac{\rho}{\rho_t} &= \left[1 - \frac{\gamma - 1}{\gamma + 1}\left(\frac{V_{Tot}}{a_*}\right)^2\right]^{\frac{1}{\gamma - 1}} \\
\frac{u}{V_{Tot}} &= \left(\frac{u}{V_{Tot}}\right)_0 = \text{specified} \\
\frac{v}{V_{Tot}} &= \left(\frac{v}{V_{Tot}}\right)_0 = \text{specified} \\
\frac{w}{V_{Tot}} &= \left(\frac{w}{V_{Tot}}\right)_0 = \text{specified}
\end{aligned} \tag{6}
$$

The first two equations above are simply the isentropic relations written in terms of the total velocity, V_{Tot}, and the speed of sound at a sonic throat, a_*. The speed of sound at a sonic throat is calculated from the specified stagnation temperature.

$$
(a_*)^2 = \frac{2R}{\gamma + 1}\frac{p_t}{\rho_t}
$$

Equations 6 are a system of five equations in five unknowns: p, ρ, u, v, and w, but one of the last three equations is redundant. To complete the system another equation is needed. This is to be expected since, as was explained earlier, only four items may be specified at the inflow boundary.

The last equation to close the system is the characteristic relation carrying information upstream to the inflow boundary.

$$
\frac{\delta p}{\delta t} - \rho c \frac{\delta u'}{\delta t} = (u' - c)\left[\frac{\delta p}{\delta x} - \rho c \frac{\delta u'}{\delta x}\right]
$$

This equation is forward differenced.

$$
\delta p_B - \rho c \delta u'_B = \frac{(u' - c)\,\Delta t}{\Delta x}\left[p_I - p_B - \rho c\,(u'_I - u'_B)\right]^n \tag{7}
$$

The subscripts I and B indicate the first interior cell and the inflow boundary cell, respectively. The prefix δ indicates the forward in time difference of the variable following it.

The number of unknowns is reduced to three if the isentropic relations, the first two of Equations 6 are written in terms of u'. This is done using the relation

$$V_{Tot}^2 = bu'^2 \tag{8}$$

where

$$\begin{aligned} b &= \frac{1}{1+(u'/V_{Tot})^2-(u/V_{Tot})^2-(v/V_{Tot})^2-(w/V_{Tot})^2} \\ &= constant. \end{aligned}$$

The modified isentropic relations are

$$\begin{aligned} \frac{p}{p_t} &= \left[1-\frac{\gamma-1}{\gamma+1}b\left(\frac{u'}{a_*}\right)^2\right]^{\frac{\gamma}{\gamma-1}} \\ \frac{\rho}{\rho_t} &= \left[1-\frac{\gamma-1}{\gamma+1}b\left(\frac{u'}{a_*}\right)^2\right]^{\frac{1}{\gamma-1}}. \end{aligned} \tag{9}$$

The modified isentropic relations, Equations 9, and the descrete form of the upstream runing characteristic relation, Equation 7 are three algebraic relations in three unknowns.

The isentropic relation for pressure, the first of Equation 9, may be placed in delta law form by considering incremental changes in the variables p and u'.

$$\delta p_B = p_t \frac{2b\gamma}{\gamma+1}\frac{u'}{a_*^2}\left[1-\frac{\gamma-1}{\gamma+1}b\left(\frac{u'}{a_*}\right)^2\right]^{\frac{1}{\gamma-1}}\delta u'_B \tag{10}$$

This equation and Equation 7 are solved for u'. The pressure and density are then obtained from the isentropic relations, Equations 9. The velocities are also calculated from u'_B using Equation 8 and the last three of Equations 6. The specification of the flow within the subsonic inflow boundary cell is then complete.

3.3.2 Subsonic Outflow

The treatment of outflow boundary conditions is also guided by the theory of characteristics. If the flow normal to the outflow boundary is supersonic then all characteristics have positive slopes and no information propogates upstream from the boundary cell to the interior cell. In this case the five locally one-dimensional characteristic equations are used to update the solution in the boundary cell. If the flow normal to the outflow boundary is subsonic then the $u'-c$ characteristic propogates information upstream from the boundary cell to the interior cell. In this case, one item must be specified at the boundary cell.

For subsonic outflow the static pressure is specified. The remaining variables in the boundary cell are calculated using the four downstream running characteristic equations As with the variables specified at the inflow boundary, the requirement of constant pressure at the outflow boundary is written in delta law form.

$$\delta p_B = 0 \tag{11}$$

This equation is combined with the four characteristic relations.

$$\begin{aligned}
\delta\rho_B + \frac{1}{c^2}\delta p_B &= -\frac{u'_I \Delta t |\vec{S}|}{Vol_I}\left[\rho_B - \rho_I + \frac{1}{c^2}(p_B - p_I)\right] = R_1 \\
\delta p_B + \rho c \delta u'_B &= -\frac{(u'+c)\,\Delta t |\vec{S}|}{Vol_I}\left[p_B - p_I + \rho c\,(u'_B - u'_I)\right] = R_2 \\
\delta v'_B &= \frac{u'_I \Delta t |\vec{S}|}{Vol_I}\left[v'_B - v'_I\right] = R_3 \\
\delta w'_B &= \frac{u'_I \Delta t |\vec{S}|}{Vol_I}\left[w'_B - w'_I\right] = R_4
\end{aligned}$$

The above four equations and Equation 11 are five linear algebraic equations in the five unknowns $\delta\rho_B$, $\delta u'_B$, $\delta v'_B$, $\delta w'_B$, and δp_B. This system is solved directly and the boundary cell solution is updated.

3.3.3 Noslip Wall

At solid adiabatic walls the pressure and temperature gradients are assumed to be zero and a no slip (zero velocity at the wall) boundary condition is applied. To resolve the boundary layer the mesh must be refined near the wall so that the cell nearest the wall is deep within the boundary layer. In this case, boundary layer theory shows that pressure gradient normal to the wall is a higher order effect and can be neglected. Similarly, for adiabatic walls the temperature gradient normal to the wall deep within the boundary layer is negligable. These boundary conditions are satisfied by setting the pressure and internal energy in each boundary cell equal to the pressure and temperature in the interior cell adjacent to the boundary. The no-slip condition is satisfied by setting the velocity in the boundary cell equal and opposite to the velocity in the adjacent interior cell.

3.3.4 Periodic Boundary Conditions

It is often possible to reduce the scale of a problem by assuming that the flow field is periodic. For example, in cascade flows the cost of calculating the flow about all airfoils of the cascade may be prohibitive, whereas the cost of calculating the flow about an isolated airfoil is reasonable. Assuming that the flow fields about the airfoils are identical except for position, the cascade problem may be reduced to the calculation of flow about one airfoil. This is done by choosing any surface passing above an airfoil and duplicating it below the airfoil, as shown in Figure 3. These two surfaces define the solution domain. The presence of the neighboring airfoils is included by allowing the solution to vary such that the solutions along the upper and lower surfaces are identical.

The periodic boundary conditions are applied as depicted in Figure 4. On each time step the solution in the two layers of interior cells nearest the upper boundary are copied into boundary cells on the lower boundary. Likewise, the solution in the two layers of interior cells nearest the lower boundary are copied into the boundary cells on the upper boundary. In this manner the solutions extrapolated to the upper and lower boundaries is identical and the solution is periodic.

For the rotor/stator interaction problem considered in this investigation the solution is periodic in cylindrical coordinates. The flow is described, however, in cartesian coordinates. Therefore, the velocity vectors must be rotated appropriately before they are placed into the boundary cells.

The periodic boundary condition is very much like an interzone boundary condition in that both involve copying the solution from the interior cells of one zone into the boundary cells of another zone. For the periodic boundary conditions, the zone copied from and the zone copied to just happen to be the same zone. The implementation of the periodic boundary conditions actually uses the same routines as the interzone boundary condition with the addition of a routine to rotate the velocity vectors. Details of the interzone boundary condition implementation are given in Section 4.

