THREE-DIMENSIONAL SOLUTION OF COMPRESSIBLE POTENTIAL FLOW IN TURBOMACHINERY BY A FINITE ELEMENT METHOD

by

Doğan Güneş

B.S. in M.E., Boğaziçi University, 1978
M.S. in M.E., Boğaziçi University, 1980

Submitted to the Institute for Graduate Studies in Science and Engineering in partial fulfillment of the requirements for the degree of

Doctor

of

Philosophy

Boğaziçi University

1983

To the Memory of my Father
KEMAL GÜNEŞ

ABSTRACT

The present study is concerned with the analysis of three-dimensional, compressible flows in axial-, radial-, or mixed-flow turbomachinery under the limitations of steady and potential flow conditions. The basic approach is to isolate a single blade row (which may be stationary or rotating at a constant speed) and consider the flow in one of the blade passages as representative of the total flow through the entire row of blades. It is assumed that the fluid is inviscid and enters this "characteristic" passage with uniform entropy, uniform total enthalpy and zero vorticity. Under these conditions, Kelvin's circulation theorem ensures irrotationality of the absolute flow throughout. The analysis is restricted to subsonic flows which may have local suporsonic spots. The fluid is either incompressible or assumed to be accurately represented by the perfect gas law.

The analysis begins with the development of the classical velocity potential formulation of the problem stated above. An equivalent variational formulation is then described. This formulation incorporates a quasilinearization concept which leads to iterative solution. Density distribution is assumed to be given by a previous solution and therefore has no variation. The problem of three dimensional, compressible, potential flow in turbomachinery is thus reduced to the determination of the absolute velocity potential distribution which minimizes an equivalent functional in

the solution domain with appropriate boundary conditions.

A computer code has been developed to solve this problem based on the finite element analysis presented in this thesis. The solution domain is discretized by using hexahedral superelements each composed of six ten-node tetrahedral elements enabling quadratic interpolation of velocity potential. The code offers the flexibility of using a combination of subparametric and isoparametric elements which provides high accuracy at reasonable cost in the treatment of turbomachinery flows that exhibit complicated design features.

Applications of the code to the Gostelow cascade, an experimental turbine stator, the first stage stator and rotor of an electric utility axial-flow turbine and finally to a mixed-flow turbine rotor are presented. The validity of the code is established by comparing the results with the exact solution, experimental data and calculations by other numerical methods.

Ö Z E T

Bu çalışmada turbomakinalardaki üç-boyutlu sıkıştırılabilir akışların daimi ve potansiyel akış şartları altındaki analizi konu edilmiştir. Temel yaklaşım olarak izole edilmiş bir pala sırasındaki toplam akış karakteristik bir pala kanalındaki akış ile temsil edilmiştir. Akışkanın viskozitesiz olduğu ve karakteristik kanala üniform entropi, üniform toplam entalpi ve sıfır çeviriyle girdiği kabul edilmiştir. Bu şartlar altında Kelvin sirkülasyon teoremi mutlak akışın döngüsüz olmasını temin eder. Sunulan analiz yerel ses hızı üstü noktaları olabilen ses hızı altı akışlar için geçerlidir. Akışkan sıkıştırılamaz veya mükemmel gaz olarak kabul edilmiştir.

Yukardaki problemin klasik hız potansiyeli formülasyonu geliştirilmiş ve eşdeğer varyasyonel formülasyon elde edilmiştir. Bu formülasyon ardışık yaklaşım işlemi gerektiren bir yaklaşık-doğrusallaştırma yöntemi içermektedir. Yoğunluk dağılımı bir önceki çözümden elde edildiği için varyasyona sahip değildir.

Bu problemin çözülmesi için bu tezde sunulan sonlu elemanlar analizine dayanan bir bilgisayar programı geliştirilmiştir. Çözüm bölgesi her biri altı adet on düğümlü tetrahedral elemanlardan oluşan heksahedral süperelemanlara bölünmüştür.

Geliştirilen programın Gostelow kaskatına, bir deneysel türbin statoruna, bir elektrik santralı eksenel türbininin birinci kademe stator ve rotoruna ve son olarak karışık-akışlı bir türbin rotoruna uygulamaları sunulmuştur. Programın geçerliliği sonuçların kesin çözüm, deneysel veriler ve başka sayısal yöntemlerin hesaplarıyla kıyaslanması ile gösterilmiştir.

ACKNOWLEDGEMENTS

I am indebted to Doç.Dr. Muhsin Mengütürk for forwarding all the necessary moral and material support which has been a continuous supply of impulse in my graduate studies. It has been a privilege for me to work with a person who has the rare ability to synthesize available data, present knowledge and theory into a masterpiece. I have always admired and appreciated the patience, endurance and goodwill that he has shown in shaping an impatient and disorganized student into a Ph.D. candidate. Words fail to describe my gratitude to "Hocam" who has first introduced me to the world of scientific research and has been a friend and a colleagu as well as a teacher.

The various projects I have been involved in during my graduate studies have served as invaluable guides in directing my scientific curiosity into fruitful scientific experience. The friendly atmosphere and the pleasure of scientific accomplishment have turned the years I have spent in the Mechanical Engineering Department into memorable days of my carrier.

I would like to take this opportunity to extend my appreciation to my friends, especially to Ruhi Kaykayoğlu, Halil Mustafa, Ali Cemal Benim and Süleyman Gökoğlu, for their being along in heart.

My thanks to Dr. Vahan Kalenderoğlu and Mr. Zafer Erbaş for their encouragement.

Support of Boğaziçi University Computer Center is gratefully acknowledged.

The last but not the least I would like to express my deepest gratitude to my family who have supported me throughout.

Doğan GÜNEŞ

TABLE OF CONTENTS

LIST OF SYMBOLS

C	Blade chord
C'	Axial blade chord
C_p	Specific heat at constant pressure
C_P	Pressure coefficient
E	Total number of elements
$\vec{e}_\theta$	The unit vector in the tangential direction
F_L	Lift force perpendicular to the blade chord
H	Stagnation enthalpy
h	Enthalpy
I	Rothalpy
I_S	Surface integral part of the functional (Eq. (43))
I_Ψ	Volume integral part of the functional (Eq. (44))
$[K]$	System property matrix
$[k]$	Element property matrix
L	Reference length
M	Total number of nodes
M_{in}	Reference Mach number on the inlet plane
m	Number of nodes in an element
$\{N\}$	Quadratic interpolation function vector
$\vec{n}$	Unit vector normal to the surface

P	Pressure
{Q}	Global nodal values of the velocity potential
{q}	Element nodal values of the velocity potential
R	Gas constant
{r}	Element right-hand side vector
r	Radial coordinate
S	Denotes surface
s	Entropy
T	Temperature
t	Time
[T]	Transformation matrix
V	Absolute velocity
W	Relative velocity
x	Cartesian coordinate
x'	Distance along blade chord
y	Cartesian coordinate
z	Axial coordinate

Subscripts

d	Denotes those portions of the boundary surface which are characterized by Dirichlet type boundary conditions
e	Denotes the element under consideration
ex	Denotes quantity on the exit plane
in	Denotes quantity on the inlet plane
n	Denotes those portions of the boundary surface which are characterized by Neumann type boundary conditions
∀	Denotes volume

Greek Symbols

α Absolute flow angle

γ Specific heat ratio

δ First variation

θ Tangential coordinate

K Functional given by Eq. (51)

$\mathcal{L}$ Differential operator defined by Eq. (38)

λ Prerotation (or inlet swirl)

ξ Blade stagger

Φ Absolute velocity potential

ρ Density

ω Angular velocity

LIST OF TABLES

LIST OF FIGURES

I. INTRODUCTION

Ten years after the outset of the so-called "energy crisis" in 1973, problems associated with the efficiency and reliability of turbomachinery are becoming more generally recognized as crucial restraints on the power plant and propulsion system designs.

Increasing cost of energy generation has imposed new demands on turbomachinery design for the following main reasons.

i. Stringent economic constraints necessitate efficient and durable turbomachinery.

ii. To meet the increasing power demand and yet limit the capital cost to a reasonable level, turbines need to be designed to operate at high inlet temperatures (in excess of $2000^{o}C$) and at high Mach numbers.

iii. The technological shift from the petroleum to the more abundant fossil fuels escorts problems of erosion, deposition and corrosion due to particle laden gas. The turbomachinery of the future must be designed to avert these problems.

Rapid technological advances made during the past decade have brought into sharper focus the need for a comprehensive and competent treatment of the aerodynamic aspects of turbomachinery design. Although the present theoretical and numerical means fall short of enabling computation of complex viscous three-dimensional effects with certainty, significant advances have been realized in inviscid flow calculation techniques.

The majority of the current inviscid flow numerical analyses are two-dimensional and applied on one of the two mutually orthogonal relative streamsurfaces of Wu [17]. Computations on the "blade-to-blade relative streamsurface (S1)" and the "hub-to-shroud (meridional) relative streamsurface (S2)" are effected with streamline curvature, finite difference or finite element formulations. The former two techniques have a long time of experience behind them and the commonly cited advantages and disadvantages of these techniques are now clearly known. The application of the finite element approach to turbine flow calculations is very recent and the comparative merits of this technique are yet to be demonstrated.

The demand for improved turbomachinery design has recently brought the three-dimensional flow problem into the agenda. Instead of attacking the full three-dimensional problem, initial attempts have usually comprised an artificial superposition of the two-dimensional solutions obtained on the S1 and S2 surfaces without iteration. Nevertheless, attention seems to be finally focusing on the full three-dimensional flow. The classical streamline curvature and the finite difference methods that have proven to be so powerful on Wu's relative

streamsurfaces are now being implemented to this case. In this regard there is substantial interest for developing three-dimensional finite element solutions.

Considering the demand for three-dimensional turbomachinery flow models together with the deservedly increasing popularity of the finite element methods in many areas of engineering, the present thesis has been concerned with calculation of three-dimensional compressible potential flows in axial-, radial-, and mixed-flow turbomachinery. The basic approach involves a finite element formulation governing the velocity potential of the absolute flow. The intent has been to develop, on the basis of this formulation, a computer code primarily for use in compressors, turbines and fans.

The work consists of five chapters. The vast effort which has been expended in development of turbomachinery flow models has warranted the presentation of a literature survey as Chapter 2, following the present chapter. The intent here has been to emphasize the present state of the art in turbomachinery calculations. The fundamental aspects of the three-dimensional potential flow theory in turbomachinery are described in Chapter 3 which also treats in sequence the variational formulation used, discretization of the flow geometry, development of the element and the system equations and finally description of the computer code evolved. Chapter 4 is concerned with the results of the test cases considered for verifying and demonstrating the applicability of the code in a variety of turbomachinery flows. Chapter 5 presents the ensuing conclusions and seeks to highlight the directions in which further research may be potentially most fruitful.

II. BACKGROUND

It is true that scientific research is usually tied closely with technological progress. The former being a prerequisite for the latter is in turn stimulated by technological advances to generate new ideas for research which leads to further advances in technology. This cyclic cause-and-effect relationship is a major driving force in science and technology.

The intention of the present chapter is to describe the present state of the art in turbomachinery flow calculations. To show how the need for such calculations has gradually developed, the reader is first taken on a hurried tour starting from the early attempts that resulted in nothing but rotating toys, past the mostly trial-and-error efforts of the late 19^{th} century engineers who endeavored to render turbines competitive with piston engines, to our own century of dazzling technological advances which have helped to establish turbomachinery as the prime mover of electric utility and aircraft industries. Research on flow calculations stimulated by these advances in turbomachine technology is then narrated in chronological order. The accelerating pace of research in this area is laid out in some detail through efforts which have resulted in the development of the conformal mapping, singularity, streamline curvature, finite difference and finite element methods.

The essential features of these methods are briefly explained. Throughout, the reader is informed as to how the sophistication of solution methods has proliferated from the simple one-dimensional theory of Euler towards complex three-dimensional analyses. It is clear that turbomachine technology and research in aerodynamics have proceeded hand in hand inspiring and shaping the development of one another. Finally, it is emphasized that, although the two-dimensional and quasi-three-dimensional methods of solution have provided a major contribution to the development of modern turbomachinery, the current needs dictate analysis of the complete three-dimensional flow.

History of turbomachinery can be traced back to Hero's rotating sphere and gas turbine (Alexandria 120 B.C.) neither of which produced any power, but were only used as toys. Although during the succeeding centuries, water wheels of the impulse and gravity-fed types were built for use in grinding mills, water supply or mining, the movement toward modern turbomachinery really started in eighteenth century. Swiss mathematician Leonhard Euler published his application of Newton's law to turbomachinery, now universally known as Euler's equation, in 1754, and thereby immediately permitted more scientific approach to design than the previous cut-and-try methods.

The first steam turbine to have a major impact on the engineering world was that of Charles Parsons, who made a multistage axial-flow reaction turbine giving 10 hp at 18,000 rpm in 1884. In France, Auguste Rateau experimented with a de Laval turbine in 1894, and developed the pressure-staged impulse turbine by 1900. Charles G. Curtis patented in 1896 the velocity-staged turbine, similar to a two-stage de Laval turbine.

Developments such as the regenerative feed heating (1920) and the reheat cycle (1925) led to more efficient and powerful (up to 500 MW) steam turbines (1958).

Development of the gas turbine lagged considerably behind that of the steam turbine, because the former was hampered by the unavailability of an efficient enough compressor. For many decades, the losses during compression, in particular, were just too high for positive work to be given at the temperatures turbines of the day could withstand. Thus, the difficulty of making efficient compressors led to the failure of early gas turbine schemes. The first gas turbine activity to result in a workin engine was started by Charles Lemale, who was granted a patent for a constant pressure (Brayton or Joule) cycle in 1901. This engine had an efficiency of barely 3.5 percent. The blades were probably stalled for the compressor was built before the first airplane had flown and at a time when aerodynamics was at its infancy.

It was not until the 1920's that compressors of high enough efficiency to be useful in a gas turbine began to be developed. The improvement in efficiency is attributable to the use of growing knowledge of aerodynamics of airfoils and airplanes.

Initially, turbines were designed on the assumption of one-dimensional flow through the blade passages. Euler's equation was used to relate the inlet and outlet velocities to the output power. Steam turbines designed by these methods were reasonably successful, because in a flow with decreasing pressure, there is no tendency for the boundary layer to separate. In the compressor, on the other hand, the positive pressure gradients in the passages between the blades tend to cause

separation. The development of two-dimensional potential flow theories in conjunction with boundary layer calculation techniques enabled prediction of the pressure distribution and the onset of separation, and thus allowed compressor blades to be designed for higher efficiency.

In 1930's a number of workers began to apply aerodynamic theory to the three-dimensional flow. It was assumed that between the blade rows the radial velocities and accelerations would quickly vanish and a condition of "radial equilibrium" would be established. The axial velocity would then be invariant with radius and the tangential velocity would vary inversely with it. Blades designed to suit these conditions were termed "free-vortex" blades. The radial equilibrium theory was first used by Whittle in 1930. Whittle designed a turbojet which had "free-vortex" blades and demonstrated that radial equilibrium outside the blade row would increase the efficiency.

The modern successful gas turbine engine resembles those constructed by Aurel Stodola (1936) and Adolf Meyer (1939). Adolf Meyer's gas turbine was also fitted in a locomotive (1942). Hence, once the scientific breakthrough (with a thermal efficiency of 3.5 percent) was achieved in 1901, it was followed by the economic breakthrough in 1936.

The computational difficulties of the full three-dimensional flow through a turbomachine blade row caused investigators to seek a simpler but less general description of the flow. Lorenz [1] first introduced the idea of an infinite number of blades of infinitesimal thickness in order to follow the flow on a given surface (1905). Later Bauersfeld [2] (1905), von Mises [3] (1909), Stodola [4] (1927) and Dreyfus [5] (1946) further clarified and strengthened the theory.

Reissner [6] (1948) gave a blade-design method in which the extension from an infinite number of blades to a finite number of blades is accomplished by the use of a power series in the circumferential direction, and the terms in the series are determined by a comparison of the equations for an infinite number of blades and a finite number of blades.

Howell [7] (1948) gave a solution, based upon a conformal transformation, and by use of suitable intermediate stages transformed the cascade of arbitrarily specified airfoil profiles into a circle, the flow around which could be determined. This method was programmed by Pollard and Wordsworth [8] (1962). Garrick [9] (1944) also gave a solution to the problem based upon the Theodersen conformal transformations, and this method of solution was developed by Hall [10] (1963).

Schlichting [11] (1955), whose method was later modified by Mellor [12] (1956), distributed sources, sinks and vortices on the chord line in order to represent a given airfoil cascade profile. This limited the application of the theory to profiles of low camber. A more sophisticated approach is that due to Martensen [13] (1959) who distributed vorticity around the profile. During the years 1940-1944 Merchant and Collar [14] produced an analysis giving a transformation linking the known potential flow around a series of ovals to that around a cascade of inclined flat plates. Gostelow [15] (1963) extended Merchant and Collar's theory to a cascade of airfoil profiles, in an analogous manner to the theory of isolated Joukowsky transform airfoils. Gostelow's procedure can be outlined as follows. The normal flow past a series of laminae lying along the imaginary axis is known and the laminae can be transformed into a series of ovals, flow around which was given by

Lamb [16] (1932). A particular case of the general flow about the ovals is that for which the flow at infinity is inclined to the axis but for which there is no circulation. In this case the ovals can immediately be transformed into a cascade of flat plates parallel to the direction of flow at infinity. Application of this transformation to ovals which are offset from the origin produces a cascade of airfoil shapes for which the aerodynamic characteristics are readily obtained.

The aforementioned works dealing with the flow through a cascade of airfoils do not represent the actual flow through a turbomachine. The current turbomachinery analysis has its fundamental roots in the classical work of Wu [17] (1952). Wu was among the first to recognize clearly the requirements, approximations and assumptions inherent in any attempt to solve the full three-dimensional flow by two-dimensional approaches. In his study, Wu succeeded in taking the three-dimensional inviscid flow on mutually orthogonal surfaces. Thus, the three-dimensional problem was broken down into managable two-dimensional flow problems. Wu proposed a finite-difference method of solution for computations on the S1 (blade-to blade) and S2 (hub-to-shroud, i.e. meridional) computational surfaces.

Since Wu, a diversity of solution methods have emerged through efforts aiming at solving Wu's equations on the S1 surface, or the S2 surface or on both by iteration.

The various finite difference procedures begin by breaking the flow field into a grid or network system with a set of node points. At each node point, finite difference expressions are written to approximate the differential operators from the governing equations prior to solution by a difference technique. Linearization is accomplished with a variety

of techniques ranging from lagging the evaluation of certain terms from one iteration to the next, to the utilization of various mathematical linearization operations. The procedures require a damping factor in order to stabilize the calculations and assure convergence.

Using finite difference methods Marsh [18] (1966) developed a computer program for the through flow problem on the midchannel S2 surface, Katsanis [19] (1969) developed a program for transonic flow on the blade-to-blade (S1) streamsurfaces, Bosman and El-Shaarawi [20] (1976) calculated the flow on the blade-to-blade (S1) and hub-to-shroud (S2) surfaces, Farrel and Adamczyk [21] (1981) solved the blade-to-blade flow on the S1 surface.

Smith [22] (1966) presented the fundamentals of the streamline curvature methods. Streamline curvature methods gained early popularity due to their intrinsic ability to handle odd geometric boundary shapes with ease, and the convenience of quasi-orthogonals to rigorously establish a general "grid network". Quasi-orthogonals are lines arbitrarily placed across the flow field, roughly perpendicular to the streamlines. The momentum equation is written and solved along the quasi-orthogonal. An important part of the streamline curvature approach is the utilization of some curve fitting technique. The curve fit procedure ties together the flow field between upstream an downstream locations thus preserving the "ellipticity" of the flow field. The approach involves first derivatives along the quasi-orthogonals while first and second order derivatives are used along the streamline. The quality of the curve fit is extremely important to this method. Accuracy, stability and speed are directly connected to the quality of the curve fit. Linearization is

not required but damping is necessary to assure stable convergence.

Katsanis [19] extrapolated a reduced flow subsonic solution, obtained by a finite difference method, to transonic flow by making use of the streamline curvature techniques (1969). Streamline curvature methods were used by Senoo and Nakase [23] (1972) to solve the flows on the S1 and S2 surfaces, by Katsanis and Mc Nally [24] to calculate the flow on the hub-to-shroud (S2) mid-channel surface (1973), by Novak [25] to calculate the mean stream sheet (S2) flow (1975). Later in 1977 Novak and Hearsey [26] used Novak's method to solve the flow on blade-to-blade (S1) streamsurfaces. In 1981 Kundig [27] presented a streamline curvature solution on the mean blade-to-blade (S1) streamsurface.

Application of the finite element method to fluid flow problems is very recent. Since 1974 several papers have been published in the open literature dealing with the application of the finite element methods to either the blade-to-blade (S1) or the hub-to-shroud (S2) flow solutions. The finite element techniques usually utilize variational principle formulations or a weighted residual Galerkin procedure. The flow field is broken into a system of node points defining a series of finite elements. On each element the field variable (either the stream function of the velocity potential) is assigned a certain variation determined by the adjacent nodal values, the form of which gives the interpolation functions. Using the interpolation function representation of the field variable in the variational principle formulation or alternately in the Galerkin procedure such that the weight functions are equal to the interpolation functions, algebraic equations are obtained

for the elements. The so-called global system equations are obtained by assembling all the equations for all elements. The resulting system of nonlinear equations is solved by conventional techniques.

Adler and Krimerman [28] (1974) were the first to apply the finite element method to through flow (S2 surface) problems. In 1976 Hirsch and Warzee [29] presented through flow (S2 surface) calculations in turbomachinery and discussed the applicability of their method and its superiorities over the previous solution techniques. Not surprisingly, this paper has received the attention of many investigators and attracted numerous follow-on discussions. Various other finite element solutions on the blade-to-blade (S1) surfaces were reported by Adler and Krimerman [30] (1977),Habashi et.al. [31] (1979) and Deconinck and Hirsch [32] (1981) In 1980 Haas and Keck [33] demonstrated the applicability of such programs on microcomputers. Baskharone and Hamed [34] (1981) and Akay and Ecer [35,36,37,38] (1981-1982) attacked the problem of transonic cascade flows. With the exception of Hirsch and Warzee and Habashi et.al. all the references cited above made use of linear elements.

Strictly speaking, flow solutions on S1 and S2 surfaces are dependent on each other and iteration is required between the two. Frequently, however, calculations are made without recognizing the connection. Methods of combining the S1 and S2 solutions yield the so-called "quasi-three-dimensional" solutions which exhibit notable improvement over the "isolated" analyses, and yet fail to give some essential features of the full three-dimensional flows. As examples to quasi-three-dimensional investigations one may cite Senoo and Nakase [23] (1972), Novak and Hearsey [26] (1977), Adler and Krimerman [39] (1978), and Hirsch and Warzee [40] (1979).

The difficulty experienced in the two-dimensional solutions, and even in the quasi-three-dimensional solutions for that matter, is the impossibility of extracting from these some essential features of the full three-dimensional flow that may be extended to apply to technological problems. On the other hand, a complete three-dimensional analysis will offer the possibility of investigating in detail very particular problems that may arise. There are a few complete three-dimensional solution programs that have been published in the open literature. Laskaris [41] published in 1978 his work on the finite element analysis of three-dimensional potential flow in turbomachines. Laskaris used second-order isoparametric Lagrangian elements in discretizing the solution domain. Laskaris' pioneering work lacks corroborating experimental evidence and comparison with other solution methods. Following the paper of Laskaris, Whirlow, et.al. [42] used four noded linear tetrahedral elements to calculate the incompressible flow in a centrifugal fan (1980). Their results compared well with experimental data. Finally, Sarathy [43] (1981) solved the three-dimensional flow field in turbomachinery by using a time-marching finite difference solution technique based on the Denton [44] flow solver. Sarathy also presented a comparison between his calculations and experimental results.

The theory of the two-dimensional flow about airfoil blades has been fundamental to the development of turbomachinery, because it can be used with fair approximation to represent the actual flow. However, the turbomachine technology has now reached a level of sophistication which requires more detail and more accuracy. The aftereffects of the recent energy crisis, the increasing standards and requirements of the aircraft

industry together with the new areas of application such as the automotive gas turbines all indicate the need for more accurate representation of the flow in turbomachinery. With the advent of remarkably fast and versatile computers, complex computational procedures can now be tailored into handy and effective design tools.

In fact, the actual three-dimensional flow in a turbomachine is neither inviscid nor steady, it is rotational and irreversible. Owing to the relative motion between successive rows of blades, the flow is far from steady. Owing to the presence of many fixed and moving boundaries there are substantial boundary layer and other secondary flow effects. The logical approach to this problem is to introduce one step of complication at a time. The present intent is to deal with the steady, compressible, three-dimensional, potential flow.

III. ANALYSIS

The present study is concerned with the analysis of three-dimensional, compressible flows in axial-, radial-, or mixed-flow turbomachinery under the limitations of steady and potential flow conditions. The basic approach is to isolate a single blade row (which may be stationary or rotating at a contant speed) and consider the flow in one of the blade passages as representative of the total flow through the entire row of blades. It is assumed that the fluid is inviscid and enters this "characteristic" passage with uniform entropy, uniform total enthalpy and zero vorticity. Under these conditions, Kelvin's circulation theorem ensures irrotationality of the absolute flow throughout. The analysis is restricted to subsonic flows which may have local supersonic spots. The fluid is either incompressible or assumed to be accurately represented by the perfect gas law.

This chapter begins with a presentation of the classical velocity potential formulation of the problem stated above. An equivalent variational formulation is then described. This formulation incorporates a quasi-linearization concept which leads to iterative solution. Density distribution is assumed to be given by a previous solution and therefore has no variation. The problem of three-dimensional, compressible, potential flow in turbomachinery is thus reduced to the determination of the

absolute velocity potential distribution which minimizes an equivalent functional in the solution domain with appropriate boundary conditions. A finite element method is developed to solve this problem. The remainder of the chapter treats the basic features of the finite element method and the computer code developed. Details are given in the Appendices.

3.1 THEORY OF THREE-DIMENSIONAL FLOW IN TURBOMACHINERY

The fundamental equations governing the three-dimensional flow of an inviscid, compressible fluid through a turbomachine will be developed with respect to the moving coordinate system shown in Figure 1. The system defined by the coordinates (rzθ) is rotating at constant angular speed, ω, about the z-axis of the fixed cartesian system (xyz). Here, z represents the axial coordinate along the centerline of the machine, r the radial coordinate and θ the tangential coordinate measured relative to the blade leading edge.

The equation of continuity is given by [45]

$$\frac{\partial \rho}{\partial t} + \vec{\nabla} \cdot (\rho \vec{W}) = 0 \qquad (1)$$

For flow through a blade passage rotating at constant angular velocity about the z-axis, Euler's equation of motion may be written as follows (e.g. see Wu [17]).

$$\frac{D\vec{W}}{Dt} - \omega^2 \vec{r} + 2\vec{\omega} \times \vec{W} = -\frac{1}{\rho} \vec{\nabla} P \qquad (2)$$

where $\vec{W}$, P and ρ denote the relative velocity vector, pressure and density respectively. It should be noted that the substantial derivative and the

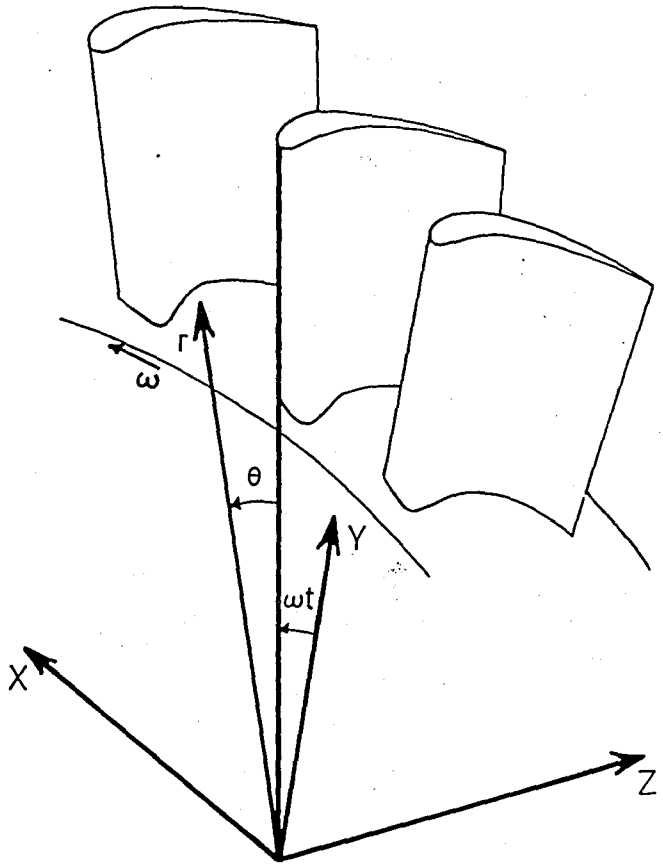

FIGURE 1 - Moving Coordinate System

gradient operator are defined here with respect to the rotating coordinate system.

Using Gibb's relationship

$$T\,ds = dh - \frac{dP}{\rho} \tag{3}$$

where T, s and h denote the temperature, entropy and enthalpy, respectively, and noting

$$\frac{D\vec{W}}{Dt} = \frac{\partial \vec{W}}{\partial t} + \vec{W}\cdot\vec{\nabla}\vec{W} \tag{4}$$

Eq. (2) becomes,

$$\frac{\partial \vec{W}}{\partial t} + W\cdot\vec{\nabla}\vec{W} - \omega^2\vec{r} + 2\vec{\omega} \times \vec{W} = -\vec{\nabla}h + T\,\vec{\nabla}s \tag{5}$$

The convective acceleration term is given by the vector identity

$$\vec{W}\cdot\vec{\nabla}\vec{W} \equiv -\vec{W} \times \vec{\nabla} \times \vec{W} + \frac{1}{2}\,\vec{\nabla}(W^2) \tag{6}$$

Substitution of the above into Eq. (5) yields

$$\frac{\partial \vec{W}}{\partial t} - \vec{W} \times (\vec{\nabla} \times \vec{W} + 2\vec{\omega}) = -\vec{\nabla}I + T\,\vec{\nabla}s \tag{7}$$

where the rothalpy, I, is defined as

$$I \equiv h + \frac{1}{2}W^2 - \frac{1}{2}(\omega r)^2 = H - \omega(r\,V_\theta) \tag{8}$$

where

$$H = \text{stagnation enthalpy} = h + \frac{1}{2}V^2 \tag{9}$$

The relative velocity, $\vec{W}$, is related to the absolute velocity by

$$\vec{W} = \vec{V} - \vec{\omega} \times \vec{r} \tag{10}$$

Hence,

$$\vec{\nabla} \times \vec{W} = \vec{\nabla} \times \vec{V} - \vec{\nabla} \times (\vec{\omega} \times \vec{r}) \tag{11}$$

But

$$\vec{\nabla} \times (\vec{\omega} \times \vec{r}) = (\vec{r} \cdot \vec{\nabla})\vec{\omega} - (\vec{\omega} \cdot \vec{\nabla})\vec{r} + \vec{\omega}(\vec{\nabla} \cdot \vec{r}) - \vec{r}(\vec{\nabla} \cdot \vec{\omega}) = 2\vec{\omega}$$

Therefore, an alternative form of Eq. (7) which involves the vorticity of the absolute motion is

$$\frac{\partial \vec{W}}{\partial t} - \vec{W} \times \vec{\nabla} \times \vec{V} = -\vec{\nabla} I + T \vec{\nabla} s \tag{12}$$

For stationary blades Eq. (12) reduces to

$$\frac{\partial \vec{V}}{\partial t} - \vec{V} \times \vec{\nabla} \times \vec{V} = -\vec{\nabla} H + T \vec{\nabla} s \tag{13}$$

If the fluid enters the stator of a turbomachine with uniform H and s and zero vorticity and if the flow is assumed to be steady and adiabatic, then Kelvin's circulation theorem ensures that the flow will remain irrotational in passing through the stator.

Under the above conditions, the flow entering the rotor has uniform H and s. If, in addition, the absolute tangential velocity of the rotor inlet flow varies inversely with radius, as in the potential vortex, then from Eq. (8) the rothalpy, I, is also uniform and Eq. (12) leads to the conclusion that the absolute flow through the rotor will also remain irrotational. It should be noted that if the blade rows are not placed too close together and no trailing vortices are shed from the

preceeding row, the relative flow through the rotor can be taken as steady.

Under the foregoing limitations, it is concluded that the absolute flow through the stationary and rotating rows of a turbomachine can be treated as potential flow in which the rothalpy remains constant throughout. Hence, the equation of motion reduces to

$$\vec{\nabla} \times \vec{V} = 0 \tag{14}$$

and

$$I = h + \frac{1}{2} V^2 - \omega r V_\theta = \text{constant} \tag{15}$$

Since the absolute flow is irrotational, an absolute velocity potential, Φ, may be defined as

$$\vec{V} = \vec{\nabla}\Phi \tag{16}$$

or in terms of the velocity components as

$$\begin{aligned} \frac{\partial \Phi}{\partial r} &= V_r = W_r \\ \frac{1}{r}\frac{\partial \Phi}{\partial \theta} &= V_\theta = W_\theta + \omega r \\ \frac{\partial \Phi}{\partial z} &= V_z = W_z \end{aligned} \tag{17}$$

Substituting Eq. (16) into Eq. (1) for steady flow we obtain

$$\vec{\nabla}\cdot(\rho\vec{\nabla}\Phi) = \omega \frac{\partial \rho}{\partial \theta} \tag{18}$$

On the other hand, from Eq. (15)

$$\frac{h}{h_{in}} = 1 + \frac{1}{2h_{in}} [V_{in}^2 - V^2 + 2\omega(r V_\theta - \lambda_{in})] \tag{19}$$

where the symbol (in) refers to some reference point on the inlet plane and

$$\lambda_{in} = (r V_\theta)_{in} = \text{prerotation (or inlet swirl)} \tag{20}$$

For isentropic flow of a perfect gas

$$\frac{\rho}{\rho_{in}} = \left(\frac{T}{T_{in}}\right)^{1/(\gamma-1)} \tag{21}$$

where $\gamma = C_p/C_v = \text{specific heat ratio}$ (22)

and

$$h = C_p T = \frac{\gamma}{\gamma - 1} RT \tag{23}$$

where R is the gas constant.

Eq. (23) allows Eq. (19) to be written in terms of the temperature.

$$\frac{T}{T_{in}} = 1 + \frac{\gamma - 1}{2RT_{in}} [V_{in}^2 - V^2 + 2\omega(r V_\theta - \lambda_{in})] \tag{24}$$

The problem of steady, three-dimensional, potential flow of a perfect gas through the stationary and rotating rows of a turbomachine is governed by Eqs. (18), (21) and (24) subject to various boundary conditions. These equations may be nondimensionalized by introducing the nondimensional quantities

$$V' = \frac{V}{V_{in}} \quad , \quad W' = \frac{W}{V_{in}}$$

$$T' = \frac{T}{T_{in}}$$

$$\omega' = \frac{\omega L}{V_{in}}$$

$$\Phi' = \frac{\Phi}{LV_{in}} \tag{25}$$

$$\rho' = \frac{\rho}{\rho_{in}}$$

$$r' = \frac{r}{L} \quad , \quad z' = \frac{z}{L}$$

where L is a reference length.

Substituting Eq. (25) into Eqs. (18), (21) and (24), and dropping out the primes for simplicity, we obtain the nondimensional equations

$$\vec{\nabla} \cdot (\rho \vec{\nabla} \Phi) = \omega \frac{\partial \rho}{\partial \theta} \tag{26}$$

$$\rho = T^{1/(\gamma-1)} \tag{27}$$

$$T = 1 + \frac{(\gamma - 1)M_{in}^2}{2} [1 - V^2 + 2\omega(r V_\theta - \lambda_{in})] \tag{28}$$

where M_{in} = reference Mach number = $\dfrac{V_{in}}{(\gamma R T_{in})^{1/2}}$

The analysis will be presented, henceforth, in the nondimensiona format unless otherwise stated.

The three-dimensional solution domain corresponding to a characteristic blade passage and the flow boundaries are illustrated in Figure 2.

At the inlet surface λ_{in} and the radial variation of V_r are given. To avoid arbitrariness, Φ is specified at some point at the inlet surface as Φ_* and its distribution at the inlet surface is then calculated from

$$\Phi_{in}(r,\theta) = \Phi_* + [\int_{r_*}^{r} (V_r)_{in}\, dr]_{\theta_*} + [\int_{\theta_*}^{\theta} \lambda_{in}\, d\theta]_r \tag{29}$$

Along the side surfaces 1 and 2 (both upstream and downstream), the flow conditions are periodic. That is, $V_1 = V_2$ at the same r and z. Hence,

$$\begin{aligned} (\frac{\partial\Phi}{\partial r})_1 &= (\frac{\partial\Phi}{\partial r})_2 \\ (\frac{1}{r}\frac{\partial\Phi}{\partial\theta})_1 &= (\frac{1}{r}\frac{\partial\Phi}{\partial\theta})_2 \\ (\frac{\partial\Phi}{\partial z})_1 &= (\frac{\partial\Phi}{\partial z})_2 \end{aligned} \tag{30}$$

The first and the last conditions in the above require that

$$\Phi_1 = \Phi_2 + C \tag{31}$$

where the constant C is to be evaluated in the upstream and the downstream regions, separately. In the upstream region C is determined directly from Eq. (29) as

$$C_{upstream} = \lambda_{in}\, \Delta\theta_p \tag{32}$$

where $\Delta\theta_p$ denotes the blade pitch.

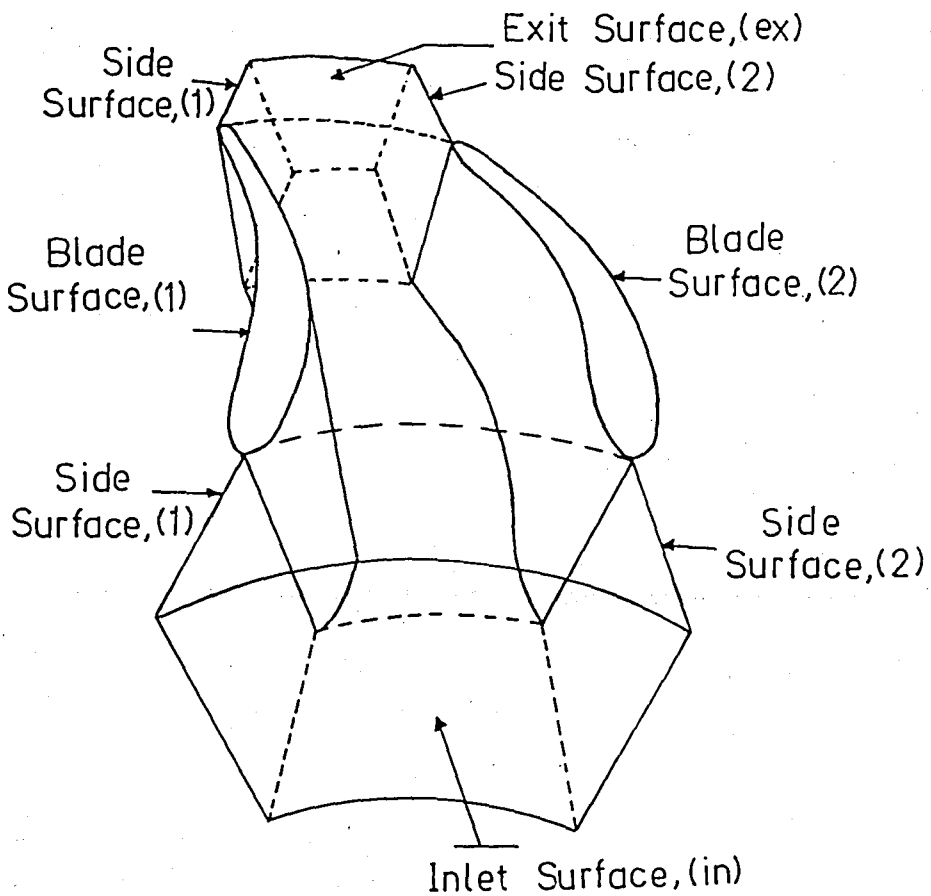

FIGURE 2 - Characteristic Blade Passage

By a similar argument, C in the downstream region is related to the outlet tangential velocity as follows

$$C_{downstream} = (r\, V_\theta)_{ex}\, \Delta\theta_p \tag{33}$$

where the symbol (ex) refers to the exit surface.

It is convenient to express the second condition in Eq. (30) as

$$\left(\frac{\partial \Phi}{\partial n}\right)_1 = -\left(\frac{\partial \Phi}{\partial n}\right)_2 \tag{34}$$

where $\vec{n}$ denotes the unit vector normal to the boundary surface.

At the exit surface, the distribution of the axial velocity component is to be specified,

$$\left(\frac{\partial \Phi}{\partial z}\right)_{ex} = [V_z(r)]_{ex} \tag{35}$$

Along the solid surfaces (blade, hub and shroud surfaces) the relative normal velocity must vanish, i.e.

$$\frac{\partial \Phi}{\partial n} = \omega r(\vec{n}\cdot\vec{e}_\theta) \tag{36}$$

where $\vec{n}$, here denotes the unit normal to the solid surface and $\vec{e}_\theta$ the unit vector in the tangential direction.

An iterative solution procedure is adopted to solve the governing system of nonlinear equations. This procedure involves a quasilinearization concept and may be outlined as follows. A flow solution is calculated from Eq. (26) with the given boundary conditions by assuming ρ = constant initially. Estimates of the temperature and the density distributions are then obtained from Eqs. (28) and (27), in sequence. This procedure is repeated until convergence, with density assumed to be given

at each time from the previous solution.

A major difficulty encountered in this potential flow problem is that the exit angle imparted to the gas stream by the blade row is usually not known in advance and $(r\ V_\theta)_{ex}$ in Eq. (33) cannot be specified. The question arises then as to what the exit angle would be in the real situation. This difficulty could be overcome by imposing the Kutta condition that stipulates the equality of the pressures on both surfaces of the blade at the trailing edge. The Kutta condition cannot be imposed explicitly. Instead, amongst several solutions obtained for a reasonable range of $(r\ V_\theta)_{ex}$ the one that satisfies the Kutta condition is identified as the true solution and the corresponding exit condition gives the actual flow deflection within limitations of the present analysis (inviscid flow).

3.2 VARIATIONAL FORMULATION

Variational analysis is used to determine an extremum problem which is equivalent to the differential formulation developed above.

The nondimensional functional, K, associated with the nondimensional differential equation (26) must satisfy

$$\delta K = \langle \delta\phi, L\phi - \omega \frac{\partial \rho}{\partial \theta} \rangle = 0 \tag{37}$$

where L is a differential operator defined as

$$L = \frac{1}{r}\frac{\partial}{\partial r}\left(\rho r \frac{\partial}{\partial r}\right) + \frac{1}{r^2}\frac{\partial}{\partial \theta}\left(\rho \frac{\partial}{\partial \theta}\right) + \frac{\partial}{\partial z}\left(\rho \frac{\partial}{\partial z}\right) \tag{38}$$

and δ denotes the first variation.