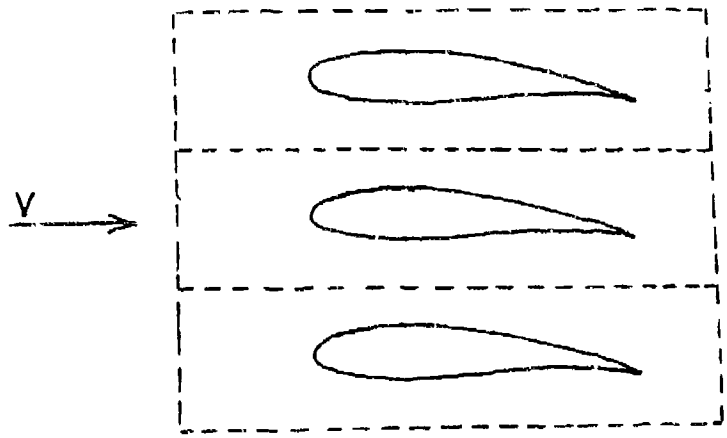


Figure 3: Cascade Flow

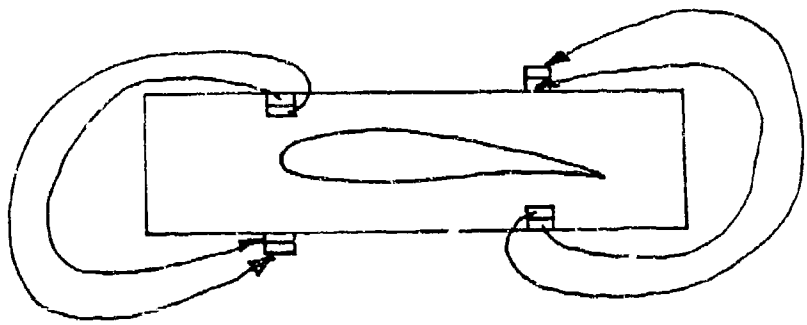


Figure 4: Application of Periodic Boundary Conditions for cascade flow

4 Zonal Boundary Conditions

The pilot code uses multiple zones of i, j, k ordered body grids. There are two main advantages of a multiple zone code over a single zone code. The first is that the multiple zones offers a greater degree of geometric flexibility than is available with a single zone. This simplifies the task of generating an adequate mesh for moderately complex geometries. For highly comples geometries, the generation of an adequate mutliple zone mesh may be possible when the generation of an adequate single zone mesh is impossible. The second advantage is that the use of memory management is simplified.[2] The code may keep only one zone within the primary memory at a time while the other zones are in secondary memory (SSD or disk). For complex problems with many zones, this can dramatically reduce the amount of primary memory required.

The multiple zone approach has disadvantages. The main disadvantage is that flowfield data must somehow be transferred without both zones being present in memory. This data transfer process is referred to a specification of zonal boundary conditions.

The data transfer at zonal boundaries requires a two step process. The first step is to extract the flow data from the interior of one zone, the 'donor' zone, and store it in a temporary location in memory, called the 'patch'. At this point the donor zone may be written to secondary memory and the adjacent zone, the 'host' zone, may be read from secondary memory. The second step is to copy the 'patch' to the boundary cells of the 'host' zone. This process transfers flow data across the zonal boundary in one direction. The process is repeated, with the zones trading 'host' and 'donor' roles, to transfer flow data across the zonal boundary in the opposite direction.

Two types of interzone zonal boundary conditions were utilized during the Phase I investigation: a simply connected butt joint boundary and a generalized, sliding, butt joint boundary. The two differ in how they extract data from the 'donor' zone. The general sliding butt joint boundary procedure was developed as part of the Phase I effort.

[2]Some form of memory management is required when memory constrained computers such as the CRAY XMP are used.

4.1 Simply Connected Butt Joint Boundary

The procedure for simply connected butt joint boundaries is that the boundary cells along an interzone segment of a zone boundary (i.e., that segment of the zone boundary which attaches to another zone) are duplicated in a patch[3]. For cases where the mesh lines are continuous across the zonal interface surface, the flow data (conservative field variables, non-conservative field variables, properties, etc.) are simply copied from the appropriate cell of the donor zone into the patch. The data from the patch are then copied into the appropriate boundary cells of the host zone.

4.2 General Sliding Boundary

The primary goal of the Phase I work was to demonstrate the multi-zone Navier-Stokes analysis on an unsteady turbine rotor/stator flow field. Figure 5 shows a computational mesh at midspan for a turbine rotor/stator problem. As shown, the stator blade is in the fixed zone on the left side of the figure and the rotor blade is in the moving zone on the right side. The two mesh zones join along a planar interface. As the rotor zone moves the rotor zone mesh points slide along this planar surface. In general, mesh points of the rotor zone do not coincide with mesh points of the stator zone at the "sliding boundary"–that is, mesh lines are discontinuous across the sliding boundary. This sliding boundary approach has been used by Rai[5], Peery[6], and others.

The calculation of fluxes at a cell face requires flow field data at the cell centers of two cells on both sides of the face. Unfortunately, there are no real cells for zone 1 that exist on the right side of the sliding boundary from which the flux at cell faces on the sliding boundary can be computed. In order to calculate fluxes at the cell faces of zone 1 on the sliding boundary flow, field data from within zone 2 is interpolated to fictitious boundary cells of zone 1 that overlay zone 2 as shown in Figure 6. Transferring data to these boundary cells is difficult for two reasons. First of all, the cell centers of the boundary cells do not coincide with the cell centers of cells in zone 2 and, therefore, require a three-dimensional interpolsation. Secondly, in flow calculations of practical use each zone has a large number of mesh points. It is not usually possible to have both zones in RAM at the same time.

[3]A separate data storage location for the patch is required to handle the situation where two large zones are to be coupled on a machine with limited memory. In this case, both zones may not fit in main memory at the same time, while a zone and a patch would.

Transferring flow data across the sliding boundary was performed in a two-step process as shown in Figure 7. The first step consists of interpolating (using tri-linear interpolation) flow field data from mesh cells in zone 2 onto the cell centers of a set of cells that correspond to the locations of the fictitious boundary cells of zone 1. This set of cells is called a 'patch.' The second step consists of simply copying the patch data onto the corresponding boundary condition cells of zone 1. Patch data is stored separately from zone data allowing the transfer of interzone boundary condition data between zones with only one zone resident in RAM at a time.

Since the interpolation must be repeated every time step that the rotor zone moves, the interpolation can become a major part of the computational work. In order to do the interpolation it is first necessary to detemine which cell of zone 2 contains the center of each one of the patch cells of zone 1. An exhaustive search for each patch cell would be very expensive. The search procedure starts by first searching in the neighborhood of the cell in zone 2 in which the previous patch cell was found. If this initial search fails then the following seach procedure is used. The cells of zone 2 are subdivided into subzones containing nearly an equal number of cells each. Each of these subzones is is in turn subdivided again into sub-subzones. The number of subzones and sub-subzones is variable. The range coordinate values in each coordinate direction of each subzone (and each sub-subzone) is calculated at the beginning of the time step prior to the interpolation process. A quick search is made to determine in which subzone(s) the patch cell might be contained. Next each sub-subzone of each selected subzone is examined to determine if the patch cell lies within it. Finally after determining which sub-subzone contains the patch cell a cell-by-cell search is performed. This procedure reduces the time substantially for interpolation as compared to an exhaustive search.

During the Phase I effort a form of trilinear interpolation was used. The interpolation is based on the cell centered coordinates so that first step is to calculate these coordinates from the cell corner point coordinates. For convenience in the following discussion, we call the point to which we are interpolating 'P'. The next step is to determine which cell the point P is in. The final step is to actually perform the interpolation. We will discuss the last step first and, to simplify the discussion, we will consider the two-dimensional case.