Eq. (37) may be rewritten more explicitly as

$$\int_{\forall} \delta\Phi(L\Phi)d\forall - \int_{\forall} \delta\Phi(\omega \frac{\partial\rho}{\partial\theta})d\forall = 0 \tag{39}$$

where $\forall$ denotes the solution domain.

It is noted that

$$\delta[\int_{\forall} \Phi\omega \frac{\partial\rho}{\partial\theta} d\forall] = \int_{\forall} \delta\Phi\omega \frac{\partial\rho}{\partial\theta} d\forall + \int_{\forall} \Phi\omega\delta(\frac{\partial\rho}{\partial\theta})d\forall \tag{40}$$

By virtue of the solution procedure described earlier, the density is assumed to be given by the previous solution and therefore has no variation. Hence, the second term on the right-hand side of Eq. (40) vanishes and Eq. (39) becomes

$$\delta K = \int_{\forall} \delta\Phi(L\Phi)d\forall - \delta[\int_{\forall} \Phi\omega \frac{\partial\rho}{\partial\theta} d\forall] = 0 \tag{41}$$

Using Eq. (38) the first integral in Eq. (41) may be integrated by parts to yield

$$\int_{\forall} \delta\Phi(L\Phi)d\forall = I_S - I_{\forall} \tag{42}$$

where

$$I_S = \int_S \rho \frac{\partial\Phi}{\partial n} \delta\Phi \, dS \tag{43}$$

S = boundary of the solution domain

and

$$I_{\forall} = \int_{\forall} \rho[\frac{\partial\Phi}{\partial r} \delta(\frac{\partial\Phi}{\partial r}) + \frac{1}{r^2} \frac{\partial\Phi}{\partial\theta} \delta(\frac{\partial\Phi}{\partial\theta}) + \frac{\partial\Phi}{\partial z} \delta(\frac{\partial\Phi}{\partial z})]d\forall \tag{44}$$

Since the density is treated as having no variation, it is possible to express Eqs. (43) and (44) as variations of surface and volume integrals, respectively.

The volume integral in Eq. (44) may be rearranged to give

$$I_{\forall} = \delta\{\frac{1}{2}\int_{\forall} \rho[(\frac{\partial\Phi}{\partial r})^2 + \frac{1}{r^2}(\frac{\partial\Phi}{\partial\theta})^2 + (\frac{\partial\Phi}{\partial z})^2]d\forall\} \tag{45}$$

On the other hand, the surface integral in Eq. (43) may be decomposed into two parts:

$$I_S = I_{S_d} + I_{S_n} \tag{46}$$

where S_d and S_n denote those portions of the boundary surface which are characterized by, respectively, Dirichlet and Neumann type boundary conditions.

Since Φ is specified on S_d, $\delta\Phi = 0$ on S_d and the Dirichlet boundary integral vanishes.

$$I_{S_d} = \int_{S_d} \rho \frac{\partial\Phi}{\partial n} dS = 0 \tag{47}$$

Expressing the Neumann boundary integral in a manner similar to Eq. (40),

$$I_{S_n} = \int_{S_n} \frac{\partial\Phi}{\partial n} \Phi dS = \delta[\int_{S_n} \rho\Phi \frac{\partial\Phi}{\partial n} dS] - \int_{S_n} \rho\Phi\delta(\frac{\partial\Phi}{\partial n})dS \tag{48}$$

and noting that $\delta(\partial\Phi/\partial n) = 0$ on S_n ($\partial\Phi/\partial n$ specified), we obtain

$$I_{S_n} = \delta[\int_{S_n} \Phi \frac{\partial\Phi}{\partial n} dS] \tag{49}$$

Substitution of Eqs. (42), (45), (46), (47) and (49) into Eq. (41) gives

$$\delta K = \delta\{\int_{S_n} \rho\Phi \frac{\partial\Phi}{\partial n}\, dS - \frac{1}{2}\int_{\forall} \rho[(\frac{\partial\Phi}{\partial r})^2 + \frac{1}{r^2}(\frac{\partial\Phi}{\partial\theta})^2 + (\frac{\partial\Phi}{\partial z})^2]d\forall$$

$$- \int_{\forall} \rho\Phi\omega \frac{\partial\rho}{\partial\theta}\, d\forall = 0 \qquad (50)$$

Hence, the problem of steady, three-dimensional, compressible, potential flow in turbomachinery is reduced to the determination of the velocity potential distribution which minimizes the functional

$$K = K_{\forall} - K_{S_n} \qquad (51)$$

where

$$K_{\forall} = \frac{1}{2}\int_{\forall} \rho[(\frac{\partial\Phi}{\partial r})^2 + \frac{1}{r^2}(\frac{\partial\Phi}{\partial\theta})^2 + (\frac{\partial\Phi}{\partial z})^2]d\forall + \int_{\forall} \rho\Phi\omega\frac{\partial\rho}{\partial\theta}\, d\forall \qquad (52)$$

and

$$K_{S_n} = \int_{S_n} \rho\Phi \frac{\partial\Phi}{\partial n}\, dS \qquad (53)$$

in the solution domain by using the Dirichlet boundary conditions given by Eqs. (29, 31-33) and the Neumann boundary conditions given by Eqs. (34-36) with the density calculated at each step of iteration from Eqs. (27,28).

3.3 FINITE ELEMENT ANALYSIS

Due to the complexity of the flow geometry and the difficulty in imposing the boundary conditions, a finite element method will be developed to solve the extremum problem described above.

3.3.1 General Development

In this method the solution domain is divided into numerous sub-domains known as finite elements. The total functional, K, may then be written in terms of the element functionals $(K_V)_e$ and $(K_{S_n})_{e_n}$ as

$$K = \sum_{e=1}^{E} (K_V)_e - \sum_{e_n=1}^{E_n} (K_{S_n})_{e_n} \tag{54}$$

In the above equation the subscript e denotes the elements under consideration, E is the total number of elements used in representing the total domain, the subscript e_n refers to those elements which have surfaces on the Neumann boundaries and E_n is the total number of such elements.

The field variable (velocity potential), Φ, is approximated within each element in terms of its values at several points (called nodes) in that element as

$$\Phi_e = \sum_{j=1}^{m} N_j q_j \tag{55}$$

where N_j are the interpolation functions, q_j the element coordinates (the nodal values of the field variable) and m is the number of nodes.

In general, the nodes of the solution domain is the ensemble of the element nodes. If each element has m nodes resulting in M nodes in the global domain, then the element nodes can be mapped into the global nodes by means of an (mxM) transformation matrix. Thus, the element nodal values of the velocity potential (element coordinates) may be related to the global nodal values (global coordinates) according to the transformation law

$$\underset{(mx1)}{\{q\}_e} = \underset{(mxM)}{[T]_e} \quad \underset{(Mx1)}{\{Q\}} \tag{56}$$

where $\{q\}_e$ is the element coordinate vector, $\{Q\}$ the global coordinate vector and $[T]_e$ the element transformation matrix.

The solution to the problem at hand is that set of nodal values of the velocity potential, Q_j, which minimizes the functional K. Minimization of K requires setting

$$\frac{\partial K}{\partial Q_j} = 0 \qquad \text{for } j = 1,\ldots,M \tag{57}$$

The above requirement may be expressed in terms of the sum of the element contributions as

$$\sum_{e=1}^{E} [T]_e^T \frac{\partial (K_V)_e}{\partial \{q\}_e} - \sum_{e_n=1}^{E_n} [T]_e^T \frac{\partial (K_{S_n})_{e_n}}{\partial \{q\}_{e_n}} = 0 \tag{58}$$

where $[T]_e^T$ denotes the transpose of the transformation matrix of element e and serves to map the element contribution into the global domain.

The two summation terms in the above equation will be treated separately. Using Eqs. (52) and (55) the volume integral contribution of element e may be written as

$$\underset{(mx1)}{\frac{\partial (K_V)_e}{\partial \{q\}_e}} = \underset{(mxm)}{[k]_e} \quad \underset{(mx1)}{\{q\}_e} + \underset{(mx1)}{\{r\}_e} \tag{59}$$

where $[k]_e$, known as the element property matrix, and $\{r\}_e$ are defined by,

$$(k_{ij})_e = \int_{\forall_e} \rho\left[\frac{\partial N_i}{\partial r}\frac{\partial N_j}{\partial r} + \frac{1}{r^2}\frac{\partial N_i}{\partial \theta}\frac{\partial N_j}{\partial \theta} + \frac{\partial N_i}{\partial z}\frac{\partial N_j}{\partial z}\right] d\forall \qquad (60)$$

and

$$(r_i)_e = \int_{\forall} \omega \frac{\partial \rho}{\partial \theta} N_i \, d\forall \qquad (61)$$

Similarly, from Eqs. (53) and (55) the surface integral contribution of a Neumann boundary element e_n is

$$\frac{\partial (K_{S_n})_{e_n}}{\partial \{q\}_{e_n}} = \{r\}_{e_n} \qquad (62)$$

where

$$(r_i)_{e_n} = \int_{(S_n)_{e_n}} \rho \, N_i \left(\frac{\partial \Phi}{\partial n}\right)_{(S_n)_{e_n}} dS \qquad (63)$$

In Eq. (63), $(\partial\Phi/\partial n)_{(S_n)_{e_n}}$ represents the boundary condition (Eqs. (34-36)) specified on the Neumann boundary surface S_n of the element e_n.

Substitution of Eqs. (59) and (62) into Eq. (58) gives

$$\sum_{e=1}^{E} [T]_e^T [k]_e \{q\}_e = \sum_{e=1}^{E_n} [T]_{e_n}^T \{r\}_{e_n} - \sum_{e=1}^{E} [T]_e^T \{r\}_e \qquad (64)$$

Using Eq. (56) in the above equation and defining a system property matrix [K] and a system right-hand side vector {R} as

$$[K] = \sum_{e=1}^{E} [T]_e^T [k]_e [T]_e \qquad (65)$$

and

$$\{R\} = \sum_{e_n=1}^{E_n} [T]_{e_n}^T \{r\}_{e_n} - \sum_{e=1}^{E} [T]_e^T \{r\}_e \qquad (66)$$

we obtain the global system equation.

$$[K]\{Q\} = \{R\} \tag{67}$$

Hence, at an internal node the algebraic sum of the volume integral contributions from all the elements having this node in common is set equal to zero. However, at a Neumann boundary node this sum is set equal to the algebraic sum of the surface integral contributions from all the Neumann boundary elements having this node in common. Repeating the same for all the nodes yields M algebraic equations for M unknowns.

The Dirichlet boundary conditions impose additional constraints to the above formulation. To introduce these constraints, let us partition the matrix equation (67) as follows

$$\begin{bmatrix} K_{gg} & K_{gd} \\ K_{dg} & K_{dd} \end{bmatrix} \begin{Bmatrix} Q_g \\ Q_d \end{Bmatrix} = \begin{Bmatrix} R_g \\ R_d \end{Bmatrix} \tag{68}$$

where d represents the Dirichlet boundary nodes where the velocity potential is prescribed and g represents the remaining nodes, including the periodic Dirichlet nodes.

Since Q_d are known, we obtain the following

$$[K_{gg}]\{Q_g\} = \{R_g\} - [K_{gd}]\{Q_d\} \tag{69}$$

The above equation set needs to be further modified to incorporat the Dirichlet type periodic boundary conditions given by Eq. (31). For this purpose Eq. (69) may be rewritten in the compact form

$$[K_{gg}]\{Q_g\} = R_g^* \tag{70}$$

and repartitioned as

$$\begin{bmatrix} K_{aa} & K_{a1} & K_{a2} \\ K_{1a} & K_{11} & K_{12} \\ K_{2a} & K_{21} & K_{22} \end{bmatrix} \begin{Bmatrix} Q_a \\ Q_1 \\ Q_2 \end{Bmatrix} = \begin{Bmatrix} R_a^* \\ R_1^* \\ R_2^* \end{Bmatrix} \tag{71}$$

where subscripts 1 and 2 represent the periodic nodes on side surfaces 1 and 2 respectively and subscript a the remaining nodes.

From Eq. (31) we have

$$\{Q_2\} = \{Q_1\} - \{C\} \tag{72}$$

Hence, Eq. (71) becomes

$$\begin{bmatrix} K_{aa} & (K_{a1}+K_{a2}) \\ (K_{1a}+K_{2a}) & (K_{11}+K_{21})+(K_{12}+K_{22}) \end{bmatrix} \begin{Bmatrix} Q_a \\ Q_1 \end{Bmatrix} = \begin{Bmatrix} R_a^*+K_{a2}\ C \\ (R_1^*+R_2^*)+(K_{12}+K_{22})C \end{Bmatrix} \tag{73}$$

Finally, we close up the formulation by showing that the net effect of the Neumann type periodic boundary conditions on the above equations is zero. In light of Eqs. (34) and (63) it is noted that the surface integral contributions from the Neumann type periodic boundary elements to R_1^* and R_2^* are equal and opposite in sign. Since R_1^* and R_2^* are added in Eq. (73), these will cancel and the net contribution of the periodic surface integrals will vanish. A similar argument conducted in the global domain with the help of Eqs. (34) and (49) leads to the same conclusion. Hence, the Neumann type periodic boundary integrals need not be calculated.

The solution to the above set of linear algebraic equations (Eq. (73)) minimizes the discretized form of the functional, K, given by Eq. (54), and therefore, it represents an approximate solution to the potential flow problem under consideration for a given density distribution. Iteration by the use of Eqs. (27) and (28) leads to the final compressible flow solution.

3.3.2 Determination of Element Contribution for Tetrahedral Elements

The solution domain is discretized by using ten-node tetrahedral elements (Figure 3) which allows quadratic interpolation of the velocity potential in Eq. (55).

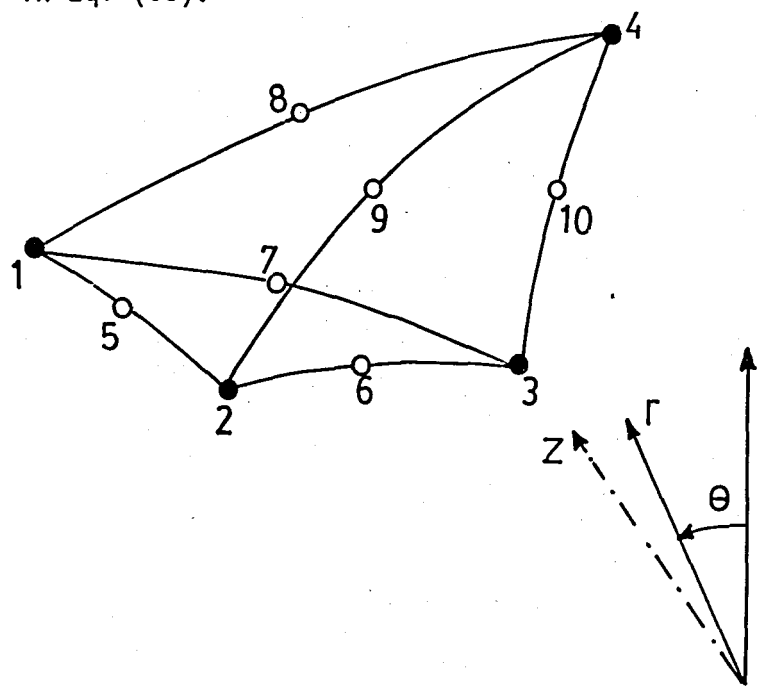

FIGURE 3 - Ten-node Tetrahedral Element

The element geometry is related to the nodal values of the coordinates r, θ and z by using either linear or quadratic interpolation functions. If the same quadratic interpolation functions that approximate the velocity potential in an element are used to interpolate the coordinates in terms of their nodal values as

$$r = \sum_{j=1}^{10} N_j r_j$$

$$\theta = \sum_{j=1}^{10} N_j \theta_j \qquad (74)$$

$$z = \sum_{j=1}^{10} N_j z_j$$

where r_j, θ_j and z_j denote the coordinates of the j'th node in the element, then curved sided elements may be implemented so as to enable better approximation of the blade profile. This formulation which uses the same order of interpolation for the element geometry as for the velocity potential is referred to as an "isoparametric" formulation.

An alternative simplified formulation may be obtained in terms of linear tetrahedral elements in which coordinates are related to the four corner nodes (nodes 1-4 in Figure 3) through the relationships

$$r = \sum_{j=1}^{10} \ell_j r_j$$

$$\theta = \sum_{j=1}^{10} \ell_j \theta_j \qquad (75)$$

$$z = \sum_{j=1}^{10} \ell_j z_j$$

where ℓ_j denote a set of linear interpolation functions. The simplified formulation in which a lower order of interpolation is used for the element geometry is said to be a "subparametric" formulation.

Use of a normalized local coordinate system will facilitate the construction of interpolation functions in developing the finite element formulations. The natural coordinates, L_i ($i = 1,2,3,4$), of a point within a tetrahedral element may be defined as the ratios of the volumes subtended by lines connecting this point and the vertices to the total volume. With reference to Figure 4 the natural coordinates are given by

$$L_i = \frac{\text{volume } (pjk\ell)}{\text{volume } (ijk\ell)} \tag{76}$$

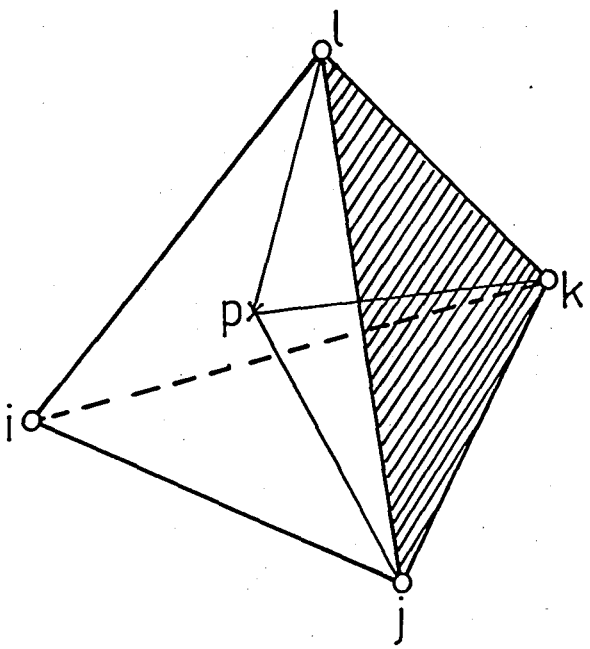

FIGURE 4 - Local Coordinates

In view of Eq. (76), L_i varies from 0 at surface jkℓ to 1 at vertex i and we have

$$L_1 + L_2 + L_3 + L_4 = 1 \tag{77}$$

Thus the natural coordinates are not independent. This difficulty may be resolved by considering L_1, L_2 and L_3 as an independent system and L_4 as a dependent coordinate given by Eq. (77). Derivatives in this system of natural coordinates may be transformed into the cylindrical coordinate system as follows

$$\begin{Bmatrix} \frac{\partial}{\partial L_1} \\ \frac{\partial}{\partial L_2} \\ \frac{\partial}{\partial L_3} \end{Bmatrix} = \begin{bmatrix} \frac{\partial r}{\partial L_1} & \frac{\partial \theta}{\partial L_1} & \frac{\partial z}{\partial L_1} \\ \frac{\partial r}{\partial L_2} & \frac{\partial \theta}{\partial L_2} & \frac{\partial z}{\partial L_2} \\ \frac{\partial r}{\partial L_3} & \frac{\partial \theta}{\partial L_3} & \frac{\partial z}{\partial L_3} \end{bmatrix} \begin{Bmatrix} \frac{\partial}{\partial r} \\ \frac{\partial}{\partial \theta} \\ \frac{\partial}{\partial z} \end{Bmatrix} \tag{78}$$

The transformation matrix in the above equation is known as the Jacobian matrix, J. The volume elements in the two coordinate systems are related through the determinant of the Jacobian matrix as

$$dr\ d\theta\ dz = \det J\ \ dL_1\ dL_2\ dL_3 \tag{79}$$

Assuming that the Jacobian matrix possesses an inverse, J^{-1}, that is, the transformation is one-to-one, we may write

$$\begin{Bmatrix} \frac{\partial}{\partial r} \\ \frac{\partial}{\partial \theta} \\ \frac{\partial}{\partial z} \end{Bmatrix} = J^{-1} \begin{Bmatrix} \frac{\partial}{\partial L_1} \\ \frac{\partial}{\partial L_2} \\ \frac{\partial}{\partial L_3} \end{Bmatrix} \tag{80}$$

The quadratic interpolation functions N_j used in approximating the velocity potential in Eq. (55) are given in terms of the local coordinates as

$$
\begin{aligned}
N_j &= L_j(2L_j - 1) \qquad\qquad j = 1,2,3,4 \\
N_5 &= 4L_1L_2 \\
N_6 &= 4L_2L_3 \\
N_7 &= 4L_1L_3 \\
N_8 &= 4L_1L_4 \\
N_9 &= 4L_2L_4 \\
N_{10} &= 4L_3L_4
\end{aligned}
\tag{81}
$$

where the side nodes (5-10) are located midway between the corner nodes (1-4). Note that we have $N_j = 1$ at node j and 0 at all other nodes.

When combined with Eq. (76) which defines the transformation between the cylindrical coordinates and the local coordinates, Eq. (81) allows the volume and surface integrals existing in the element equations (Eqs. (60), (61) and (63)) to be easily evaluated with respect to the local coordinates. The details of the coordinate transformation for the cases of subparametric and isoparametric tetrahedral elements and the development leading to the corresponding element equations are presented in Appendices A and B, respectively.

3.3.3 Automatic Discretization of Solution Domain and Computer Code

The discretization of the solution domain directly into tetrahedral elements presents difficulties of visualization and could lead to errors in mesh construction. These difficulties may be avoided by performing the discretization in two stages. First, the domain is subdivided into eight-cornered hexahedral elements. Figure 5 illustrates the use of these "superelements" as the building bricks of the finite element mesh. Each hexahedral superelement is then subdivided into six basic tetrahedral elements as shown in Figure 6.

A general finite element computer code, TURBO, has been developed to solve three-dimensional, compressible, potential flows in axial-, radial- or mixed-flow turbomachines. The code handles subsonic flows with local supersonic spots. It has an automatic mesh generation capability. The solution domain is automatically subdivided into hexahedral superelements each consisting of six internal tetrahedra.

The code user has the option of choosing between subparametric (linear) and isoparametric (quadratic) tetrahedral elements. This flexibility enables high accuracy in the treatment of turbomachinery flows that exhibit complicated design features, such as highly twisted, curved and staggered blades, by the use of curved-sided elements that best fit the shape of the physical boundaries. Although it is true that isoparametric elements can better approximate the physical boundaries and the dynamic characteristics of the flow (i.e. the orientation and curvature of streamlines) subparametric elements are more economical with respect to computation time. For example, on the UNIVAC 1106 computer the

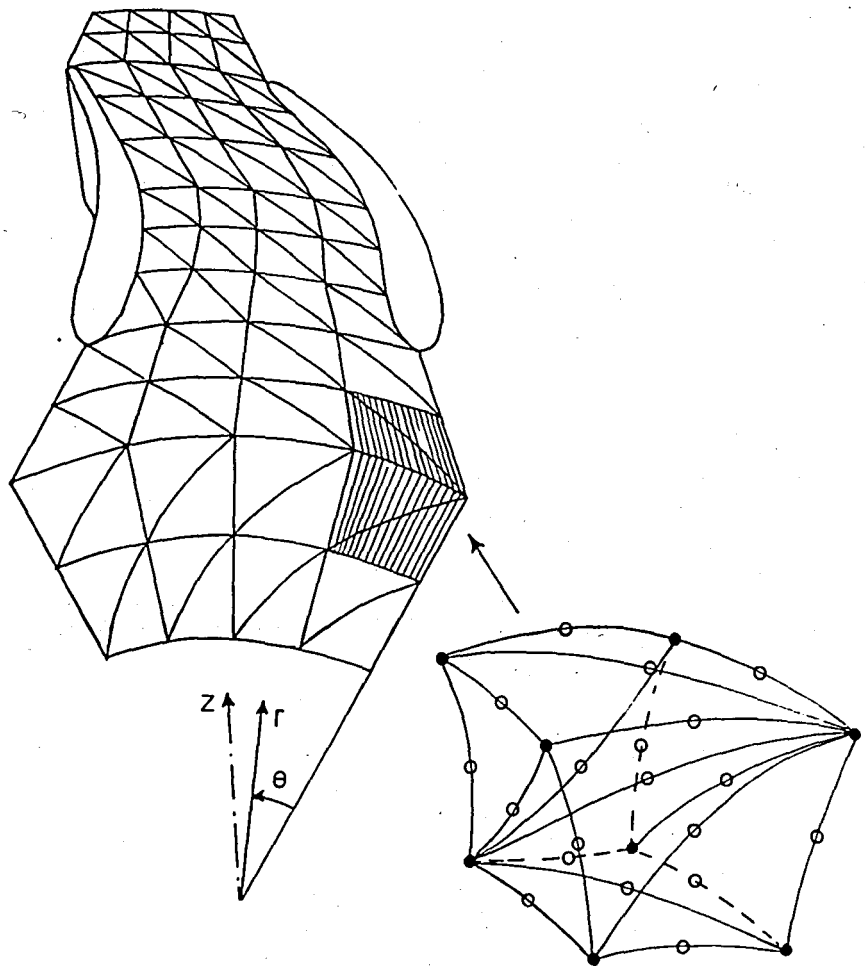

FIGURE 5 - Domain Discretization

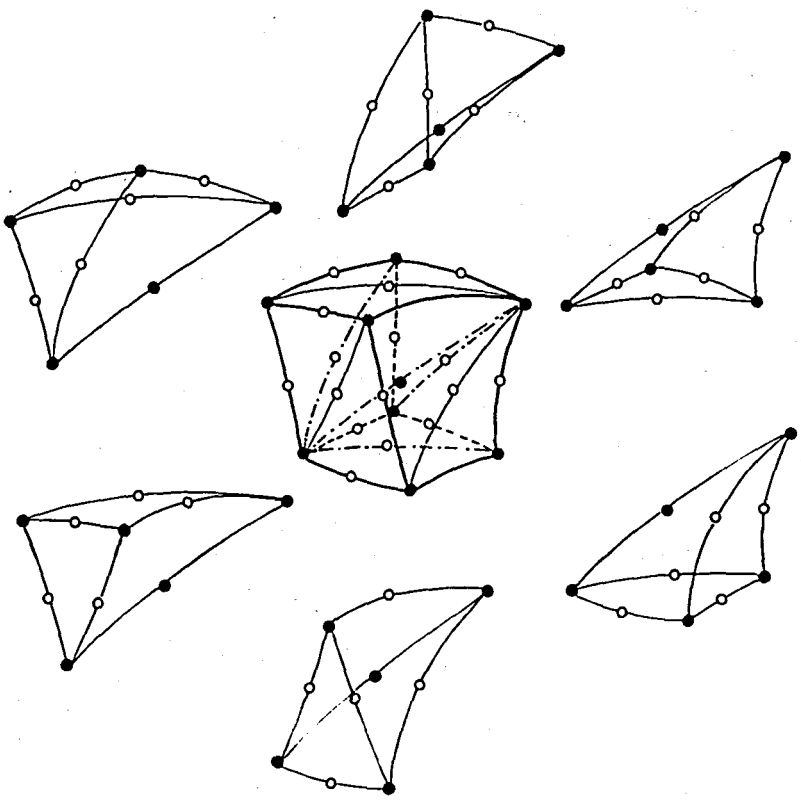

FIGURE 6 - Hexahedral Superelement

calculation of the system property matrix corresponding to 700 tetrahedra requires about 30 seconds with subparametric elements and about 6 minutes with isoparametric elements. It might be pointed out that the code is structured to accept any combination of these elements. An optimum tradeoff between accuracy and computation time may be obtained by using isoparametric elements at the blade boundaries and subparametric elements in the internal domain.

The user's manual for the code TURBO is given in Appendix D. Appendix E contains the code listing.

IV. APPLICATIONS

This chapter contains the applications considered for the verification of the finite element code developed.

The two dimensional verification of the code is established by application to the Gostelow cascade. The results are compared with the exact solution.

An experimental turbine stator and the first stage stator and rotor of a large electric utility gas turbine are the bases of comparison between the three-dimensional solution of the present code and the two-dimensional midstream surface solution of another code available in the public domain. This study provides a test against a code widely used in turbomachine calculations. Surface velocities measured in the experimental turbine provide empirical evidence for the validity of the calculated results.

A mixed-flow turbine rotor is considered to demonstrate the full three-dimensional capability of the code. Results are assessed upon comparison with another three-dimensional solution previously published.

4.1 APPLICATIONS TO TWO-DIMENSIONAL INCOMPRESSIBLE CASCADE FLOW AND COMPARISON WITH EXACT SOLUTION

The two-dimensional cascade shown in Figure 7 has been used to provide a partial validation of the computer code. The airfoil shape and the solution to the incompressible flow in this cascade have been obtained by Gostelow [15] through the application of Merchant and Collar transformation [14] to a set of ovals. The cascade parameters and the blade coordinates are presented in Table 1 and Appendix C, respectively. The notation used in Table 1 is defined in Figure 8.

A total of seven runs was made using the Gostelow cascade. The key input parameters to these runs are given in Table 2.

Since the cascade is two dimensional, the solution domain is represented by a single layer of hexahedral superelements. Figure 7 shows the top view of the finite element mesh structure employed in the first four runs. A total of 105 hexahedral and 630 tetrahedral elements is used in these runs giving rise to a total of 837 nodes, 66 of which fall on the blade surfaces. This represents the finest mesh possible on the UNIVAC 1106 of the Boğaziçi University Computer Center. Runs No. 1-3 are the primary test cases which provide comparison of the results from the present code with the exact solution for three different inlet flow angles. In all the runs except Run No. 4 the blade surfaces are approximated by isoparametric elements while the internal domain is described in terms of subparametric elements. Run No. 4 employs subparametric elements throughout and aims to reveal the resulting effect on accuracy. Run No. 5 involves the same grid configuration as in the

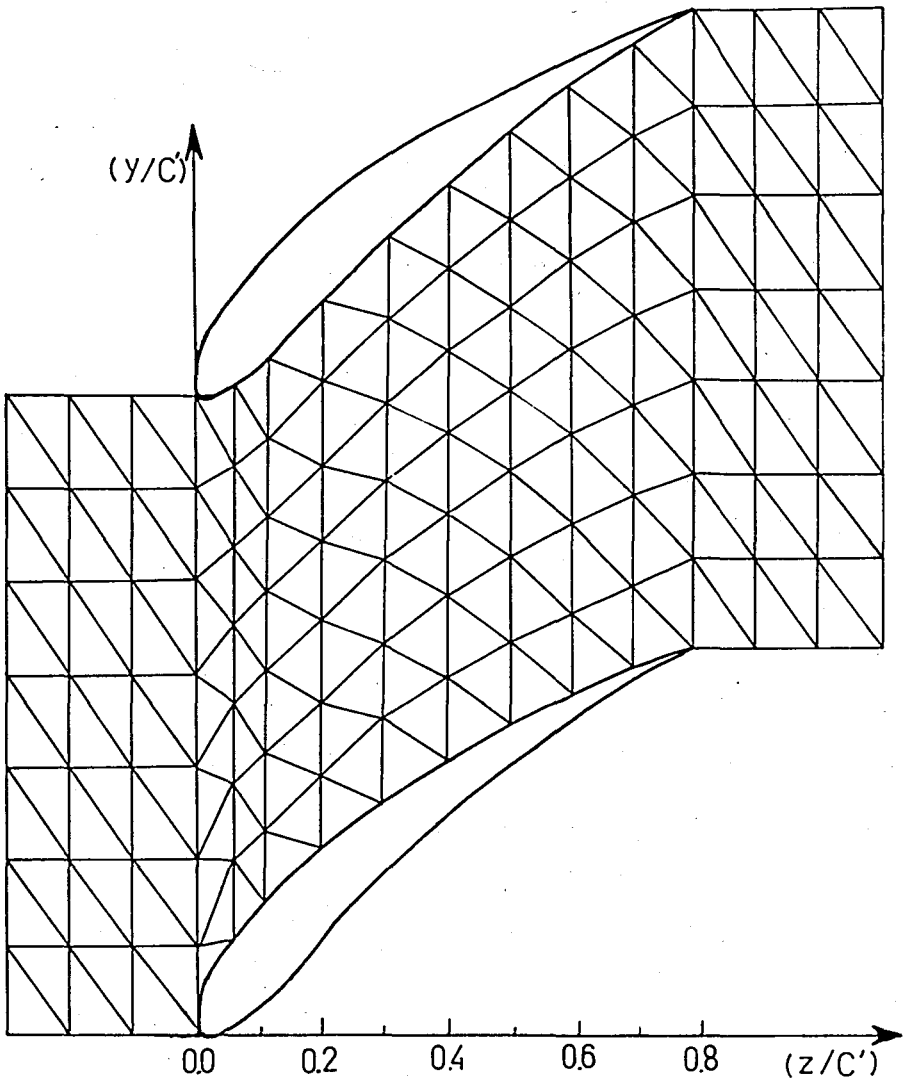

FIGURE 7 - Top View of the Finite Element Mesh Structure Employed for the Gostelow Cascade

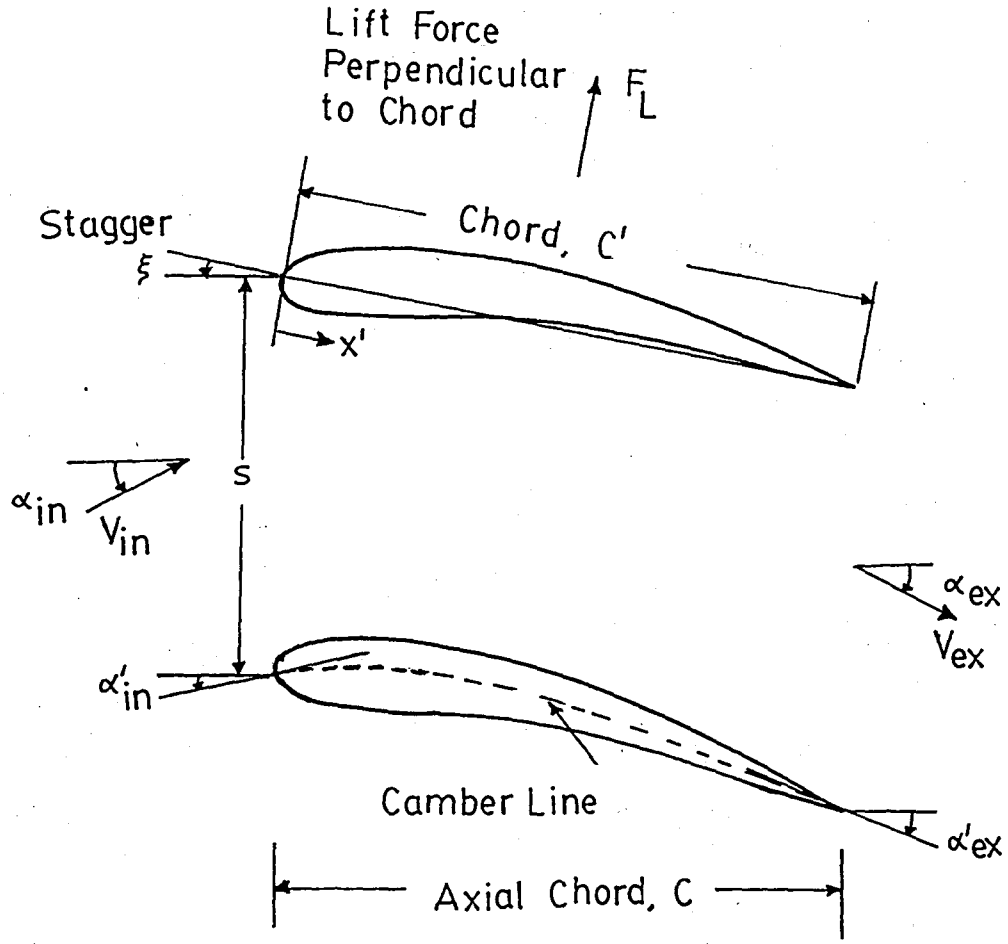

FIGURE 8 - Cascade Nomenclature

TABLE 1 - Gostelow Cascade Parameters

Axial Chord to Chord Ratio (C/C')	Space to Chord Ratio (S/C')	Inlet Blade Angle α'_{in} (deg.)	Outlet Blade Angle α'_{ex} (deg.)	Stagger Angle ξ	Blade Profile
0.79335	0.9901573	52.9	-20.8	-37.5	See Appendix C

TABLE 2 - Key Parameters Used in the Gostelow Cascade Validation Runs

Run No.	Inlet Mach No.	Inlet Flow Angle α_1 (deg.)	No. of Hexahedra	Subdivision of Hexahedra	No. of Tetrahedra	Type of Tetrahedra	No. of Nodes
1	Incomp.	47.5	105	R	630	I + S	837
2	Incomp.	53.5	105	R	630	I + S	837
3	Incomp.	59.0	105	R	630	I + S	837
4	Incomp.	53.5	105	R	630	S	837
5	Incomp.	53.5	105	X	630	I + S	837
6	Incomp.	53.5	15	R	90	I + S	231
7	Incomp.	53.5	50	R	150	I + S	693

I : Isoparametric

S : Subparametric

R : Regular subdivision - Top view

X : Irregular subdivision - Top view

preceding four cases. The only difference is in the subdivision pattern of the superelements. Each hexahedral superelement is subdivided along its longer diagonal thus producing tetrahedral elements with high aspect ratio. This run is intended to establish the effect of element aspect ratio. Runs No. 6 and 7 have coarser meshes (See Figure 9) and thus reflect the effect of grid resolution.

The key output parameters for these computer runs are given in Table 3. Also shown in the table are the numbers of the figures in which the calculated blade surface pressure distributions are compared with the corresponding exact solution. The lift coefficient, C_L, is defined as follows,

$$C_L = \frac{F_L}{\frac{1}{2} \rho V_{in}^2 C'} \tag{82}$$

where F_L denotes the lift force perpendicular to the blade chord and C' denotes the blade chord (See Figure 8).

C_L is calculated by integrating the pressure distribution over the blade surface and using the above definition.

The surface pressure distributions are expressed in terms of the pressure coefficient, C_p, defined as follows,

$$C_p = \frac{P - P_{in}}{\frac{1}{2} \rho V_{in}^2} \tag{83}$$

Using the nondimensional quantities introduced in Chapter III, the above definition may be rewritten as

$$C_p = (1 - V^2) \tag{84}$$

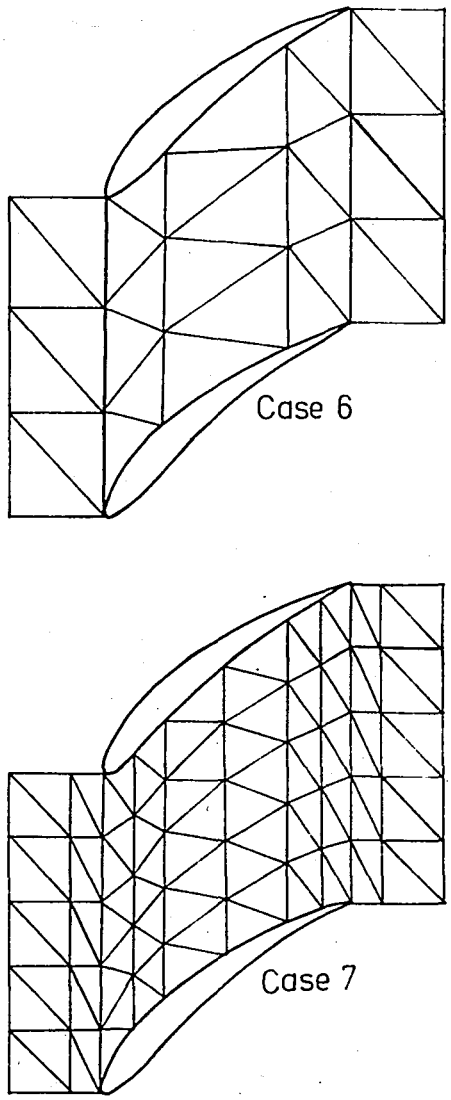

FIGURE 9 - Top View of the Coarse Meshes Employed in Cases 6 and 7 for the Gostelow Cascade

TABLE 3 - Key Results of the Gostelow Cascade Validation Runs

Run No.	Figure No.	Flow Exit Angle α_{ex} (deg.)	Flow Deflection $\delta = \alpha_{in} + \alpha_{ex}$	Lift Coefficient, C_L		% Error	Computer Execution Time (min)
				Present Code	Exact Solution		
1	10	-29.5	18.0	0.59	0.616	4.22	35
2	11	-30.0	23.5	0.71	0.7448	4.67	35
3	12	-30.4	28.6	0.72	0.84	14.29	35
4	13	-30.0	23.5	0.666	0.7448	10.58	31
5	14	-30.0	23.5	0.652	0.7448	12.19	35
6	15	-30.0	23.5	0.654	0.7448	12.19	3
7	15	-30.0	23.5	0.706	0.7448	5.2	12

where again, for simplicity, no symbolic distinction is drawn between dimensional and nondimensional quantities.

Now let us turn our attention to the comparison of calculated surface pressure distributions and exact solution for the aforementioned computer runs. Figure 10 shows that good agreement exists for the inlet angle of 47.5^{o}. As indicated by Figures 11 and 12, the agreement is also good for the inlet angles of 53.5^{o} and 59^{o} on the pressure side. However, as the inlet angle increases, the calculated pressure distribution on the suction surface tends to deviate from the exact solution. This is not surprising in light of the fact that the corresponding increase in flow deflection makes it difficult to calculate the accelerating flow near the leading edge of the suction surface. If it were possible to improve the grid further, the suction surface calculation would more accurately match the exact solution.

Figure 13 gives an interesting comparison between Runs No. 2 and 4 which approximate the blade surface with isoparametric and subparametric elements, respectively. Results indicate that the use of subparametric elements deteriorates the solution near the leading edge considerably. Evidently, this is a critical region which requires high grid resolution due to rapidly changing flow and forbids use of subparametric elements due to large surface curvature.

Figure 14 shows the adverse effect of changing the internal division pattern of the hexahedral superelements. The resulting increase in the aspect ratios of the tetrahedral elements causes a reduction in the accuracy of calculations. As before, the region most affected is the blade leading edge.