The interpolation is developed using a geometric interpretation. As shown in Figure 8, each quadr lateral cell in the mesh may be divided into a pair of triangles in two different ways. The first pair of triangles, the A-triangles, are obtained by drawing a line between nodes (i, j) and $(i + 1, j + 1)$. The second pair of triangles, the B-triangles, are obtained by drawing a line between the nodes $(i, j + 1)$ and $(i + 1, j)$. If the point, P, is within the quadralateral it is within one of the A-

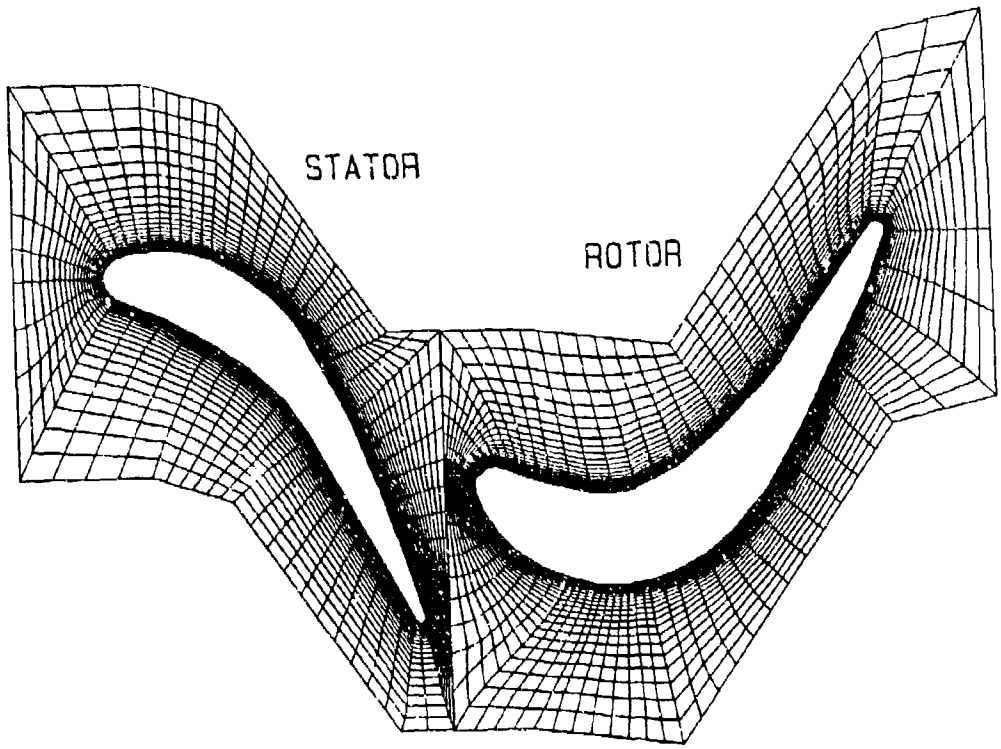


Figure 5: Midspan Mesh for Turbine Rotor/Stator calculation

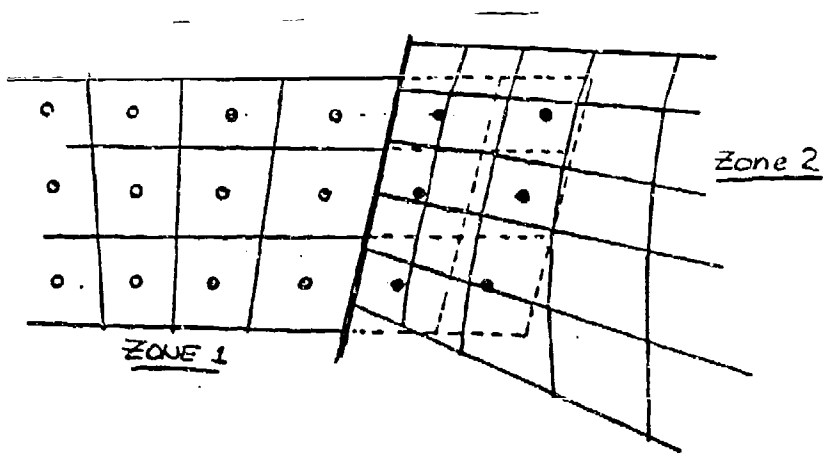


Figure 6: Fictitious Boundary Cells of Zone 1 Overlap Sliding Boundary

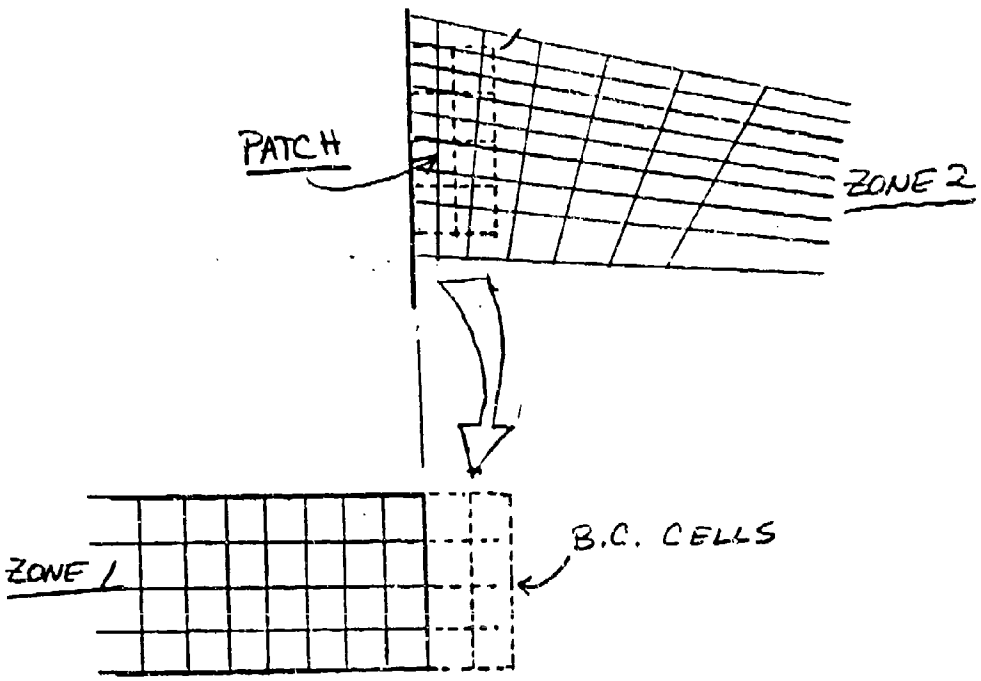


Figure 7: Two-Step Interzone Data Transfer Process

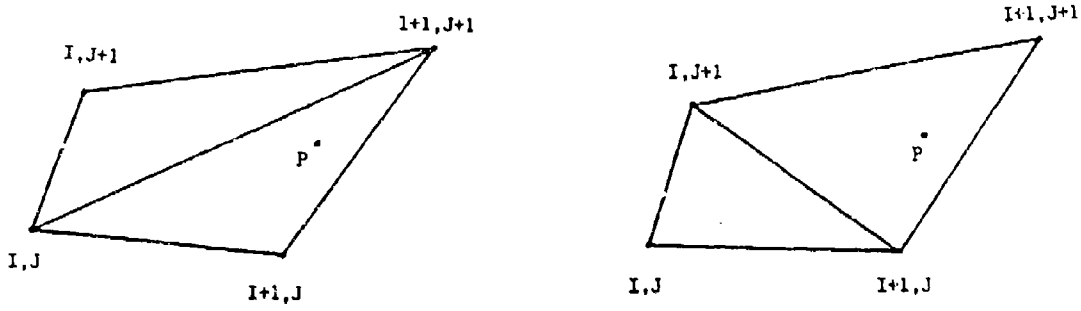


Figure 8: Two possible divisions of a quadralateral cell into triangles

triangles and one of the B-triangles. The first step is to do a linear interpolation on each of the two triangles that P is in. The linear interpolation gives the value at P as a weighted average of the values at the vertices of the triangles. The average of the weights for the two triangles containing P give the weights for the bilinear interpolation.

The test to see if a point is within a triangle is done by considering the areas of the three new triangles created by drawing lines from P to the vertices of the original triangle. As shown in Figure 9, P is outside the original triangle if the sum of the areas of the three new triangles is greater than the area of the original triangle. If P is within the triangle, the weights for the linear interpolation over the triangle are obtained by dividing the area of the triangle into the areas of the three subtriangles. The weight obtained from a given subtriangle corresponds to the vertex opposing the subtriangle.

The search to determine which quadralateral contains P is potentially very costly, especially when the total number of cells is large. To simplify the process the zone is broken into multiple levels of subzones. The search begins by determining which subzone contains P, then which sub-subzone contains P, and so on until the subzone which contains P at the lowest level is found. Each subzone (sub-subzone, etc.) is defined by the smallest horizontal rectangle which envelops the cells contained in the subzone, so the search through the subzones is very efficient. Finally, an exhaustive search is made through the appropriate lowest level subzone

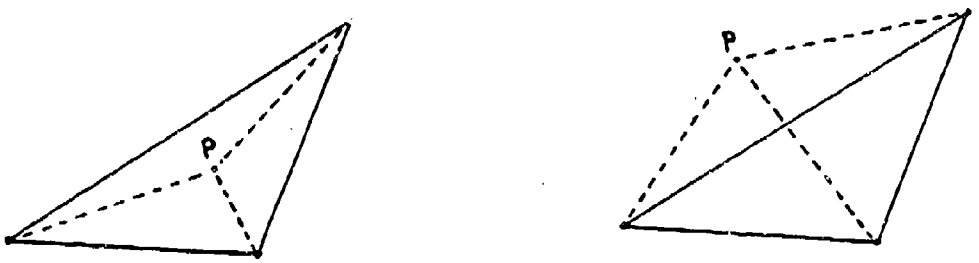


Figure 9: Subtriangles with P within (left) and outside of (right) the triangle

until the actual cell containing P is found. The number of cells in this lowest level subzone (call it the target subzone) is relatively small so the final search is inexpensive. The total cost for this overall search is much less than the cost for a single exhaustive search, especially when the number of mesh points is large.

The development of the trilinear interpolation procedure is similar to the above development of the bilinear interpolation procedure except that the triangles are replaced by tetrahedrons.

5 Turbine Rotor/Stator Calculation

5.1 Approach

As shown in Figure 10, zone 1 extends over one complete period of the stator blades. Zone 2 extends over one complete period of the rotor blades. For this analysis the blade period of the rotor blades was assumed equal to the blade period of the stators. Periodic boundary conditions are appled between faces A and B for each zone. Subsonic inflow boundary of zone 1. Total pressure, total temperature, and flow angle were specified.

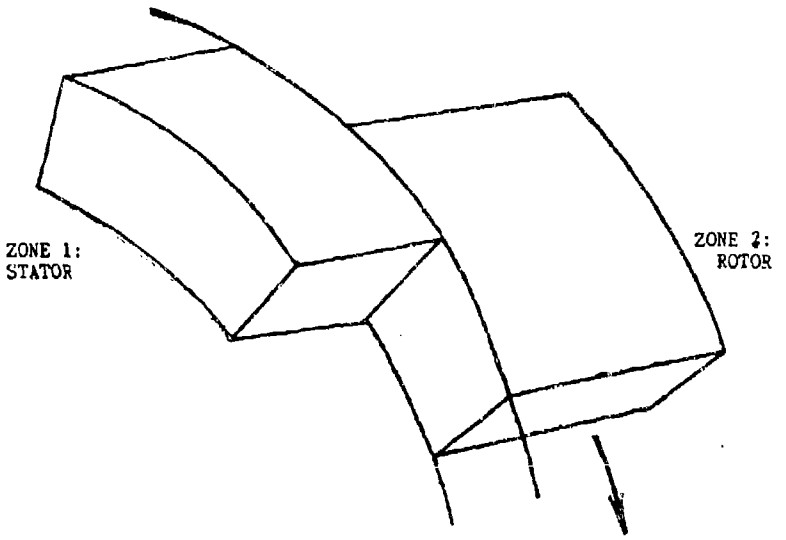


Figure 10: Zonal Configuration for Turbine Rotor/Stator Calculation

As shown in Figure 10 the stator zone and the rotor zone are not circumferentially aligned at the sliding boundary. This presents a problem in setting flow field data in the interzone patches. Setting the flow field data in patch 1 consists of interpolating data from zone 2. Unfortunately, some cells in patch 1 do not lie within zone 2. A typical fix for this problem would be to add some coding to the interpolation routine that would rotate the patch cell forward by one blade period to a corresponding position in zone 2 before performing the interpolation. The interpolated data would then be rotated back to its original position. This method would work, but it would also complicate the structure of the interpolation sub-

routine. Since this rotation would not always be done, a special test would have to be added before the coding that performs the forward and reverse rotations. The interpolation routine would lose its usefullness for other applications and could introduce bugs.

Since one of the intents of this work was to develop an analysis procedure suitable for general flow problems involving solid objects in relative motion, we prefered not to write any coding that is specific to this test problem. That is, no subroutines were written that would be applicable to only the turbine rotor/stator problem. All additional coding in the multi-zone 3D Navier-Stokes analysis must perform a simple function, be of general use, and conform to the layered code structure shown in Figure 11. The procedure for applying interzone boundary conditions across the sliding boundary would have to be built out of simpler basic data operations.

The present multi-zone 3D Navier-Stokes analysis is a result of an effort to develop a layered code structure. The planning of its design required a substantial amount of time. The development of the coding, however, is surprisingly fast because of its logical, uncomplicated code structure. The layered structure requires highly modular programming and clean data paths into and out of subroutines. This allows the subroutine modules to be easily testable and bugs easily traced to offending subroutines. Routines may call routines in the same or any lower layer but not routines in higher layers.

As shown in Figure 11 the highest level of layered structure is the APPLICATION LAYER. This layer consists of a series of subroutine calls. It is almost entirely user definable and contains some local processing for applications specific functions such as inputting control vriables. The subroutine calls from the APPLICATION LAYER are primarily to the TASK LAYER. Example TASK LAYER routines are described below:

CALL LOADZ(IZ)	Check that data for zone IZ is located in the program controlled heap in RAM. If not, locate data and load it into RAM.
CALL SOLVE(IZ)	Apply one cycle of the solution procedure of the type currently assigned to zone IZ, This would be one time-step of the N.S. solution procedure.
CALL UPDATBC(IZ)	Update all boundary conditions on zone IZ including interzone boundary conditions.