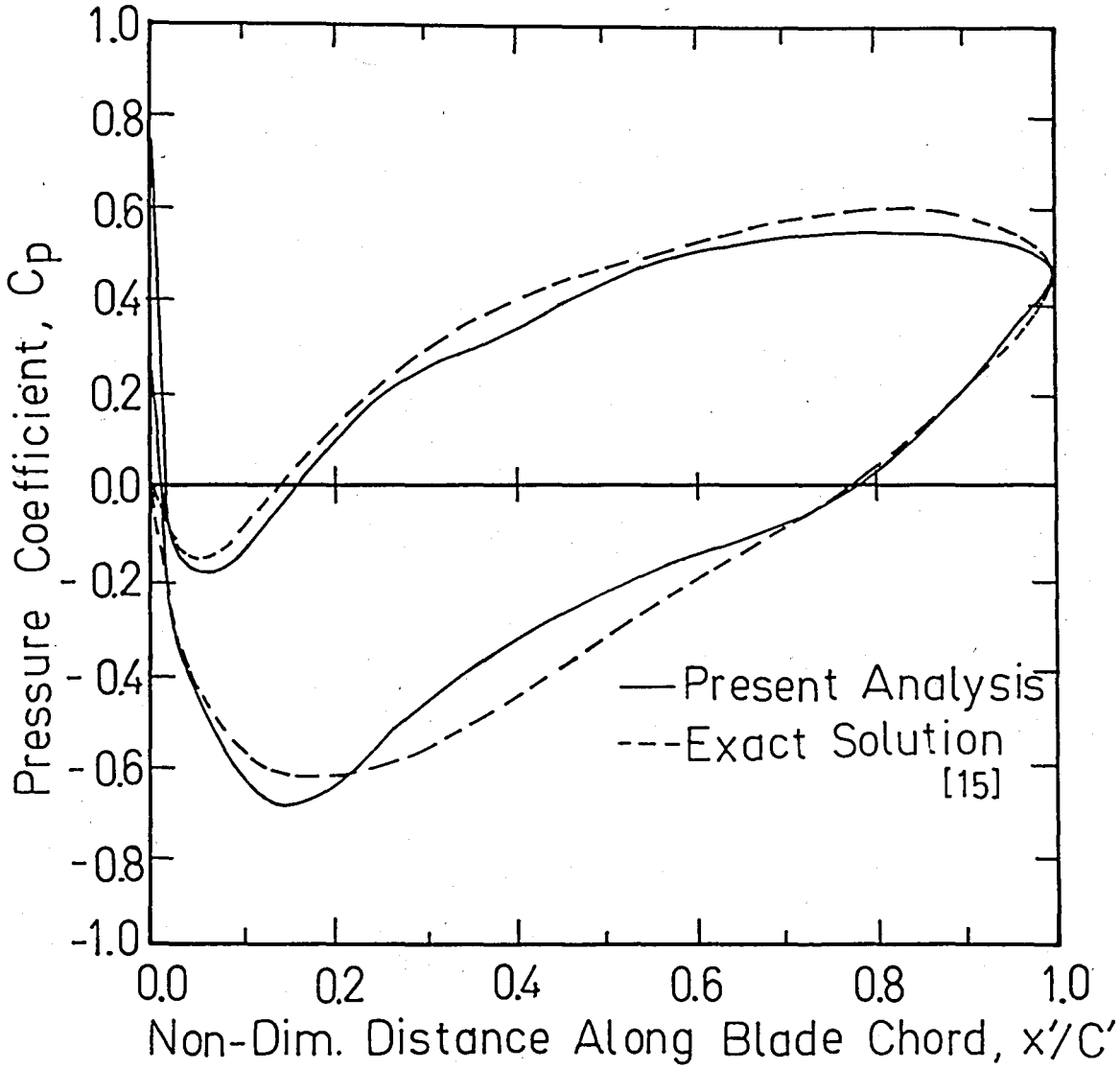

FIGURE 10 - Comparison of Pressure Coefficient Distributions Calculated in the Present Analysis and the Exact Solution for $\alpha_{in} = 47.5^{o}$ (Case 1)

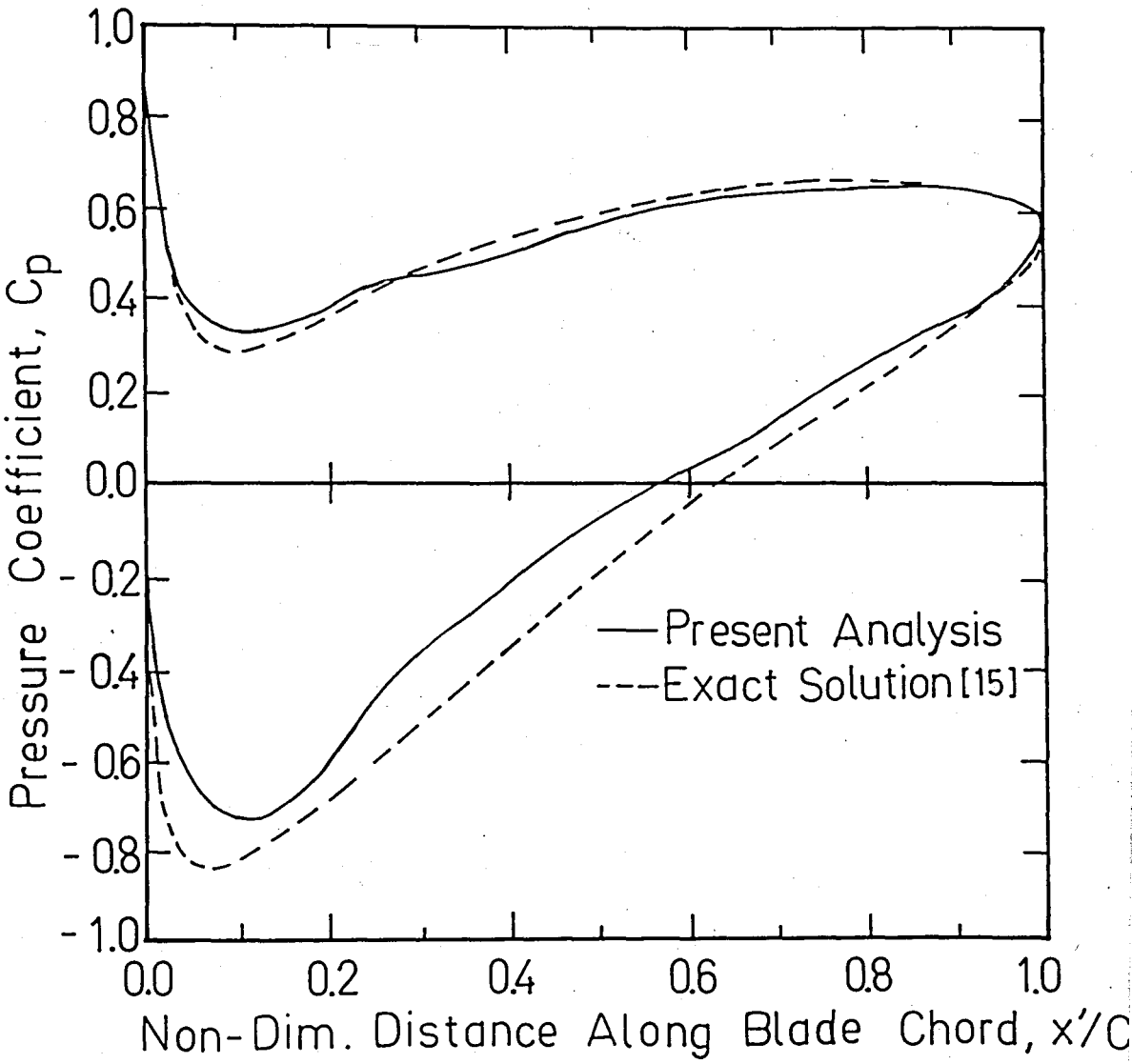

FIGURE 11 - Comparison of Pressure Coefficient Distributions Calculated in the Present Analysis and the Exact Solution for $\alpha_{in} = 53.5^{0}$ (Case 2)

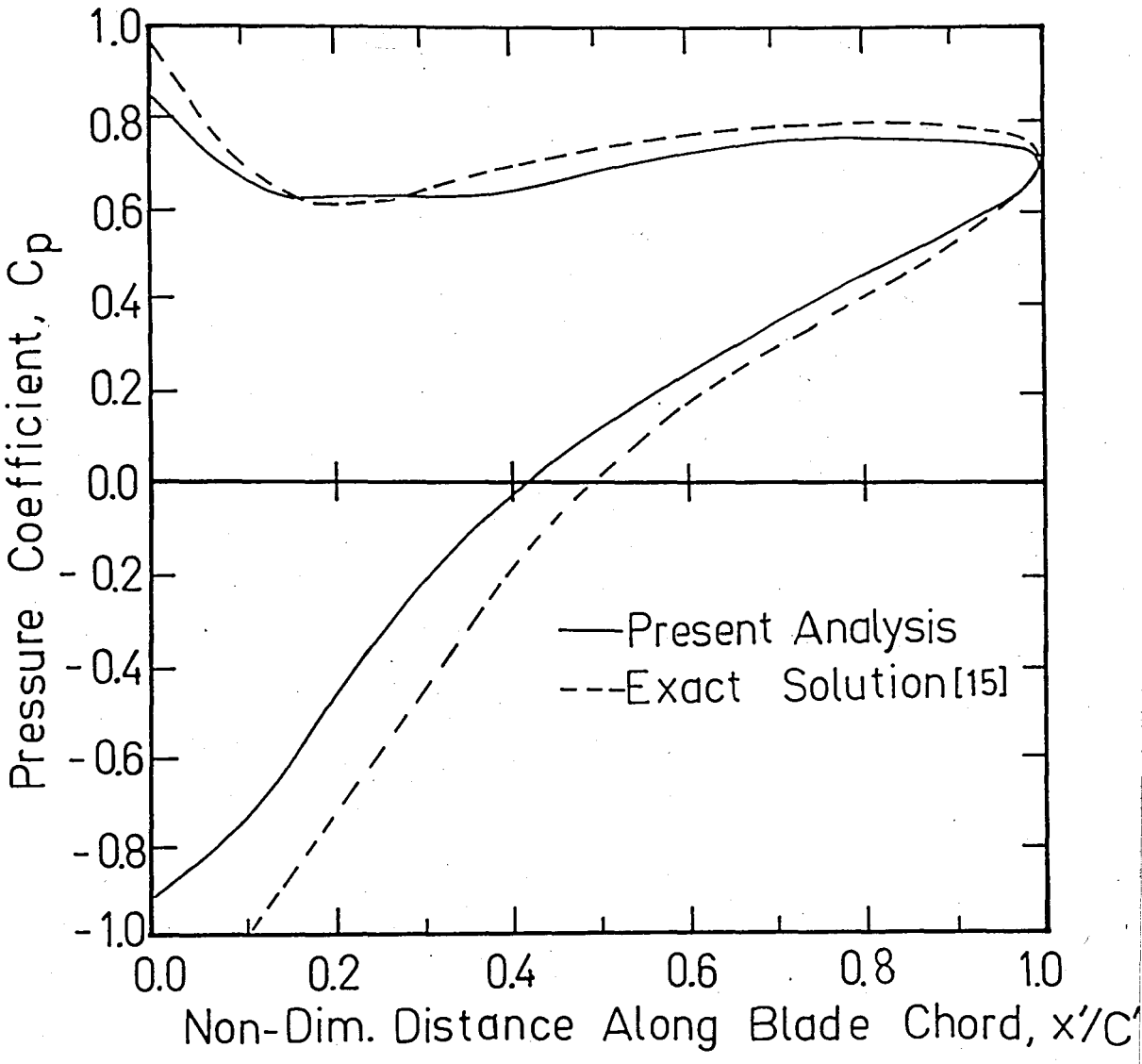

FIGURE 12 - Comparison of Pressure Coefficient Distributions Calculated in the Present Analysis and the Exact Solution for $\alpha_{in} = 59^{o}$ (Case 3)

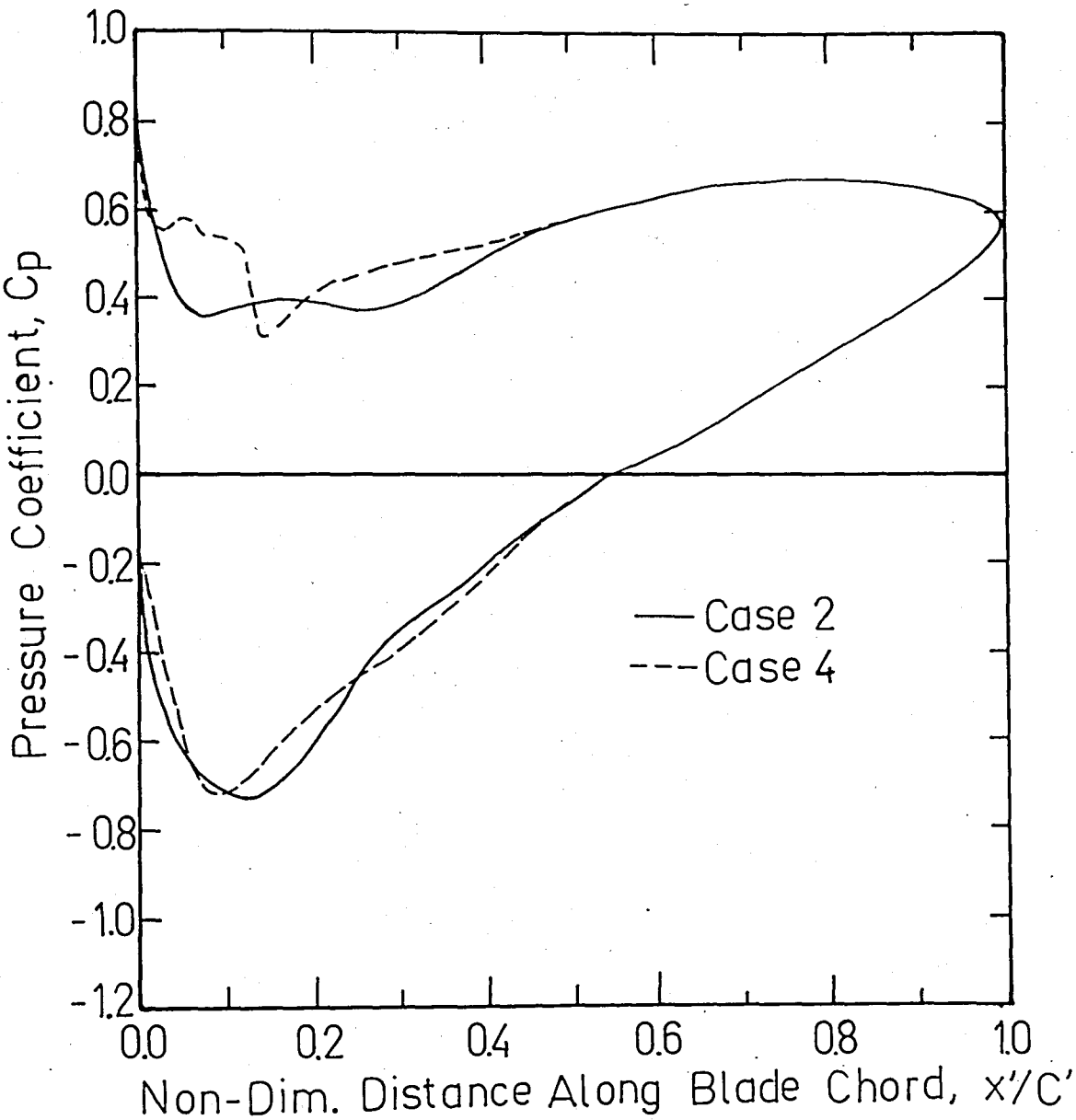

FIGURE 13 - Comparison of Pressure Coefficient Distributions for Cases 2 and 4 ($\alpha_{in} = 53.5^{o}$)

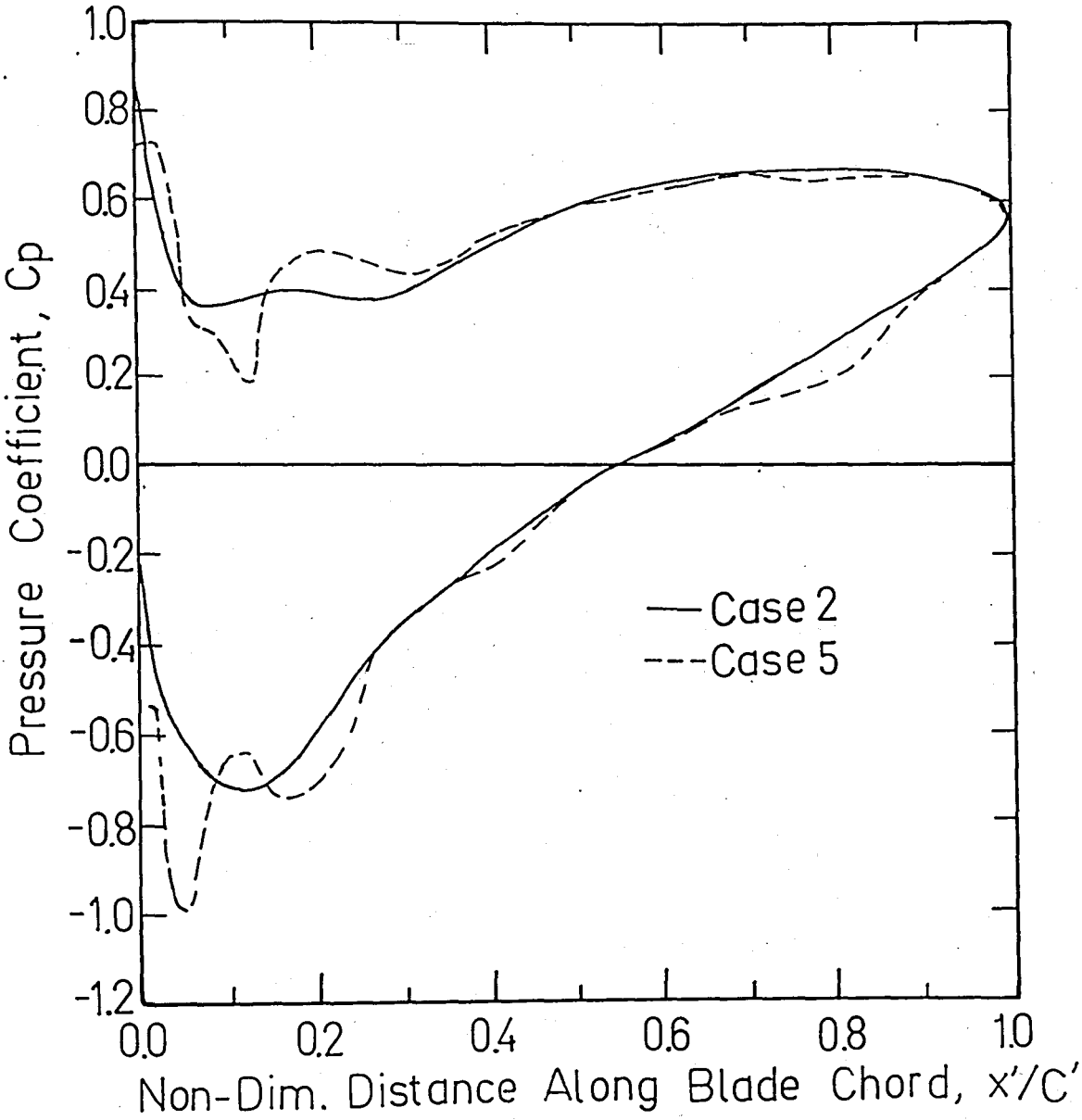

FIGURE 14 - Comparison of Pressure Coefficient Distributions for Cases 2 and 5 ($\alpha_{in} = 53.5^{o}$)

The remaining two cases involve coarser mesh configurations which highlight the effect of grid resolution. As indicated in Figure 15, the effect of reducing resolution is to degrade the predictions at the leading edge. The accuracy at the trailing edge is surprisingly good even for the coarsest mesh. It is interesting to note that between Runs No. 2 and 7 there corresponds only a 0.5 % improvement in lift coefficient prediction to a threefold increase in execution time (See Table 4).

The results with the Gostelow cascade suggest that to obtain good numerical results four items become critical:

1. Sufficiently fine mesh configuration within the core memory limitations of the existing computer,
2. Finer resolution of the grid near the blade leading edge,
3. Isoparametric elements at the blade surface, and
4. Domain subdivision into elements with aspect ratio as near unity as possible.

4.2 APPLICATIONS TO AXIAL-FLOW TURBINES AND COMPARISON WITH EXPERIMENTS AND TWO-DIMENSIONAL "STREAMSURFACE" CALCULATIONS

The calculations with the Gostelow cascade have verified the applicability of the code to two-dimensional incompressible flows. We now proceed to show its validity in two-dimensional compressible flow calculations.

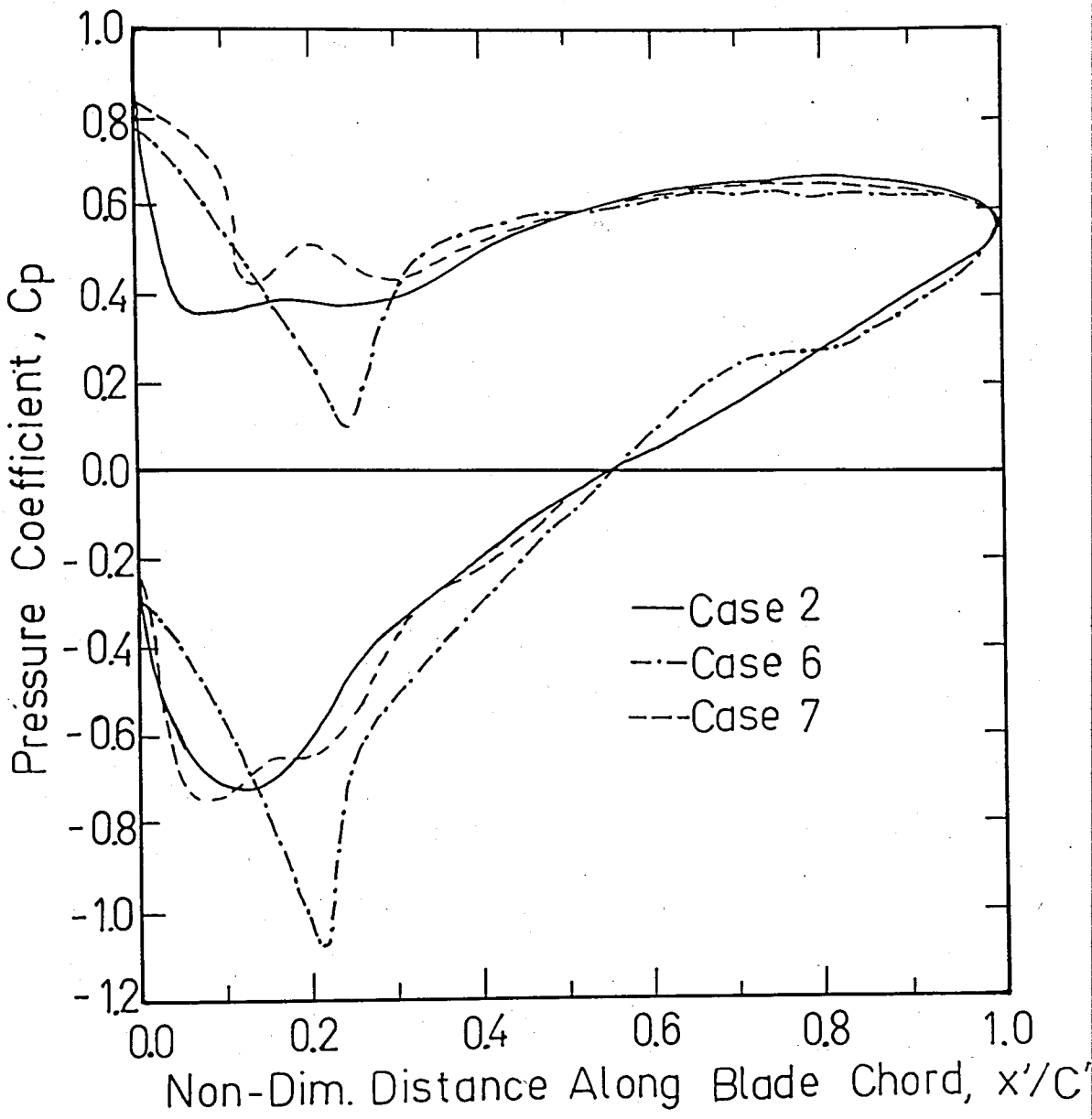

FIGURE 15 - Comparison of Pressure Coefficient Distributions for Cases 2, 6 and 7 ($\alpha_{in} = 53.5^{o}$)

For this purpose we consider three runs concerning an experimental turbine stator [45] used at the NASA Lewis Research Center and the first stage stator and rotor [46] of a large electric utility turbine. The validity of the code is tested by using the surface velocities measured at the midstreamsurface of the experimental turbine [45] and the two-dimensional midstreamsurface calculations of Katsanis' code [19]. Experimental verification is not possible for the latter two cases (the first stage rows of the electric utility turbine) as no data are available at present. However, comparison with the TSONIC code widely used in turbomachine calculations lends confidence in the compressible flow capability of the present code. The TSONIC code obtains transonic flow solution in two stages: (1) the weight flow is reduced sufficiently so the flow is completely subsonic throughout the passage and a solution is obtained to the streamfunction equation based on this reduced weight flow, and (2) the transonic velocity distribution is determined based on the actual weight flow by means of a velocity gradient solution.

Figures 16-18 show the mesh configurations used in the three runs. These configurations composed of two layers of 4 x 15 hexahedral superelements are the finest that could be achieved on the existing computer. Since the results are to be compared with data on the midstreamsurface, it is not necessary to treat the entire span of the blade-to-blade channel. The solution region considered in each case is indicated in the corresponding figure. It is worth mentioning that subparametric elements are used everywhere except at blade surfaces which are approximated by isoparametric elements. It is also important to note that from the two alternative subdivision patterns of hexahedral superelements the one which produces the

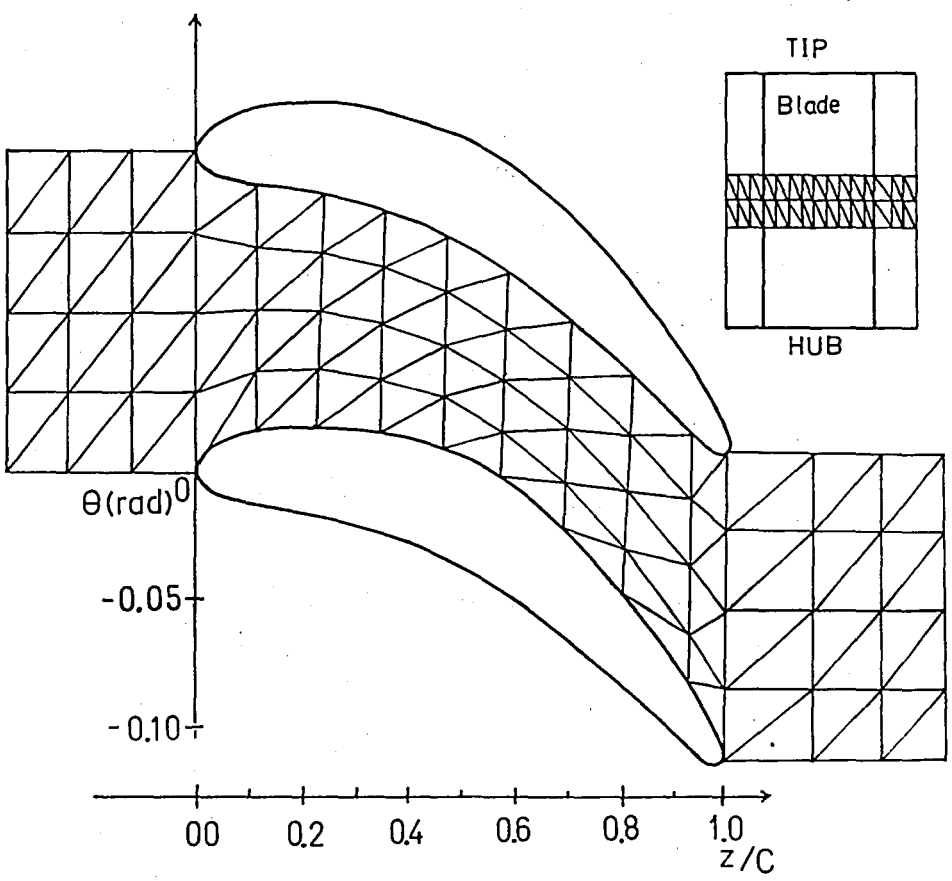

FIGURE 16 - Top View of the Finite Element Mesh Employed for the Experimental Turbine Stator

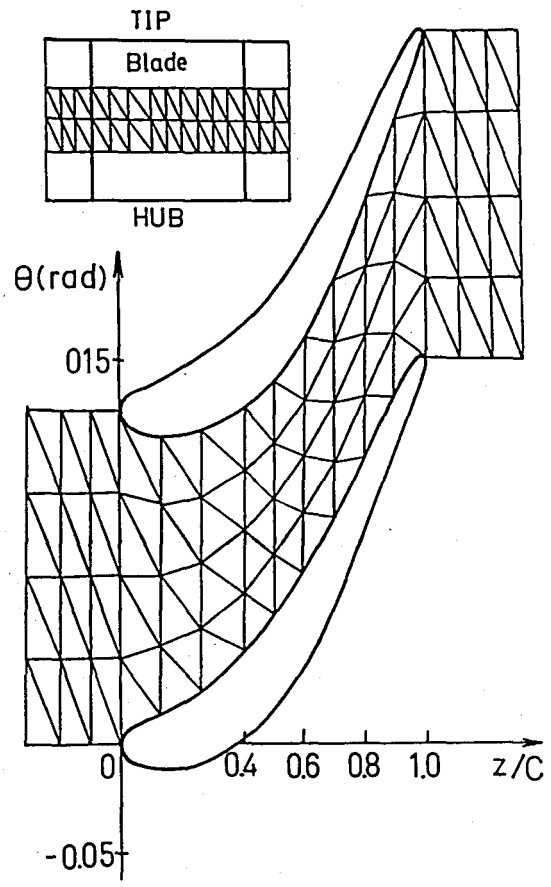

FIGURE 17 - Top View of the Finite Element Mesh Employed for the First Stage Stator of the Electric Utility Turbine

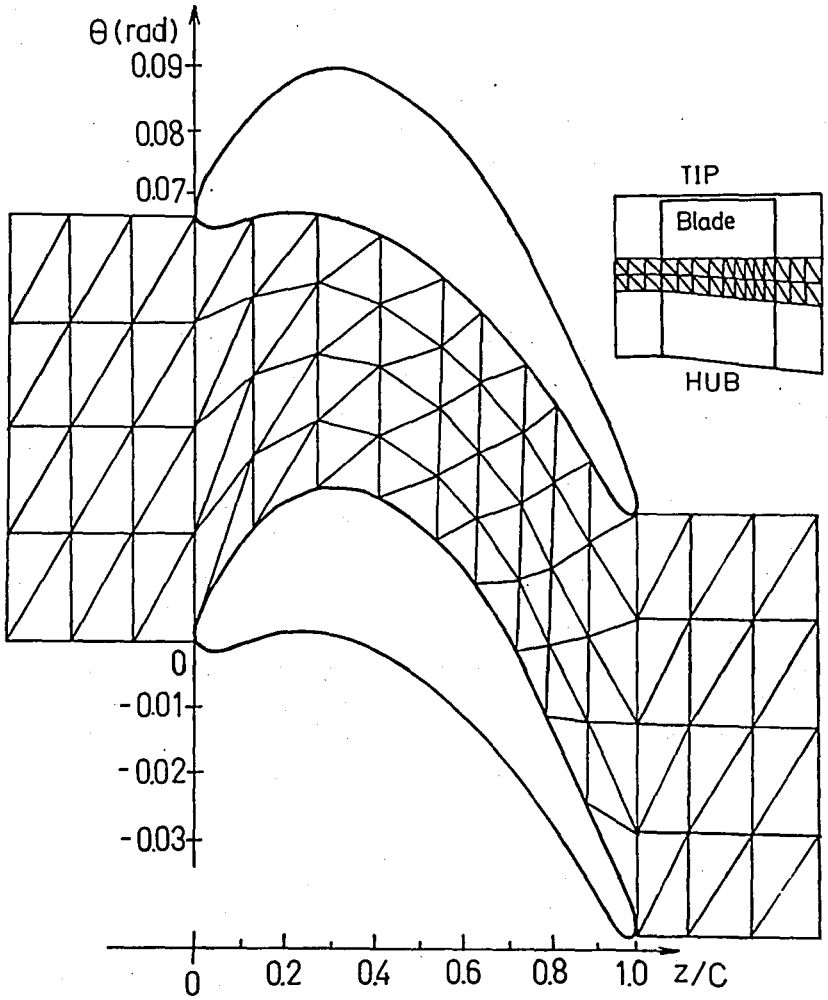

FIGURE 18 - Top View of the Computational Mesh Employed for the First Stage Rotor of the Electric Utility Turbine

optimum aspect ratio is chosen. Some important geometric parameters and the blade coordinates are given in Table 4 and Appendix C, respectively. Table 5 summarizes the key input parameters to these runs.

The key output parameters obtained from these computer runs are given in Table 6. This table also shows the numbers of the figures that compare the blade surface velocity distributions calculated by the present code with the experimental data and/or the results from the TSONIC code. It might be mentioned that all of the three runs have converged after four density iterations, the maximum relative change in velocity potential in the final iteration being less than 2 %.

Figure 19 shows that the present results for Run No. 1 agree very well with the experiments and the TSONIC predictions obtained on a 20x63 finite difference grid. The truly excellent agreement with the measured velocities near the leading and the trailing edges is especially noteworthy in view of the usual difficulty for potential flow methods to treat these regions. In fact, the trailing edge calculation of the TSONIC code is way off. The pressure and suction surface velocities calculated by the present code merge smoothly at the trailing edge showing that the Kutta condition is fulfilled.

Figure 20 shows the constant Mach number contours corresponding to Run No. 1. The Mach level is low at the inlet but increases in the blade passage going up to 0.9 on the suction side of the blade trailing edge. There is a considerable decrease in the Mach level right after the trailing edge probably due to the absence of any viscous blockage effects in the analysis.

TABLE 4 - Geometric Parameters at Midstreamsurface for Two-Dimensional Compressible Flow Runs

Run Description No.	No. of Blades	Chord C',(ft)	Axial Chord C', (ft)	Blade Inlet Angle α_{in} (deg.)	Blade Exit Angle α_{ex} (deg.)	Stagger	Blade Profile
1 Experimental Stator	50	0.0576	0.043	0	59	Clock-wise	See Appen-
2 Turbine Stator	48	0.579	0.417	0	-60	Counter-clock-wise	dix C
3 Turbine Rotor	95	0.323	0.297	32	49	Clock-wise	

TABLE 5 - Key Input Parameters to Two-Dimensional Compressible Flow Runs

Run No.	Rotational Speed,(rpm)	M_{in}	α_{in}(deg.)	No. of Hexahedra	No. of Tetrahedra	Subdivision of Hexahedra	No. of Nodes
1	0.0	0.21	0.0	120	720	C	1395
2	0.0	0.21	0.0	120	720	CC	1395
3	3600.0	0.90	66.7	120	720	C	1395

C: Subdivision for clockwise stagger - Top view

CC: Subdivision for counterclockwise stagger - Top view

TABLE 6 - Key Output Parameters from Two-Dimensional Compressible Flow Runs

Run No.	Figure No.	M_{ex}	α_{ex}(deg.)	$\delta = \alpha_{in} + \alpha_{ex}$	Computer Execution Time (min)
1	19	0.64	-67.0	-67.7	155
2	21	0.67	-66.7	-66.7	155
3	22	0.24	15.5	82.2	155

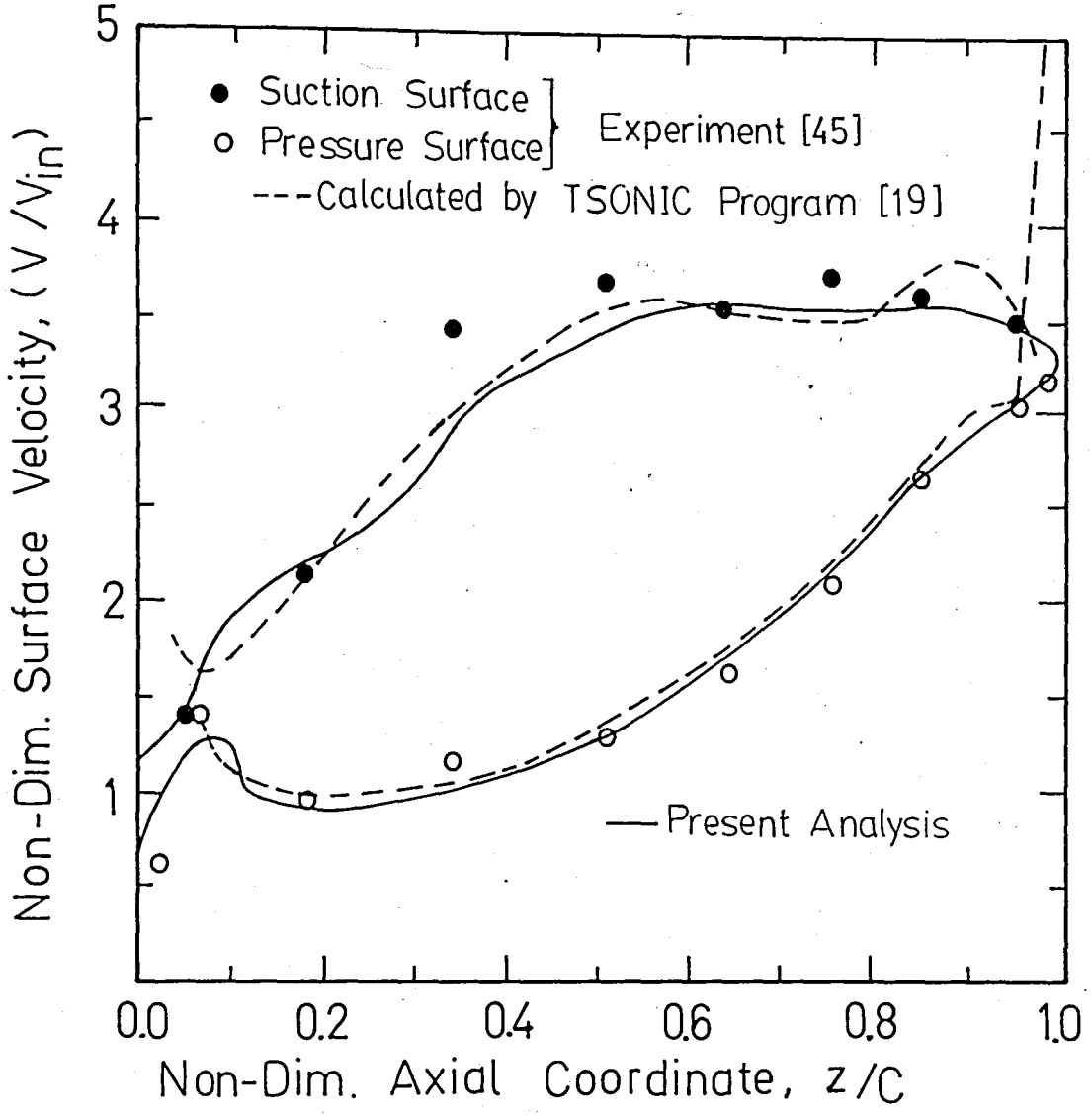

FIGURE 19 - Comparison of Measured Surface Velocities with Results Calculated by the Present Code and Those Calculated by TSONIC Code

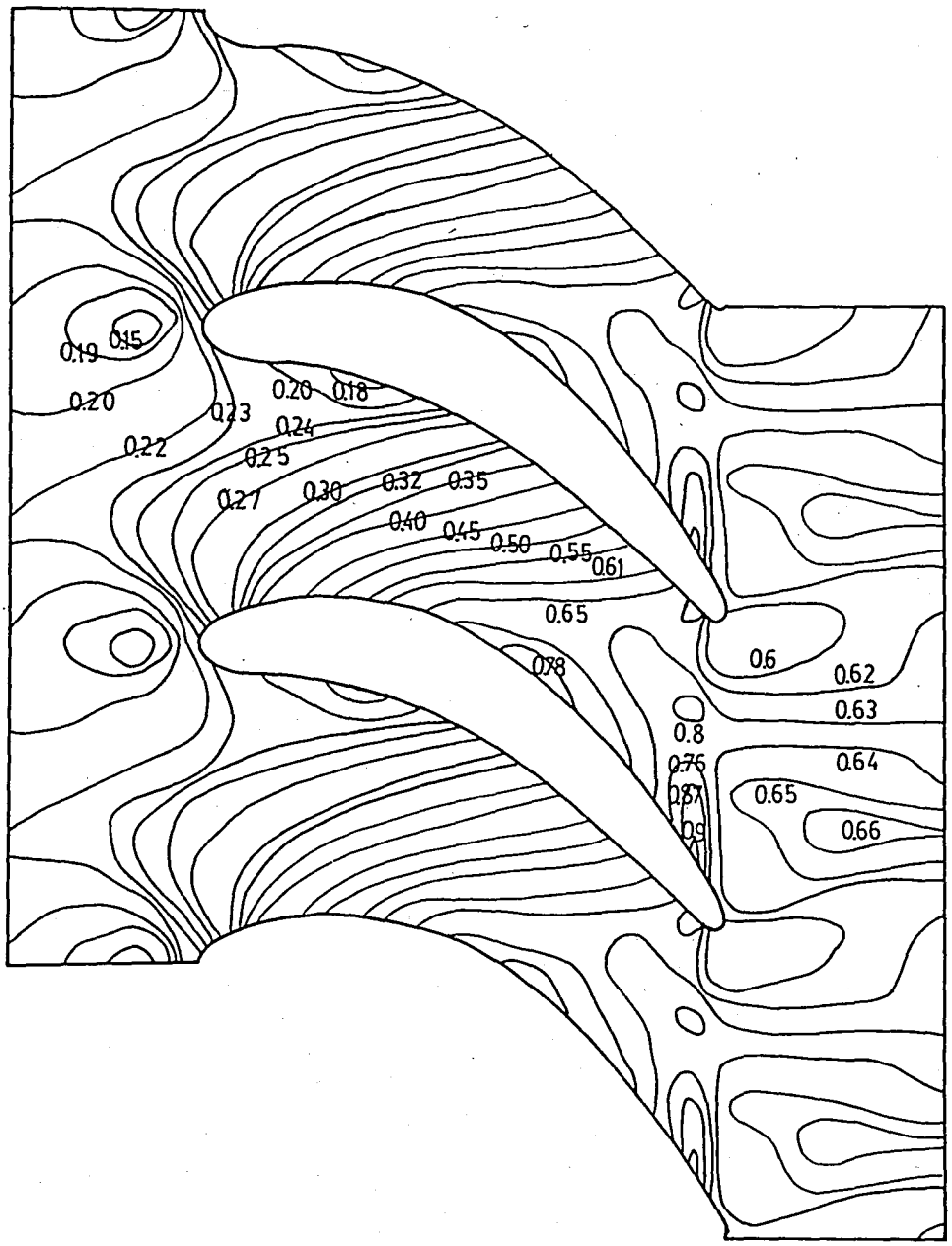

FIGURE 20 - Constant Mach Number Contours on the Midspan Blade-to-Blade Surface of Revolution of the Experimental Turbine Stator

Comparisons of the blade surface velocity distributions calculated by the present code and the TSONIC code for the first stage stator and rotor rows of the electric utility turbine are shown in Figures 21 and 22. The TSONIC calculations are based on a 20x43 finite difference grid. The agreement between the two methods is generally satisfactory. Results suggest that the maximum discrepancy occurs around the leading and trailing edges. At the trailing edges the TSONIC predictions are clearly in error on account of not meeting the Kutta condition. In the case of the rotor, the present code predicts the location of stagnation as somewhat detached from the leading edge. The validity of this prediction has been tested by applying Katsanis' MAGNFY code [47]. This code allows the coarse-mesh solution from the TSONIC code to be magnified by a chosen magnification factor in a small rectangular region. The MAGNFY code used by Gunes and Mengütürk [48] to obtain a detailed fine-mesh solution around the leading edge confirms the location of rotor blade stagnation indicated by the present code.