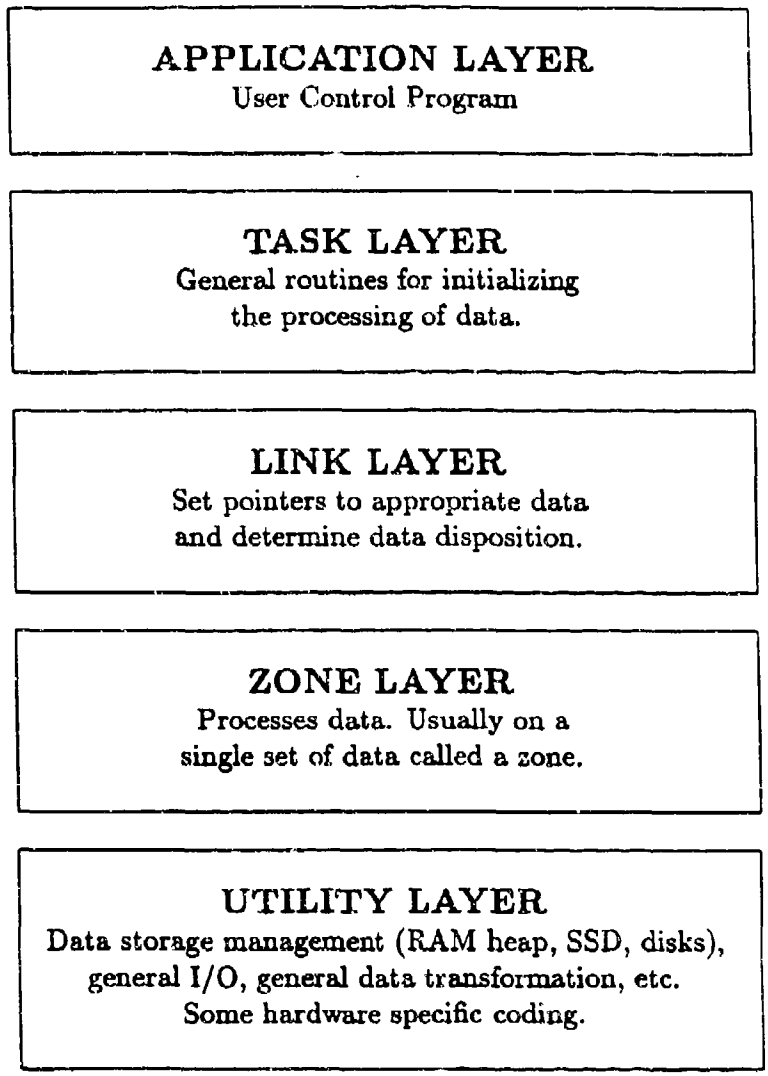


Figure 11: Layered Code Structure of Pilot Code

TASK LAYER routines primarily call LINK LAYER routines. LINK LAYER routines look at various control data to determine the disposition of the data (in RAM, SSD, etc.) and the form of the data, i.e., which solver class (3D N.S., Full-Potential, etc.). The appropriate pointer variables are then passed on to the appropriate routine in the next lower layer for processing.

The routines in the ZONE LAYER process data. This includes integration of a zone of field variables by one time step, setting boundary conditions along zone boundaries (e.g. solid-wall no-slip), and calculating mesh metrics for example. All routines in the ZONE LAYER operate on only a single zone or single set of data at a time.

Data transfer between zones is handled as described previously in a two-step procedure using 'patches.' ZONE LAYER routines exist for: 1) interpolating data from one zone into cells of a patch, and 2) copying patch data into zone boundary condition data. The patch serves as a buffer for temporary data storage. The ZONE LAYER routines can be relatively simple to write, since each routine operates on only one zone at a time. The problems in managing the multi-zone data are handled by routines in the APPLIATION, TASK, and LINK LAYERS.

The lowest layer is called the UTILITY LAYER. The routines in this layer manage the computer resources. The perform operations that are sometimes machine specific. These include 1) initializing the RAM heap, requesting and returning blocks of data to the heap, 2) moving zone and patch data between RAM, SSD, and disks, and 3) generalized input and output of user data.

The procedure for applying the interzone boundary condition is being developed with the layered structure in mind. The layered structure separates the complexities of managing multiple zones of data from the nitty gritty low-level data manipulations. This simplifies the coding greatly, but must be well thought out to have general application. The present layered structure allows the interzone boundary condition to be applied very easily. To do this, the two following basic routines are added to the TASK LAYER (plus the auxilliary lower-level routines that perform the data manipulation).

DUPZON(IZ,IDZ)	This routine creates a duplicate copy of zone IZ and assignes it the zone number IDZ.
ROTZON(IZ,ANG).	This routine rotates the coordinates and vector components about an axis passing through the origin in the Cartesian x-direction.

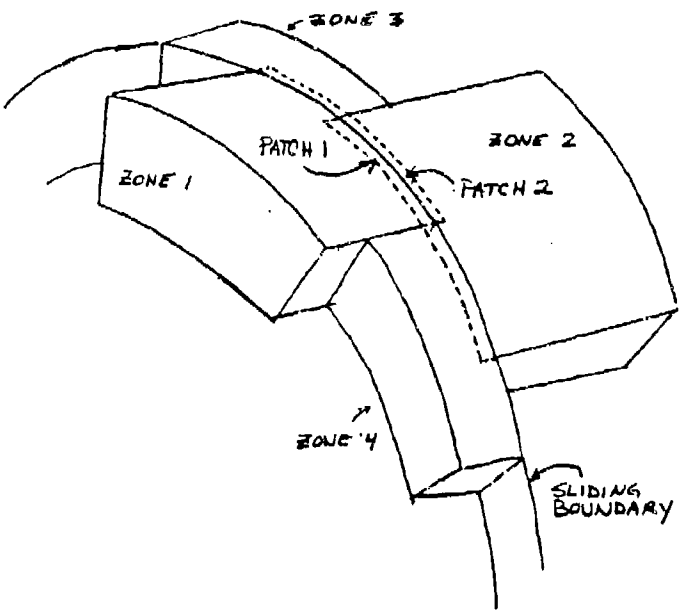


Figure 12: Interzone Data Transfer Between Non-aligned Zones

Setting the interzone boundary conditions across the sliding boundary becomes trivial with the use of the above routines. Consider setting the interzone boundary conditions on zone 1. The first step is to interpolate flow field data from zone 2 into patch 1 (which is associated with the boundary cells of zone 1). Prior to setting the cells in patch 1, a third zone is temporarily created by duplicating zone 2 using the TASK LAYER routine DUPZON. This temporary zone is given the number 3 and rotated counterclockwise one blade period so that it trails behind zone 2 as shown in Figure 12.

Setting data into patch 1 is now straight forward and proceeds as follows. Zone 2 and patch 1 are loaded into RAM. Flow field data is interpolated from zone 2 into the cells of patch 1 that lie within zone 2. Next, zone 2 is moved from RAM to the SSD and zone 3 is loaded into RAM. Flow field data in zone 3 is interpolated into the remaining cells in patch 1 which lie within zone 3. Zone 3 is then deleted. As shown in Figure 12, the process occurs in a similar manner for setting cells in patch 2 for the boundary conditions of zone 2.

The application of periodic boundary conditions on zones 1 and 2 was also performed with the use if simple TASK LAYER routines. Zone 1 is shown in isolation in Figure 13. Data from two planes of internal cells parallel to face A are transferred into patch 4 using a TASK Layer routine called UPDZIP(IP,IZ).

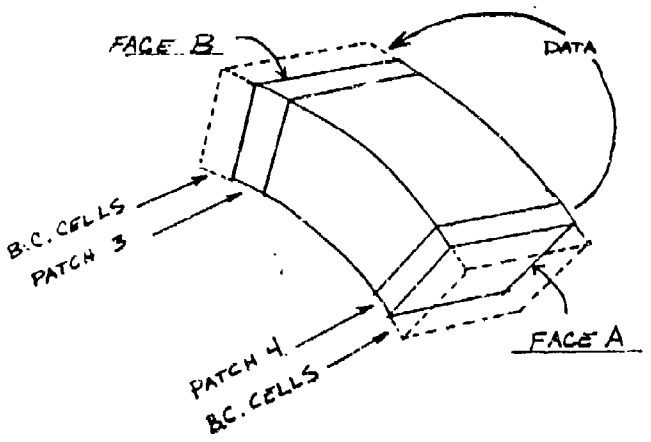


Figure 13: Periodic Boundary Conditions for Stator Zone

UPDZIP(IP,IZ) extracts data from zone 1 (IZ=1) and copies it into patch 4 (IP=4). Another TASK LAYER routine called ROTPCH(IP,IZ,ANG) then rotates patch 4 (IP=4) by an angle of ANG about an axis through the origin in the Cartesian x-direction. Setting ANG equal to the negative of the angular blade period will rotate the patch (and vector components in its data) so that it coincides with the boundary condition cells labeled B in Figure 13. Then using a third TASK LAYER routine called UPDIPZ(IP,IZ) data transferred from patch 4 (IP=4) to the boundary cells in zone (IZ=1).

The above TASK LAYER routines DUPZON and ROTZON can be used in other applications as well. They are now building blocks within a library of TASK LAYER routines. They do not have to be bugged again. Once a large library of TASK LAYER routines have been built, the user can create applications to many flow problems without coding a single low-level routine. This layered approach also reduced the probability of programming errors, since the low-level routines are relatively simple functions that can be easily tested, the complexities of data and control management are confined to the high-level layers, and new applications may require only changes to routines at the TASK LAYER.

A significant contribution of the Phase I work is the demonstration of the benefits of a layered code structure. The development of a sophisicated well-engineered protocol for a layered code structure would be of great usefulness to CFD users and developers of flow analysis programs and other physical process simulation

programs.