The midstreamsurface Mach contours for Runs No. 2 and 3 are plotted in Figures 23 and 24, respectively. It is important to note that the Mach levels calculated at the stator outlet do not match the rotor inlet conditions. It is believed that this particular behaviour can be attributed to the absence of viscous blockage effects in the analysis. In reality, the boundary layer build-up on the blades, the casing and the hub and the wake flow formation greatly reduce the effective flow area behind the blades resulting in increased outlet velocities. Thus, the inviscid flow solution fails to predict accurate Mach numbers at the outlet.

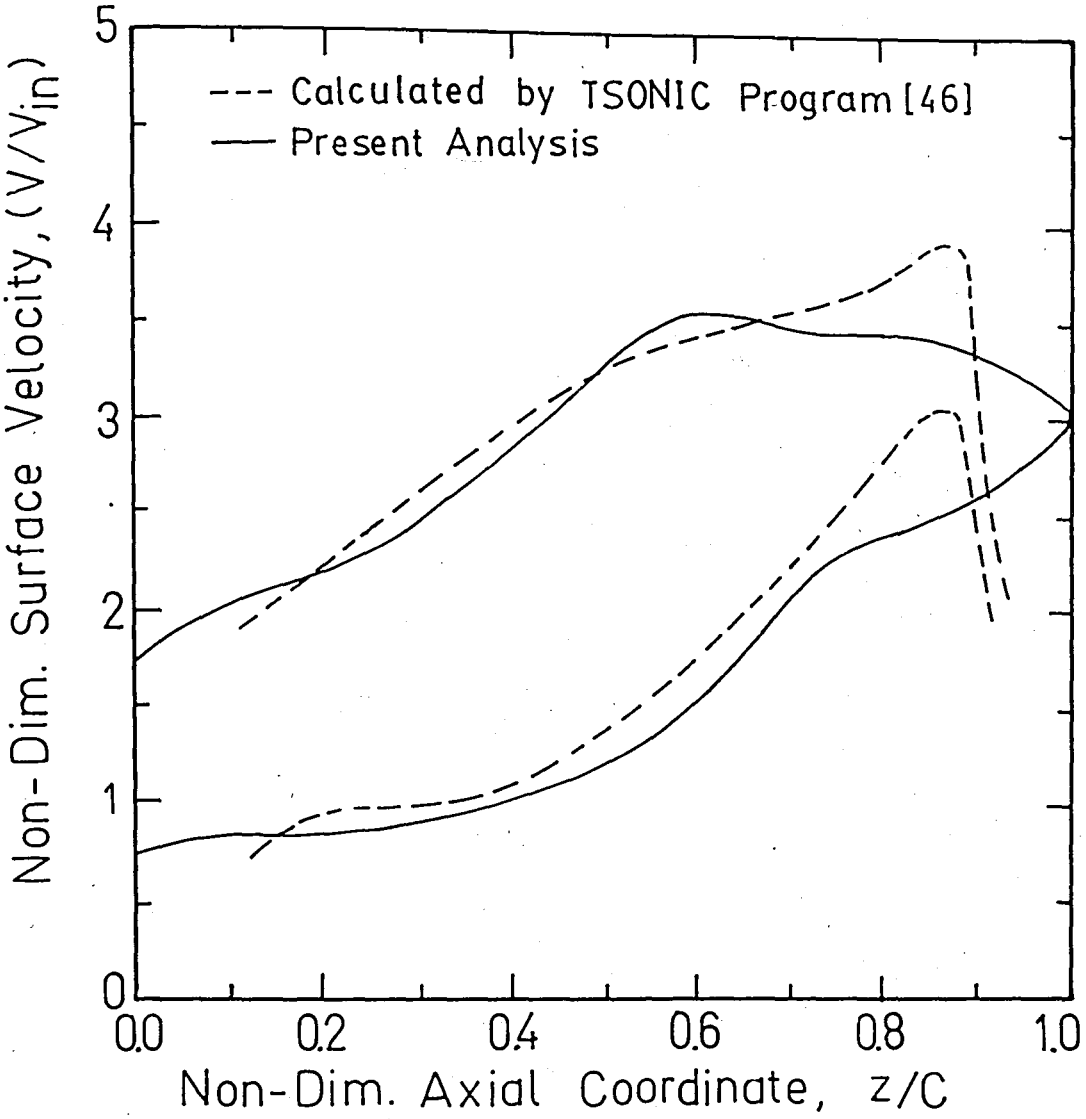

FIGURE 21 - Comparison of Blade Surface Velocity Distributions Calculated by the Present Code and the TSONIC Code for the Stator Row of the Electric Utility Turbine

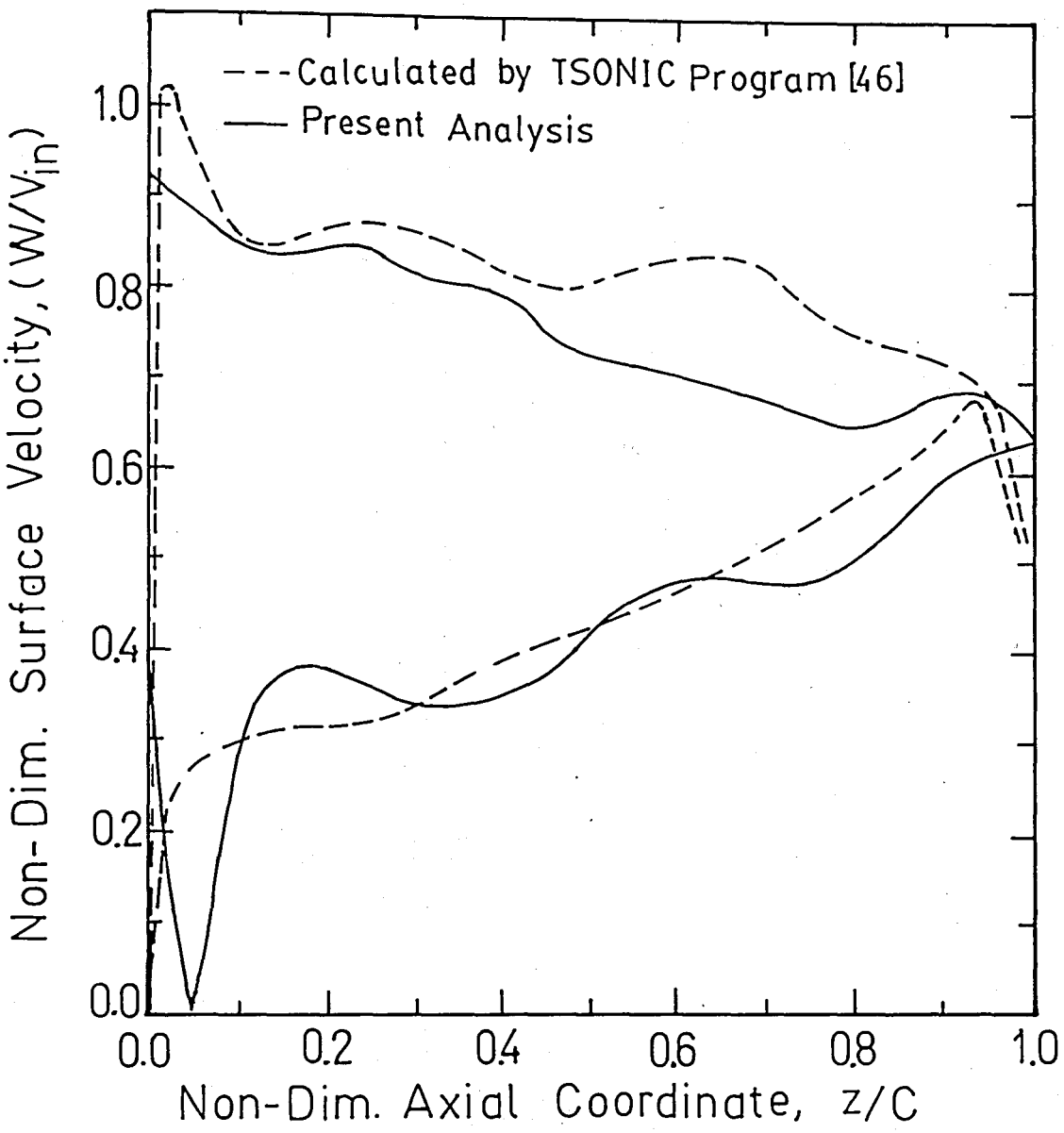

FIGURE 22 - Comparison of Blade Surface Velocity Distributions Calculated by the Present Code and the TSONIC Code for the Rotor Row of the Electric Utility Turbine

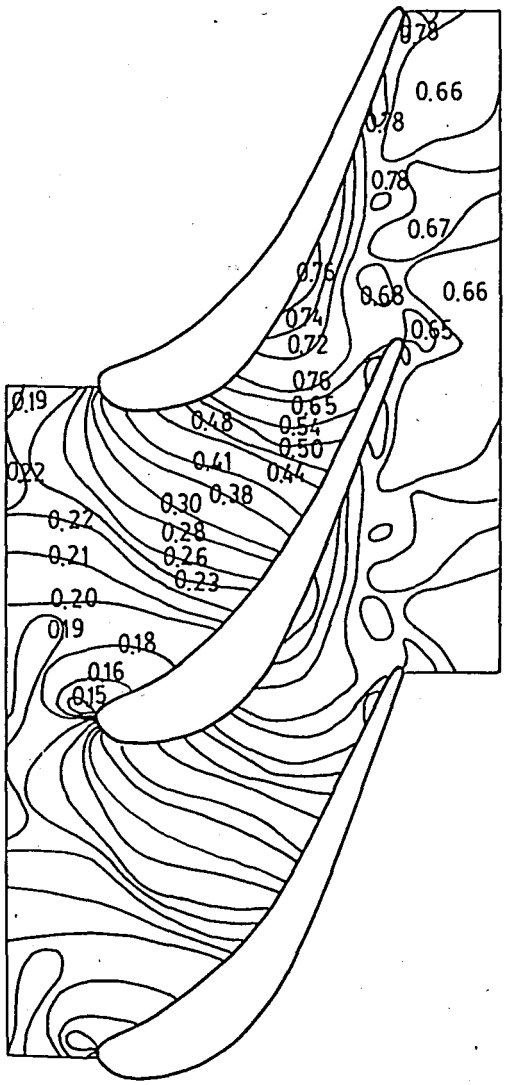

FIGURE 23 - Mach Number Contours on the Midspan Blade-to-Blade Surface of Revolution for the Stator Row of the Electric Utility Turbine

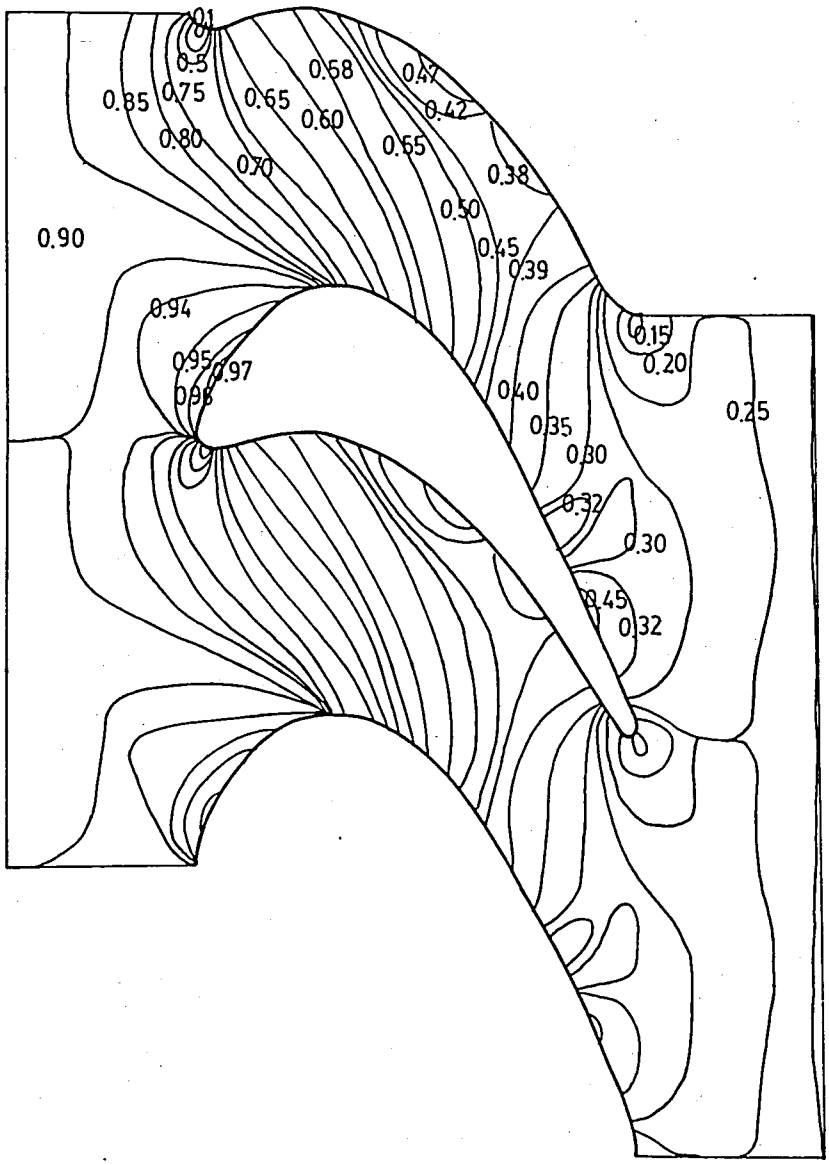

FIGURE 24 - Mach Number Contours on the Midspan Blade-to-Blade Surface of Revolution for the Rotor Row of the Electric Utility Turbine

4.3 APPLICATION TO A MIXED-FLOW TURBINE AND COMPARISON WITH THREE-DIMENSIONAL CALCULATIONS

The final test of the code is the demonstration of its ability to predict the three-dimensional flow. This is achieved by applying the code to the mixed-flow turbine rotor considered by Laskaris [41] and comparing the results obtained by both methods. Since Laskaris disclosed no geometric data on this turbine rotor, the coordinates of the blade profile and the hub-to-shroud walls had to be measured from the figures supplied in his paper. These measurements which inevitably involve some degree of uncertainty (± 0.25 cm) are given in Appendix C. The geometry constructed based on the measured coordinates is reproduced in Figures 25 and 26 which show, respectively, the blade-to-blade and the hub-to-shroud sections of the blade passage. Also shown in these figures is the mesh configuration used in the present calculations. This configuration consists of two layers of 4x15 hexahedral superelements which represents, as before, the best possible mesh on the existing computer. To cover the entire passage each hexahedral block is extended half the blade span making element aspect ratios too high. Had the machine core storage limitations permitted, it would have been appropriate to increase the number of layers used. The present mesh configuration involves 5x9x31 nodal points in the r, θ and z directions, respectively. This compares favorably well with the 5x9x29 noded mesh configuration used by Laskaris. It should be pointed out that Laskaris' analysis is based on Galerkin's method applied over 27 node second-order Lagrangian elements whereas the present method relies upon variational formulation over 27-node hexahedral superelements each made up of six basic tetrahedral elements.

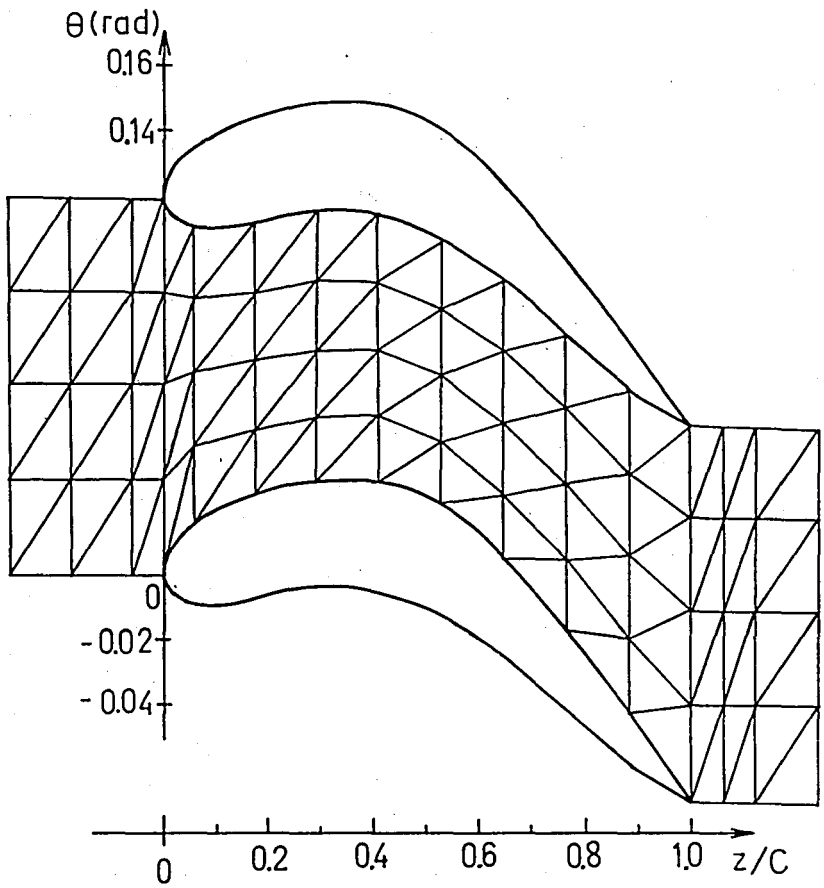

FIGURE 25 - Top View of the Finite Element Mesh Used for the Mixed-Flow Turbine

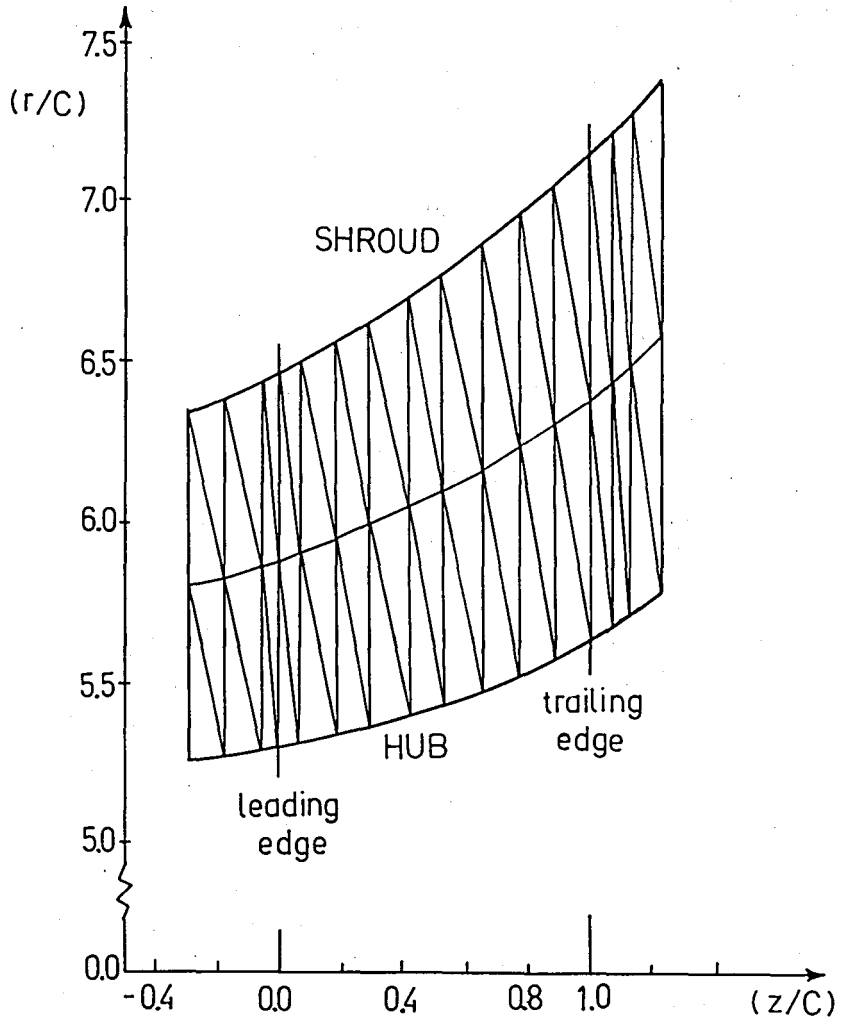

FIGURE 26 - Side View of the Finite Element Mesh Used for the Mixed-Flow Turbine

The key input parameters used in the computer run were estimated from the data given by Laskaris. These are shown in Table 7 together with some important output parameters. The total execution time of 164 minutes is to be compared with the 160 minutes required by the Laskaris code on the GE 605 computer.

TABLE 7 - Key Input-Output Parameters for Mixed-Flow Rotor

Key Input Parameters:

Mid-span						
$(V_r)_{in}$	$(V_\theta)_{in}$	$(V_z)_{in}$	ω(rpm)	No. of Hexahedra	No. of Tetrahedra	No. of Nodes
70 (m/s)	283 (m/s)	273 (m/s)	3600	120	720	1395

Key Output Parameters:

Mid-span			No. of Iterations	Computer Execution Time
$(V_r)_{ex}$	$(V_\theta)_{ex}$	$(V_z)_{ex}$		
140 (m/s)	140 (m/s)	170 (m/s)	4	164 min

Comparisons of blade surface pressure distributions are presented in Figures 27-29 at three different sections: hub, mid-span and shroud. In these figures pressures are nondimensional with respect to the rotor inlet pressure (1.04×10^6 N/m^2). Laskaris' results originally given as normalized with respect to a reference pressure of 1.0×10^6 N/m^2 are reproduced here in the new format. It is noted that the present results are qualitatively in agreement with Laskaris' calculations. Essentially similar trends are indicated by both methods. For example, it is observed that

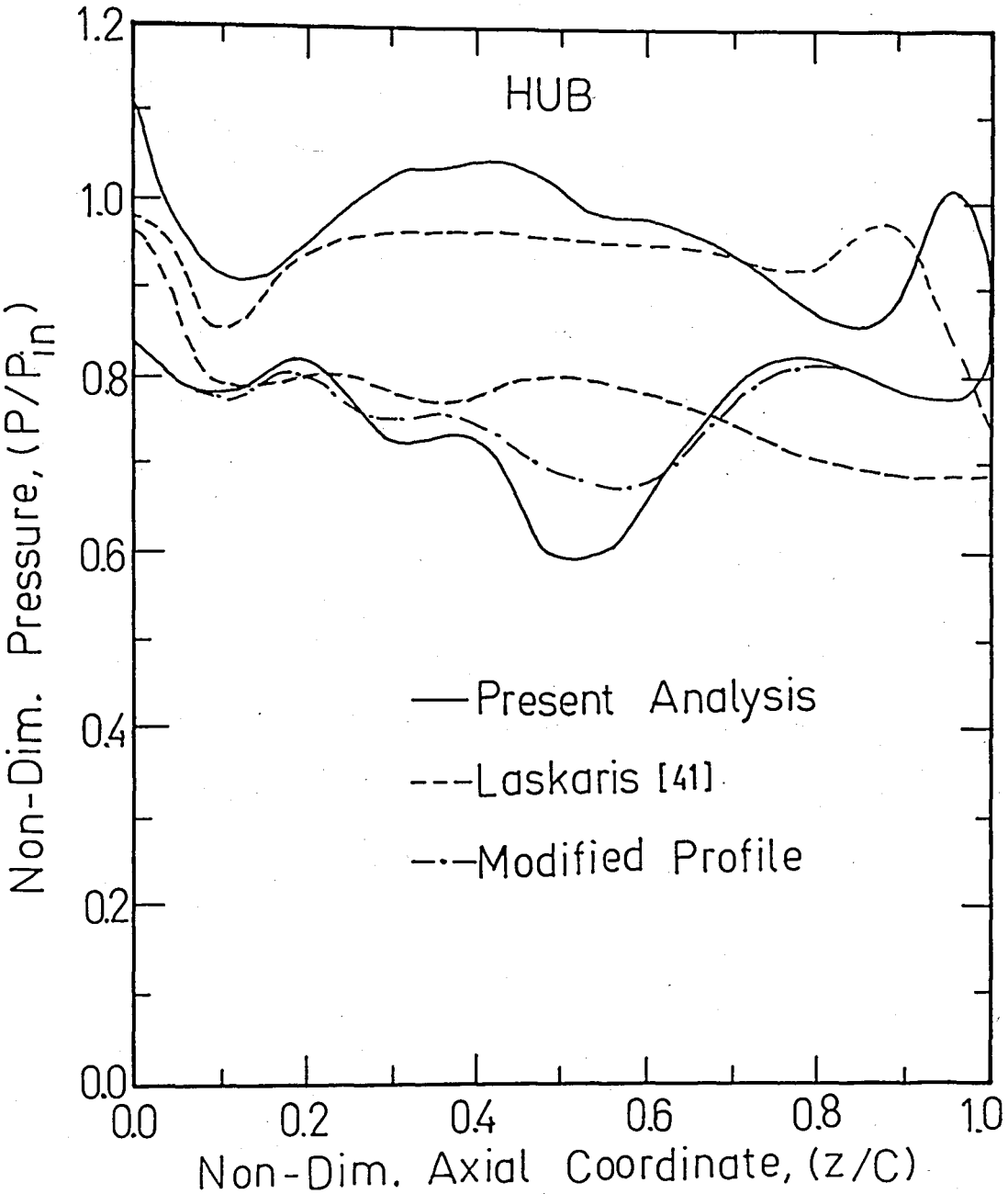

FIGURE 27 - Comparison of Surface Pressure Distributions Obtained by the Present Code and Laskaris' Code for the Mixed-Flow Turbine at the Hub

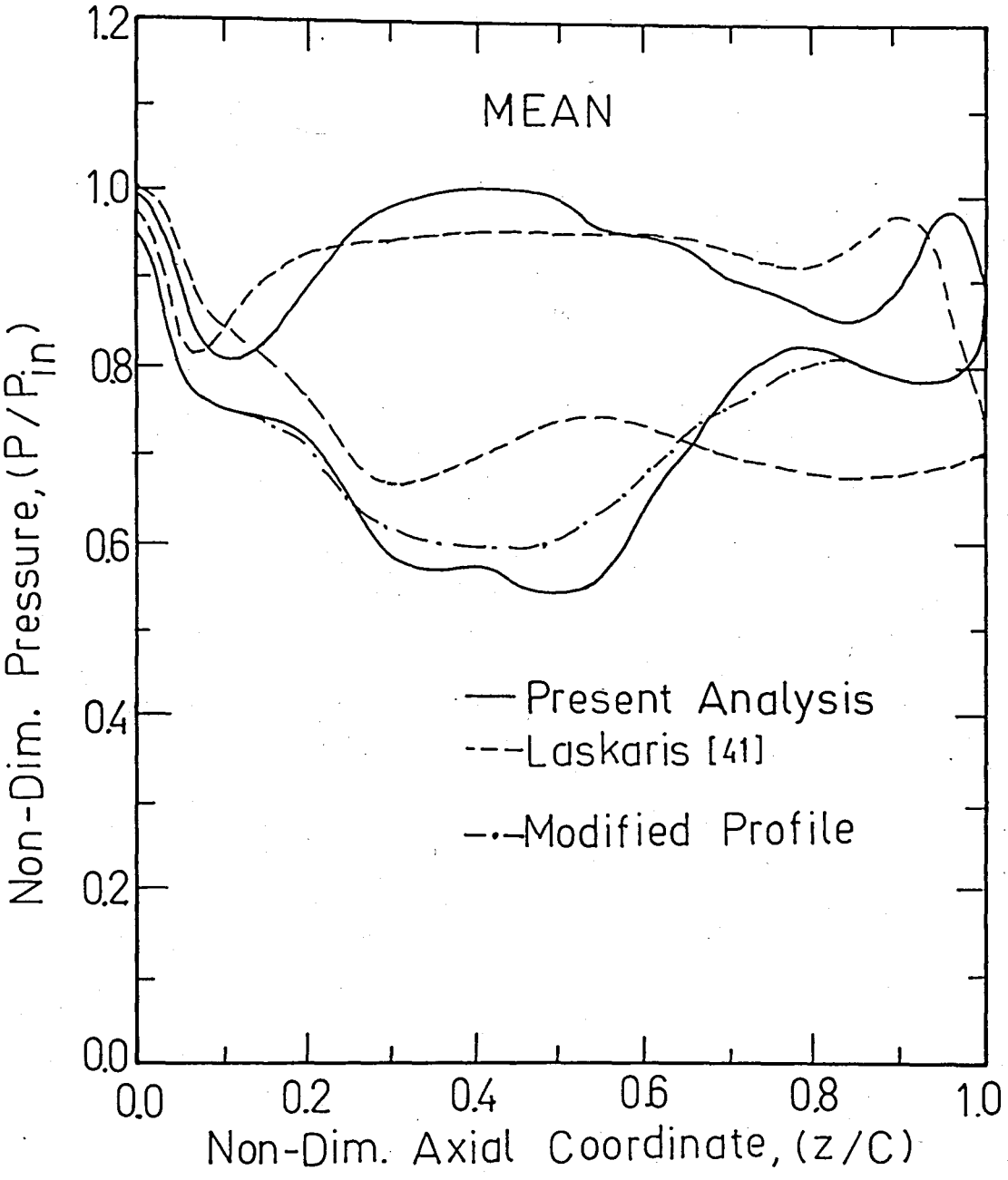

FIGURE 28 - Comparison of Surface Pressure Distributions Obtained by the Present Code and Laskaris' Code for the Mixed-Flow Turbine at the Midspan

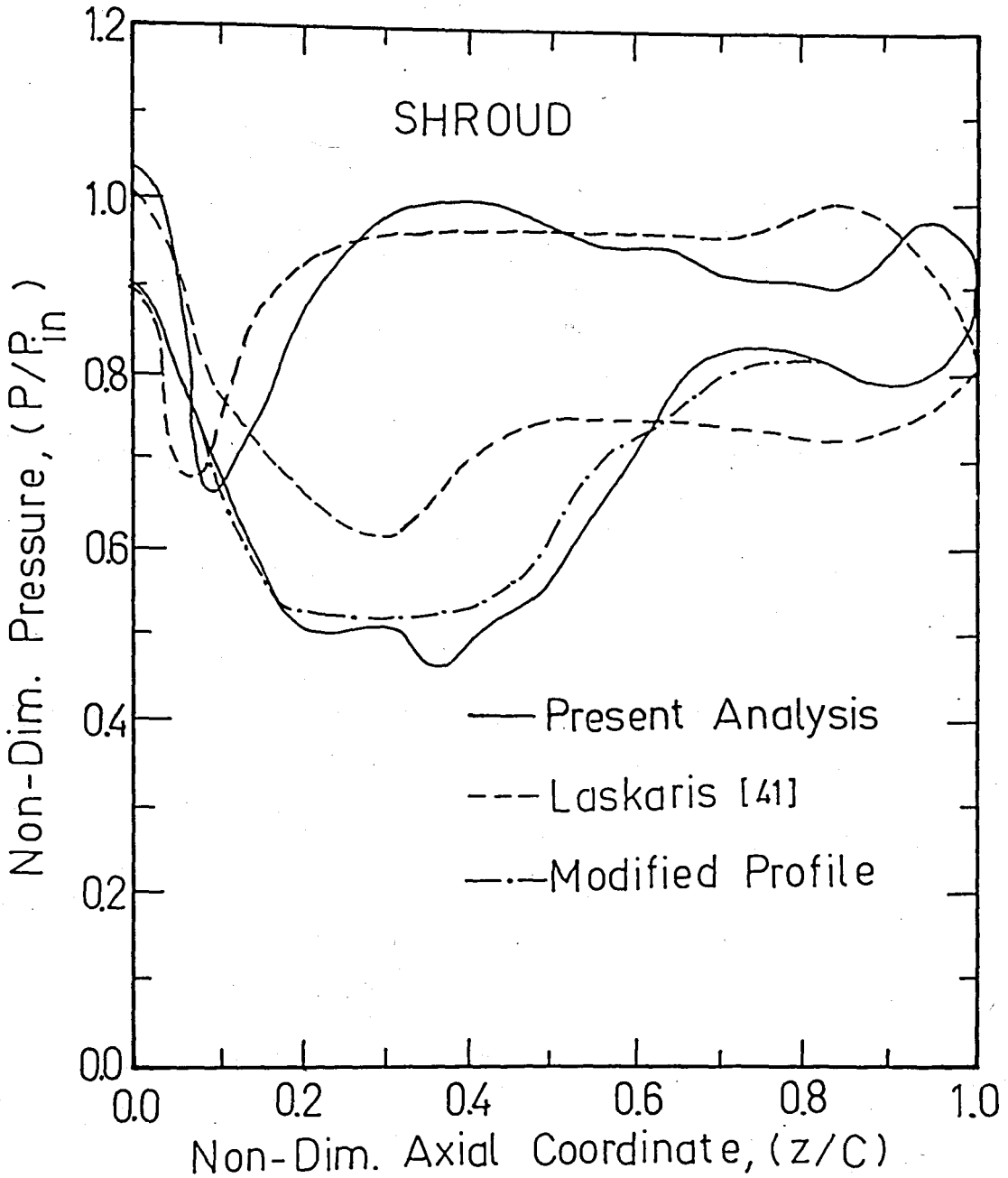

FIGURE 29 - Comparison of the Surface Pressure Distributions Obtained by the Present Code and Laskaris' Code for the Mixed-Flow Turbine at the Shroud

surface pressure gradients tend to increase toward the shroud. This is true for both the positive and the negative pressure gradients. Since the former indicates possibility of flow separation, it may be concluded that the separation tendency increases with radius in this turbine. At the shroud section Laskaris predicts a critical zone with adverse pressure gradient on the pressure surface near the blade nose whereas our calculations indicate an additional one on the suction surface halfway between the leading and trailing edges. Interestingly, the two solution methods display satisfactory agreement around the blade nose in spite of the fact that this is often the most problematic area in numerical calculations. The differences increase after approximately 20 % axial chord from the leading edge. The maximum deviations occur on the suction surface at about 60 % axial chord at the hub section, 50 % at the mid-span and 40 % at the shroud. The agreement on the pressure surface is relatively satisfactory except toward the trailing edge where the Kutta condition appears to indicate somewhat different exit conditions. As an overall observation our calculations seem to be consistently overpredicting the variations in pressure. This consistent pattern leads us to suspect that the profiles considered by the two codes are slightly different as a result of the errors that may be involved in reproducing Laskaris' blade profile. For example, it may be argued that the over-acceleration occuring over the suction surface may be reduced by shaving off this surface slightly. To support this idea and also to demonstrate the ability of the code as a design tool, calculations have been repeated using the modified profile shown in Figure 30. The maximum change made in the blade profile is -0.004 radians (corresponding to -0.23 cm at the

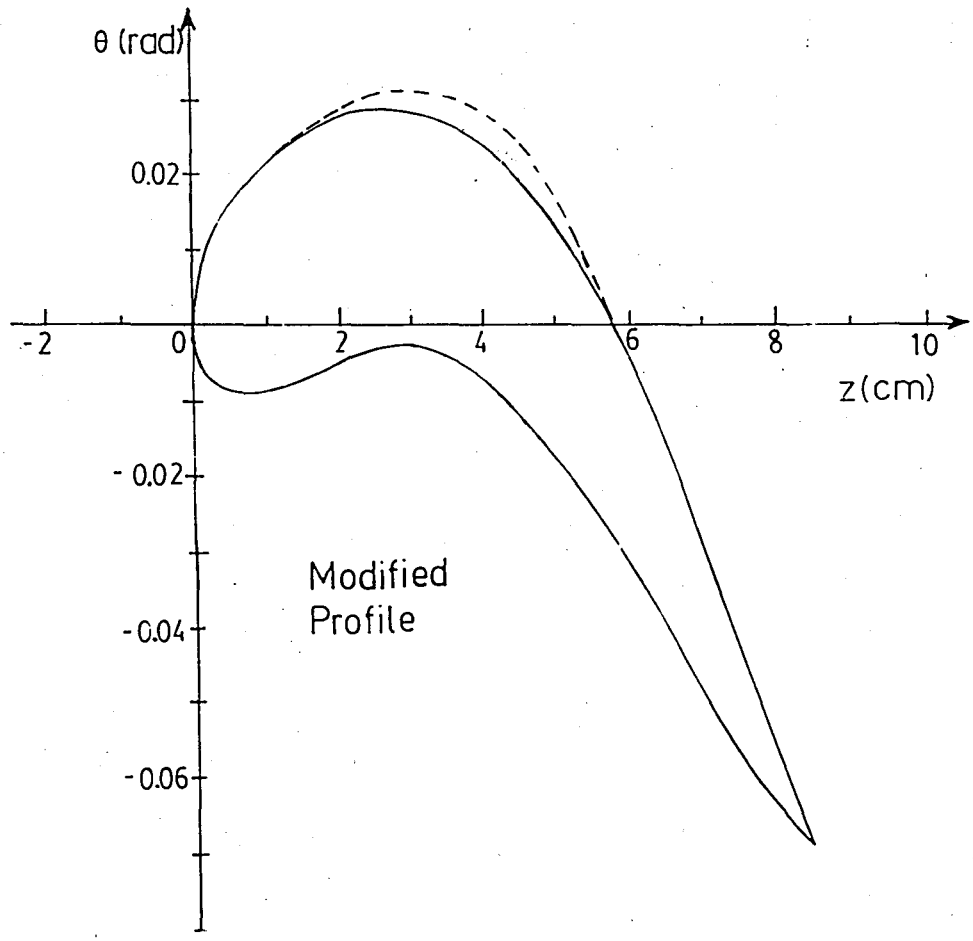

FIGURE 30 - Modified Profile for the Mixed-Flow Turbine

shroud) which is within the estimated error band of ±0.25 cm. Results, also included in Figures 27-29, exhibit considerable improvement.

Figures 31, 32 and 33 show the relative velocity vectors calculated by the present code, respectively, on the surface midway between two successive blades extending from hub to shroud, on the midspan blade-to-blade surface of revolution and on the passage cross-sectional plane at the blade trailing edge. These figures illustrate some of the three-dimensional aspects of the flow.

The blade-to-blade Mach number contours are shown in Figures 34 and 35 for the hub and shroud sections, respectively. Figure 36 shows the Mach number contours calculated on the surface midway between blades. The contours calculated by Laskaris are also plotted at the bottom of each figure, for comparison. It is seen that results are in good agreement in both pattern and magnitude.

Finally, Figure 37 depicts the Mach number contours on the passage cross-sectional plane at the trailing edge for which no data is available for comparison.

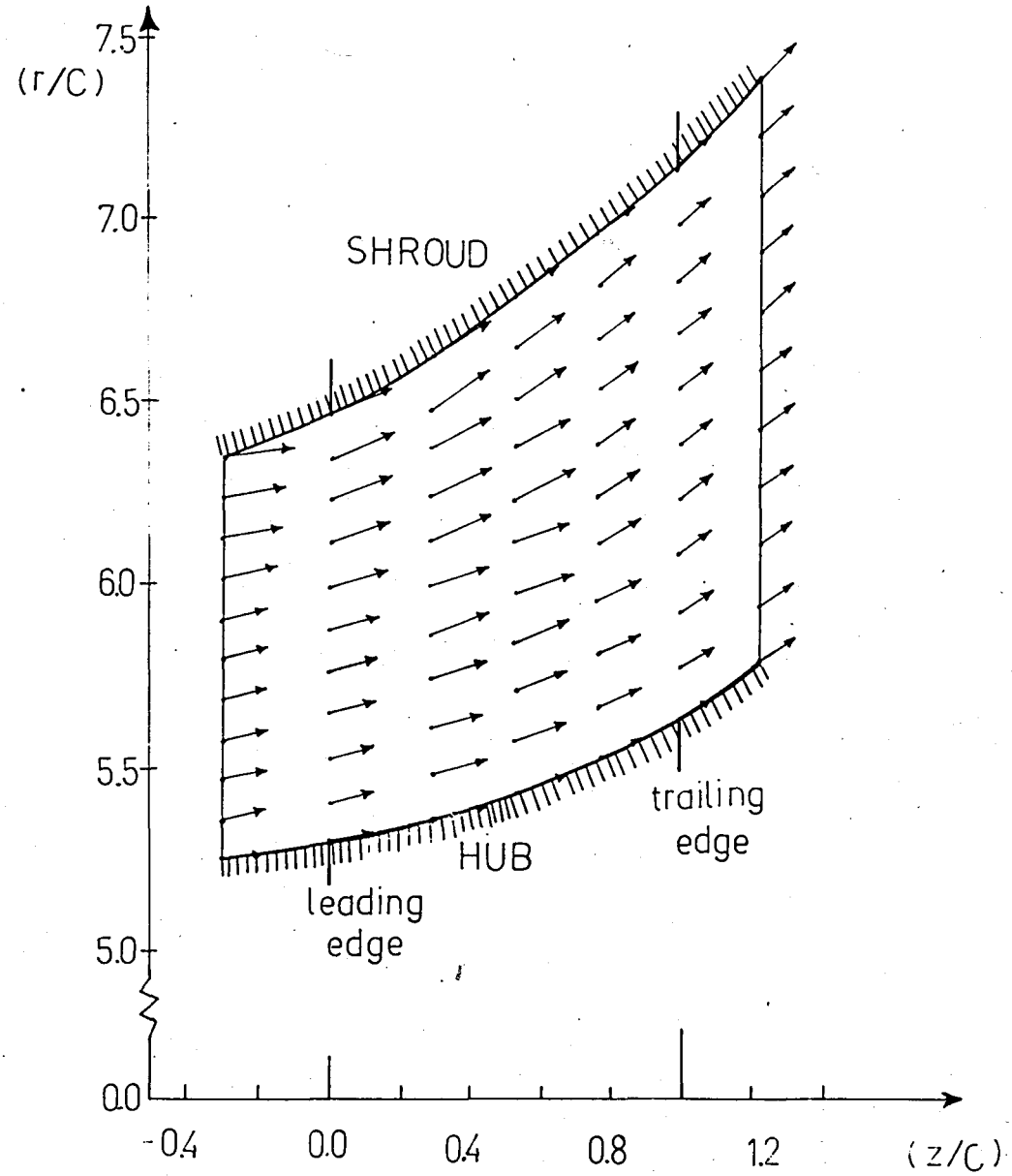

FIGURE 31 - Relative Velocity Vectors Calculated by the Present Code on the Midway Between Two Blades for the Mixed-Flow Turbine

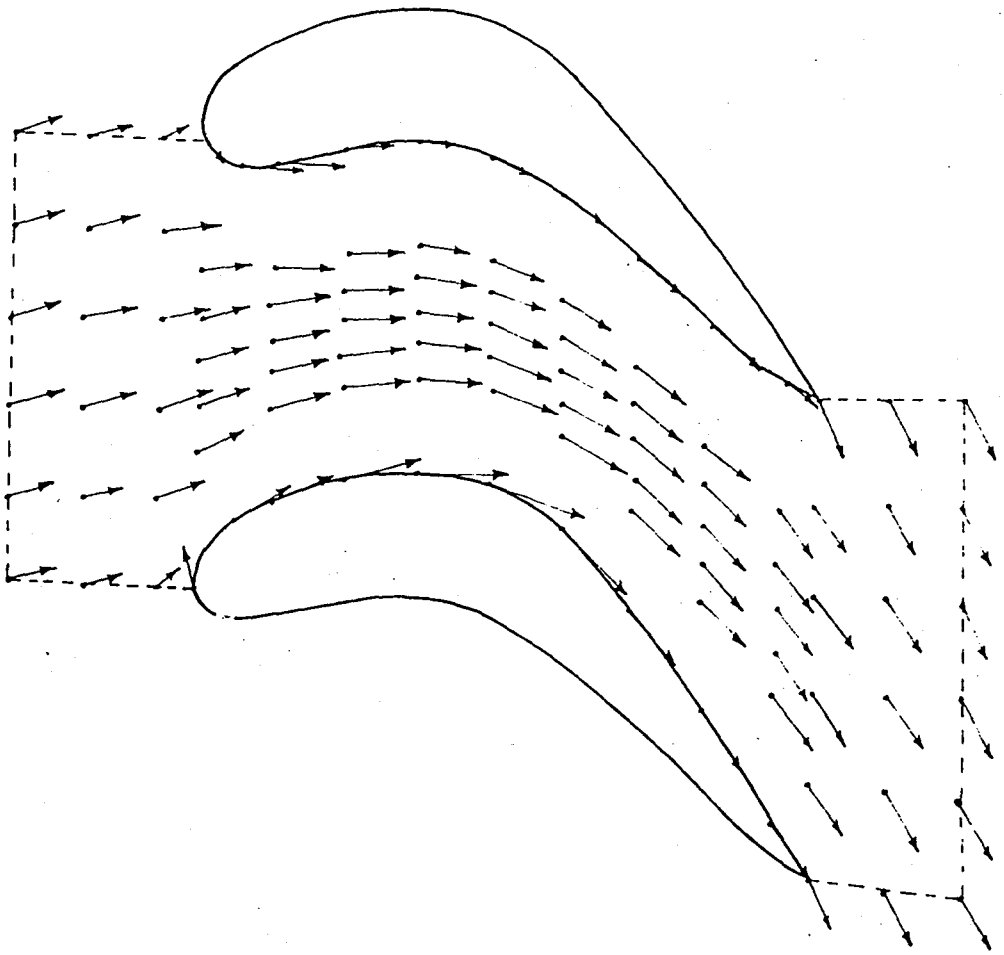

FIGURE 32 - Relative Velocity Vectors Calculated by the Present Code on the Midspan Blade-to-Blade Surface of Revolution for the Mixed-Flow Turbine

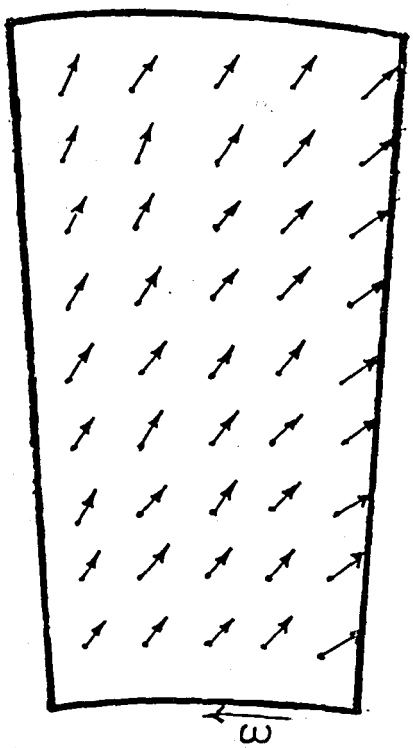

FIGURE 33 - Relative Velocity Vectors Calculated by the Present Code on the Passage Cross-Sectional Plane at the Blade Trailing Edge for the Mixed-Flow Turbine

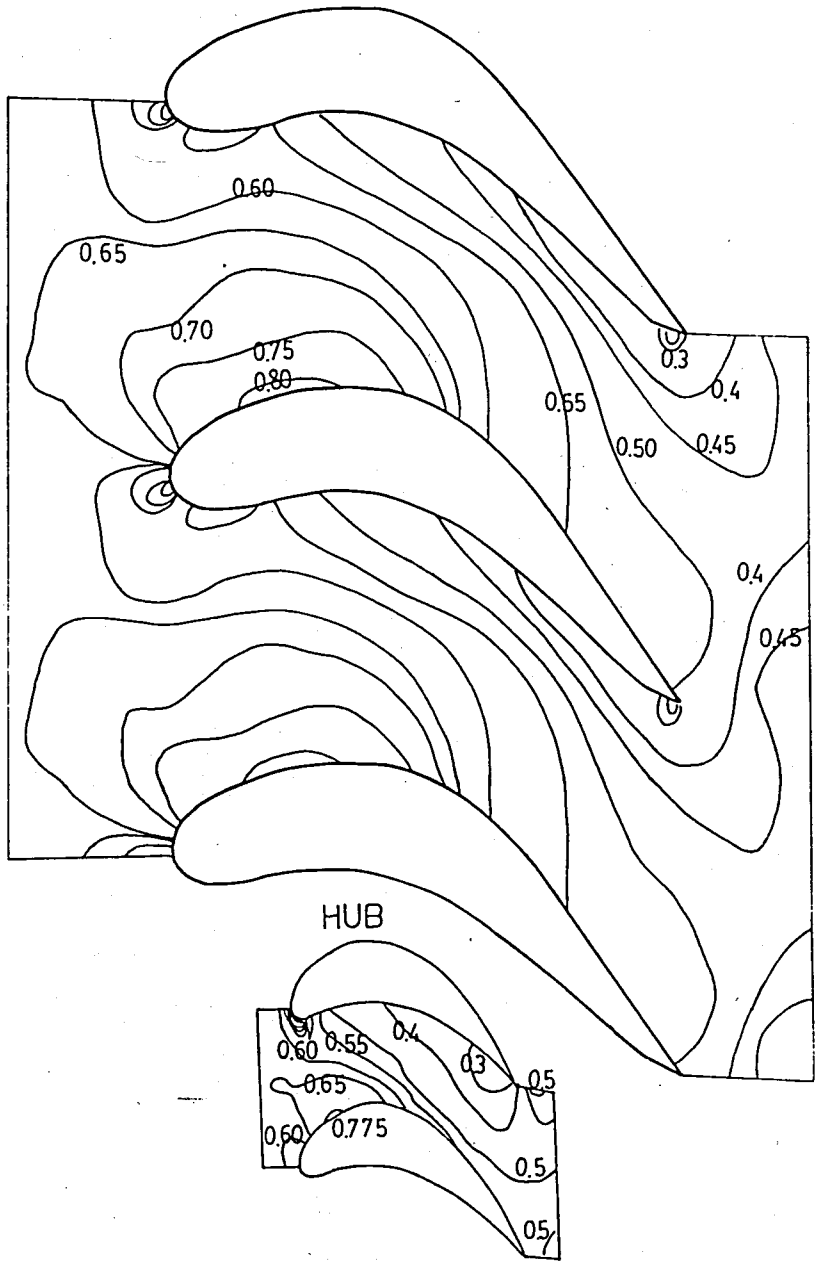

FIGURE 34 - Comparison of Blade-to-Blade Mach Number Contours Calculated by the Present Code and Laskaris' Code at the Hub Section of the Mixed-Flow Turbine

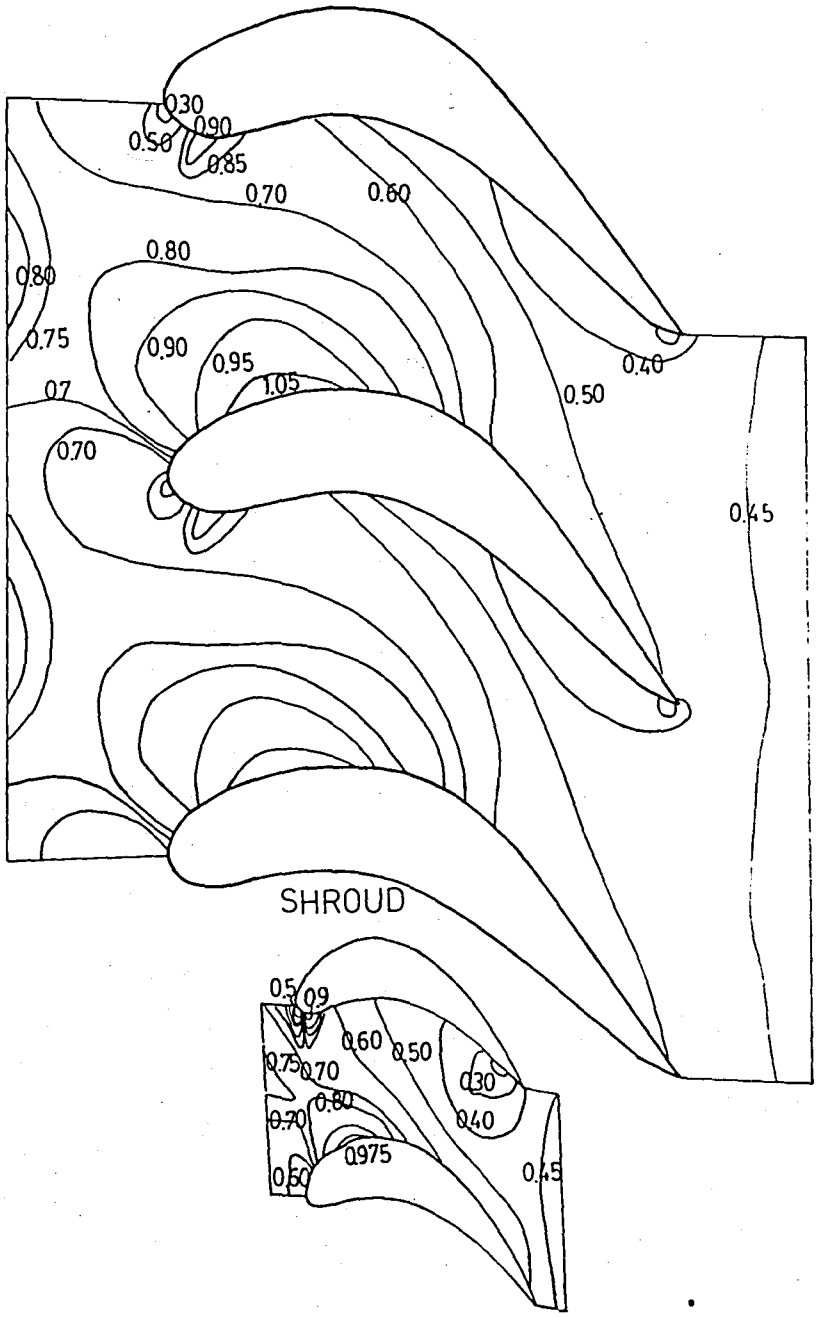

FIGURE 35 - Comparison of Blade-to-Blade Mach Number Contours Calculated by the Present Code and Laskaris' Code at the Shroud Section of the Mixed-Flow Turbine

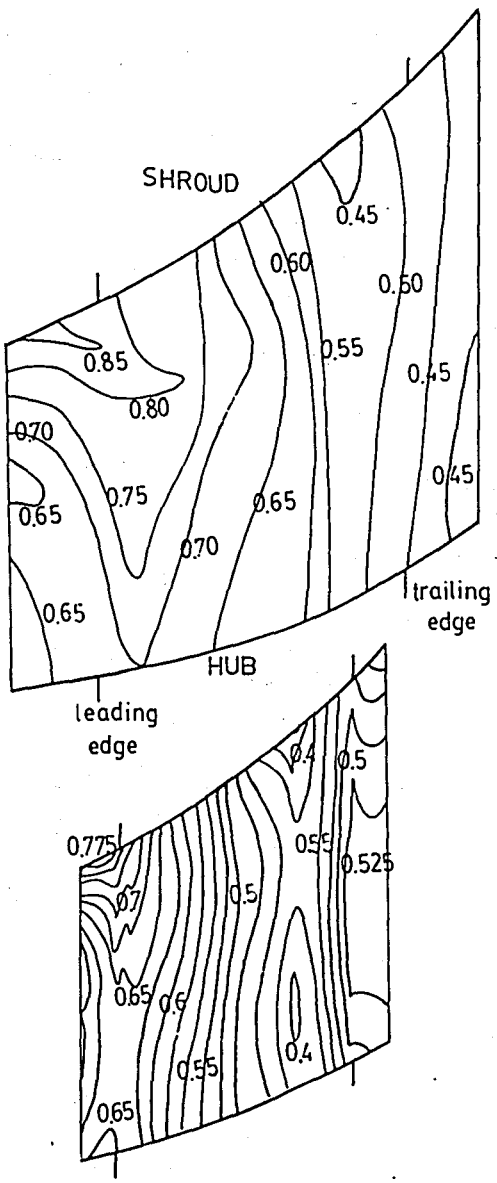

FIGURE 36 - Comparison of Mach Number Contours on the Surface Midway Between Blades by the Present Code and Laskaris' Code for the Mixed-Flow Turbine

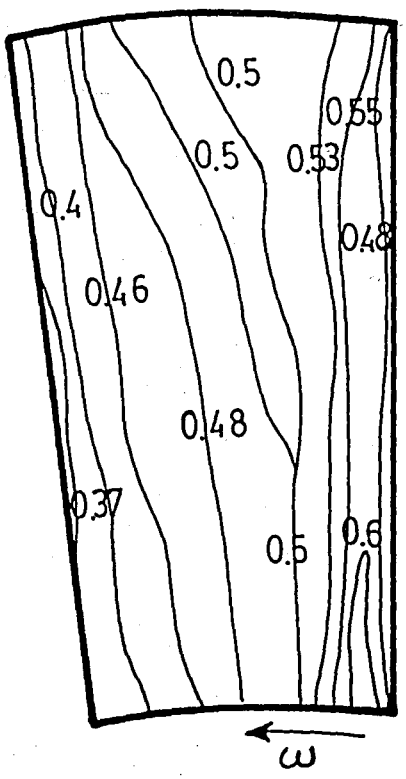

FIGURE 37 - Mach Number Contours on the Passage Cross-Sectional Plane at the Trailing Edge of the Mixed-Flow Turbine Calculated by the Present Code

V. CONCLUSIONS AND RECOMMENDATIONS

1. Based on the finite element analysis presented in this thesis, a computer code has been developed for calculating three-dimensional, compressible, potential flows in axial-, radial- or mixed-flow turbomachinery.

2. Applications of the code to the Gostelow cascade, an experimental turbine stator, the first stage stator and rotor of an electric utility turbine and finally to a mixed-flow turbine rotor are presented. The validity of the code is established by comparing the results with the exact solution, experimental data and calculations by other numerical methods.

3. The present code handles subsonic flows with local supersonic spots. The analysis may be extended to include shock fitting and shock capturing techniques so as to enable the analysis of transonic flows.

4. The analysis should be extended to the solution of rotational flows.

5. Viscous effects are not included in the present code. It is possible to account for these effects by superposing

secondary flows and loss models onto the solution obtained by the code.

6. In parallel with rapid advances in the computer technology, the code developed in this study will be a powerful and economical design and evaluation tool not totally substituting the need for experimentation but providing insight and direction towards better ideas in turbomachinery design.

APPENDICES

APPENDIX A

SUBPARAMETRIC TETRAHEDRAL ELEMENT FORMULATION

The natural coordinates of a linear tetrahedron are the linear interpolation functions used in Eq. (75) of Chapter III. i.e. $\ell_j = L_j$, $j = 1,2,3,4$. Hence,

$$\begin{aligned} r &= \sum_{j=1}^{10} L_j\, r_j \\ \theta &= \sum_{j=1}^{10} L_j\, \theta_j \\ z &= \sum_{j=1}^{10} L_j\, z_j \end{aligned} \tag{A1}$$

Here it will be convenient to incorporate Eq. (77) of Chapter III into Eq. (A1) above yielding

$$\begin{Bmatrix} r \\ \theta \\ z \\ 1 \end{Bmatrix} = \begin{bmatrix} r_1 & r_2 & r_3 & r_4 \\ \theta_1 & \theta_2 & \theta_3 & \theta_4 \\ z_1 & z_2 & z_3 & z_4 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{Bmatrix} L_1 \\ L_2 \\ L_3 \\ L_4 \end{Bmatrix} \tag{A2}$$

Denoting the coefficient matrix by C we obtain

$$\begin{Bmatrix} L_1 \\ L_2 \\ L_3 \\ L_4 \end{Bmatrix} = C^{-1} \begin{Bmatrix} r \\ \theta \\ z \\ 1 \end{Bmatrix} \tag{A3}$$

The inverse coefficient matrix may be written as

$$C^{-1} = \frac{1}{\det C} \begin{bmatrix} a_1 & b_1 & c_1 & d_1 \\ a_2 & b_2 & c_2 & d_2 \\ a_3 & b_3 & c_3 & d_3 \\ a_4 & b_4 & c_4 & d_4 \end{bmatrix} \tag{A4}$$

where

$$a_1 = \det \begin{bmatrix} \theta_2 & \theta_3 & \theta_4 \\ z_2 & z_3 & z_4 \\ 1 & 1 & 1 \end{bmatrix}$$

$$b_1 = -\det \begin{bmatrix} r_2 & r_3 & r_4 \\ z_2 & z_3 & z_4 \\ 1 & 1 & 1 \end{bmatrix} \tag{A5}$$

$$c_1 = \det \begin{bmatrix} r_2 & r_3 & r_4 \\ \theta_2 & \theta_3 & \theta_4 \\ 1 & 1 & 1 \end{bmatrix}$$

$$d_1 = -\det \begin{bmatrix} r_2 & r_3 & r_4 \\ \theta_2 & \theta_3 & \theta_4 \\ z_2 & z_3 & z_4 \end{bmatrix}$$

The remaining coefficients can be obtained by permutation of indices.

It follows from Eqs. (A3) and (A4) that

$$L_j = \frac{a_j r + b_j \theta + c_j z + d_j}{\det C} \quad , \quad j = 1,2,3,4 \qquad \text{(A6)}$$

Note that for a given element a_j, b_j, c_j, d_j and det C are constants that depend on the coordinates of the corner nodes.

It can be shown that det C = det J, and therefore, from Eq. (79) of Chapter III we have

$$dr\, d\theta\, dz = \det C\, dL_1\, dL_2\, dL_3 \qquad \text{(A7)}$$

Using Eq. (A6) derivatives of the quadratic functions, N_j, with respect to the cylindrical coordinates r, θ and z within an element may be related to the derivatives with respect to the natural coordinates as

$$\begin{aligned}
\frac{\partial N_j}{\partial r} &= \sum_{i=1}^{4} \frac{\partial N_j}{\partial L_i} \frac{a_i}{\det C} \\
\frac{\partial N_j}{\partial \theta} &= \sum_{i=1}^{4} \frac{\partial N_j}{\partial L_i} \frac{b_i}{\det C} \\
\frac{\partial N_j}{\partial z} &= \sum_{i=1}^{4} \frac{\partial N_j}{\partial L_i} \frac{c_i}{\det C}
\end{aligned} \qquad \text{(A8)}$$

Hence the element property matrix, $(k_{ij})_e$, and the element right-hand side vector, $(r_i)_e$, given by Eqs. (60) and (61) in Chapter III, respectively, may be written in terms of the local coordinates as

$$(k_{ij})_e = \frac{1}{\det C} \int_0^1 \int_0^{1-L_3} \int_0^{1-L_2-L_3} \left(\sum_{\ell=1}^{10} N_\ell \rho_\ell \right)$$

$$\left[\sum_{n=1}^{4} \sum_{m=1}^{4} \left(\frac{\partial N_i}{\partial L_n} \frac{\partial N_j}{\partial L_m} \right) \left(a_n a_m + \frac{b_n b_m}{\sum_{p=1}^{4} L_p r_p} + c_n c_m \right) \right]$$

$$\left(\sum_{p=1}^{4} L_p r_p \right) dL_1 \, dL_2 \, dL_3 \qquad \text{(A9)}$$

and

$$(r_i)_e = \int_0^1 \int_0^{1-L_3} \int_0^{1-L_2-L_3} \omega N_i \left(\sum_{k=1}^{4} \sum_{\ell=1}^{10} \frac{\partial N_\ell}{\partial L_k} b_k \rho_\ell \right) \left(\sum_{j=1}^{4} L_j r_j \right) dL_1 \, dL_2 \, dL_3$$

(A10)

Here the density, ρ, is assumed to be given by the same quadratic interpolation rule

$$\rho = \sum_{\ell=1}^{10} N_\ell \rho_\ell \qquad \text{(A11)}$$

where ρ_ℓ are the known values of the density at the element nodes.

To evaluate Eq. (A9) and Eq. (A10) we resort to numerical integration formulae derived by Hammer et al [49]. For an integration of order 3 (cubic) and an error of $R = O(h^4)$ we have

$$\int_0^1 \int_0^{1-L_3} \int_0^{1-L_2-L_3} f(L_1, L_2, L_3) dL_1 \, dL_2 \, dL_3 =$$

$$[f(a)W_a + f(b)W_b + f(c)W_c + f(d)W_d + f(e)W_e] \qquad \text{(A12)}$$

where a, b, c, d, e are locations in the element and W are the weighting factors given below

Loc	L_1	L_2	L_3	L_4	W
a	1/4	1/4	1/4	1/4	-4/5
b	1/2	1/6	1/6	1/6	9/20
c	1/6	1/2	1/6	1/6	9/20
d	1/6	1/6	1/2	1/6	9/20
e	1/6	1/6	1/6	1/2	9/20

(A13)

If it is assumed that $\rho = \bar{\rho}$ (density at the centroid of the element and $r = \bar{r}$ (radial coordinate of the centroid of the element) then Eq. (A9) may be integrated analytically by using

$$\int_0^1 \int_0^{1-L_3} \int_0^{1-L_2-L_3} L_1^a L_2^b L_3^c L_4^d \, dL_1 \, dL_2 \, dL_3 = \frac{a!b!c!d!}{(a+b+c+d+3)!} \tag{A14}$$

The element matrix, $(k_{ij})_e$, for this case is given in Table A-1.

TABLE A-1. The Element Property Matrix

$k_{ij} = k_{ji}$ for all i,j

i = 1,2,3,4

$$k_{ii} = \frac{\bar{\rho}}{10 \det C}(a_i^2\bar{r} + b_i^2/\bar{r} + c_i^2\bar{r})$$

i = 1,2,3 j = 2,3,4

$$k_{ij} = -\frac{\bar{\rho}}{30 \det C}(a_i a_j\bar{r} + b_i b_j/\bar{r} + c_i c_j\bar{r})$$

i = 5	k = 1	ℓ = 2
i = 6	k = 2	ℓ = 3
i = 7	k = 1	ℓ = 3
i = 8	k = 1	ℓ = 4
i = 9	k = 2	ℓ = 4
i = 10	k = 3	ℓ = 4

$$k_{ii} = \frac{4\bar{\rho}\bar{r}}{15 \det C}[(a_k^2 + b_k^2/\bar{r}^2 + c_k) + (a_\ell^2 + b_\ell^2/\bar{r}^2 + c_\ell^2) + (a_k a_\ell + b_k b_\ell/\bar{r}^2 + c_k c_\ell)]$$

i = 1	j = 5	k = 2	i = 3	j = 6	k = 2
	j = 7	k = 3		j = 7	k = 1
	j = 8	k = 4		j = 10	k = 4
i = 2	j = 5	k = 1	i = 4	j = 8	k = 1
	j = 6	k = 3		j = 9	k = 1
	j = 9	k = 4		j = 10	k = 3

$$k_{ij} = \frac{\bar{\rho}\bar{r}}{30 \det C}[3(a_i a_k + b_i b_k/\bar{r}^2 + c_i c_k) - (a_i^2 + b_i^2/\bar{r}^2 + c_i^2)]$$

(Table A-1. Continued)

i = 5	j = 10	k = 1	ℓ = 2	m = 3	n = 4
i = 6	j = 8	k = 2	ℓ = 3	m = 1	n = 4
i = 7	j = 9	k = 1	ℓ = 3	m = 2	n = 4

$$k_{ij} = \frac{2\bar{\rho}\bar{r}}{15 \det C}[(a_\ell a_m + a_k a_n + a_\ell a_n) + (b_\ell b_m + b_k b_n + b_\ell b_n)/\bar{r}^2 + (c_\ell c_m + c_k c_n + c_\ell c_n)]$$

i = 1	j = 6	k = 3	ℓ = 2	i = 3	j = 5	k = 1	ℓ = 2
	j = 9	k = 2	ℓ = 2		j = 8	k = 1	ℓ = 4
	j = 10	k = 3	ℓ = 4		j = 9	k = 2	ℓ = 4
i = 2	j = 7	k = 1	ℓ = 3	i = 4	j = 5	k = 1	ℓ = 2
	j = 8	k = 4	ℓ = 1		j = 6	k = 2	ℓ = 3
	j = 10	k = 4	ℓ = 3		j = 7	k = 1	ℓ = 3

$$k_{ij} = -\frac{\bar{\rho}\bar{r}}{30 \det C}[(a_i a_k + b_i b_k/\bar{r}^2 + c_i c_k) + (a_i a_\ell + b_i b_\ell/\bar{r}^2 + c_i c_\ell)]$$

i = 5	j = 6	k = 2	ℓ = 1	m = 3	i = 7	j = 8	k = 1	ℓ = 3	m = 4
	j = 7	k = 1	ℓ = 2	m = 3		j = 10	k = 3	ℓ = 1	m = 4
	j = 8	k = 1	ℓ = 2	m = 4	i = 8	j = 9	k = 4	ℓ = 1	m = 2
i = 6	j = 7	k = 3	ℓ = 1	m = 2	i = 9	j = 10	k = 4	ℓ = 2	m = 3
	j = 9	k = 2	ℓ = 3	m = 4					
	j = 10	k = 3	ℓ = 2	m = 4					

$$k_{ij} = \frac{2\bar{\rho}\bar{r}}{15 \det C}[(a_k^2 + b_k^2/\bar{r}^2 + c_k^2) + (a_k a_\ell + a_k a_m + (b_k b_\ell + b_k b_m)/\bar{r}^2 + (c_k c_\ell + c_k c_m) + 2(a_\ell a_m + b_\ell b_m/\bar{r}^2 + c_\ell c_m)]$$

The surface integral contribution, $(r_i)_{e_n}$, of a Neumann boundary element, e_n, given by Eq. (63) of Chapter III is evaluated as follows

$$(r_i)_{e_n} = \int_{(S_n)_{e_n}} N_i \left(\frac{\partial \Phi}{\partial n}\right)_{(S_n)_{e_n}} dS \tag{A15}$$

where $(\partial\Phi/\partial n)_{(S_n)_{e_n}}$ is given by Eqs. (35) and (36) of Chapter III for the exit surface and along the blade surfaces, respectively, as

$$\left(\frac{\partial \Phi}{\partial n}\right)_{ex} = V_z(r)_{ex} \tag{A16}$$

and

$$\left(\frac{\partial \Phi}{\partial n}\right)_{blade} = \omega r(\vec{n}\cdot\vec{e}_\theta) \tag{A17}$$

Note that the surface integral contributions of the periodic elements cancel each other as discussed in Chapter III.

Since the natural coordinate L_i vanishes on the surface jkℓ (refer to Figure 4 of Chapter III) of an element, the surface integral may be transformed into the local coordinate system by the Jacobian matrix for the surface transformation, J'.

If the exit plane is taken to be normal to z, assuming L_1 and L_2 to be the independent local coordinates ($L_3 = 1 - L_1 - L_2$ and $L_4 = 0$) and that nodes (1-3) are on the exit surface we may write

$$dr\, d\theta = \det J'\, dL_1\, dL_2 \tag{A19}$$

where, for subparametric elements,

$$\det J' = \det \begin{bmatrix} r_1 & r_2 & r_3 \\ \theta_1 & \theta_2 & \theta_3 \\ 1 & 1 & 1 \end{bmatrix} \tag{A20}$$

Hence,

$$dS = r \det J' \, dL_1 \, dL_2 \tag{A21}$$

It follows that

$$[(r_i)_{e_n}]_{ex} = \int_0^1 \int_0^{1-L_2} \left(\sum_{\ell=1}^{10} N_\ell \rho_\ell \right) N_i V_z(r)_{ex} \left(\sum_{j=1}^{3} L_j r_j \right) \det J' \, dL_1 \, dL_2 \tag{A22}$$

For the blade surfaces we write, by using Eq. (A17), that

$$[(r_i)_{e_n}]_{blade} = \int_{(S_n)_{e_n}} \rho N_i \omega r (\vec{n} \cdot \vec{e}_\theta) dS \tag{A23}$$

The term $(\vec{n} \cdot \vec{e}_\theta)dS$ is the projection of the element blade surface in the rz plane, which allows us to write

$$[(r_i)_{e_n}]_{blade} = \int_0^1 \int_0^{1-L_2} \left(\sum_{\ell=1}^{10} N_\ell \rho_\ell \right) N_i \omega \left(\sum_{j=1}^{3} L_j r_j \right) \det J' \, dL_1 \, dL_2 \tag{A24}$$

where J' here is the Jacobian matrix which relates the cylindrical coordinates r and z to the local coordinates L_1 and L_2 and its determinant is given by

$$\det J' = \det \begin{bmatrix} r_1 & r_2 & r_3 \\ z_1 & z_2 & z_3 \\ 1 & 1 & 1 \end{bmatrix} \tag{A25}$$

APPENDIX B

ISOPARAMETRIC TETRAHEDRAL ELEMENT FORMULATION

In the treatment of flows that exhibit complicated design features such as highly twisted, curved and staggered blades, the subparametric formulation founded upon linear tetrahedral elements may be insufficient to yield the desired accuracy. It is desirable to have the capability of specifying elements with surfaces curved to best fit the shape of the physical boundaries and to best approximate the dynamic characteristics of the flow (i.e. the orientation and curvature of streamlines). An alternative isoparametric formulation that uses curved sided elements will definitely enhance the power and flexibility of the solution method.

The coordinates r, θ, z in an isoparametric element are related to the nodal coordinates through the quadratic interpolation functions, N_j, as follows.

$$\begin{aligned} r &= \sum_{j=1}^{10} N_j r_j \\ \theta &= \sum_{j=1}^{10} N_j \theta_j \\ z &= \sum_{j=1}^{10} N_j z_j \end{aligned} \tag{B1}$$

From Eq. (78) of Chapter III and Eq. (B1), the Jacobian matrix is obtained as

$$J = \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ J_{31} & J_{32} & J_{33} \end{bmatrix} \tag{B2}$$

where

$$\begin{aligned} J_{i1} &= \sum_{j=1}^{10} \frac{\partial N_j}{\partial L_i} r_j \\ J_{i2} &= \sum_{j=1}^{10} \frac{\partial N_j}{\partial L_i} \theta_j \qquad i = 1,2,3 \\ J_{i3} &= \sum_{j=1}^{10} \frac{\partial N_j}{\partial L_i} z_j \end{aligned} \tag{B3}$$

Hence, the inverse Jacobian matrix may be written as

$$J^{-1} = \frac{1}{\det J} \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{bmatrix} \tag{B4}$$

where

$$\begin{aligned} H_{11} &= (J_{22}J_{33} - J_{23}J_{32}) & H_{32} &= -(J_{11}J_{32} - J_{12}J_{31}) \\ H_{21} &= -(J_{21}J_{33} - J_{23}J_{31}) & H_{13} &= (J_{12}J_{23} - J_{13}J_{22}) \\ H_{31} &= (J_{21}J_{32} - J_{22}J_{31}) & H_{23} &= -(J_{11}J_{23} - J_{13}J_{21}) \\ H_{12} &= -(J_{12}J_{33} - J_{13}J_{32}) & H_{33} &= (J_{11}J_{22} - J_{12}J_{21}) \\ H_{22} &= (J_{11}J_{33} - J_{13}J_{31}) \end{aligned} \tag{B5}$$

and

$$\det J = \sum_{k=1}^{3} J_{1k} H_{k1} \tag{B6}$$

Using Eqs. (60), (61), (79), (80) of Chapter III, Eq. (A11), Eq. (B1) and Eqs. (B4-B6) we write

$$(k_{ij})_e = \int_0^1 \int_0^{1-L_3} \int_0^{1-L_2-L_3} \left(\sum_{\ell=1}^{10} N_\ell \rho_\ell\right)[(A_{ij})_1 + r^2(A_{ij})_2 + (A_{ij})_3] \left(\sum_{\ell=1}^{10} N_\ell r_\ell\right) \det J \, dL_1 \, dL_2 \, dL_3 \tag{B7}$$

and

$$(r_i)_e = \int_0^1 \int_0^{1-L_3} \int_0^{1-L_2-L_3} \omega N_i \left[\sum_{\ell=1}^{10} \rho_\ell \left(\sum_{k=1}^{4} H_{2\ell} \frac{\partial N_\ell}{\partial L_k}\right)\right]\left(\sum_{\ell=1}^{10} N_\ell r_\ell\right) dL_1 \, dL_2 \, dL_3 \tag{B8}$$

where

$$(A_{ij})_k = \frac{1}{(\det J)^2}\left(\sum_{\ell=1}^{4} H_{k\ell} \frac{\partial N_i}{\partial L_\ell}\right)\left(\sum_{\ell=1}^{4} H_{k\ell} \frac{\partial N_j}{\partial L_\ell}\right) \quad , k = 1,2,3 \tag{B9}$$

The surface integral contribution, $(r_i)_{e_n}$, of a Neumann boundary element, e_n, given by Eq. (63) of Chapter III is obtained from expressions similar to Eqs. (A22) and (A24).

On the exit plane we have,

$$[(r_i)_{e_n}]_{ex} = \int_0^1 \int_0^{1-L_2} \left(\sum_{\ell=1}^{10} N_\ell \rho_\ell\right) N_i V_z(r)_{ex} \left(\sum_{j=1}^{10} N_j r_j\right) \det J' \, dL_1 dL_2. \tag{B10}$$

where

$$\det J' = \det \begin{bmatrix} \sum_{j=1}^{10} \frac{\partial N_j}{\partial L_1} r_j & \sum_{j=1}^{10} \frac{\partial N_j}{\partial L_1} \theta_j \\ \sum_{j=1}^{10} \frac{\partial N_j}{\partial L_2} r_j & \sum_{j=1}^{10} \frac{\partial N_j}{\partial L_2} \theta_j \end{bmatrix} \tag{B11}$$

For the blade surfaces we write,

$$[(r_i)_{e_n}]_{blade} = \int_0^1 \int_0^{1-L_2} \left(\sum_{\ell=1}^{10} N_\ell \rho_\ell \right) N_i \omega \left(\sum_{j=1}^{10} N_j r_j \right) \det J' \, dL_1 \, dL_2 \tag{B12}$$

where

$$\det J' = \det \begin{bmatrix} \sum_{j=1}^{10} \frac{\partial N_j}{\partial L_1} r_j & \sum_{j=1}^{10} \frac{\partial N_j}{\partial L_1} z_j \\ \sum_{j=1}^{10} \frac{\partial N_j}{\partial L_2} r_j & \sum_{j=1}^{10} \frac{\partial N_j}{\partial L_2} z_j \end{bmatrix} \tag{B13}$$

APPENDIX C

BLADE PROFILE AND CHANNEL COORDINATES USED IN TEST RUNS

TABLE C-1. Gostelow Cascade Blade Geometry

z/C'	Y/C'	
	Surface 1	Surface 2
0.0	0.0	0.0
0.025	0.1025	-0.0025
0.050	0.14	0.012
0.075	0.17	0.03
0.10	0.20	0.05
0.15	0.255	0.115
0.20	0.295	0.1525
0.25	0.332	0.202
0.30	0.3675	0.25
0.35	0.40	0.30
0.40	0.43	0.345
0.45	0.4575	0.3875
0.50	0.485	0.43
0.55	0.51	0.465
0.60	0.532	0.50
0.65	0.5525	0.535
0.70	0.525	0.564
0.7467	0.591	0.586
0.7934	0.6088	0.6088

TABLE C-2. Experimental Turbine Stator Blade Geometry

z(ft)	θ (rad)	
	Surface 1	Surface 2
0.0	0.0	0.0
0.0025	0.01257	-0.0116
0.005	0.01676	-0.01289
0.0075	0.01805	-0.0145
0.010	0.01882	-0.01676
0.0125	0.01882	-0.02011
0.015	0.01805	-0.02449
0.0175	0.01547	-0.02965
0.0200	0.01096	-0.03545
0.0225	0.00516	-0.04189
0.025	-0.00258	-0.04898
0.0275	-0.01289	-0.05814
0.030	-0.02449	-0.06703
0.0325	-0.03996	-0.07734
0.035	-0.05543	-0.08766
0.0375	-0.0709	-0.09797
0.04	-0.08637	-0.1083
0.0415	-0.09797	-0.1134
0.043	-0.116	-0.116

Channel Geometry:

Hub Radius = 0.2794 (ft)

Tip Radius = 0.381 (ft)

Solution domain boundaries used in the present analysis:

Lower Boundary Radius = 0.3202 (ft)

Upper Boundary Radius = 0.3402 (ft)

TABLE C-3. Electric Utility Turbine First Stage Stator Geometry

z (in)	θ (rad)	
	Surface 1	Surface 2
0.0	0.0	0.0
0.32815	0.00762	-0.00762
0.6563	0.01113	-0.00936
0.97815	0.0165	-0.00904
1.3	0.02093	-0.00739
1.65	0.0292	-0.00508
2.0	0.03595	0.0
2.25	0.04316	0.00508
2.5	0.04925	0.01034
2.75	0.0584	0.01778
3.0	0.06648	0.02708
3.25	0.07029	0.04138
3.5	0.08729	0.05318
3.75	0.1022	0.07122
4.0	0.1108	0.08495
4.25	0.1289	0.1079
4.5	0.1391	0.1243
4.75	0.1503	0.1479
5.0	0.1477	0.1477

Channel Geometry:

Hub Radius = 30.03 (in)

Tip Radius = 35.19 (in)

Solution domain boundaries used in the present analysis:

Lower Boundary Radius = 32.11 (in)

Upper Boundary Radius = 33.11 (in)

TABLE C-4. Electric Utility Turbine First Stage Rotor Blade and Channel Geometry

z (in)	Blade Geometry		Channel Geometry			
	θ (rad)		Radius (in)			
	Surface 1	Surface 2	Actual		Solution	
			Hub	Tip	Lower	Upper
0.00	0.0	0.0	30.03	35.19	32.11	33.11
0.25	0.01118	-0.002033	30.00	35.19	32.085	33.11
0.50	0.01831	0.0	29.98	35.19	32.06	33.11
0.75	0.0224	0.00102	29.95	35.19	32.035	33.11
1.00	0.02395	0.000917	29.92	35.19	32.01	33.11
1.25	0.02448	-0.000204	29.90	35.19	31.98	33.11
1.50	0.02246	-0.000204	29.87	35.19	31.95	33.11
1.75	0.01972	-0.005874	29.84	35.19	31.92	33.11
2.00	0.01531	-0.009186	29.81	35.19	31.90	33.11
2.15	0.01229	-0.01126	29.80	35.19	31.885	33.11
2.30	0.008188	-0.01433	29.78	35.19	31.87	33.11
2.45	0.003554	-0.01741	29.77	35.19	31.855	33.11
2.60	-0.001025	-0.0205	29.75	35.19	31.84	33.11
2.75	-0.007175	-0.0244	29.73	35.19	31.82	33.11
2.90	-0.01333	-0.02871	29.71	35.19	31.80	33.11
3.05	-0.02052	-0.0328	29.69	35.19	31.78	33.11
3.20	-0.02669	-0.03798	29.67	35.19	31.76	33.11
3.38	-0.03493	-0.04417	29.65	35.19	31.75	33.11
3.56	-0.0473	-0.0473	29.63	35.19	31.73	33.11

TABLE C-5. Laskaris' Turbine Rotor Blade and Channel Geometry

z (cm)	Blade Geometry		Channel Geometry	
	θ (rad)		Radius (cm)	
	Surface 1	Surface 2	Hub	Tip
0.0	0.0	0.0	45.0	55.0
0.25	0.01303	-0.0078	45.105	55.23
0.50	0.0166	-0.0082	45.21	55.46
1.00	0.0218	-0.0085	45.27	55.58
1.50	0.02511	-0.00734	45.33	55.75
2.5	0.0305	-0.003	45.59	56.42
3.0	0.031	-0.003	45.74	56.67
3.5	0.0302	-0.004	45.88	56.92
4.0	0.028	-0.0075	46.09	57.34
4.5	0.02424	-0.01258	46.29	57.75
5.0	0.016	-0.018	46.46	58.12
5.5	0.00547	-0.0242	46.63	58.50
6.0	-0.005	-0.0328	46.84	58.13
6.5	-0.01686	-0.041	47.04	59.33
7.0	-0.03	-0.05	47.27	59.67
7.5	-0.0433	-0.0586	47.5	60.00
8.0	-0.058	-0.064	47.73	60.44
8.5	-0.07058	-0.07058	47.96	60.88

APPENDIX D

COMPUTER CODE USER'S MANUAL

A computer code TURBO has been developed to calculate the three-dimensional, compressible, potential flow in turbomachines of the axial-, radial- or mixed-flow type.

TURBO consists of a main program and twenty-two subroutines. An organization chart showing the relationship between these subroutines and the main program is given in Figure D.1. Short descriptions of the subroutines are presented in Table D-1.

In the present version of the code there are four main option categories to identify the case under consideration:

1. Compressible or incompressible flow
2. Two-dimensional cascade or three-dimensional turbomachine
3. Subparametric and/or isoparametric element use in domain discretization
4. An output option enabling calculation of velocity, density and Mach number distributions from the velocity potentials either calculated in the present run or calculated and stored in an earlier run.