This layered approach has a number of benefits: 1) it greatly reduces probability of programming errors, 2) substantially reduces programmimg time in applying codes to new problems, 3) has a potential for becoming a programming standard allowing networked users to benefit from other users' programming (i.e. share routines), and 4) allows researchers interested in algorithm development a convenient environment to conduct research wothout spending time developing a complete program.

5.2 Results of Calculations

The above code is being applied to the solution of a turbine rotor/stator interaction problem. Geometry is based on the rotor/stator model tested experimentally by Dring et al. [4]. Calculations of this flow have been presented by Rai[1,3]. The model tested experimentally by Dring had 22 stator blades. Calculations of this flow have been presented by Rai[1,3]. The model tested experimentally by Dring had 22 stator blades and 28 rotor blades. A calculation using the model geometry would require that 11 stator blades and 14 rotor blades be modeled before periodic boundary conditions could be used. To mimimize the computational expense we make the same assumption as Rai[3] and model the case with 22 rotor and 22 stator blades by enlarging the rotor blade. No attempt is being made to model the rotor blade tip effects.

Two calculations were performed. The first was with the rotor blade stationary and the second was with the rotor blade moving. In both calculations the mesh lines were discontinuous across the boundary between the rotor and stator zones, and the general sliding zonal boundary condition was used. The first calculation, with the rotor fixed, provided an excellent test of the zonal boundary condition, without the added complexity of the moving mesh.

The mesh used for the stationary rotor calculation is shown in Figure 14. An O–mesh is used for both the stator and the rotor. The dimensions of the mesh are 66x23x4 for the stator zone and 74x23x4 for the rotor zone. The initial conditions for the calculation were a uniform axial flow of 100 m/s. The flow at the inflow boundary was specified as axial with a stagnation pressure of 1.0 MPa and a stagnation temperature of 1000.0 degrees Kelvin. The outflow pressure was specified

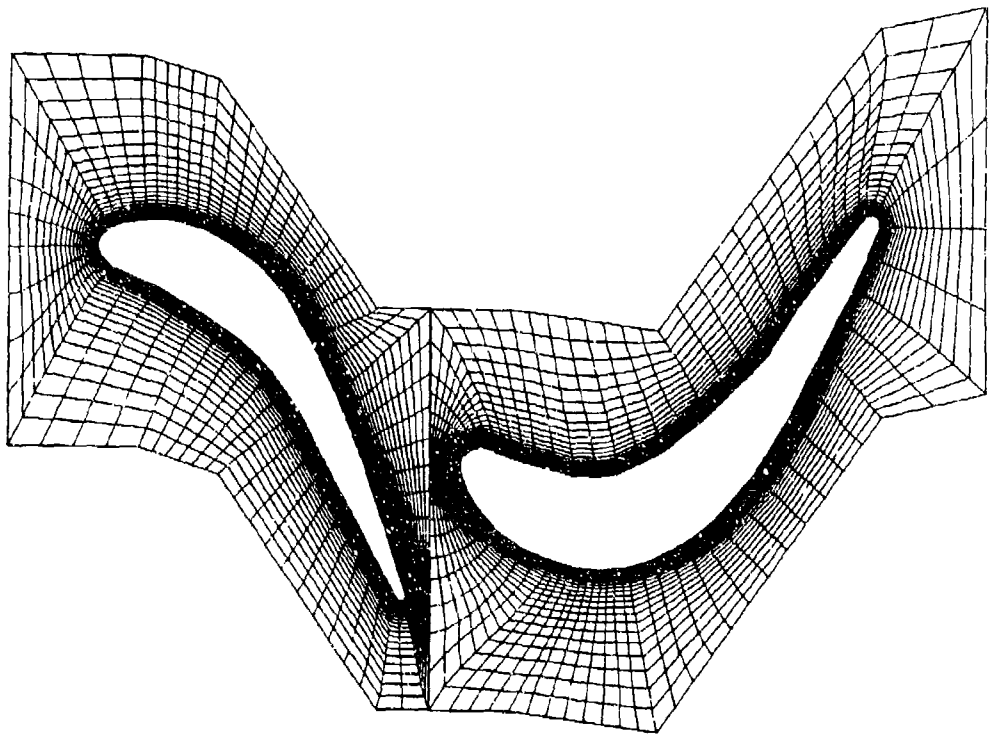


Figure 14: Mesh used for stationary rotor calculation

as 0.896 MPa. The solution was obtained after 8100 explicit time steps [4]. At this point the L2–norm of the residual had dropped four orders of magnitude. The pressure contours at the median radial surface (midspan) are shown in Figure 15, the velocitiy vectors are shown in Figure 16, and the streamlines are shown in Figure 17. These results are qualitatively correct.

The mesh used for the moving rotor calculation is shown in Figure 18 at three times during the cycle. A coarse mesh was used for this calculation because of limited computer resources. Shortly after the last time in Figure 18 the rotor zone is rotated back [5] and the solution is continued. The velocity vectors at these three times are shown in Figure 19 and the streamlines at the first two times are shown in Figure 20.

[4] Local time stepping was used to accelerate the convergence.

[5] The physical interpretation is that the calculation proceeds with the next rotor in line.

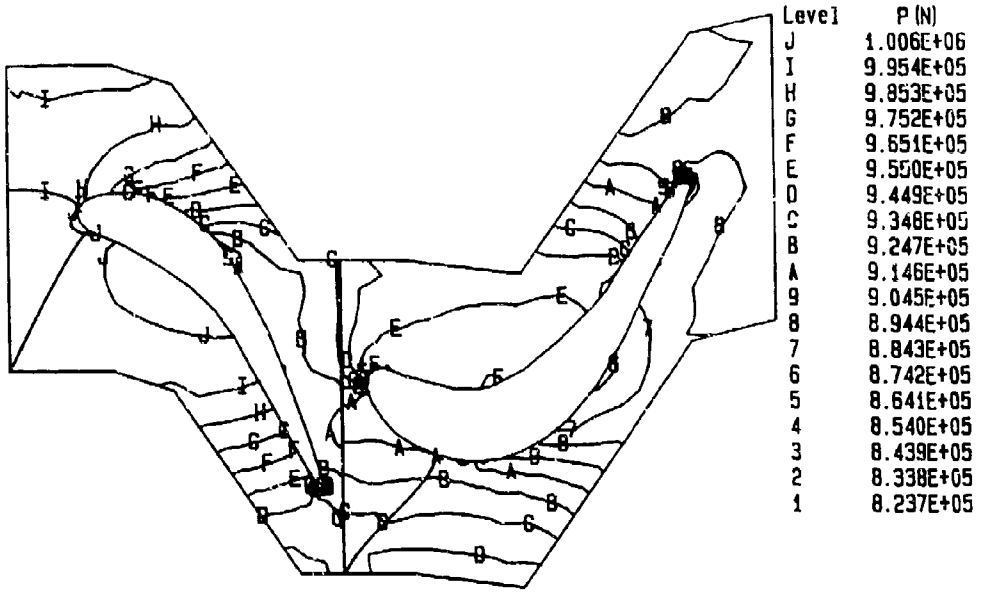


Figure 15: Pressure contours from stationary rotor calculation

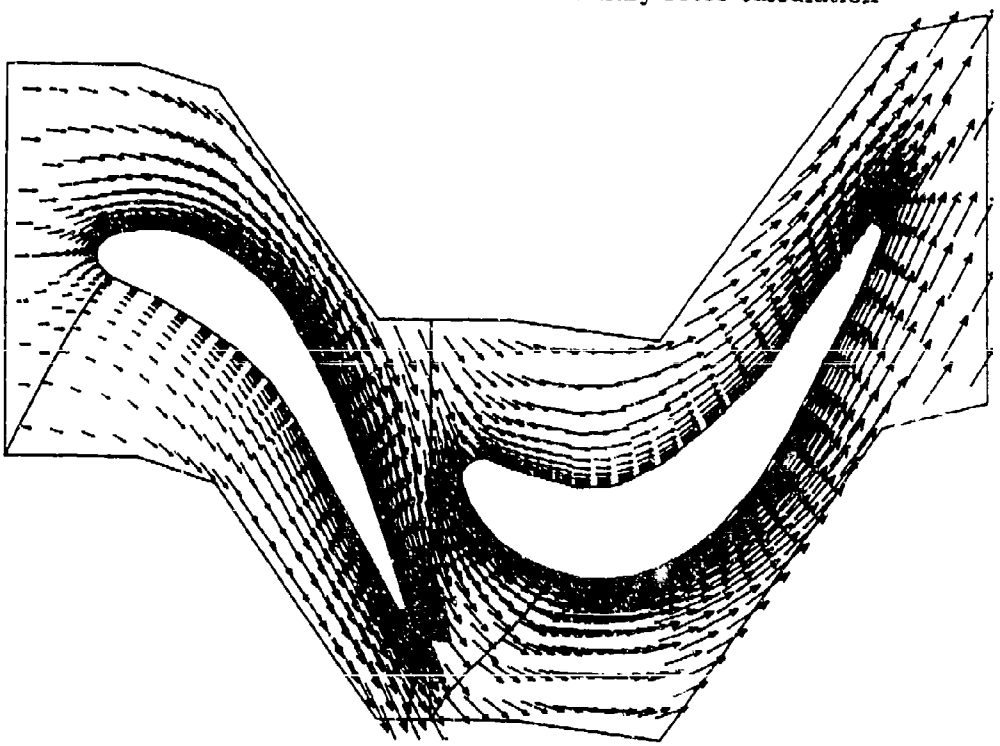


Figure 16: Velocity vectors from stationary rotor calculation

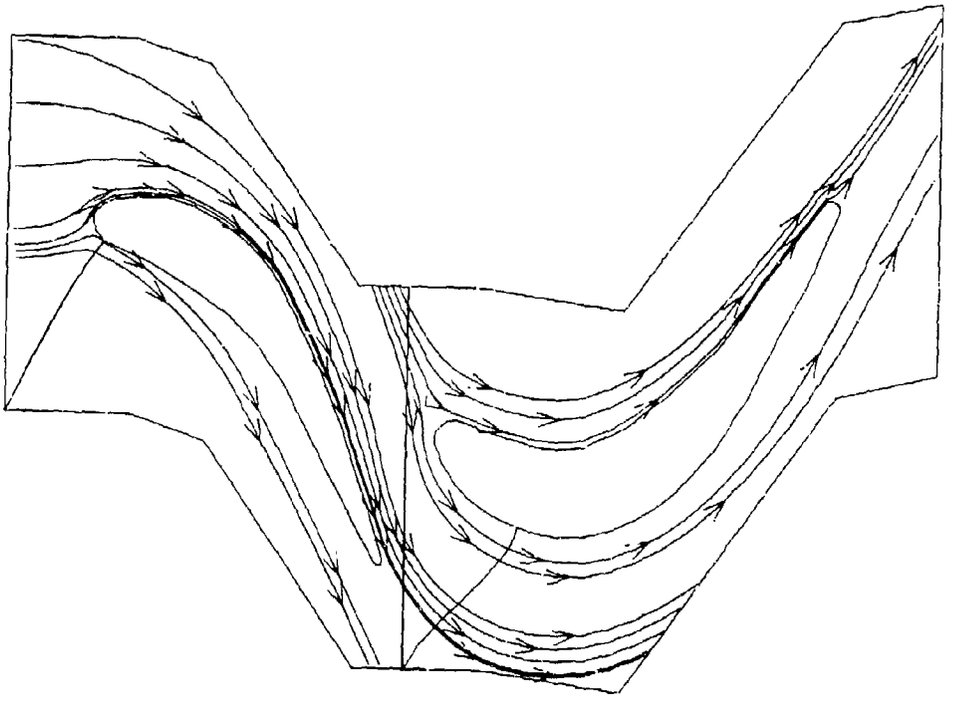


Figure 17: Streamlines from stationary rotor calculation

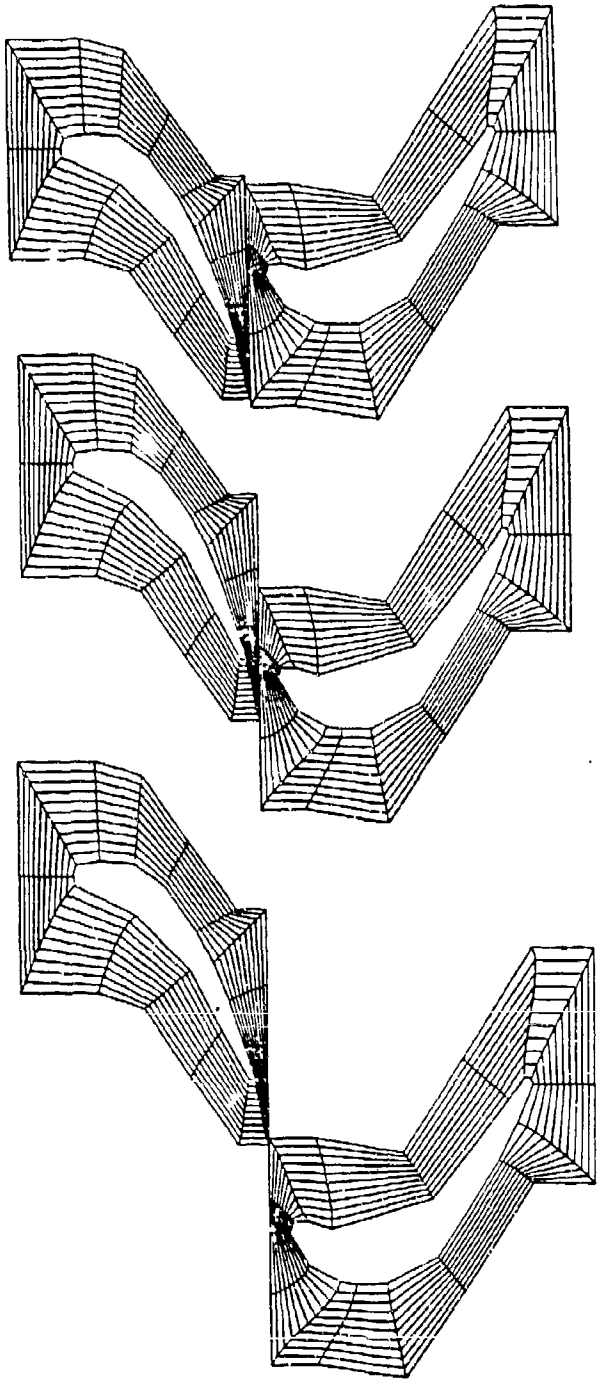


Figure 18: Turbine Rotor/Stator calculation — coarse midspan meshes at three times during the calculation