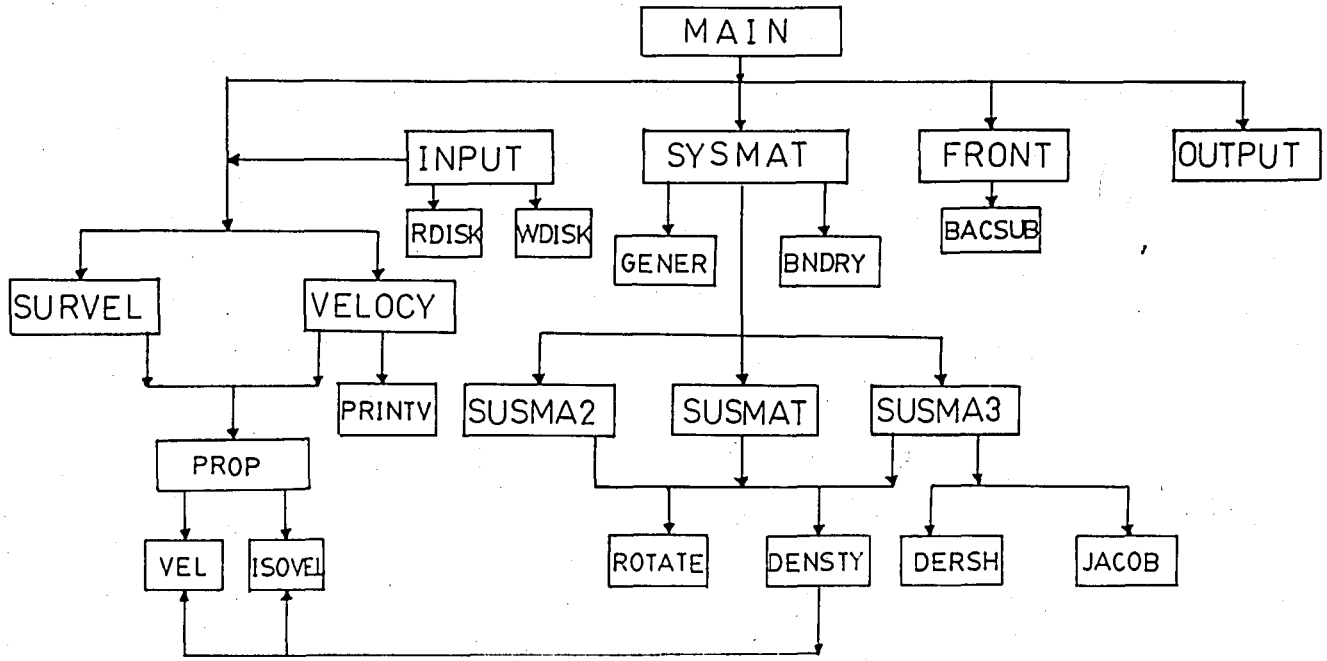

FIGURE D.1 - Subroutine Relationship Chart

The solution to the incompressible flow problem is obtained in a single run, whereas the compressible flow problem requires several runs each corresponding to a solution update. The first run for the latter problem obtains an incompressible flow solution as the initial guess and therefore has the same input as the former problem. The geometry is specified only in the first run that we shall refer to as the incompressible flow run. The remaining runs that we shall call the iteration runs are used to successively improve the solution based on the results and the geometric data transferred from the preceeding iteration run or the incompressible flow run, as the case may be.

TABLE D-1. Subroutine Descriptions

Subroutine Name	Called From	Description
MAIN	-	Controls the sequence of calculation routines
INPUT	MAIN	Reads, prints and stores all Input Data
RDISK	INPUT	Reads data from disk file No. 7
WDISK	INPUT	Writes data on disk file No. 8
SYSMAT	MAIN	Constructs the coordinates and nodal numbers for elements and writes element equations and transformation matrix on disk file No. 9
GENER	SYSMAT	Calculates the coordinates of the nodes on a specified hexahedral prism row
SUSMAT	SYSMAT	Calculates the subparametric tetrahedral element equations and assembles the hexahedral superelement (subsystem) equations by using analytical integration results
SUSMA2	SYSMAT	Calculates the subparametric tetrahedral element equations and assembles the hexahedral superelement (subsystem) equations by using numerical integration
SUSMA3	SYSMAT	Calculates the isoparametric tetrahedral element equations and assembles the hexahedral superelement (subsystem) equations by using numerical integration
DERSH	SUSMA3	Calculates the derivatives of the shape functions with respect to the local coordinates for an isoparametric tetrahedral element

Table D.1 continued..

Subroutine Name	Called From	Description
JACOB	SUSMA3	Calculates the inverse of the coordinate transformation matrix and its determinant (Jacobian of transformation)
DENSTY	SUSMAT SUSMA2 SUSMA3	Calculates densities at the nodes of a hexahedral superelement
ROTATE	SUSMAT SUSMA2 SUSMA3	Calculates the right-hand side vector for a tetrahedral element which has a side on one of the surfaces of a rotating blade
BNDRY	SYSMAT	Constructs the right-hand side vector of the global system equations resulting from Neumann type boundary conditions at the exit plane
FRONT	MAIN	Solves the global system equations using a Gaussian (Frontal) elimination backsubstitution method. Uses disk file No. 10 to write the eliminated equations on disk.
BACSUB	FRONT	Reads information from disk file No. 10 and obtains the resulting global velocity potential vector by backsubstitution
OUTPUT	MAIN	Prints the nodal values of the velocity potential in each iteration
VELOCY	MAIN INPUT	Calculates the velocities on a specified cross-sectional plane
PROP	VELOCY SURVEL	Calculates the velocities at a specified location within a hexahedral superelement
SURVEL	MAIN INPUT	Calculates and prints the blade surface velocities
VEL	DENSTY PROP	Calculates the velocities at a specified location within a subparametric tetrahedral element
ISOVEL	DENSTY PROP	Calculates the velocities at a specified location within an isoparametric tetrahedral element
PRINTV	VELOCY	Calculates and prints Mach No. and density at a given location with known velocities

PROGRAM TURBO INPUT DATA

All formats free

Card.(1). HEADR

HEADR: Heading

Card.(2). ICOMP,IDIM

ICOMP: Compressibility flag

= 0: for an incompressible flow run

> 0: for an iteration run

IDIM : Dimension flag

= 2: two-dimensional cascade

= 3: three-dimensional turbomachine

Card.(3). INTG,IVEL

INTG : Element type and integration flag

= 1: subparametric elements are used and integrations are performed analytically

= 2: subparametric elements are used and integrations are performed numerically

= 3: isoparametric elements are used and integrations are performed numerically

= 4: subparametric elements are used internally and isoparametric elements are used on the boundaries and the integrations are performed numerically

IVEL : Velocity flag

= 0: no velocity calculation

= 1: velocity calculation based on the present solution

= 2: velocity calculation based on a solution stored in disk

If ICOMP = 0 (incompressible flow run) then

Card.(4). NDT,NDR (NDT*NDR $\leq$ 8 on the existing computer)

NDT : Number of hexahedra in the tangential direction

NDR : Number of hexahedra in the radial direction

Note: Due to the present core storage limitations, the number of hexahedral blocks in the z-direction is fixed as follows; 3 blocks upstream of the blade row, 9 blocks in the blade row, 3 blocks downstream of the blade row

Card.(5). VIN

VIN : Inlet velocity at channel midspan

If IDIM = 2 (2-D Cascade) then

Card.(6). BETAI,BETAO,OMEGA,SC,CHORD

BETAI : Inlet absolute flow angle, α_{in}(deg), at channel midspan

BETAO : Outlet absolute flow angle, α_{ex}(deg), at channel midspan

OMEGA : Dummy parameter, leave it blank

SC : Space-to-chord ratio, s/C'

CHORD : Blade chord, C'

Card.(7). ((YBL(I,J),(J=1,NX1+1,NX2), I=1,2)

YBL(I,J): y-coordinates of the corner nodes of the elements on the blade surface,

J= no. of hexahedral blocks to the left of the node+1,

I= blade surface index (1=upper, 2=lower)

Note: NX1 = No. of hexahedral blocks upstream of the blade row

NX2 = No. of hexahedral blocks up to the blade trailing edge

Presently NX1 and NX2 are fixed at 3 and 12

If INTG > 2 (isoparametric) then

Card Set.(8). ((YBLI(I,J),(J=1,NX2-NX1),I=1,2)

YBLI(I,J): y-coordinates of the mid-side nodes of the element on the blade surface,

J= No. of hexahedral blocks to the left of the node-NX1

I= surface index as before

Card Set.(9). (ZBL(I),I=1,NX3+1)

ZBL(i) : z-coordinate of the left face of the ith hexahedral block

Note: NX3 denotes the total number of hexahedral blocks which is internally fixed at 15

If IDIM = 3 (3-D Turbomachine) then

Card.(6). BETAI,BETAO,OMEGA,NBL,CHORD

BETAI : Inlet absolute flow angle, α_{in}(deg), at channel midspan

BETAO : Outlet absolute flow angle, α_{ex}(deg) at channel midspan

OMEGA : Rotational speed (rpm)

NBL : Number of blades

CHORD : Axial blade chord, C

Card Set.(7). ((THBLH(I,J),J=NX1+1,NX2),I=1,2)

THBLH(I,J): Tangential coordinates of the corner nodes of the elements on the blade surface at the hub,
J= no. of hexahedral blocks to the left of the node+1,
I= blade surface index (1=upper, 2=lower)

Card Set.(8). ((THBLT(I,J),J=NX1+1,NX2,I=1,2)

THBLT(I,J): Tangential coordinates of the corner nodes of the element on the blade surface at the casing,
I and J defined as in Card Set.(7).

If INTG > 2 (isoparametric) then

Card Set.(9). ((THETAH(I,J),J=1,NX2-NX1),I=1,2)

THETAH(I,J): Tangential coordinates of the mid-side nodes of the element on the blade surface at the hub,
J= No. of hexahedral blocks to theleft of the node-NX1
I= surface index as before

Card Set.(10). ((THETAT(I,J),J=1,NX2-NX1),I=1,2)

THETAT(I,J): Tangential coordinates of the mid-side nodes of the element on the blade surface at the tip,
I and J defined as in Card Set.(9).

Card Set.(11). (ZBLH(I),RH(I),ZBLT(I),RT(I),I=1,NX3+1)

ZBLH(I) : Axial coordinate of the left face of the ith hexahedral block at the hub

RH(I) : Radial coordinate of the left face of the ith hexahedral block at the hub

ZBLT(I) : Axial coordinate of the left face of the ith hexahedral block at the casing

RT(I) : Radial coordinate of the left face of the ith hexahedral block at the casing

Card.(12). REFL

REFL : Reference length

If IVEL > 0 then

Card.(12) or (10). NVELP

NVELP : Number of cross-sectional planes on which velocities, densities and Mach numbers are to be calculated

Card Set.(13) or (11).(XVP(I),I=1,NVELP)

XVP(I) : Axial coordinates of the ith cross-sectional plane

Card.(14) or (12). NVR,NVT

NVR : Number of equally spaced intervals in the radial direction

NVT : Number of equally spaced intervals in the radial direction

Note: There are (NVR+1)x(NVT+1) points equally spaced on each cross-sectional plane

If ICOMP > 0 (compressible flow iteration run) then

Card.(4). MIN,GAMA

MIN : Inlet Mach number at channel midspan, M_{in}

GAMA : Specific heat ratio,

Card.(5). ITER,CREL

ITER : Iteration number (1 for first iteration)

CREL : Relative error tolerance (percent)

Note: Card sequence 12-14 follows the above if IVEL>0

In all options, unless IVEL=2, the card deck ends with the following card

Last Card. IDEBUG

IDEBUG : Program debugging flag

= 0 : no debug

> 0 : global code numbers and coordinates of hexahedra, which define the finite element mesh topology generated are printed

= 1000 : complete debug of the frontal solution. The computer output is very long and detailed. Therefore, this option should be use only when necessary.

APPENDIX E

COMPUTER CODE LISTINGS

```
C
C
C
C
C          P R O G R A M     T U R B O
C
C        FINITE ELEMENT COMPUTER PROGRAM FOR CALCULATING
C        3-D  POTENTIAL FLOW IN TURBOMACHINERY
C
C        MARCH   1983,BOGAZICI UNIVERSITY
C
      COMMON/PRINT/IDEBUG,IVEL,ICOMP,NP,IMESH
C
C      SUBROUTINE INPUT READS AND STORES ALL INPUT DATA
C
      CALL INPUT
C
C        SUBROUTINE SYSMAT IS USED TO CONSTRUCT THE SYSTEM MATRIX
C
      CALL SYSMAT
C
C        SUBROUTINE FRONT IS USED TO
C        SOLVE THE GOVERNING EQUATIONS
C
      CALL FRONT
C
C        NODAL VALUES OF THE POTENTIAL ARE PRINTED BY
C        SUBROUTINE OUTPUT
C
C
      CALL OUTPUT
C
C
C     VELOCITIES ARE CALCULATED AND PRINTED IN
C           SUBROUTINE VELOCY
C
C
      IF(IVEL.GT.0) CALL VELOCY
C
C
C      SURFACE VELOCITIES ARE CALCULATED AND PRINTED IN
C                  SUBROUTINE SURVEL
C
C
      IF(IVEL.GT.0) CALL SURVEL
      STOP
    1 FORMAT()
      END
```

```
      SUBROUTINE INPUT
C
C     THIS SUBROUTINE READS AND STORES ALL INPUT DATA
C
C
      PARAMETER NX1=3,NX2=12,NX3=15
      PARAMETER NPO=45 , NE2=8 , NCP=NX3+1 , NET=NE2*NX3
      PARAMETER N= NPO*(2*NX3+1)
      REAL MO,LAMBDA,LAMBD1,LAMBD2,MACHO
      COMMON/INPU1/LAMBD1,LAMBD2,OMEGA,NBL,CHORD,PITCH
      COMMON/INPU2/THBLH(2,NCR),ZBLH(NCR),THBLT(2,NCR),ZBLT(NCR),
     1  NGB1(NE2,9),NGB2(NE2,9),RH(NCR),RT(NCR)
      COMMON /PRINT/ IDEBUG,IVLL,ICOMP,NP,IMESH
      COMMON/BNDR/DELIDX,BETAIN,BETOUT
      COMMON/CMPRSS/POT(N),MO,GAMA,LAMBDA
      COMMON/ITER/ITER,CREL
      COMMON/FRON1/NOP(NET,27),NOPP(N),MDF(N),NCOD(N),BC(N),R(N)
      COMMON/VELO/XVP(100),NVELP,NVB,NVT
      COMMON/PERE/NCODT(6,10),NPER(9),LE,RC
      COMMON/PARAM/NDT,NDR,NE1,NP1,NP2,NPOIN
      COMMON/ISBL/THETAH(2,NX2-NX1),THETAT(2,NX2-NX1)
      COMMON/TWDIM/YBL(2,NCR),YBLI(2,NCR),XBL(NCR)
      COMMON/DIM/IDIM,INTG
      COMMON/DISK/UO,BETAI,BETAO,SC,REFL
      COMMON/MIXED/VRIN
      DIMENSION V(100),VT(100)
      INTEGER HEADR(80)
C
C
C     READ CASE IDENTIFICATION HEADING
C
C
      WRITE(6,10)
      READ(5,2) HEADR
      WRITE(6,20) (HEADR(I),I=1,80)
C
C     READ PROGRAM  CONTROL FLAGS
C
      READ(5,1) ICOMP,IDIM
      WRITE(6,30) ICOMP,IDIM
      IF(ICOMP.EQ.0) THEN
      IF(IDIM.EQ.2) THEN
      WRITE(6,40)
      END IF
      IF(IDIM.EQ.3) THEN
      WRITE(6,50)
      END IF
      ELSE
      IF(IDIM.EQ.2) THEN
      WRITE(6,60)
      END IF
      IF(IDIM.EQ.3) THEN
      WRITE(6,70)
      END IF
      END IF
      WRITE(6,80)
      WRITE(6,81)
      READ(5,1) INTG,IVEL
      WRITE(6,90) INTG,IVEL
      IF(INTG.EQ.1) WRITE(6,100)
      IF(INTG.EQ.2) WRITE(6,110)
      IF(INTG.EQ.3) WRITE(6,120)
      IF(INTG.EQ.4) WRITE(6,130)
      IF(IVEL.EQ.0) WRITE(6,140)
      IF(IVEL.EQ.1) WRITE(6,150)
      IF(IVEL.EQ.2) WRITE(6,160)
C
C     IVEL > 1    OR    ICOMP > 0
C
C     READ   INPUT DATA FROM DISK FILE NO. 7
C
C
      IF(IVEL.GT.1.OR.ICOMP.GT.0) CALL RDISK
      IF(ICOMP.EQ.0.AND.IVEL.LT.2) THEN
      WRITE(6,340)
      READ(5,1) NDT,NDR
      IF(IDIM.EQ.2) NDR=1
```

```
      WRITE(6,570) NDT,NDR
      WRITE(6,170)
      READ(5,1) UO
      WRITE(6,350) UO
      IF(IDIM.EQ.2) THEN
C
CCC
CCC     2 - D   CASCADE INPUT DATA
CCC
C
      WRITE(6,180)
      READ(5,1) BETAI,BETAO,OMEGA,SC,CHORD
      WRITE(6,360) BETAI,BETAO,OMEGA,SC,CHORD
      WRITE(6,190)
      READ(5,1) ((YBL(I,J),J=NX1+1,NX2+1),I=1,2)
      WRITE(6,370)
      WRITE(6,420) (J,(YBL(I,J),I=1,2),J=NX1+1,NX2+1)
      IF(INTG.GT.2) THEN
      WRITE(6,200)
      READ(5,1) ((YBLI(I,J),J=1,NX2-NX1),I=1,2)
      WRITE(6,380)
      WRITE(6,420) (J,(YBLI(I,J),I=1,2),J=1,NX2-NX1)
      END IF
      WRITE(6,210)
      READ(5,1) (XBL(I),I=1,NCR)
      WRITE(6,390)(XBL(I),I=1,NCR)
      END IF
      IF(IDIM.EQ.3) THEN
C
CCC     3 - D TURBOMACHINE INPUT DATA
CCC
C
      WRITE(6,220)
      READ(5,1) BETAI,BETAO,OMEGA,NBL,CHORD
      WRITE(6,400) BETAI,BETAO,OMEGA,NBL,CHORD
      WRITE(6,230)
      READ(5,1) ((THBLH(I,J),J=NX1+1,NX2+1),I=1,2)
      WRITE(6,410)
      WRITE(6,420) (J,(THBLH(I,J),I=1,2),J=NX1+1,NX2+1)
      WRITE(6,240)
      READ(5,1) ((THBLT(I,J),J=NX1+1,NX2+1),I=1,2)
      WRITE(6,430)
      WRITE(6,420) (J,(THBLT(I,J),I=1,2),J=NX1+1,NX2+1)
      IF(INTG.GT.2) THEN
      WRITE(6,250)
      READ(5,1) ((THETAH(I,J),J=1,NX2-NX1),I=1,2)
      WRITE(6,440)
      WRITE(6,420) (J,(THETAH(I,J),I=1,2),J=1,NX2-NX1)
      WRITE(6,260)
      READ(5,1) ((THETAT(I,J),J=1,NX2-NX1),I=1,2)
      WRITE(6,450)
      WRITE(6,420) (J,(THETAT(I,J),I=1,2),J=1,NX2-NX1)
      END IF
      WRITE(6,270)
      READ(5,1) (ZBLH(I),RH(I),ZBLT(I),RT(I),I=1,NCR)
      WRITE(6,460)
      WRITE(6,470) (I,ZBLH(I),RH(I),ZBLT(I),RT(I),I=1,NCR)
      END IF
      IF(IDIM.EQ.3) THEN
      WRITE(6,330)
      READ(5,1) REFL
      ELSE
      REFL=CHORD
      END IF
      END IF
C
      IF(ICOMP.GT.0) THEN
C
CCC
CCC     COMPRESSIBLE  FLOW  :
CCC     READ INLET MACH NO. AND SPECIFIC HEAT RATIO
CCC
C
      WRITE(6,280)
      READ(5,1) MO,GAMA
      WRITE(6,480) MO,GAMA
      END IF
      IF(IVFL.GT.0) THEN
C
C
```

```
C       READ AXIAL COORDINATES OF THE VELOCITY PLANES
C
C
        WRITE(6,290)
        READ(5,1) NVELP
        WRITE(6,490) NVELP
        WRITE(6,300)
        READ(5,1) (XVP(IV),IV=1,NVELP)
        WRITE(6,500)
        WRITE(6,510) (IV,XVP(IV),IV=1,NVELP)
        WRITE(6,310)
        READ(5,1) NVR,NVT
        WRITE(6,520) NVR,NVT
        END IF
C
C     WRITE INPUT DATA ON DISK FILE NO. 8
C
        IF(IVEL.LT.2) CALL WDISK
C
        IF(IVEL.GT.0) THEN
        DO 12 I=1,NVELP
        XVP(I)=XVP(I)/REFL
     12 CONTINUE
        END IF
        IF(IDIM.EQ.2) STAGER=YBL(1,NX2+1)
        IF(IDIM.EQ.3) STAGER=THBLH(1,NX2+1)
        IMESH=1
        IF(STAGER.GT.0) IMESH=2
        DO 111 K=1,3
        IF(IMESH.EQ.1) KK=K+6
        IF(IMESH.EQ.2) KK=K
        NPER(K)=KK
        NPER(K+3)=NPER(K)+9
        NPER(K+6)=NPER(K+3)+9
    111 CONTINUE
        NP1=(2*NDR+1)*(2*NDT)
        NP2=(2*NDR+1)*(2*NDT+1)
        NPOIN=NP1*(2*NX1+1)+NP2*(2*(NX2-NX1)-1)+NP1*(2*(NX3-NX2)+1)
        NE1=NDT*NDR
        IF(ICOMP.GT.0.AND.IVEL.LT.2) THEN
        WRITE(6,320)
        READ(5,1) ITER,CREL
        WRITE(6,530) ITER,CREL
        END IF
        IF(IDIM.EQ.2) THEN
        DO 3 I=1,2
        DO 3 J=NX1+1,NX2+1
        THBLH(I,J)=YBLI(I,J)/REFL
        THBLT(I,J)=YBLI(I,J)/REFL
     3  CONTINUE
        IF(INTG.GT.2) THEN
        DO 4 I=1,2
        DO 4 J=1,NX2-NX1
        THETAH(I,J)=YRLI(I,J)/REFL
        THETAT(I,J)=YRLI(I,J)/REFL
     4  CONTINUE
        END IF
        DO 5 I=1,NCR
        ZBLH(I)=XBL(I)/REFL
        ZBLT(I)=XBL(I)/REFL
        RH(I)=0.0
        RT(I)=SC/NDT
     5  CONTINUE
        END IF
        IF(IDIM.EQ.3) THEN
        DO 11 I=1,NCR
        RH(I)=RH(I)/REFL
        RT(I)=RT(I)/REFL
        ZBLH(I)=ZBLH(I)/REFL
        ZBLT(I)=ZBLT(I)/REFL
    11  CONTINUE
        END IF
        PI=ACOS(-1)
        IF(IDIM.EQ.3) THEN
        PITCH=2*PI/NBL
        END IF
        IF(IDIM.EQ.2) PITCH=SC
        DO 6 I=1,2
        DO 6 J=1,NX1
        THBLH(I,J)=0.0
        THBLT(I,J)=0.0
     6  CONTINUE
```

```
      DO 7 I=1,2
      DO 7 J=NX2+2,NX3+1
      THBLH(I,J)=THBLH(I,NX2+1)
      THBLT(I,J)=THBLT(I,NX2+1)
    7 CONTINUE
      DO 8 I=1,NCR
      THBLH(2,I)=THBLH(2,I)+PITCH
      THBLT(2,I)=THBLT(2,I)+PITCH
    8 CONTINUE
      IF(INTG.GT.2) THEN
      DO 9 I=1,NX2-NX1
      THETAH(2,I)=THETAH(2,I)+PITCH
      THETAT(2,I)=THETAT(2,I)+PITCH
    9 CONTINUE
      END IF
      BETAIN=BETAI*PI/180.
      BETOUT=BETAO*PI/180.
      RMEANI=(RH(1)+RT(1))/2.
      BETIR1=(RH(2)-RH(1))/(ZBLH(2)-ZBLH(1))
      BETIR2=(RT(2)-RT(1))/(ZBLT(2)-ZBLT(1))
      BETIR1=ATAN(BETIR1)
      BETIR2=ATAN(BETIR2)
      BETIR=(BETIR1+BETIR2)/2.
      FAC=(RT(1)**2-RH(1)**2)/(RT(NCR)**2-RH(NCR)**2)
      VZIN=SQRT(1./(TAN(BETIR)**2+TAN(BETAIN)**2+1))
      VTIN=TAN(BETAIN)*VZIN
      VRIN=TAN(BETIR)*VZIN
      LAMBD1=RMEANI*VTIN*PITCH
      IF(IDIM.EQ.2) LAMBD1=VTIN*PITCH
      DFIDX=VZIN*FAC
      VZOUT=VZIN*FAC
      VTOUT=TAN(BETOUT)*VZOUT
      RMEANO=(RH(NX3+1)+RT(NX3+1))/2.
      LAMBD2=RMEANO*VTOUT*PITCH
      IF(IDIM.EQ.2) LAMBD2=VTOUT*PITCH
      BETOR1=(RH(NCR)-RH(NX3))/(ZBLH(NCR)-ZBLH(NX3))
      BETOR2=(RT(NCR)-RT(NX3))/(ZBLT(NCR)-ZBLT(NX3))
      BETOR1=ATAN(BETOR1)
      BETOR2=ATAN(BETOR2)
      BETOR=(BETOR1+BETOR2)/2.
      VROUT=TAN(BETOR)*VZOUT
      VELOUT=SQRT(VROUT**2+VTOUT**2+VZOUT**2)
      IF(ICOMP.GT.0) THEN
      I=1
  113 V(I)=VELOUT
      VT(I)=VTOUT
      TTO=1+(GAMA-1)*MO**2*((1-VELOUT**2)/2.
     1    +OMEGA*(VTOUT*RMEANO-LAMBD1/PITCH))
      MACHO=MO*VELOUT/SQRT(TTO)
      DENSTO=(TTO)**(1/(GAMA-1))
      VELOUT=V(1)/DENSTO
      VTOUT=VT(1)/DENSTO
      I=I+1
      V(I)=VELOUT
      RELERR=(V(I)-V(I-1))/V(I)
      IF(RELERR.GT.0.01) GO TO 113
      LAMBDA=LAMBD2
      LAMBD2=LAMBD2/DENSTO
      IF(ITER.GT.1) LAMBDA=LAMBD2
      END IF
      WRITE(6,550)
      WRITE(6,560) LAMBD1,DENSTO,VELOUT,LAMBD2
      NRAD=2*NDR+1
      DO 112 J=1,NDT
      DO 112 I=1,NDR
      DO 112 K=1,3
      II=(J-1)*NDR+I
      IF(IMESH.EQ.1) KK=K
      IF(IMESH.EQ.1) KK3=KK+3
      IF(IMESH.EQ.1) KK6=KK+6
      IF(IMESH.EQ.2) KK=10-K
      IF(IMESH.EQ.2) KK3=KK-3
      IF(IMESH.EQ.2) KK6=KK-6
      NGB1(II,KK)=(I-1)*2+(J-1)*2*NRAD+K
      NGB2(II,KK)=(I-1)*2+(J-1)*2*NRAD+K
      NGB2(II,KK3)=NGB2(II,KK)+NRAD
      NGB1(II,KK3)=NGB1(II,KK)+NRAD
      NGB2(II,KK6)=NGB2(II,KK3)+NRAD
      NGB1(II,KK6)=NGB1(II,KK3)+NRAD
      IF(J.EQ.NDT) THEN
      NGB1(II,KK6)=(I-1)*2+K
      END IF
```

```
  112 CONTINUE
C
C
      IF(IVEL.EQ.2) THEN
      CALL VELOCY
      CALL SURVEL
      STOP
      END IF
      WRITE(6,580)
      WRITE(6,590) ((NGB1(I,J),J=1,9),I=1,NE1)
      WRITE(6,600)
      WRITE(6,590) ((NGB2(I,J),J=1,9),I=1,NE1)
    1 FORMAT()
    2 FORMAT(80A1)
   10 FORMAT(1H1,///,10X,,3-D FLOW PROGRAM TURBO EXECUTION,,//,
     1/,5X,,ENTER CASE IDENTIFICATION HEADING,,5X,,FORMAT 80A1,,//)
   20 FORMAT(10X,,TURBO EXECUTION FOR CASE,,/,3X,80A1,///,
     1/,5X,,  ALL FORMATS FREE  ,,///,
     2/,5X,,ICOMP    : COMPRESSIBILITY INDICATION FLAG,,/,
     3/,5X,,         =0: INCOMPRESSIBLE FLOW,,
     4/,5X,,         >0: COMPRESSIBLE FLOW,,/,
     5/,5X,,IDIM     : CASCADE DIMENSION FLAG,,/,
     6/,5X,,         =2: 2-D CASCADE,,
     7/,5X,,         =3: 3-D TURBOMACHINE,,/,
     8/,5X,,ENTER  ICOMP,IDIM  ,,/)
   30 FORMAT(//,10X,,ICOMP=,,I2,2X,,IDIM=,,I2,/)
   40 FORMAT(10X,,INCOMPRESSIBLE FLOW IN A 2-D CASCADE,,/)
   50 FORMAT(10X,,INCOMPRESSIBLE FLOW IN A 3-D TURBOMACHINE,,/)
   60 FORMAT(10X,,COMPRESSIBLE FLOW IN A 2-D CASCADE,,/)
   70 FORMAT(10X,,COMPRESSIBLE FLOW IN A 3-D TURBOMACHINE,,/)
   80 FORMAT(/,5X,,INTG     : INTEGRATION CONTROL FLAG,,/,
     1/,5X,,         =1: ANALYTICAL INTEGRATION (SUBPARAMETRIC),,
     2/,5X,,         =2: NUMERICAL INTEGRATION (SUBPARAMETRIC),,
     3/,5X,,         =3: NUMERICAL INTEGRATION (ISOPARAMETRIC),,
     4/,5X,,         =4: NUMERICAL INTEGRATION (BOUNDARIES ISOPARAMETRIC),,
     5//,5X,,IVEL     : VELOCITY CALCULATION FLAG,,/,
     6//,5X,,        =0: VELOCITIES WILL NOT BE CALCULATED,,
     7/,5X,,         =1: VELOCITIES WILL BE CALCULATED BASED ON THE,)
   81 FORMAT(10X,,      VELOCITY POTENTIALS COMPUTED IN THE PRESENT,,
     1,  EXECUTION,,
     1/,6X,,         =2: CALCULATE VELOCITIES BASED ON THE VELOCITY,,
     2/,7X,,          POTENTIALS STORED IN DISK FILE NO. 7,,/,
     2/,5X,,ENTER  INTG,IVEL,,/)
   90 FORMAT(//,10X,,INTG=,,I2,2X,,IVEL=,,I2,/)
  100 FORMAT(//,10X,,DENSITY AND RADIUS WILL BE AVERAGED IN THE ,,
     1,ELEMENT,,//,10X,,ELEMENT MATRICES WILL BE CALCULATED BASED,,
     2,  ANALYTICAL INTEGRATION RESULTS,,/)
  110 FORMAT(//,10X,,SUBPARAMETRIC ELEMENTS WILL BE USED,,/,
     110X,, AND ELEMENT MATRICES WIL BE CALCULATED NUMERICALY,,/)
  120 FORMAT(//,10X,,ISOPARAMETRIC ELEMENTS WILL BE USED,,/,
     110X,, IN THE ENTIRE SOLUTION DOMAIN,,/)
  130 FORMAT(//,10X,,ISOPARAMETRIC ELEMENTS WILL BE USED,,/,
     110X,, ONLY AT THE BLADE BOUNDARIES,,/)
  140 FORMAT(//,10X,,VELOCITIES WILL NOT BE CALCULATED,,/)
  150 FORMAT(//,10X,,VELOCITIES WILL BE CALCULATED BASED ON,,/,
     110X,, VELOCITY POTENTIALS COMPUTED IN THE PRESENT EXECUTION,,/)
  160 FORMAT(//,10X,,VELOCITIES WILL BE CALCULATED BASED ON,,/,
     110X,, VELOCITY POTENTIALS STORED IN DISK FILE NO. 7,,/)
  170 FORMAT(//,10X,,UO= INLET VELOCITY (FPS),,/,
     1/,5X,,ENTER UO,,/)
  180 FORMAT(//,10X,,2-D CASCADE INPUT DATA,,//,
     1/,5X,,BETAI    : INLET ABSOLUTE FLOW ANGLE (DEGREES),,
     2/,5X,,BETAO    : OUTLET ABSOLUTE FLOW ANGLE (DEGREES),,
     3/,5X,,OMEGA    : ANGULAR VELOCITY (ROTATIONAL SPEED),(RPM),,
     4/,5X,,SC       : SPACE TO CHORD RATIO,,
     5/,5X,,CHORD    : AXIAL CHORD LENGTH (FT),,/,
     6/,5X,,ENTER  BETAI,BETAO,OMEGA,SC,CHORD,,/)
  190 FORMAT(//,5X,,YBL(I,J) : CASCADE Y-COORDINATES (FT) ,,/,
     1 5X,,ENTER  ((YBL(I,J),J=NX1+1,NX2),I=1,2),,/)
  200 FORMAT(//,5X,,ISOPARAMETRIC ELEMENTS WILL BE USED,,/,
     15X,,YBLI(I,J) : Y-COORDINATES OF THE MID SIDE NODES,,/,
     25X,,            ON THE BLADE,,/,
     35X,,ENTER  ((YBLI(I,J),J=1,NX2-NX1),I=1,2),,/)
  210 FORMAT(//,5X,,XBL(I)   : X-COORDINATES OF THE INPUT DATA,,/,
     15X,,ENTER  (XBL(I),I=1,NX3+1),,/)
  220 FORMAT(//,10X,,3-D TURBOMACHINE INPUT DATA,,///,
     1/,5X,,BETAI    : INLET ABSOLUTE FLOW ANGLE (DEGREES),,
     2/,5X,,BETAO    : OUTLET ABSOLUTE FLOW ANGLE (DEGREES),,
     3/,5X,,OMEGA    : ANGULAR VELOCITY (ROTATIONAL SPEED),(RPM),,
     4/,5X,,NBL      : NO. OF BLADES,,
     5/,5X,,CHORD    : AXIAL CHORD LENGTH (FT),,/,
     6/,5X,,ENTER  BETAI,BETAO,OMEGA,NBL,CHORD,,/)
```

```
  230 FORMAT(//,5X,,THBLH(I,J) :BLADE TANGENTIAL COORDINATES,,
     1/,5X,,                    AT THE HUB,,/,
     2/,5X,,ENTER ((THBLH(I,J),J=NX1+1,NX2),I=1,2,,/)
  240 FORMAT(//,5X,,THBLT(I,J) :BLADE TANGENTIAL COORDINATES,,
     1/,5X,,                    AT THE TIP,,/,
     2/,5X,,ENTER ((THBLH(I,J),J=NX1+1,NX2),I=1,2,,/)
  250 FORMAT(//,5X,,ISOPARAMETRIC ELEMENTS WILL BE USED,,/,
     1 5X,,THETAH(I,J) : BLADE TANGENTIAL COORDINATES ,,
     2/5X,,                     OF THE MID-SIDE NODES AT THE HUB,,/,
     3/5X,,ENTER ((THETAH(I,J),J=1,NX2-NX1),I=1,2),,/)
  260 FORMAT(//,5X,,THETAT(I,J) : BLADE TANGENTIAL COORDINATES ,,
     2/5X,,                     OF THE MID-SIDE NODES AT THE TIP,,/,
     3/5X,,ENTER ((THETAT(I,J),J=1,NX2-NX1),I=1,2),,/)
  270 FORMAT(//,5X,,ZBLH(I) : AXIAL COORDINATES OF THE INPUT DATA,,
     1/,5X,,                                AT THE HUB,,
     2/,5X,,RH(I)    : RADIAL COORDINATES OF THE INPUT DATA,,
     3/,5X,,                                AT THE HUB,,
     4/,5X,,ZBLT(I) : AXIAL COORDINATES OF THE INPUT DATA,,
     5/,5X,,                                AT THE TIP,,
     6/,5X,,RT(I)    : RADIAL COORDINATES OF THE INPUT DATA,,
     7/,5X,,                                AT THE TIP,,
     8//,5X,,ENTER (ZBLH(I),RH(I),ZBLT(I),RT(I),I=1,NX3+1),,/)
  280 FORMAT(//,5X,,FLOW IS COMPRESSIBLE,,/,
     1 5X,,MO     : INLET MACH NUMBER,,
     2/5X,,GAMA   : SPECIFIC HEAT RATIO,,
     2/5X,,ENTER MO,GAMA,,/)
  290 FORMAT(//,10X,,VELOCITIES WILL BE CALCULATED ON VARIOUS,,
     1 ,CROSS-SECTIONAL PLANES,,/,
     2/,5X,,NVELP : NUMBER OF CROSS-SECTIONAL PLANES,,
     3/,5X,,ENTER NVELP,,/)
  300 FORMAT(//,5X,,XVP(I) : AXIAL-CORDINATES OF THE ,,
     1/,5X,,                    CROSS-SECTIONAL PLANES,,
     2/,5X,,ENTER (XVP(I),I=1,NVELP),,/)
  310 FORMAT(//,5X,,THERE WILL BE NVR*NVT POINTS EQUALLY SPACED,,
     1/,5X,,ON EACH CROSS-SECTIONAL PLANE,,
     2/,5X,,NVR   : NUMBER OF POINTS IN THE RADIAL DIRECTION,,
     3/,5X,,NVT   : NUMBER OF POINTS IN THE TANGENTIAL DIRECTION,,
     4/,5X,,ENTER NVR,NVT,,/)
  320 FORMAT(//,5X,,COMPRESSIBLE FLOW ITERATION ,,
     2/,5X,,ITER  : ITERATION NUMBER ,,
     3/,5X,,CREL  : RELATIVE ERROR TOLERANCE,,
     4/,5X,,ENTER ITER,CREL,,/)
  330 FORMAT(//,5X,,REFL  : REFERENCE LEGTH  (FT) ,,
     1/,5X,,ENTER REFL,,/)
  340 FORMAT(//,5X,,SOLUTION DOMAIN WILL BE CROSS-SECTIONALLY,,
     1 ,DISCRETIZED INTO  NDT*NDR HEXAHEDRA,,
     2/,5X,,NDT   : NUMBER OF HEXAHEDRA IN THE TANGENTIAL DIRECTION,,
     3/,5X,,NDR   : NUMBER OF HEXAHEDRA IN THE RADIAL DIRECTION,,
     4/,5X,,FOR 2-D CASCADES NDR=1,,
     5/,5X,,ENTER NDT,NDR,,/)
  350 FORMAT(//,5X,,INLET VELOCITY UO=,,F10.4,, (FPS),,/)
  360 FORMAT(//,5X,,BETAI,,5X,,BETAO,,5X,,OMEGA,,5X,,  SC ,,
     1 5X,,CHORD,,/,1X,5(F9.2),/)
  370 FORMAT(//,5X,,  J  ,,5X,, YBL(1,J) ,,5X,, YBL(2,J) ,,/,
     1 5X,3(1X,,2(6X,10(1H-),),/)
  380 FORMAT(//,5X,,  J  ,,5X,, YBLI(1,J),,5X,, YBLI(2,J),,/,
     1 5X,3(1X,,2(6X,10(1H-),),/)
  390 FORMAT(//,5X,,XBL  ARRAY,,/,(8(E11.4)))
  400 FORMAT(//,5X,,BETAI,,5X,,BETAO,,5X,,OMEGA,,5X,,NBL,,
     1 5X,,CHORD,,/,1X,3(F9.2),5X,I3,1X,F9.2,/)
  410 FORMAT(//,5X,,  J  ,,5X,,THBLH(1,J),,5X,,THBLH(2,J),,/,
     1 5X,3(1X,,2(6X,10(1H-),),/)
  420 FORMAT((5X,I3,2(5X,E10.4)))
  430 FORMAT(//,5X,,  J  ,,5X,,THBLT(1,J),,5X,,THBLT(2,J),,/,
     1 5X,3(1X,,2(6X,10(1H-),),/)
  440 FORMAT(//,5X,,  J  ,,5X,,THETAH(1,J),,5X,,THETAH(2,J),,/,
     1 5X,3(1X,,2(6X,10(1H-),),/)
  450 FORMAT(//,5X,,  J  ,,5X,,THETAT(1,J),,5X,,THETAT(2,J),,/,
     1 5X,3(1X,,2(6X,10(1H-),),/)
  460 FORMAT(//,5X,,  I  ,,5X,,  ZBLH(I)  ,,5X,,  RH(I)   ,,
     15X,,  ZBLT(I)  ,,5X,,  RT(I)   ,,/,5X,3(1H-),4(5X,10(1H-)),//)
  470 FORMAT((5X,I3,4(5X,E10.4)))
  480 FORMAT(//,5X,,INLET MACH NUMBER MO =,,F10.4,2X,
     1 ,GAMA=,,F10.4,/)
  490 FORMAT(//,5X,,VELOCITIES WILL BE CALCULATED ON,,I4,2X,
     1 ,CROSS-SECTIONAL PLANES,,/)
  500 FORMAT(/,5X,,  I  ,,5X,,  XVP(I)  ,,/,5X,3(1H-),5X,10(1H-),/)
  510 FORMAT(5X,I3,5X,E10.4)
  520 FORMAT(/,10X,,NO. OF POINTS IN THE RADIAL DIRECTION (NVR)=,,
     1 I4,/,10X,,NO. OF POINTS IN THE TANGENTIAL DIRECTION (NVT)=,,
     2 I4,/)
  530 FORMAT(///,,COMPRESSIBLL FLOW ITERATION NO.,,I4,/,10X,
```

```
     1 ,RELATIVE ERROR TOLERANCE (PERCENT) =,,F10.4,/)
  540 FORMAT(//,10X,,REFERENCE LENGTH =,,F10.4,/)
  550 FORMAT(//,5X,,LAMBD1,,5X,,OUTLET DENSITY,,5X,,COMP.,,
     1 , OUTLET VEL.,,5X,,LAMBD2,,/)
  560 FORMAT(4X,F6.3,8X,F11.4,11X,F11.4,4X,F6.3,/)
  570 FORMAT(4X,,NDT=,,I3,2X,,NDR=,,I3,/)
  580 FORMAT(//,5X,,ELMT NO.,,5X,,NGB1 ARRAY,,/)
  590 FORMAT(7X,I4,8X,9I4)
  600 FORMAT(//,5X,,ELMT NO.,,5X,,NGB2 ARRAY,,/)
      RETURN
      END
```

```
      SUBROUTINE RDTSK
C
C     THIS SUBROUTINE READS AND STORES ALL INPUT DATA.
C
      PARAMETER NX1=3,NX2=12,NX3=15
      PARAMETER NPO=45 , NE2=8 , NCR=NX3+1 , NET=NE2*NX3
      PARAMETER NE= NPO*(2*NX3+1)
      REAL MO,LAMBDA,LAMBD1,LAMBD2,MACHO
      COMMON/INPU1/LAMBD1,LAMBD2,OMEGA,NBL,CHORD,PITCH
      COMMON/INPU2/THBLH(2,NCR),ZBLH(NCR),THBLT(2,NCR),ZBLT(NCR),
     1   NGB1(NE2,9),NGB2(NE2,9),RH(NCR),RT(NCR)
      COMMON /PRINT/ IDEBUG,IVEL,ICOMP,NP,IMESH
      COMMON/UNDR/DELIDX,BETAIN,BETOUT
      COMMON/CMPRSS/POT(N),MO,GAMA,LAMBDA
      COMMON/ITEP/ITER,CREL
      COMMON/FRON1/NOP(NET,27),NOPP(N),MDF(N),NCOD(N),RC(N),R(N)
      COMMON/VELO/XVP(100),NVELP,NVP,NVI
      COMMON/PERE/NCODT(6,10),NPER(9),LE,RC
      COMMON/PARAM/NDT,NDR,NE1,NP1,NP2,NPOIN
      COMMON/ISBL/THETAH(2,NX2-NX1),THETAT(2,NX2-NX1)
      COMMON/DIM/IDIM,INTG
      COMMON/DISK/U0,BETAI,BETA0,SC,REFL
      COMMON/TWDIM/YBL(2,NCR),YBLI(2,NCR),XBL(NCR)

      READ(7) NDT,NDR
      NP1=(2*NDR+1)*(2*NDT)
      NP2=(2*NDR+1)*(2*NDT+1)
      NPOIN=NP1*(2*NX1+1)+NP2*(2*(NX2-NX1)-1)+NP1*(2*(NX3-NX2)+1)
      READ(7) U0
      IF(IDIM.EQ.2) THEN
      READ(7) BETAI,BETA0,SC,CHORD
      READ(7) ((YBL(I,J),J=NX1+1,NX2+1),I=1,2)
      IF(INTG.GT.2) THEN
      READ(7) ((YBLI(I,J),J=1,NX2-NX1),I=1,2)
      END IF
      READ(7) (XBL(I),I=1,NCR)
      END IF
      IF(IDIM.EQ.3) THEN
      READ(7) BETAI,BETA0,OMEGA,NBL,CHORD
      READ(7) ((THBLH(I,J),J=NX1+1,NX2+1),I=1,2)
      READ(7) ((THBLT(I,J),J=NX1+1,NX2+1),I=1,2)
      IF(INTG.GT.2) THEN
      READ(7) ((THETAH(I,J),J=1,NX2-NX1),I=1,2)
      READ(7) ((THETAT(I,J),J=1,NX2-NX1),I=1,2)
      END IF
      READ(7) (ZBLH(I),RH(I),ZBLT(I),RT(I),I=1,NCR)
      END IF
      IF(IDIM.EQ.3) THEN
      READ(7)  REFL
      ELSE
      REFL=CHORD
      END IF
      IF(IVEL.LT.2) READ(7) (I,POT(I),J=1,NPOIN)
      IF(IVEL.GT.1) READ(7) (I,R(I),J=1,NPOIN)
      READ(7) NOP
      RETURN
      END
```

```
      SUBROUTINE WDTSK
C
C     THIS SUBROUTINE READS AND STORES ALL INPUT DATA
C
C
      PARAMETER NX1=3,NX2=12,NX3=15
      PARAMETER NPO=45 , NE2=8 , NCR=NX3+1 , NFT=NP2*NX3
      PARAMETER NE= NPO*(2*NX3+1)
      REAL MO,LAMBDA,LAMBD1,LAMBD2,MACHO
      COMMON/INPU1/ AMBD1,LAMBL2,OMEGA,NBL,CHORD,PITCH
      COMMON/INPU2/THBLH(2,NCR),ZBLH(NCR),THBLT(2,NCR),ZBLT(NCR),
     1  NGB1(NE2,9),NGB2(NE2,9),RH(NCR),RT(NCR)
      COMMON /PRINT/ IDEBUG,IVEL,ICOMP,NP,IMESH
      COMMON/BNDR/DFIDX,BETAIN,BETOUT
      COMMON/CMPRSS/POT(N),MO,GAMA,LAMBDA
      COMMON/ITER/ITER,CREL
      COMMON/FRON1/NOP(NET,27),NOPP(N),MDF(N),NCOU(N),BC(N),R(N)
      COMMON/VELO/XVP(100),NVELP,NVP,NVT
      COMMON/PERE/NCODT(6,10),NPER(6),LE,RC
      COMMON/PARAM/NDT,NDR,NE1,NP1,NP2,NPOIN
      COMMON/ISBL/THETAH(2,NX2-NX1),THETAT(2,NX2-NX1)
      COMMON/DIM/IDIM,INTG
      COMMON/DISK/UO,BETAI,BETAO,SC,REFL
      COMMON/TWDIM/YBL(2,NCR),YBLI(2,NCR),XBL(NCR)

      WRITE(8) NDT,NDR
      WRITE(8) UO
      IF(IDIM.EQ.2) THEN
      WRITE(8) BETAT,BETAO,SC,CHORD
      WRITE(8) ((YBL(I,J),J=NX1+1,NX2+1),I=1,2)
      IF(INTG.GT.2) THEN
      WRITE(8) ((YBLI(I,J),J=1,NX2-NX1),I=1,2)
      END IF
      WRITE(8) (XBL(I),I=1,NCR)
      END IF
      IF(IDIM.EQ.3) THEN
      WRITE(8) BETAT,BETAO,OMEGA,NBL,CHORD
      WRITE(8) ((THBLH(I,J),J=NX1+1,NX2+1),I=1,2)
      WRITE(8) ((THBLT(I,J),J=NX1+1,NX2+1),I=1,2)
      IF(INTG.GT.2) THEN
      WRITE(8) ((THETAH(I,J),J=1,NX2-NX1),I=1,2)
      WRITE(8) ((THETAT(I,J),J=1,NX2-NX1),I=1,2)
      END IF
      WRITE(8) (ZBLH(I),RH(I),ZBLT(I),RT(I),I=1,NCR)
      END IF
      IF(IDIM.EQ.3) THEN
      WRITE(8)  REFL
      ELSE
      REFL=CHORD
      END IF
      RETURN
      END
```

```
      SUBROUTINE  SYSMAT
C
C        THIS SUBROUTINE CONSTRUCTS THE SYSTEM MATRIX
C
      PARAMETER NX1=3,NX2=12,NX3=15
      PARAMETER NPO=45 , NE2=8 , NCR=NX3+1 , NET=NE2*NX3
      PARAMETER NN=NPO*(2*NX3+1)
      REAL MO,LAMBDA,LAMBD1,LAMBD2
      COMMON ELMP(27,27),NG(27),X(27),Y(27),Z(27)
      COMMON/FROM1/NOP(NET,27),NOPP(NN),MDF(NN),NCOD(NN),BC(NN),R(NN)
      COMMON/INPU1/LAMBD1,LAMBD2,OMEGA,CHORD,NBL,PITCH
      COMMON/INPU2/THBLH(2,NCR),ZBLH(NCR),THBLT(2,NCR),ZBLT(NCR),
     1 NGB1(NE2,9),NGB2(NE2,9),RH(NCR),RT(NCR)
      COMMON/COORD/RBL(2,NPO),TBL(2,NPO),ZBL(2,NPO)
      COMMON/BNDR/DFIDX,BETAIN,BETOUT
      COMMON/PRINT/ IDEBUG,IVEL,ICOMP,NP,IMESH
      COMMON/CMPRSS/POT(NN),MO,GAMA
      COMMON/ISBL/THETAH(2*NX2-NX1),THETAT(2*NX2-NX1)
      COMMON/DIM/IDIM,INTG
      COMMON/PARAM/NDT,NDR,NE1,NP1,NP2,NPOIN
      COMMON/MIXED/VRIN
      DIMENSION NCODT(6,10)
      DATA NCODT/27,27,27,1,1,1,1,7,9,25,19,19,21,1,1,7,25,27,3,9,3,
     1 27,27,21,14,17,18,13,10,10,11,4,5,16,22,23,24,14,14,4,13,14,15,
     2 18,15,14,14,11,2,8,6,26,23,20,12,5,2,17,26,24/
C
C
      IEXIT=0
      PRINT *,,    ENTER  IDEBUG,
      READ *,IDEBUG
      DO 1 I=1,NPOIN
      R(I)=0.0
      BC(I)=0.0
      NOPP(I)=I
      MDF(I)=0
    1 NCOD(I)=0
      DO 90 I=1,NP1
      NCOD(I)=1
      IF(IEXIT.EQ.1) THEN
      BC(I+NPOIN-NP1)=PEXIT
      NCOD(I+NPOIN-NP1)=1
      END IF
   90 CONTINUE
      DP=VRIN*(RT(1)-RH(1))/(2*NDR)
      DO 91 K=1,2*NDT
      IV=0
      DO 91 KK=(K-1)*(2*NDR+1)+1,K*(2*NDR+1)
      IV=IV+1
      BC(KK)=-LAMBD1/(2*NDT)*(K-1)-DP*(IV-1)
   91 CONTINUE
C
C       NG(I)=GLOBAL NODE NO. CORRESPONDING TO
C             THE ELEMENT NODE NO. OF I
C
C
      PI=ACOS(-1)
      NE=NE1
      NP=0
      DO 10 I=1,NX3
      CALL GENER(I)
      DO 10 J=1,NE1
      DO 30 K=1,9
      IF(I.GT.NX1.AND.I.LT.NX2) THEN
      IF(I.EQ.(NX1+1)) THEN
      NG(K)=NGB1(J,K)+2*NX1*NP1
      NG(K+9)= NGB2(J,K)+(2*NX1+1)*NP1
      NG(K+18)=NG(K+9)+NP2
      ELSE
      NG(K)=NGB2(J,K)+(2*NX1+1)*NP1+(2*(I-NX1-2)+1)*NP2
      NG(K+9)=NG(K)+NP2
      NG(K+18)=NG(K+9)+NP2
      END IF
      ELSE
      IF(I.EQ.NX2) THEN
      NG(K)=NGB2(J,K)+(2*NX1+1)*NP1+(2*(I-NX1-2)+1)*NP2
      NG(K+9)=NG(K)+NP2
      NG(K+18)=NGB1(J,K)+(2*NX1+1)*NP1+(2*(I-NX1-2)+1)*NP2+2*NP2
      ELSE
      IF(I.GT.NX2) THEN
      NG(K)=NGB1(J,K)+(2*NX1+1)*NP1+(2*(NX2-NX1)-1)*NP2+2*(I-NX2-1)*NP1
```

```
            NG(K+9)=NG(K)+NP1
            NG(K+18)=NG(K+9)+NP1
            ELSE
            NG(K)=NGB1(J,K)+2*(I-1)*NP1
            NG(K+9)=NG(K)+NP1
            NG(K+18)=NG(K+9)+NP1
            END IF
            END IF
            END IF
C
C
C           ÖX(I),Y(I),Z(I), I=1,27Ö  ARE THE NODAL
C           COÖRDINATES OF THE HEXAHEDRA
C
C
            Z(K)=RBL(1,NGB2(J,K))
            Z(K+18)=RBL(2,NGB2(J,K))
            Z(K+9)=(Z(K)+Z(K+18))/2.
            Y(K)=TBL(1,NGB2(J,K))
            Y(K+18)=TBL(2,NGB2(J,K))
            Y(K+9)=(Y(K)+Y(K+18))/2.
            X(K)=ZBL(1,NGB1(J,K))
            X(K+18)=ZBL(2,NGB2(J,K))
            X(K+9)=(X(K)+X(K+18))/2.
         30 CONTINUE
            IF(INTG.GT.2.AND.I.GT.NX1.AND.I.LE.NX2) THEN
            R1=(RBL(1,1)+RBL(2,1))/2.
            R2=(RBL(1,2*NDR+1)+RBL(2,2*NDR+1))/2.
            DR=R2-R1
            IF(J.LE.NDR) THEN
            IF(IMESH.EQ.1) THEN
            AZ1=(Z(10)-R1)/DR
            AZ2=(Z(11)-R1)/DR
            AZ3=(Z(12)-R1)/DR
            Y(10)=THETAT(1,(I-NX1))*AZ1+THETAH(1,(I-NX1))*(1.-AZ1)
            Y(11)=THETAT(1,(I-NX1))*AZ2+THETAH(1,(I-NX1))*(1.-AZ2)
            Y(12)=THETAT(1,(I-NX1))*AZ3+THETAH(1,(I-NX1))*(1.-AZ3)
            ELSE
            AZ1=(Z(16)-R1)/DR
            AZ2=(Z(17)-R1)/DR
            AZ3=(Z(18)-R1)/DR
            Y(16)=THETAT(1,(I-NX1))*AZ1+THETAH(1,(I-NX1))*(1.-AZ1)
            Y(17)=THETAT(1,(I-NX1))*AZ2+THETAH(1,(I-NX1))*(1.-AZ2)
            Y(18)=THETAT(1,(I-NX1))*AZ3+THETAH(1,(I-NX1))*(1.-AZ3)
            END IF
            END IF
            IF(J.GT.(NDT-1)*NDR) THEN
            IF(IMESH.EQ.1) THEN
            AZ1=(Z(16)-R1)/DR
            AZ2=(Z(17)-R1)/DR
            AZ3=(Z(18)-R1)/DR
            Y(16)=THETAT(2,(I-NX1))*AZ1+THETAH(2,(I-NX1))*(1.-AZ1)
            Y(17)=THETAT(2,(I-NX1))*AZ2+THETAH(2,(I-NX1))*(1.-AZ2)
            Y(18)=THETAT(2,(I-NX1))*AZ3+THETAH(2,(I-NX1))*(1.-AZ3)
            ELSE
            AZ1=(Z(10)-R1)/DR
            AZ2=(Z(11)-R1)/DR
            AZ3=(Z(12)-R1)/DR
            Y(10)=THETAT(2,(I-NX1))*AZ1+THETAH(2,(I-NX1))*(1.-AZ1)
            Y(11)=THETAT(2,(I-NX1))*AZ2+THETAH(2,(I-NX1))*(1.-AZ2)
            Y(12)=THETAT(2,(I-NX1))*AZ3+THETAH(2,(I-NX1))*(1.-AZ3)
            END IF
            END IF
            END IF
C
C           GLÖBAL CÖDE NUMBERS AND CÖÖRDINATES ÖF THE NÖDES
C
C           ARE PRINTED FÖR EACH PRISM
C
            NP=NP+1
            IF(IDEBUG.GT.0) THEN
            WRITE(6,100) NP
            DO 81 L=1,27
            WRITE(6,110) L,NG(L),Z(L),Y(L),X(L)
         81 CONTINUE
            END IF
            DO 80 L=1,27
         80 NOP(NP,L)=NG(L)
C
C           SUBSYSTEM MATRIX IS CONSTRUCTED
C           IN SUBROUTINE SUSMAT
C
C
```

```
      *     IF(INTG.EQ.1)  CALL SUSMAT(I,J)
            IF(INTG.EQ.2)  CALL SUSMA2(I,J)
            IF(INTG.EQ.3)  CALL SUSMA3(I,J)
            IF(INTG.EQ.4) THEN
            IF(I.LE.NX1.OR.I.GT.NX2) THEN
            CALL  SUSMA2(I,J)
            ELSE
            IF(J.LE.NDR.OR.J.GT.(NDT-1)*NDR) THEN
            CALL SUSMA3(I,J)
            ELSE
            CALL SUSMA2(I,J)
            END IF
            END IF
            END IF

      C
            IF(IEXIT.EQ.0.AND.I.EQ.NX3)  CALL BNDRY


            WRITE(9) ELMP
         10 CONTINUE
          2 FORMAT(5X,,PRISM NO.,,1X,I3,5X,,CODE NUMBERS,,2X,8(I3,1X))

          3 FORMAT(4(4X,,,ODE,,I2,1X,,R=,,F5.2,,T=,,F5.2,,Z=,,F5.2))
            RETURN
        100 FORMAT(/,5X,,,EXAHEDRAL PRISM NO.,,I4,//,
           1 5X,,  I  ,,2X,,NG(I),,5X,,   R   ,,5X,,   TH  ,,5X,,   Z   ,,/)
        110 FORMAT(4X,I3,3X,I4,2X,3F10.4)

            END
```

```
      SUBROUTINE GENER(NX)
C
C       THIS SUBROUTINE CONSTRUCTS THE COORDINATES OF THE NODES
C       FOR A SPECIFIED PRISM ROW
C
      PARAMETER NX1=3,NX2=12,NX3=15
      PARAMETER NPO=45 , NCR=NX3+1 , NE2= 8
      COMMON/INPU1/AMBD1,LAMBD2,OMEGA,CHORD,NBL,PITCH
      COMMON/INPU2/THBLH(2,NCR),ZBLH(NCR),THBLT(2,NCR),ZBLT(NCR),
     1 NGB1(NE2,9),NGB2(NE2,9),RH(NCR),RT(NCR)
      COMMON/PARAM/NDT,NDR,NE1,NP1,NP2,NPOIN
      COMMON/COORD/RBL(2,NPO),TBL(2,NPO),ZBL(2,NPO)
      DT1H=(THBLH(2,NX)-THBLH(1,NX))/NDT
      DT1T=(THBLT(2,NX)-THBLT(1,NX))/NDT
      DT2H=(THBLH(2,NX+1)-THBLH(1,NX+1))/NDT
      DT2T=(THBLT(2,NX+1)-THBLT(1,NX+1))/NDT
      DR1=(RT(NX)-RH(NX))/NDR
      DR2=(RT(NX+1)-RH(NX+1))/NDR
      DZ1=(ZBLT(NX)-ZBLH(NX))/NDR
      DZ2=(ZBLT(NX+1)-ZBLH(NX+1))/NDR
      DO 1 I=1,NDT+1
      DO 1 J=1,NDR+1
      RBL(1,(2*J-1+(2*NDR+1)*(I-1)*2))
     1                 = RH(NX)+DR1*(J-1)
      RBL(2,(2*J-1+(2*NDR+1)*(I-1)*2))
     1                 = RH(NX+1)+DR2*(J-1)
      ZBL(1,(2*J-1+(2*NDR+1)*(I-1)*2))
     1                 = ZBLH(NX)+DZ1*(J-1)
      ZBL(2,(2*J-1+(2*NDR+1)*(I-1)*2))
     1                 = ZBLH(NX+1)+DZ2*(J-1)
      AZ1=(RT(NX)-RBL(1,(2*J-1+(2*NDR+1)*(I-1)*2)))/(RT(NX)-RH(NX))
      AZ2=(RT(NX+1)-RBL(2,(2*J-1+(2*NDR+1)*(I-1)*2)))/
     1     (RT(NX+1)-RH(NX+1))
      TBL(1,(2*J-1+(2*NDR+1)*(I-1)*2))
     1                 = THBLH(1,NX)*AZ1+THBLT(1,NX)*(1.-AZ1)+
     1                  (DT1H*AZ1+DT1T*(1.-AZ1))*(I-1)
      TBL(2,(2*J-1+(2*NDR+1)*(I-1)*2))
     1                 = THBLH(1,NX+1)*AZ2+THBLT(1,NX+1)*(1.-AZ2)+
     1                  (DT2H*AZ2+DT2T*(1.-AZ2))*(I-1)
    1 CONTINUE
      DO 2 J=1,NE1
      RBL(1,NGB2(J,2))=(RBL(1,NGB2(J,1))+RBL(1,NGB2(J,3)))/2.
      RBL(2,NGB2(J,2))=(RBL(2,NGB2(J,1))+RBL(2,NGB2(J,3)))/2.
      TBL(1,NGB2(J,2))=(TBL(1,NGB2(J,1))+TBL(1,NGB2(J,3)))/2.
      TBL(2,NGB2(J,2))=(TBL(2,NGB2(J,1))+TBL(2,NGB2(J,3)))/2.
      ZBL(1,NGB2(J,2))=(ZBL(1,NGB2(J,1))+ZBL(1,NGB2(J,3)))/2.
      ZBL(2,NGB2(J,2))=(ZBL(2,NGB2(J,1))+ZBL(2,NGB2(J,3)))/2.
      RBL(1,NGB2(J,4))=(RBL(1,NGB2(J,1))+RBL(1,NGB2(J,7)))/2.
      RBL(2,NGB2(J,4))=(RBL(2,NGB2(J,1))+RBL(2,NGB2(J,7)))/2.
      TBL(1,NGB2(J,4))=(TBL(1,NGB2(J,1))+TBL(1,NGB2(J,7)))/2.
      TBL(2,NGB2(J,4))=(TBL(2,NGB2(J,1))+TBL(2,NGB2(J,7)))/2.
      ZBL(1,NGB2(J,4))=(ZBL(1,NGB2(J,1))+ZBL(1,NGB2(J,7)))/2.
      ZBL(2,NGB2(J,4))=(ZBL(2,NGB2(J,1))+ZBL(2,NGB2(J,7)))/2.
      RBL(1,NGB2(J,5))=(RBL(1,NGB2(J,1))+RBL(1,NGB2(J,9)))/2.
      RBL(2,NGB2(J,5))=(RBL(2,NGB2(J,1))+RBL(2,NGB2(J,9)))/2.
      TBL(1,NGB2(J,5))=(TBL(1,NGB2(J,1))+TBL(1,NGB2(J,9)))/2.
      TBL(2,NGB2(J,5))=(TBL(2,NGB2(J,1))+TBL(2,NGB2(J,9)))/2.
      ZBL(1,NGB2(J,5))=(ZBL(1,NGB2(J,1))+ZBL(1,NGB2(J,9)))/2.
      ZBL(2,NGB2(J,5))=(ZBL(2,NGB2(J,1))+ZBL(2,NGB2(J,9)))/2.
      RBL(1,NGB2(J,6))=(RBL(1,NGB2(J,3))+RBL(1,NGB2(J,9)))/2.
      RBL(2,NGB2(J,6))=(RBL(2,NGB2(J,3))+RBL(2,NGB2(J,9)))/2.
      TBL(1,NGB2(J,6))=(TBL(1,NGB2(J,3))+TBL(1,NGB2(J,9)))/2.
      TBL(2,NGB2(J,6))=(TBL(2,NGB2(J,3))+TBL(2,NGB2(J,9)))/2.
      ZBL(1,NGB2(J,6))=(ZBL(1,NGB2(J,3))+ZBL(1,NGB2(J,9)))/2.
      ZBL(2,NGB2(J,6))=(ZBL(2,NGB2(J,3))+ZBL(2,NGB2(J,9)))/2.
      RBL(1,NGB2(J,8))=(RBL(1,NGB2(J,7))+RBL(1,NGB2(J,9)))/2.
      RBL(2,NGB2(J,8))=(RBL(2,NGB2(J,7))+RBL(2,NGB2(J,9)))/2.
      TBL(1,NGB2(J,8))=(TBL(1,NGB2(J,7))+TBL(1,NGB2(J,9)))/2.
      TBL(2,NGB2(J,8))=(TBL(2,NGB2(J,7))+TBL(2,NGB2(J,9)))/2.
      ZBL(1,NGB2(J,8))=(ZBL(1,NGB2(J,7))+ZBL(1,NGB2(J,9)))/2.
      ZBL(2,NGB2(J,8))=(ZBL(2,NGB2(J,7))+ZBL(2,NGB2(J,9)))/2.
    2 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SUBMAT(IR,JE)
C
C
C        THIS SUBROUTINE CONSTRUCTS
C        THE RECTANGULAR-PRISM ELEMENT ( SUBSYSTEM )
C        MATRICES
C
C        SUBSYTEMS ARE COMPOSED OF
C        6 TETRAHEDRAL ELEMENTS
C        THE CODE NUMBERS FOR THE TETRAHEDRAL ELEMENTS
C        ARE DETERMINED APRIORI AND
C        ARE GIVEN AS FIXED DATA TO THE PROGRAM (NCODT)
C
      REAL LAMBDA,LAMBD1,LAMBD2,MO
      PARAMETER NX1=3,NX2=12,NX3=15
      PARAMETER NPO=45 , NE2=8 , NCP=NX3+1 , NET=NE2*NX3
      PARAMETER NN=NPO*(2*NX3+1)
      COMMON ELMP(27,27),NG(27),X(27),Y(27),Z(27)
      COMMON/PRINT/IDEBUG,IVEL,ICOMP,NP,IMESH
      COMMON/INPU1/LAMBD1,LAMBD2,OMEGA,CHORD,NBL,PITCH
      COMMON/FRON1/NOP(NET,27),NOPP(NN),MDF(NN),NCOD(NN),BC(NN),R(NN)
      DIMENSION ELMT(10,10),RT(10),PP(27)
      COMMON/PERE/NCODT(6,10),NPER(9),LE,RC
      DIMENSION XX(4),YY(4),ZZ(4),A(4),B(4),C(4)
      COMMON/PARAM/NDT,NDR,NE1,NP1,NP2,NPOIN
      COMMON/DIM/IDIM,INTG
      DATA NCODT/27,27,27,1,1,1,1,7,9,25,19,19,21,1,1,7,25,27,3,9,3,
     1 27,27,21,14,17,18,13,10,10,11,4,5,16,22,23,24,14,14,4,13,14,15,
     2 18,15,14,14,1,2,8,6,26,23,20,12,5,2,17,26,24/
      DO 1 I=1,27
      RP(I)=0.0
      DO 1 J=1,27
    1 ELMP(I,J)=0.0
      DO 5 I=1,10
      DO 5 J=1,10
    5 ELMT(I,J)=0.0
C
C        ELEMENT MATRIX FOR EACH TETRAHEDRAL ELEMENT
C        IS OBTAINED AND IS USED TO
C        CONSTRUCT THE SUBSYSTEM ELEMENT MATRIX
C        BY USING THE CODE NUMBERS ( NCODT )
C
C
C        A(I),B(I),C(I),D(I), I=1,4 ARE
C        THE COEFFICIENTS OF THE NATURAL COORDINATES
C        OF THE TETRAHEDRAL ELEMENTS AND
C        ARE USED TO CONSTRUCT
C        THE ELEMENT MATRICES
C
C        DET=DETERMINANT OF THE JACOBIAN OF TRANSFORMATION
C
      DO 2 LE=1,6
      DO 3 I=1,4
      J=NCODT(LE,I)
      XX(I)=X(J)
      YY(I)=Y(J)
    3 ZZ(I)=Z(J)
      RC=(ZZ(1)+ZZ(2)+ZZ(3)+ZZ(4))/4.
      IF(IDIM.EQ.2) RC=1.0
      DET=0.0
      DO 4 I=1,4
      J=2
      IF(I.GT.1) J=1
      K=3
      IF(I.GT.2) K=2
      LL=4
      IF(I.GT.3) LL=3
      A(I)=(-1)**I*((YY(K)*ZZ(LL)-YY(LL)*ZZ(K))+(YY(LL)*ZZ(J)-
     $YY(J)*ZZ(LL))+(YY(J)*ZZ(K)-YY(K)*ZZ(J)))
      B(I)=(-1)**(I+1)*((XX(K)*ZZ(LL)-XX(LL)*ZZ(K))+(XX(LL)*ZZ(J)-
     $XX(J)*ZZ(LL))+(XX(J)*ZZ(K)-XX(K)*ZZ(J)))
      C(I)=(-1)**I*((XX(K)*YY(LL)-XX(LL)*YY(K))+(XX(LL)*YY(J)-