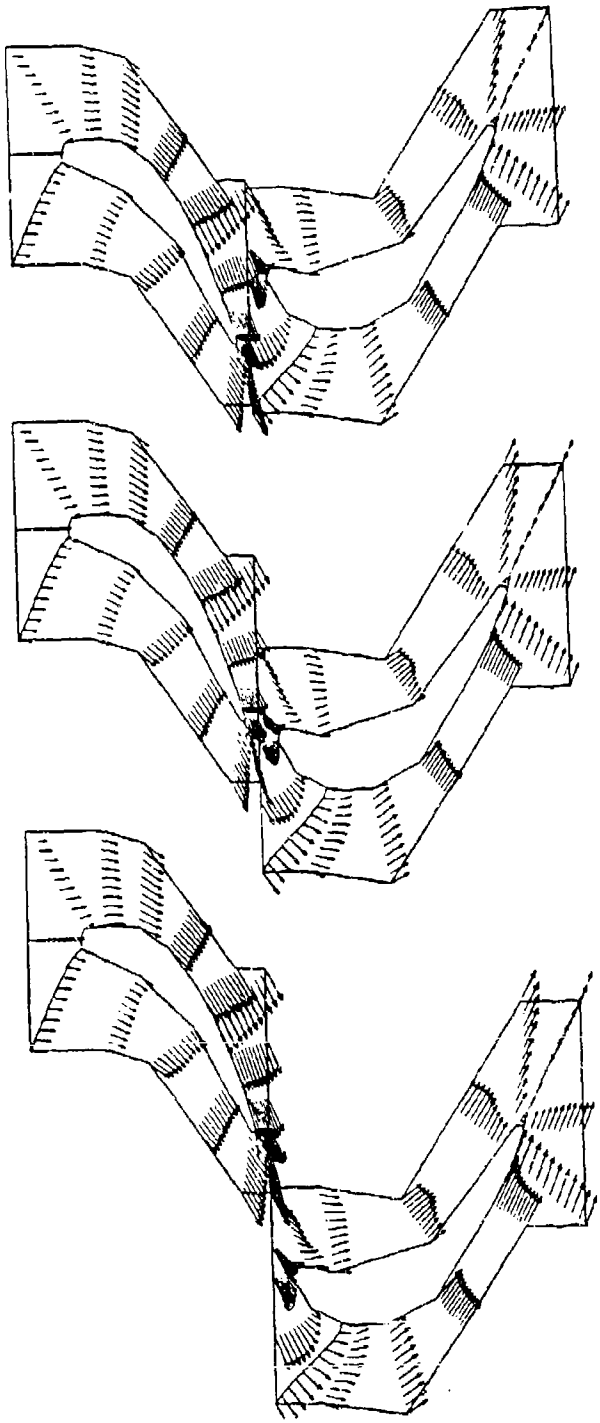


Figure 19: Turbine Rotor/Stator calculation — midspan velocity vectors at three times during the calculation

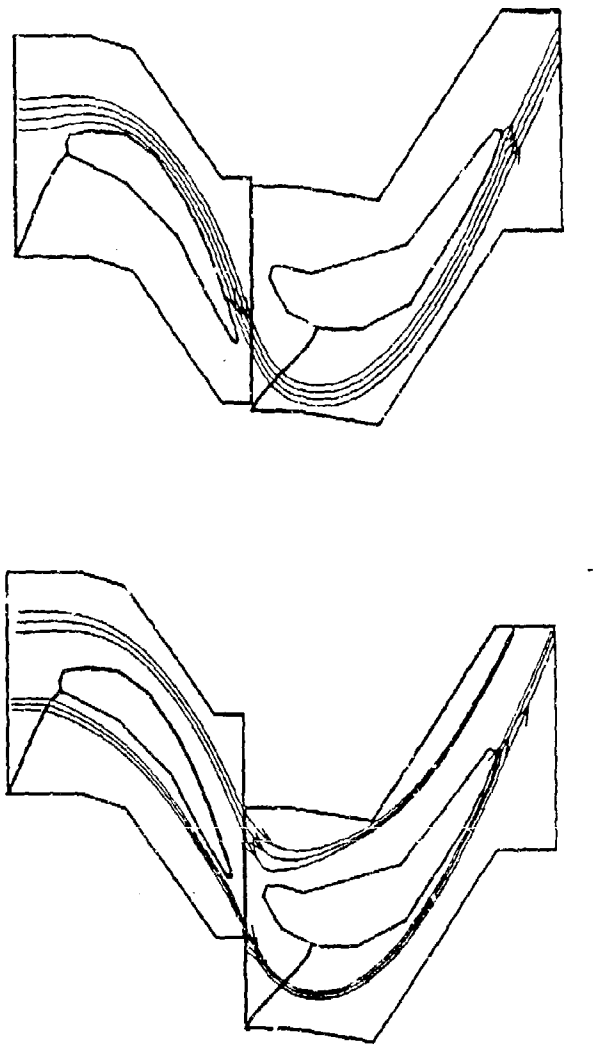


Figure 20: Turbine Rotor/Stator calculation — midspan streamlines at two times during the calculation

6 Conclusions

The goal of the Phase I effort was to demonstrate the feasibility of calculating flows with bodies in relative motion. With this goal in mind, the results of the Phase I effort are encouraging. It has shown the benefits of developing a layered software structure, and demonstrated the feasibility of the coupling procedure for interzone boundary conditions between zones in relative motion. It is felt that the procedures developed during Phase I can form the basis of a versatile code for calculating complex flows containing bodies in relative motion.

7 Recommendations

Since the primary objectives of the Phase I work were achieved, the following recommendations are made for follow up work. The ultimate goal of this effort is to develop a versatile computer program for calculating complex flows containing bodies in relative motion.

1. Software Architecture: Further study is needed to determine the optimal software architecture. The objective is to develop a software architectural plan for building an integrated modular analysis system that is capable of efficiently modeling the complex physical interactions which occur when neighboring bodies are in relative motion. The data structure of the analysis system will be zonal in nature. Each zone may have a different solver class and, for that matter, a completely different data structure. For instance, the zones used to model the fluid dynamics (Navier-Stokes, Euler, full-potential equations) may have a structured grid (a grid with a logical i,j,k ordering), and the zones used to model structural dynamics may have an unstructured finite-element grid.

2. Multiple Solvers: There are many advantages to incorporating many dissimilar solvers into a single code. The engineer needs to learn only one user interface and the various physical models can share resources such as plotting routines, I/O routines, linear algebra routines, etc. In addition, it facilitates the coupling of different physical models, such as a fluid dynamics and structural dynamic solvers. The successful integration of widely differing solution procedures into a single program requires careful consideration of the software architecture, as described in Section 5. A layered software architecture, such as that developed during Phase I, is a likely candidate for isolating the difference between solvers.

3. Zonal Coupling: The next step is to couple the different solvers described above. This would allow complex physical interactions, such as fluid/structural interaction, to be studied. Again, the goal would like to keep the program as versatile as possible, so that it would not be limited to any one category of problems.

Amtec Engineering, Inc. feels that the development of such versatile analysis software would benefit the U.S. Army and the engineering community as a whole.

References

[1] Anderson, D.A., Tannehill, J.C., and Pletcher, R.H. *Computational Fluid Mechanics and Heat Transfer*, McGraw-Hill, 1984.

[2] Baldwin, B. and Lomax, H., "Thin-layer Approximation and Algebraic Model for Separated Turbulent Flows," AIAA PAper No. 78-257, Jan. 1978.

[3] Steger, J.L. and Warming, R.F., "Flux Vector Splitting of the Inviscid Gasdynamic Equations with Applications to Finite-Difference Methods," *Journal of Comp. Phys.*, Vol. 40, pp263-293, 1981.

[4] Pulliam, T.H. and Seger, J.L., "On Implicit Finite Difference Simulations of Three-Dimensional Flows" AIAA Paper No. 78-10, Jan. 1978.

[5] Rai, M.M. "Navier-Stokes Simulations of Rotor-Stator Interaction Patched and Overlaid Grids," AIAA Paper No. 85-1519, July 1985.

[6] Peery, K.M., Ponten, B.D., and Roberts, D.W., "Simulation of Unsteady Two-Dimensional Inviscid Flow Fields Around Geometrically Complex Objects," AIAA Paper No. 85-1273.

[7] "Unsteady Three Dimensional Navier-Stokes Simulations of Turbine Rotor-Stator Interactions Including Tip Effects," AIAA/ASME/SAE 23rd Joint Propulsion Conference, San Diego, CA, June 1987.

[8] Dring, R.P., Joslyn, H.D., Hardin, L.W., and Wagner, J.H., "Turbine Rotor Stator Interaction," *J. of Engrg. for Power*, Vol. 104, Oct. 1982.

[9] Chima, R.V., "Development of an Explicit Multigrid Algorithm for Quasi-Three-Dimensional Viscous flows in Turbonmachinery," NASA TM-87128, 186.

[10] Davis, R.L., Ni, R.H., and Carter, J.E., "Cascade Viscous Flow Analysis Using the Navier-Stokes Equations," AIAA Paper No. 86-0033, Jan. 1986.

[11] Pullium, T.H. and Steger, J.L. "On Implicit Finite Difference Simulations of Three Dimensional Flows," AIAA Paper No. 78-10, January 1978.