     $XX(J)*YY(LL))+(XX(J)*YY(K)-XX(K)*YY(J)))
    4 DET=DET+XX(I)*A(I)
      K=0
      L=0
      M=0
      N=0
      DO 10 I=1,10
      IF(I.LE.4) THEN
```

```
      ELMT(I,I)=(RC*A(I)**2+B(I)**2/RC+RC*C(I)**2)*(12./120.)/DET
      ELSE
      K=1
      L=2
      IF(I.GT.5) L=3
      IF(I.EQ.6) K=2
      IF(I.GT.7) L=4
      IF(I.EQ.9) K=3
      IF(I.EQ.10) K=3
      ELMT(I,I)=(((RC*A(K)**2+B(K)**2/RC+RC*C(K)**2)+(RC*A(L)**2+B(L)**2
     1 /RC+RC*C(L)**2))*(32./120.)+(RC*A(K)*A(L)+B(K)*B(L)/RC
     2 +RC*C(K)*C(L))*(32./120.))/DET
      END IF
      DO 10 J=I+1,10
      IF(J.GT.10) GO TO 10
      IF(J.LE.4) THEN
      ELMT(I,J)=(RC*A(I)*A(J)+B(I)*B(J)/RC+RC*C(I)*C(J))*(-4./120.)/DET
      ELSE
      IF(I.LE.4) THEN
      K=0
      IF(J.EQ.5) K=2
      IF(J.EQ.7) K=3
      IF(J.EQ.8) K=4
      IF(I.EQ.2) THEN
      K=0
      IF(J.EQ.5) K=1
      IF(J.EQ.6) K=3
      IF(J.EQ.9) K=4
      END IF
      IF(I.EQ.3) THEN
      K=0
      IF(J.EQ.6) K=2
      IF(J.EQ.7) K=1
      IF(J.EQ.10) K=4
      END IF
      IF(I.EQ.4) THEN
      K=0
      IF(J.GT.7) K=J-7
      END IF
      IF(K.EQ.0) GO TO 20
      ELMT(I,J)=((RC*A(I)**2+B(I)**2/RC+RC*C(I)**2)*(-4./120.)+
     1 (RC*A(I)*A(K)+B(I)*B(K)/RC+RC*C(I)*C(K))*(12./120.))/DET
      GO TO 10
   20 CONTINUE
      K=3
      L=2
      IF(J.EQ.9) K=2
      IF(J.GE.9) L=4
      IF(I.EQ.2) THEN
      K=4
      L=3
      IF(J.EQ.7) K=1
      IF(J.EQ.8) L=1
      END IF
      IF(I.EQ.3) THEN
      K=1
      L=4
      IF(J.EQ.5) L=2
      IF(J.EQ.9) K=2
      END IF
      IF (I.EQ.4) THEN
      K=1
      L=3
      IF(J.EQ.5) L=2
      IF(J.EQ.6) K=2
      END IF
      ELMT(I,J)=((RC*A(I)*A(K)+B(I)*B(K)/RC+RC*C(I)*C(K))+
     1          (RC*A(I)*A(L)+B(I)*B(L)/RC+RC*C(I)*C(L)))*(-4./120.)/DET
      ELSE
      K=0
      IF(J.EQ.6) THEN
      K=2
      L=1
      M=3
      END IF
      IF(J.EQ.7) THEN
      K=1
      L=2
      M=3
      END IF
      IF(J.EQ.8) THEN
      K=1
```

```
      L=2
      M=4
      END IF
      IF(J.EQ.9) THEN
      K=2
      L=1
      M=4
      END IF
      IF(I.EQ.6) THEN
      K=0
      M=4
      IF(J.EQ.7) THEN
      K=3
      L=1
      M=2
      END IF
      IF(J.EQ.9) THEN
      K=2
      L=3
      END IF
      IF(J.EQ.10) THEN
      K=3
      L=2
      END IF
      END IF
      IF(I.EQ.7) THEN
      K=0
      M=4
      IF(J.EQ.8) THEN
      K=1
      L=3
      END IF
      IF(J.EQ.10) THEN
      K=3
      L=1
      END IF
      END IF
      IF(I.EQ.8) THEN
      K=4
      L=1
      M=2
      IF(J.EQ.10) M=3
      END IF
      IF(I.EQ.9) THEN
      K=4
      L=2
      M=3
      END IF
      IF(K.EQ.0) GO TO 30
      ELMT(I,J)=(((RC*A(K)*A(K)+B(K)*B(K)/RC+RC*C(K)*C(K))+
     1 (RC*(A(K)*A(L)+A(K)*A(M))+(B(K)*B(L)+B(K)*B(M))/RC+
     2 RC*(C(K)*C(L)+C(K)*C(M)))*(16./120.)+(RC*A(L)*A(M)+
     3 B(L)*B(M)/RC+RC*C(L)*C(M))*(32./120.))/DET
      GO TO 10
   30 CONTINUE
      K=1
      L=2
      M=3
      N=4
      IF(I.EQ.6) THEN
      K=2
      L=3
      M=1
      END IF
      IF(I.EQ.7) THEN
      L=3
      M=2
      END IF
      ELMT(I,J)=((RC*(A(K)*A(M)+A(K)*A(N)+A(L)*A(M)+A(L)*A(N)))+
     1          (B(K)*B(M)+B(K)*B(N)+B(L)*B(M)+B(L)*B(N))/RC+
     2          RC*(C(K)*C(M)+C(K)*C(N)+C(L)*C(M)+C(L)*C(N)))*          8TH
     3 (16./120.)/DET
      END IF
      END IF
   10 CONTINUE
      DO 40 I=1,10
      DO 40 J=1,I
   40 ELMT(I,J)=ELMT(J,I)
      IF(ICOMP.EQ.1) THEN
      COEFF=DENSTY(IE,IP,IR,JE)
      ELSE
      COEFF=1.0
```

```
      END IF
      DO 50 I=1,10
      DO 50 J=1,10
   50 ELMT(I,J)=COEFF*ELMT(I,J)
      DO 6 I=1,10
    6 RT(I)=0.0
      IF(OMEGA.EQ.0.0) GO TO 100
      IF(IR.GT.NX1.AND.IR.LE.NX2) THEN
      IF(IMESH.EQ.1) THEN
      I1=1
      I2=2
      I4=4
      I6=6
      ELSE
      I1=2
      I2=1
      I4=6
      I6=4
      END IF
      IF(JE.LE.NDR) THEN
      IF(LE.EQ.I1.OR.LE.EQ.I6) CALL ROTATE(RT)
      END IF
      IF(JE.GT.(NDT-1)*NDR) THEN
      IF(LE.EQ.I2.OR.LE.EQ.I4) CALL ROTATE(RT)
      END IF
      END IF
  100 CONTINUE
      IF(IR.LE.NX1.OR.IR.GT.NX2) THEN
      LAMBDA=LAMBD1
      IF(IR.GT.NX2) LAMBDA=LAMBD2
      IF(JE.GT.(NDT-1)*NDR) THEN
      DO 8 I=1,10
      DO 8 J=1,9
      IF(NCODT(LE,I).EQ.NPER(J)) THEN
      DO 9 II=1,10
      RT(II)=RT(II)+LAMBDA*ELMT(II,I)
    9 CONTINUE
      END IF
    8 CONTINUE
      END IF
      END IF
      IF(IR.EQ.NX1+1.AND.JE.GT.(NDT-1)*NDR) THEN
      DO 80 I=1,10
      I1=NCODT(LE,I)
      N1=NPER(1)
      N2=NPER(2)
      N3=NPER(3)
      N7=NPER(7)
      N8=NPER(8)
      N9=NPER(9)
      IF(I1.EQ.N1.OR.I1.EQ.N2.OR.I1.EQ.N3) THEN
      DO 90 II=1,10
   90 RT(II)=RT(II)+LAMBD1*ELMT(II,I)
      END IF
   80 CONTINUE
      END IF
      IF(IR.EQ.NX2.AND.JE.GT.(NDT-1)*NDR) THEN
      DO 81 I=1,10
      I1=NCODT(LE,I)
      IF(I1.EQ.N7.OR.I1.EQ.N8.OR.I1.EQ.N9) THEN
      DO 91 II=1,10
   91 RT(II)=RT(II)+LAMBD2*ELMT(II,I)
      END IF
   81 CONTINUE
      END IF
      DO 2 I=1,10
      IP=NCODT(LE,I)
      RP(IP)=RP(IP)+RT(I)
      DO 2 J=1,10

      JP=NCODT(LE,J)
    2 ELMP(IP,JP)=ELMP(IP,JP)+ELMT(I,J)
      DO 7 I=1,27
      IP=NG(I)
    7 R(IP)=R(IP)+RP(I)
      RETURN
      END
```

```
      SUBROUTINE SUBMA2(IR,JE)
C
C
C     THIS SUBROUTINE CONSTRUCTS
C     THE RECTANGULAR-PRISM ELEMENT ( SUBSYSTEM )
C     MATRICES
C
C     SUBSYTEMS ARE COMPOSED OF
C     6 TETRAHEDRAL ELEMENTS
C     THE CODE NUMBERS FOR THE TETRAHEDRAL ELEMENTS
C     ARE DETERMINED APRIORI AND
C     ARE GIVEN AS FIXED DATA TO THE PROGRAM (NCODT)
C
      REAL LAMBDA,LAMBD1,LAMBD2,MO
      PARAMETER NX1=3,NX2=12,NX3=15
      PARAMETER NPO=45
      PARAMETER NE2=8 ,NCR=NX3+1 ,NET= NE2*NX3
      PARAMETER NN=NPO*(2*NX3+1)
      COMMON ELMP(27,27),NG(27),X(27),Y(27),Z(27)
      COMMON/PRINT/IDEBUG,IVEL,ICOMP,NP,IMESH
      COMMON/INPUT1/LAMBD1,LAMBD2,OMEGA,CHORD,NBL,PITCH
      COMMON/FRON1/NOP(NET,27),NOPP(NN),MDF(NN),NCOD(NN),BC(NN),R(NN)
      DIMENSION ELMT(10,10),RT(10),RP(27)
      COMMON/PERE/NCODT(6,10),NPER(9),LE,RC
      COMMON/PARAM/NDT,NOR,NE1,NP1,NP2,NPOIN
      DIMENSION XX(4),YY(4),ZZ(4),A(4),B(4),C(4)
      DIMENSION RAD(5)
      COMMON/NUMER/CL(4,5),W(5),CN(10,5),CNP(10,5),CNT(10,5)
     1            ,CNZ(10,5)
      COMMON/RHOE/RHOT(10)
      COMMON/DIM/IDIM,INTG
      DIMENSION COE(5),DEN(5),CNP(10,5)
      DATA CL/4*0.25,0.5,4*0.1666666666,0.5,4*0.1666666666,
     1   0.5,4*0.1666666666,0.5/
      DATA NCODT/27,27,27,1,1,1,1,7,9,25,19,19,21,1,1,7,25,27,3,9,3,
     1 27,27,21,14,17,18,13,10,10,11,4,5,16,22,23,24,14,14,4,13,14,15,
     2 18,15,14,14,1,2,8,6,26,23,20,12,5,2,17,26,24/
      W(1)=-4./5.
      W(2)=9./20.
      W(3)=9./20.
      W(4)=9./20.
      W(5)=9./20.
      DO 12 LC=1,5
      DO 13 II=1,4
      CN(II,LC)=CL(II,LC)*(2.*CL(II,LC)-1.)
   13 CONTINUE
      CN(5,LC)=4.*CL(1,LC)*CL(2,LC)
      CN(6,LC)=4.*CL(2,LC)*CL(3,LC)
      CN(7,LC)=4.*CL(1,LC)*CL(3,LC)
      CN(8,LC)=4.*CL(1,LC)*CL(4,LC)
      CN(9,LC)=4.*CL(2,LC)*CL(4,LC)
      CN(10,LC)=4.*CL(3,LC)*CL(4,LC)
   12 CONTINUE
      DO 1 I=1,27
      RP(I)=0.0
      DO 1 J=1,27
    1 ELMP(I,J)=0.0
      DO 5 I=1,10
      DO 5 J=1,10
    5 ELMT(I,J)=0.0
C
C     ELEMENT MATRIX FOR EACH TETRAHEDRAL ELEMENT
C     IS OBTAINED AND IS USED TO
C     CONSTRUCT THE SUBSYSTEM ELEMENT MATRIX
C     BY USING THE CODE NUMBERS ( NCODT )
C
C
C     A(I),B(I),C(I),D(I), I=1,4 ARE
C     THE COEFFICIENTS OF THE NATURAL COORDINATES
C     OF THE TETRAHEDRAL ELEMENTS AND
C     ARE USED TO CONSTRUCT
C     THE ELEMENT MATRICES
C
C     DET=DETERMINANT OF THE JACOBIAN OF TRANSFORMATION
C
      DO 2 LE=1,6
      IF(ICOMP.EQ.1) COEFF=DENSTY(LE,NP,IR,JE)
      IF(ICOMP.EQ.0) THEN
      COEFF=1.0
      DO 22 I=1,10
```

```
C
C
C           DENSITIES AT THE NODES ARE CALCULATED IN
C           SUBROUTINE DENSTY
C
C
      IF(ICOMP.EQ.1) COEFF=DENSTY(LF,NP,IR,JE)
C
C
      IF(ICOMP.EQ.0) THEN
      COEFF=1.0
      DO 22 I=1,10
      RHO(I)=1.0
   22 CONTINUE
      END IF
      DO 3 I=1,10
      J=NCONT(LC,I)
      ZX(I)=X(J)
      T(I)=Y(J)
    3 RX(I)=Z(J)
      RC=(RX(1)+RX(2)+RX(3)+RX(4))/4.
      DO 11 LC=1,5
      RAD(LC)=0.0
      DEN(LC)=0.0
      DO 14 II=1,10
      RAD(LC)=RAD(LC)+RX(II)*CN(II,LC)
      DEN(LC)=DEN(LC)+RHO(II)*CN(II,LC)
   14 CONTINUE
   11 CONTINUE
C
C           THE INVERSE OF THE JACOBIAN MATRIX AND
C           ITS DETERMINANT  (JACOBIAN OF TRANSFORMATION)
C           ARE CALCULATED IN SUBROUTINE  JACOB
C
C
      CALL JACOB
C
C
      DO 21 I=1,10
      DO 21 J=I,10
      ELMT(I,J)=0.0
      DO 21 LC=1,5
      IF(IOTM.EQ.2) RAD(LC)=1.0
      FELM=0.0
      DO 23 K=1,3
      AMATI=AX(K,LC)*CNR(I,LC)+BX(K,LC)*CNT(I,LC)+CX(K,LC)*CNZ(I,LC)
      AMATJ=AX(K,LC)*CNR(J,LC)+BX(K,LC)*CNT(J,LC)+CX(K,LC)*CNZ(J,LC)
      IF(K.EQ.2) AMATI=AMATI/RAD(LC)**2
      FELM=FELM+AMATI*AMATJ
   23 CONTINUE
      ELMT(I,J)=FELM*DEN(LC)*RAD(LC)*W(LC)/6./DETJ(LC)+ELMT(I,J)
   21 CONTINUE
      DO 40 I=1,10
      DO 40 J=1,I
   40 ELMT(I,J)=ELMT(J,I)
      DO 6 I=1,10
      RT(I)=0.0
    6 CONTINUE
      IF(OMEGA.NE.0.) THEN
C
C           OMEGA .NE. 0.0  (ROTATING BLADE ROW)
C
C           VARIATION OF DENSITY CONTRIBUTES TO THE
C           RIGHT HAND SIDE OF EACH ELEMENT
C
C           ADDITIONAL CONTRIBUTIONS TO THE RIGHT HAND SIDES OF
C           THOSE ELEMENTS WHICH HAVE SIDES ON THE
C           BLADE BOUNDARIES ARE CALCULATES IN
C           SUBROUTINE ROTATE
C
C
      DO 24 I=1,10
      RT(I)=0.0
      DO 24 K=1,10
      DO 25 LC=1,5
      DNKDT=AX(2,LC)*CNR(K,LC)+BX(2,LC)*CNT(K,LC)+CX(2,LC)*CNZ(K,LC)
      RT(I)=RT(I)+OMEGA*DNKDT*RHO(K)*CN(I,LC)*W(LC)*RAD(LC)/6.
   25 CONTINUE
   24 CONTINUE
      IF(IR.GT.NX1.AND.IR.LE.NX2) THEN
```

```
      IF(IMESH.EQ.1) THEN
      I1=1
      I2=2
      I4=4
      I6=6
      ELSE
      I1=2
      I2=1
      I4=6
      I6=4
      END IF
      IF(JE.LE.NDR) THEN
      IF(LE.EQ.I1.OR.LE.EQ.I6) CALL ROTATE(RT)
      END IF
      IF(JE.GT.(NDT-1)*NDR) THEN
      IF(LE.EQ.I2.OR.LE.EQ.I4) CALL ROTATE(RT)
      END IF
      END IF
      END IF
      IF(IR.LE.NX1.OR.IR.GT.NX2) THEN
      LAMBDA=LAMBD1
      IF(IR.GT.NX2) LAMBDA=LAMBD2
      IF(JE.GT.(NDT-1)*NDR) THEN
      DO 8 I=1,10
      DO 8 J=1,9
      IF(NCODT(LE,I).EQ.NPER(J)) THEN
      DO 9 II=1,10
      RT(II)=RT(II)+LAMBDA*ELMT(II,I)
    9 CONTINUE
      END IF
    8 CONTINUE
      END IF
      END IF
      IF(IR.EQ.NX1+1.AND.JE.GT.(NDT-1)*NDR) THEN
      DO 80 I=1,10
      II=NCODT(LE,I)
      N1=NPER(1)
      N2=NPER(2)
      N3=NPER(3)
      N7=NPER(7)
      N8=NPER(8)
      N9=NPER(9)
      IF(II.EQ.N1.OR.II.EQ.N2.OR.II.EQ.N3) THEN
      DO 90 II=1,10
   90 RT(II)=RT(II)+LAMBD1*ELMT(II,I)
      END IF
   80 CONTINUE
      END IF
      IF(IR.EQ.NX2.AND.JE.GT.(NDT-1)*NDR) THEN
      DO 81 I=1,10
      II=NCODT(LE,I)
      IF(II.EQ.N7.OR.II.EQ.N8.OR.II.EQ.N9) THEN
      DO 91 II=1,10
   91 RT(II)=RT(II)+LAMBD2*ELMT(II,I)
      END IF
   81 CONTINUE
      END IF
      DO 2 I=1,10
      IP=NCODT(LE,I)
      RP(IP)=RP(IP)+RT(I)
      DO 2 J=1,10

      JP=NCODT(LE,J)
    2 ELMP(IP,JP)=ELMP(IP,JP)+ELMT(I,J)
      DO 7 I=1,27
      IP=NG(I)
    7 R(IP)=R(IP)+RP(I)
      RETURN
      END
```

```
      SUBROUTINE SUBMA3(IR,NE)
C
C
C        THIS SUBROUTINE CONSTRUCTS
C        THE RECTANGULAR-PRISM ELEMENT ( SUBSYSTEM )
C        MATRICES
C
C        SUBSYTEMS ARE COMPOSED OF
C        6 TETRAHEDRAL ELEMENTS
C        THE CODE NUMBERS FOR THE TETRAHEDRAL ELEMENTS
C        ARE DETERMINED APRIORI AND
C        ARE GIVEN AS FIXED DATA TO THE PROGRAM (NCODT)
C
      REAL LAMBDA,LAMBD1,LAMBD2,MO
      PARAMETER NX1=3,NX2=12,NX3=15
      PARAMETER NPO=45
      PARAMETER NE2=8 ,NCR=NX3+1 ,NET= NE2*NX3
      PARAMETER NN=NPO*(2*NX3+1)
      COMMON ELMP(27,27),NG(27),X(27),Y(27),Z(27)
      COMMON/ROT/COEFF
      COMMON/PRINT/IDEBUG,IVEL,ICOMP,NP,IMESH
      COMMON/INPU1/LAMBD1,LAMBD2,OMEGA,CHORD,NBL,PITCH
      COMMON/FRONT/NOP(NET,27),NOPP(NN),MDF(NN),NCOD(NN),BC(NN),R(NN)
      COMMON/PERE/NCODT(6,10),NPER(9),LE,RC
      DIMENSION ELMT(10,10),RT(10),RP(27)
      DIMENSION RAD(5)
      COMMON/RHOE/RHO(10)
      COMMON/NUMER/CL(4,5),W(5),CN(10,5),CNR(10,5),CNT(10,5)
     1            ,CNZ(10,5)
      COMMON/JACO/RX(10),RT(10),ZX(10),AX(3,5),BX(3,5),CX(3,5),DETJ(5)
      COMMON/PARAM/NDT,NDR,NE1,NP1,NP2,NPOIN
      DIMENSION COE(5),DEN(5),CNP(10,5)
      COMMON/DIM/IDIM,INTG
      DATA CL/4*0.25,0.5,4*0.1666666666,0.5,4*0.1666666666,
     1        0.5,4*0.1666666666,0.5/
      DATA NCODT/27,27,27,1,1,1,1,7,9,25,19,19,21,1,1,7,25,27,3,9,3,
     1 27,27,21,14,17,18,13,10,10,11,4,5,16,22,23,24,14,14,4,13,14,15,
     2 18,15,14,14,11,2,8,6,26,23,20,12,5,2,17,26,24/
      w(1)=-4./5.
      w(2)=9./20.
      w(3)=9./20.
      w(4)=9./20.
      w(5)=9./20.
      DO 12 LC=1,5
      DO 13 II=1,4
      CN(II,LC)=CL(II,LC)*(2.*CL(II,LC)-1.)
   13 CONTINUE
      CN(5,LC)=4.*CL(1,LC)*CL(2,LC)
      CN(6,LC)=4.*CL(2,LC)*CL(3,LC)
      CN(7,LC)=4.*CL(1,LC)*CL(3,LC)
      CN(8,LC)=4.*CL(1,LC)*CL(4,LC)
      CN(9,LC)=4.*CL(2,LC)*CL(4,LC)
      CN(10,LC)=4.*CL(3,LC)*CL(4,LC)
   12 CONTINUE
      DO 1 I=1,27
      RP(I)=0.0
      DO 1 J=1,27
    1 ELMP(I,J)=0.0
      DO 5 I=1,10
      DO 5 J=1,10
    5 ELMT(I,J)=0.0
C
C        ELEMENT MATRIX FOR EACH TETRAHEDRAL ELEMENT
C        IS OBTAINED AND IS USED TO
C        CONSTRUCT THE SUBSYSTEM ELEMENT MATRIX
C        BY USING THE CODE NUMBERS ( NCODT )
C
C
C        DET=DETERMINANT OF THE JACOBIAN OF TRANSFORMATION
C
C         THE DERIVATIVES OF THE SHAPE FUNCTIONS  N(I),I=1,10
C         WITH RESPECT TO THE LOCAL COORDINATES ARE
C         CALCULATED IN SUBROUTINE  DERSH
C
C
      CALL DERSH
C
C
      DO 2 LE=1,6
```

```
      RHO(I)=1.0
   22 CONTINUE
      END IF
      DO 3 I=1,4
      J=NCODT(LE,I)
      XX(I)=X(J)
      YY(I)=Y(J)
    3 ZZ(I)=Z(J)
      DET=0.0
      DO 4 I=1,4
      J=2
      IF(I.GT.1) J=1
      K=3
      IF(I.GT.2) K=2
      LL=4
      IF(I.GT.3) LL=3
      A(I)=(-1)**I*((YY(K)*ZZ(LL)-YY(LL)*ZZ(K))+(YY(LL)*ZZ(J)-
     $YY(J)*ZZ(LL))+(YY(J)*ZZ(K)-YY(K)*ZZ(J)))
      B(I)=(-1)**(I+1)*((XX(K)*ZZ(LL)-XX(LL)*ZZ(K))+(XX(LL)*ZZ(J)-
     $XX(J)*ZZ(LL))+(XX(J)*ZZ(K)-XX(K)*ZZ(J)))
      C(I)=(-1)**I*((XX(K)*YY(LL)-XX(LL)*YY(K))+(XX(LL)*YY(J)-
     $XX(J)*YY(LL))+(XX(J)*YY(K)-XX(K)*YY(J)))
    4 DET=DET+XX(I)*A(I)
      DO 11 LC=1,5
      RAD(LC)=0.0
      DEN(LC)=0.0
      DO 14 II=1,10
      DEN(LC)=DEN(LC)+RHO(II)*CN(II,LC)
   14 CONTINUE
      CNP(5,LC)=4.*(CL(1,LC)*B(2)+CL(2,LC)*B(1))
      CNP(6,LC)=4.*(CL(2,LC)*B(3)+CL(3,LC)*B(2))
      CNP(7,LC)=4.*(CL(1,LC)*B(3)+CL(3,LC)*B(1))
      CNP(8,LC)=4.*(CL(1,LC)*B(4)+CL(4,LC)*B(1))
      CNP(9,LC)=4.*(CL(2,LC)*B(4)+CL(4,LC)*B(2))
      CNP(10,LC)=4.*(CL(3,LC)*B(4)+CL(4,LC)*B(3))
      DO 11 IJ=1,4
      RAD(LC)=RAD(LC)+ZZ(IJ)*CL(IJ,LC)
      IF(IDIM.EQ.2) RAD(LC)=1.0
      CNP(IJ,LC)=(4.*CL(IJ,LC)-1.)*B(IJ)
   11 CONTINUE
      DO 10 I=1,10
      K=0
      L=0
      M=0
      N=0
      IF(I.LE.4) THEN
      ELMT(I,I)=0.0
      DO 15 LC=1,5
      COE(LC)=A(I)**2*RAD(LC)+B(I)**2/RAD(LC)+C(I)**2*RAD(LC)
      FELM=(4.*CL(I,LC)-1.)**2*COE(LC)*DEN(LC)
      ELMT(I,I)=FELM*W(LC)/DET/6.+ELMT(I,I)
   15 CONTINUE
      ELSE
      ELMT(I,I)=0.0
      K=2
      IF(I.EQ.5.OR.I.EQ.7.OR.I.EQ.8) K=1
      IF(I.EQ.10) K=3
      L=3
      IF(I.GT.7) L=4
      IF(I.EQ.5) L=2
      DO 16 LC=1,5
      COE(LC)=16.*((CL(K,LC)*A(L)+CL(L,LC)*A(K))**2*RAD(LC)
     1        +(CL(K,LC)*B(L)+CL(L,LC)*B(K))**2/RAD(LC)
     2        +(CL(K,LC)*C(L)+CL(L,LC)*C(K))**2*RAD(LC))
      ELMT(I,I)=COE(LC)*DEN(LC)*W(LC)/DET/6.+ELMT(I,I)
   16 CONTINUE
      END IF
      DO 10 J=I+1,10
      IF(J.GT.10) GO TO 10
      IF(J.LE.4) THEN
      ELMT(I,J)=0.0
      DO 17 LC=1,5
      COE(LC)=A(I)*A(J)*RAD(LC)+B(I)*B(J)/RAD(LC)+C(I)*C(J)*RAD(LC)
      FELM=(4.*CL(I,LC)-1)*(4.*CL(J,LC)-1.)*COE(LC)*DEN(LC)
      ELMT(I,J)=FELM*W(LC)/DET/6.+ELMT(I,J)
   17 CONTINUE
      ELSE
      M=2
      IF(J.EQ.5.OR.J.EQ.7.OR.J.EQ.8) M=1
      IF(J.EQ.10) M=3
      N=3
      IF(J.GT.7) N=4
```

```
      IF(J.EQ.5) N=2
      END IF
      IF(I.LE.4.AND.J.GE.5) THEN
      ELMT(I,J)=0.0
      DO 18 LC=1,5
      COE(LC)=4.*((CL(M,LC)*A(N)+CL(N,LC)*A(M))*A(I)*RAD(LC)
     1          +(CL(N,LC)*B(N)+CL(N,LC)*B(M))*B(I)/RAD(LC)
     2          +(CL(M,LC)*C(N)+CL(N,LC)*C(M))*C(I)*RAD(LC))
      FELM=(4.*CL(I,LC)-1.)*COE(LC)
      ELMT(I,J)=FELM*DEN(LC)*W(LC)/DET/6.+ELMT(I,J)
   18 CONTINUE
      END IF
      IF(I.GT.4.AND.J.GE.6) THEN
      ELMT(I,J)=0.0
      DO 19 LC=1,5
      COE1=(CL(K,LC)*A(L)+CL(L,LC)*A(K))*
     1     (CL(M,LC)*A(N)+CL(N,LC)*A(M))*RAD(LC)
      COE2=(CL(K,LC)*B(L)+CL(L,LC)*B(K))*
     1     (CL(M,LC)*B(N)+CL(N,LC)*B(M))/RAD(LC)
      COE3=(CL(K,LC)*C(L)+CL(L,LC)*C(K))*
     1     (CL(M,LC)*C(N)+CL(N,LC)*C(M))*RAD(LC)
      COE(LC)=(COE1+COE2+COE3)*16.
      ELMT(I,J)=COE(LC)*DEN(LC)*W(LC)/DET/6.+ELMT(I,J)
   19 CONTINUE
      END IF
   10 CONTINUE
      DO 40 I=1,10
      DO 40 J=1,I
   40 ELMT(I,J)=ELMT(J,I)
      DO 6 I=1,10
      RT(I)=0.0
      DO 24 K=1,10
      DO 21 LC=1,5
      RT(I)=-OMEGA*RHO(K)*W(LC)*RAD(LC)*CNP(K,LC)/6.*CN(I,LC)+RT(I)
   21 CONTINUE
   24 CONTINUE
    6 CONTINUE
      IF(OMEGA.GT.0.) THEN
      IF(IR.GT.NX1.AND.IR.LE.NX2) THEN
      IF(IMESH.EQ.1) THEN
      I1=1
      I2=2
      I4=4
      I6=6
      ELSE
      I1=2
      I2=1
      I4=6
      I6=4
      END IF
      IF(JE.LE.NDR) THEN
      IF(LE.EQ.I1.OR.LE.EQ.I6) CALL ROTATE
      END IF
      IF(JE.GT.(NDT-1)*NDR) THEN
      IF(LE.EQ.I2.OR.LE.EQ.I4) CALL ROTATE
      END IF
      END IF
      END IF
      IF(IR.LE.NX1.OR.IR.GT.NX2) THEN
      LAMBDA=LAMBD1
      IF(IR.GT.NX2) LAMBDA=LAMBD2
      IF(JE.GT.(NDT-1)*NDR) THEN
      DO 8 I=1,10
      DO 8 J=1,9
      IF(NODT(LE,I).EQ.NPER(J)) THEN
      DO 9 II=1,10
      RT(II)=RT(II)+LAMBDA*ELMT(II,I)
    9 CONTINUE
      END IF
    8 CONTINUE
      END IF
      END IF
      IF(IR.EQ.NX1+1.AND.JE.GT.(NDT-1)*NDR) THEN
      DO 80 I=1,10
      I1=NODT(LE,I)
      N1=NPER(1)
      N2=NPER(2)
      N3=NPER(3)
      N7=NPER(7)
      N8=NPER(8)
      N9=NPER(9)
      IF(I1.EQ.N1.OR.I1.EQ.N2.OR.I1.EQ.N3) THEN
```

```
246             DO 90 II=1,10
247          90 RT(II)=RT(II)+LAMBD1*ELMT(II,I)
248             END IF
249          80 CONTINUE
250             END IF
251             IF(IR.EQ.NX2.AND.JE.GT.(NDT-1)*NDR) THEN
252             DO 81 I=1,10
253             I1=NCODT(LE,I)
254             IF(I1.EQ.N7.OR.I1.EQ.N8.OR.I1.EQ.N9) THEN
255             DO 91 II=1,10
256          91 RT(II)=RT(II)+LAMBD2*ELMT(II,I)
257             END IF
258          81 CONTINUE
259             END IF
260             DO 2 I=1,10
261             IP=NCODT(LE,I)
262             RP(IP)=RP(IP)+RT(I)
263             DO 2 J=1,10
264
265             JP=NCODT(LE,J)
266           2 ELMP(IP,JP)=ELMP(IP,JP)+LLMT(I,J)
267             DO 7 I=1,27
268             IP=NG(I)
269           7 R(IP)=R(IP)+RP(I)
270             RETURN
271             END
```

```
      SUBROUTINE DERSH
C
C       THIS SUBROUTINE CALCULATES THE DERIVATIVES OF
C       THE SHAPE FUNCTIONS .. N(I),I=1,10  ..
C       WITH RESPECT TO THE LOCAL COORDINATES
C
      COMMON/NUMER/CL(4,5),W(5),CN(10,5),CNR(10,5),CNT(10,5),CNZ(10,5)
      DO 10 I=1,10
      DO 10 J=1,5
      CNR(I,J)=0.0
      CNT(I,J)=0.0
      CNZ(I,J)=0.0
   10 CONTINUE
      DO 20 LC=1,5
      CNR(1,LC)=4.*CL(1,LC)-1.
      CNR(4,LC)=-4.*CL(4,LC)+1.
      CNR(5,LC)=4.*CL(2,LC)
      CNR(7,LC)=4.*CL(3,LC)
      CNR(8,LC)=4.*(CL(4,LC)-CL(1,LC))
      CNR(9,LC)=-4.*CL(2,LC)
      CNR(10,LC)=-4.*CL(3,LC)
      CNT(2,LC)=4.*CL(2,LC)-1.
      CNT(4,LC)=-4.*CL(4,LC)+1.
      CNT(5,LC)=4.*CL(1,LC)
      CNT(6,LC)=4.*CL(3,LC)
      CNT(8,LC)=-4*CL(1,LC)
      CNT(9,LC)=4.*(CL(4,LC)-CL(2,LC))
      CNT(10,LC)=-4.*CL(3,LC)
      CNZ(3,LC)=4.*CL(3,LC)-1.
      CNZ(4,LC)=-4.*CL(4,LC)+1.
      CNZ(6,LC)=4.*CL(2,LC)
      CNZ(7,LC)=4.*CL(1,LC)
      CNZ(8,LC)=-4.*CL(1,LC)
      CNZ(9,LC)=-4.*CL(2,LC)
      CNZ(10,LC)=4.*(CL(4,LC)-CL(3,LC))
   20 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE JACOB
C
C       THIS SUBROUTINE CALCULATES THE INVERSE OF
C       THE JACOBIAN MATRIX AND ITS
C       DETERMINANT (JACOBIAN OF TRANSFORMATION)
C
      REAL J11,J12,J13,J21,J22,J23,J31,J32,J33
      COMMON/JACO/R(10),T(10),Z(10),A(3,5),B(3,5),C(3,5),DETJ(5)
      COMMON/NUMER/GL(4,5),W(5),CNR(10,5),CNT(10,5),CNZ(10,5)
      DO 10 LC=1,5
      J11=0.0
      J12=0.0
      J13=0.0
      J21=0.0
      J22=0.0
      J23=0.0
      J31=0.0
      J32=0.0
      J33=0.0
      DO 20 I=1,10
      J11=J11+CNR(I,LC)*R(I)
      J21=J21+CNT(I,LC)*R(I)
      J31=J31+CNZ(I,LC)*R(I)
      J12=J12+CNR(I,LC)*T(I)
      J22=J22+CNT(I,LC)*T(I)
      J32=J32+CNZ(I,LC)*T(I)
      J13=J13+CNR(I,LC)*Z(I)
      J23=J23+CNT(I,LC)*Z(I)
      J33=J33+CNZ(I,LC)*Z(I)
   20 CONTINUE
      A(1,LC)=J22*J33-J23*J32
      A(2,LC)=J23*J31-J21*J33
      A(3,LC)=J21*J32-J22*J31
      B(1,LC)=J13*J32-J12*J33
      B(2,LC)=J11*J33-J13*J31
      B(3,LC)=J12*J31-J11*J32
      C(1,LC)=J12*J23-J13*J22
      C(2,LC)=J13*J21-J11*J23
      C(3,LC)=J11*J22-J12*J21
      DETJ(LC)=J11*A(1,LC)+J12*A(2,LC)+J13*A(3,LC)
   10 CONTINUE
      RETURN
      END
```

```
      FUNCTION DENSITY(LE,NP,IX,JX)
      REAL MACHT,MO,LAMBD,LAMBD1,LAMBDA,LAMBD2
C
C
C     THIS FUNCTION CALCULATES THE DENSITIES
C     AT THE NODES AND THE CENTROID OF THE
C     TETRAHEDRAL ELEMENTS
C
C
C     FIELD VARIABLE(POTENTIAL) MODEL USED IS
C     PHI=N(I)*Q(I)
C     N(I)=SHAPE FUNCTION    ...I=1,10
C     N(1)=L(1)*(2*L(1)-1)
C     N(2)=L(2)*(2*L(2)-1)
C     N(3)=L(3)*(2*L(3)-1)
C     N(4)=L(4)*(2*L(4)-1)
C     N(5)=4*L(1)*L(2)
C     N(6)=4*L(2)*L(3)
C     N(7)=4*L(1)*L(3)
C     N(8)=4*L(1)*L(4)
C     N(9)=4*L(2)*L(4)
C     N(10)=4*L(3)*L(4)
C     L(I)=NATURAL COORDINATE    ...I=1,4
C     Q(I)=NODAL VALUE OF THE POTENTIAL
C     U(I)=DPHIDX=D(N(I))/DX * Q(I)    ...I=1,10
C     V(I)=DPHIDY=D(N(I))/DY * Q(I)    ...I=1,10
C     W(I)=DPHIDZ=D(N(I))/DZ * Q(I)    ...I=1,10
C
      PARAMETER NX1=3,NX2=12,NX3=15
      PARAMETER NPO=45
      PARAMETER N=NPO*(2*NX3+1)
      COMMON ELMP(27,27),NG(27),X(27),Y(27),Z(27)
      DIMENSION D(4),UT(10),VT(10),WT(10)
      DIMENSION A(4),B(4),C(4),RP(27)
      COMMON/PERE/UCORDI(9,10),NPER(9),LLE,RC
      COMMON/CHPRSS/POT(N),MO,LAMA,LAMBDA
      COMMON/RHO/RHOG(6)
      COMMON/INPU1/LAMBD1,LAMBD2,OMEGA,CHORD,NBL,PITCH
      DIMENSION VELOC(10),MACHT(10)
      COMMON/RHOF/DENST(10)
      COMMON/DIM/IDIM,INTG
      COMMON/PARAM/NDT,NDR,NE1,NP1,NP2,NPOIN
      COMMON/VELJAC/ZZ(10),YY(10),XX(10),AA(3),BB(3),CC(3),DETJ,RT(10)
      COMMON/NUMER/CL(4),CN(10),CN1(10),CN2(10),CN3(10)
      COMMON/PROV2/AR,DT,DZ,UTR,PHI
      DO 1 I=1,27
      RP(I)=0.0
      IF(NG(I).NE.0) THEN
      RP(I)=POT(NG(I))
      IF(IX.LE.NX1.OR.IX.GT.NX2) THEN
      LAMBD=LAMBD1
      IF(IX.GT.NX2) LAMBD=LAMBDA
      IF(JX.GT.(NDT-1)*NDR) THEN
      DO 12 J=1,9
      IF(I.EQ.NPER(J)) THEN
      RP(I)=RP(I)-LAMBD
      END IF
   12 CONTINUE
      END IF
      END IF
      END IF
    1 CONTINUE
      N1=NPER(1)
      N2=NPER(2)
      N3=NPER(3)
      N7=NPER(7)
      N8=NPER(8)
      N9=NPER(9)
      IF(IX.EQ.NX1+1.AND.JX.GT.(NDT-1)*NDR) THEN
      RP(N1)=RP(N1)-LAMBD1
      RP(N2)=RP(N2)-LAMBD1
      RP(N3)=RP(N3)-LAMBD1
      END IF
      IF(IX.EQ.NX2.AND.JX.GT.(NDT-1)*NDR) THEN
      RP(N7)=RP(N7)-LAMBDA
      RP(N8)=RP(N8)-LAMBDA
      RP(N9)=RP(N9)-LAMBDA
      END IF
      DO 2 I=1,10
      J=NCORT(LE,I)
```

```
      RT(I)=RP(J)
    2 CONTINUE
      DO 3 I=1,10
      J=NCONT(LE,I)
      XX(I)=X(J)
      YY(I)=Y(J)
    3 ZZ(I)=Z(J)
      RC=(ZZ(1)+ZZ(2)+ZZ(3)+ZZ(4))/4.
      IF(IDIM.EQ.2) RC=1.0
      DET=0.0
      DO 4 I=1,4
      CL(I)=0.0
      J=2
      IF(I.GT.1) J=1
      K=3
      IF(I.GT.2) K=2
      LL=4
      IF(I.GT.3) LL=3
      A(I)=(-1)**I*((YY(K)*ZZ(LL)-YY(LL)*ZZ(K))+(YY(LL)*ZZ(J)-
     $YY(J)*ZZ(LL))+(YY(J)*ZZ(K)-YY(K)*ZZ(J)))
      B(I)=(-1)**(I+1)*((XX(K)*ZZ(LL)-XX(LL)*ZZ(K))+(XX(LL)*ZZ(J)-
     $XX(J)*ZZ(LL))+(XX(J)*ZZ(K)-XX(K)*ZZ(J)))
      C(I)=(-1)**I*((XX(K)*YY(LL)-XX(LL)*YY(K))+(XX(LL)*YY(J)-
     $XX(J)*YY(LL))+(XX(J)*YY(K)-XX(K)*YY(J)))
      D(I)=(-1)**(I+1)*((XX(K)*YY(LL)-XX(LL)*YY(K))*ZZ(J)
     1 +(XX(LL)*YY(J)-XX(J)*YY(LL))*ZZ(K)
     2 +(XX(J)*YY(K)-XX(K)*YY(J))*ZZ(LL))
      DET=DET-XX(I)*A(I)
    4 CONTINUE
      DO 5 I=1,4
      A(I)=A(I)/DET
      B(I)=B(I)/DET
      C(I)=C(I)/DET
      D(I)=D(I)/DET
    5 CONTINUE
      DO 6 I=1,10
      IF(I.EQ.1)CL(1)=1.0
      IF(I.EQ.2) THEN
      CL(1)=0.0
      CL(2)=1.0
      END IF
      IF(I.EQ.3) THEN
      CL(2)=0.0
      CL(3)=1.0
      END IF
      IF(I.EQ.4) THEN
      CL(3)=0.0
      CL(4)=1.0
      END IF
      IF(I.EQ.5) THEN
      CL(4)=0.0
      CL(1)=0.5
      CL(2)=0.5
      END IF
      IF(I.EQ.6) THEN
      CL(1)=0.0
      CL(3)=0.5
      END IF
      IF(I.EQ.7) THEN
      CL(2)=0.0
      CL(1)=0.5
      END IF
      IF(I.EQ.8) THEN
      CL(3)=0.0
      CL(4)=0.5
      END IF
      IF(I.EQ.9) THEN
      CL(1)=0.0
      CL(2)=0.5
      END IF
      IF(I.EQ.10) THEN
      CL(2)=0.0
      CL(3)=0.5
      END IF
      IF(INTG.GT.2) THEN
      CALL ISOVEL
      UT(I)=DR
      VT(I)=DT/ZZ(I)
      IF(IDIM.EQ.2)VT(I)=VT(I)*ZZ(I)
      WT(I)=DZ
      ELSE
      UT(I)=VEL(CL,A,RT)
```

```
            VT(I)=VEL(CL,R,RT)/ZZ(I)
            IF(IDIM.EQ.2) VT(I)=VT(I)*ZZ(I)
            WT(I)=VEL(CL,C,RT)
            END IF
          6 CONTINUE
            DO 7 I=1,10
            VELOC(I)=SQRT(UT(I)**2+VT(I)**2+WT(I)**2)
            TTO=1.+(GAMA-1.)*MO**2*((1.-VELOC(I)**2)/2.
           1          +OMEGA*(VT(I)*RC-LAMBDA1/PITCH))
            MACHT(I)=MO*VELOC(I)/SQRT(TTO)
          7 DENST(I)=(TTO)**(1./(GAMA-1))
            AMACHT=0.0
            DENSTY=0.0
            UG=0.0
            VG=0.0
            WG=0.0
            VELC=0.0
            DO 8 I=1,4
            AMACHT=AMACHT+(-1./8.)*MACHT(I)
            UG=UG+(-1./8.)*UT(I)
            VG=VG+(-1./8.)*VT(I)
            WG=WG+(-1./8.)*WT(I)
            VELC=VELC+(-1./8.)*VELOC(I)
          8 DENSTY=DENSTY+(-1./8.)*DENST(I)
            DO 9 I=5,10
            AMACHT=AMACHT+(1./4.)*MACHT(I)
            UG=UG+(1./4.)*UT(I)
            VG=VG+(1./4.)*VT(I)
            WG=WG+(1./4.)*WT(I)
            VELC=VELC+(1./4.)*VELOC(I)
          9 DENSTY=DENSTY+(1./4.)*DENST(I)
            RHOG(LE)=DENSTY
            XC=(XX(1)+XX(2)+XX(3)+XX(4))/4.
            YC=(YY(1)+YY(2)+YY(3)+YY(4))/4.
            ZC=(ZZ(1)+ZZ(2)+ZZ(3)+ZZ(4))/4.
            IF(LE.EQ.1) THEN
            PRINT*,,PRISM NO.,,NP
            WRITE(6,10)
            END IF
            WRITE(6,11) LE,XC,YC,ZC,UG,VG,WG,VELC,AMACHT,DENSTY
            RETURN
         10 FORMAT(2X,,IT,,5X,,XC,,9X,,YC,,9X,,ZC,,9X,,UG,,9X,,VG,,
           1 9X,,WG,,8X,,VEL,,9X,,MACH NO.,,
           1 4X,,DENSITY,,/)
         11 FORMAT(2X,I2,3X,9E11.4)
            END
```

```
      FUNCTION VEL(CL,A,Q)
C
C       THIS FUNCTION CALCULATES VELOCITIES AT A GIVEN LOCATION
C       WITHIN A SUBPARAMETRIC TETRAHEDRAL ELEMENT
C
      DIMENSION CL(4),A(4),Q(10)
      VEL=(4*CL(1)-1)*A(1)*Q(1)+(4*CL(2)-1)*A(2)*Q(2)
     1    +(4*CL(3)-1)*A(3)*Q(3)+(4*CL(4)-1)*A(4)*Q(4)
     2    +4*(CL(1)*A(2)+CL(2)*A(1))*Q(5)
     3    +4*(CL(2)*A(3)+CL(3)*A(2))*Q(6)
     4    +4*(CL(1)*A(3)+CL(3)*A(1))*Q(7)
     5    +4*(CL(1)*A(4)+CL(4)*A(1))*Q(8)
     6    +4*(CL(2)*A(4)+CL(4)*A(2))*Q(9)
     7    +4*(CL(3)*A(4)+CL(4)*A(3))*Q(10)
      RETURN
      END
```

```
      SUBROUTINE ISOVEL
C
CCC     THIS SUBROUTINE CALCULATES VELOCITIES
CCC     AT A GIVEN LOCATION WITHIN AN
CCC     ISOPARAMETRIC TETRAHEDRAL ELEMENT
C
      REAL J11,J12,J13,J21,J22,J23,J31,J32,J33
      COMMON/VELJAC/R(10),T(10),Z(10),A(3),B(3),C(3),DETJ,RT(10)
      COMMON/VNUMER/CL(4),CN1(10),CN2(10),CN3(10)
      COMMON/PROV2/DR,DT,DZ,DTH,PHI
      COMMON/DIM/IDIM,INTG
      DO 10 I=1,10
      CN1(I)=0.0
      CN2(I)=0.0
      CN3(I)=0.0
   10 CONTINUE
      CN1(1)=4.*CL(1)-1.
      CN1(4)=4.*CL(4)+1.
      CN1(5)=4.*CL(2)
      CN1(7)=4.*CL(3)
      CN1(8)=4.*(CL(4)-CL(1))
      CN1(9)=-4.*CL(2)
      CN1(10)=-4.*CL(3)
      CN2(2)=4.*CL(2)-1.
      CN2(4)=4.*CL(4)+1.
      CN2(5)=4.*CL(1)
      CN2(6)=4.*CL(3)
      CN2(8)=-4*CL(1)
      CN2(9)=4.*(CL(4)-CL(2))
      CN2(10)=-4.*CL(3)
      CN3(3)=4.*CL(3)-1.
      CN3(4)=4.*CL(4)+1.
      CN3(6)=4.*CL(2)
      CN3(7)=4.*CL(1)
      CN3(8)=-4.*CL(1)
      CN3(9)=-4.*CL(2)
      CN3(10)=4.*(CL(4)-CL(3))
   20 CONTINUE
      J11=0.0
      J12=0.0
      J13=0.0
      J21=0.0
      J22=0.0
      J23=0.0
      J31=0.0
      J32=0.0
      J33=0.0
      DO 30 I=1,10
      J11=J11+CN1(I)*R(I)
      J21=J21+CN2(I)*R(I)
      J31=J31+CN3(I)*R(I)
      J12=J12+CN1(I)*T(I)
      J22=J22+CN2(I)*T(I)
      J32=J32+CN3(I)*T(I)
      J13=J13+CN1(I)*Z(I)
      J23=J23+CN2(I)*Z(I)
      J33=J33+CN3(I)*Z(I)
   30 CONTINUE
      A(1)=J22*J33-J23*J32
      A(2)=J23*J31-J21*J33
      A(3)=J21*J32-J22*J31
      B(1)=J13*J32-J12*J33
      B(2)=J11*J33-J13*J31
      B(3)=J12*J31-J11*J32
      C(1)=J12*J23-J13*J22
      C(2)=J13*J21-J11*J23
      C(3)=J11*J22-J12*J21
      DETJ=J11*A(1)+J12*A(2)+J13*A(3)
      DETJ=1./DETJ
      DR=0.0
      DZ=0.0
      DT=0.0
      DO 40 I=1,10
      DR=DR+RT(I)/DETJ*(A(1)*CN1(I)+B(1)*CN2(I)+C(1)*CN3(I))
      DT=DT+RT(I)/DETJ*(A(2)*CN1(I)+B(2)*CN2(I)+C(2)*CN3(I))
      DZ=DZ+RT(I)/DETJ*(A(3)*CN1(I)+B(3)*CN2(I)+C(3)*CN3(I))
   40 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE ROTATE(RT)
C
C       THIS SUBROUTINE IS USED FOR ROTATING BLADE ROWS TO
C       CALCULATE THE ADDITIONAL RIGHT HAND SIDE VECTOR FOR
C       ELEMENTS WHICH HAVE SIDES ON ONE OF THE BLADE SURFACES
C
      COMMON/INPU1/XAMBD1,XAMBD2,OMEGA,CHORD,UNL,PITCH
      DIMENSION RT(1,0)
      COMMON/PERE/NODT(6,10),UPER(9),LE,RC
      COMMON ELMP(27,27),NG(27),X(27),Y(27),Z(27)
      COMMON/ROT/COEFF
      DIMENSION XV(27),YV(27),ZV(27)
      IF(LE.EQ.1) THEN
      I1=1
      I2=3
      I3=21
      END IF
      IF(LE.EQ.6) THEN
      I1=1
      I2=21
      I3=19
      END IF
      IF(LE.EQ.2) THEN
      I1=7
      I2=27
      I3=9
      END IF
      IF(LE.EQ.4) THEN
      I1=7
      I2=25
      I3=27
      END IF
      DET=(X(I1)*Z(I2)-Z(I1)*X(I2))+(X(I2)*Z(I3)-X(I3)*Z(I2))+
     1    (X(I3)*Z(I1)-X(I1)*Z(I3))
      DET=ABS(DET)
      THETA=(Y(I1)+Y(I2)+Y(I3))/3.
      DO 10 I=1,27
      XV(I)=Z(I)*COS(Y(I))
      YV(I)=Z(I)*SIN(Y(I))
      ZV(I)=X(I)
   10 CONTINUE
      DX1=XV(I2)-XV(I1)
      DX2=XV(I3)-XV(I1)
      DY1=YV(I2)-YV(I1)
      DY2=YV(I3)-YV(I1)
      DZ1=ZV(I2)-ZV(I1)
      DZ2=ZV(I3)-ZV(I1)
      VEC1=SQRT(DX1**2+DY1**2+DZ1**2)
      VEC2=SQRT(DX2**2+DY2**2+DZ2**2)
      VEC=VEC1*VEC2
      A=DY1*DZ2-DY2*DZ1
      B=DX2*DZ1-DX1*DZ2
      C=DY2*DX1-DX2*DY1
      VEC=SQRT(A**2+B**2+C**2)
      A=A/VEC
      B=B/VEC
      DNT=-SIN(THETA)*A+COS(THETA)*B
      DFIDT=-RC*OMEGA*(DNT/ABS(DNT))*COEFF
      IF(LE.EQ.1.OR.LE.EQ.4) THEN
      RT(6)=RT(6)+DFIDT*DET/6.
      RT(9)=RT(9)+DFIDT*DET/6.
      RT(10)=RT(10)+DFIDT*DET/6.
      END IF
      IF(LE.EQ.2.OR.LE.EQ.6) THEN
      RT(5)=RT(5)+DFIDT*DET/6.
      RT(8)=RT(8)+DFIDT*DET/6.
      RT(9)=RT(9)+DFIDT*DET/6.
      END IF
      RETURN
      END
```

```
      SUBROUTINE BNDRY
C
C
C         THIS SUBROUTINE MODIFIES THE RIGHT-HAND SIDE OF
C         THE GOVERNING EQUATIONS FOR
C         NEUMANN TYPE BOUNDARY CONDITIONS AT THE EXIT PLANE
C
C
      PARAMETER NX1=3,NX2=12,NX3=15,NE2=8
      PARAMETER NPO=45
      PARAMETER NE1=NPO*(2*NX3+1)
      PARAMETER NET=NE2*NX3
      COMMON/PARAM/NDT,NDR,NE1,NP1,NP2,NPOIN
      COMMON ELMP(27,27),NG(27),X(27),Y(27),Z(27)
      COMMON/FRON1/NOP(NET,27),NOPP(N),MOF(N),NCOD(N),RC(N),R(N)
      COMMON/BNDP/DFIDX,BETAIN,BETOUT
      COMMON/DIM/IDIM,INTG
      DIMENSION RP(27),XNOD(4),YNOD(4),ZNOD(4),NCODT(6,10),RT(10)
      DATA NCODT/27,27,27,27,1,1,1,1,7,9,25,19,19,21,1,1,7,25,27,3,9,3,
     1 27,27,21,14,17,18,13,10,10,11,4,5,16,22,23,24,14,14,4,13,14,15,
     2 18,15,14,14,1,2,8,6,26,23,20,12,5,2,17,26,24/
      DO 20 J=1,27
   20 RP(J)=0.0
      DO 2 L1=5,6
      DO 3 L2=1,4
      L3=NCODT(L1,L2)
      XNOD(L2)=X(L3)
      YNOD(L2)=Y(L3)
    3 ZNOD(L2)=Z(L3)
      DET=(YNOD(3)*ZNOD(4)-YNOD(4)*ZNOD(3))+
     1(YNOD(4)*ZNOD(2)-YNOD(2)*ZNOD(4))+
     2(YNOD(2)*ZNOD(3)-YNOD(3)*ZNOD(2))
      IF(IDIM.EQ.2) THEN
      ZNOD(2)=1.0
      ZNOD(3)=1.0
      ZNOD(4)=1.0
      END IF
      RT(1)=0.0
      RT(2)=-(2*ZNOD(2)-ZNOD(3)-ZNOD(4))*DFIDX*DET/120.
      RT(3)=-(2*ZNOD(3)-ZNOD(2)-ZNOD(4))*DFIDX*DET/120.
      RT(4)=-(2*ZNOD(4)-ZNOD(2)-ZNOD(3))*DFIDX*DET/120.
      RT(5)=0.0
      RT(6)=-(8*ZNOD(2)+8*ZNOD(3)+4*ZNOD(4))*DFIDX*DET/120.
      RT(7)=0.0
      RT(8)=0.0
      RT(9)=-(8*ZNOD(2)+4*ZNOD(3)+8*ZNOD(4))*DFIDX*DET/120.
      RT(10)=-(4*ZNOD(2)+8*ZNOD(3)+8*ZNOD(4))*DFIDX*DET/120.
      DO 2 L2=1,10
      IP=NCODT(L1,L2)
    2 RP(IP)=RP(IP)+RT(L2)
      DO 30 M=1,27
      IP=NG(M)
   30 R(IP)=R(IP)+RP(M)
      RETURN
      END
```

```
      SUBROUTINE FRONT
      PARAMETER NX1=3,NX2=12,NX3=15
      PARAMETER NPO=45
      PARAMETER NP=NPO*(2*NX3+1)
      PARAMETER NE2=8
      PARAMETER NE=NE2*NX3 ,NN=NP,NBN=27,NCOL=27
      PARAMETER NMAX=3*NPO ,NCRIT=3*NPO-20,NWGA=1,NTRA=1
C
C             FRONTAL ELIMINATION SUBROUTINE USING FULL PIVOTING
C
      COMMON/FRON1/NOP(NE,NBN),NOPP(NH),MDF(NH),NCOR(NP),BC(NP),R1(NP)
      COMMON/FRON2/AA(NCN,NCN),EQ(NMAX,NMAX)
      COMMON/PRINT/IDEBUG,IVEL,ICOMP,INPP,IMESH
      DIMENSION LDEST(NCN),KDEST(NCN),NK(NCN),LHED(NMAX),KHED(NMAX),
     1KPIV(NMAX),LPIV(NMAX),JMOD(NMAX),QQ(NMAX),PVKOL(NMAX)
      COMMON/PARAM/NDT,NDR,NE1,NP1,NP2,NPOIN
C
C
C
C
C             PREFRONT
C
C
      NELL=0
      REWIND 9
C
      IF(IDEBUG.EQ.1000) WRITE(6,400)
      IF(NTRA.EQ.0)GO TO 14
C
C             FIND LAST APPEARANCE OF EACH NODE
C
      NLAST=0
      NET1=NE1*NX3
      DO 12 L=1,NPOIN
      DO 8 N=1,NET1
      DO 4 I=1,NBN
      IF(NOP(N,I).NE.L)GO TO 4
      NLAST1=N
      IF(NLAST.NE.NLAST1)GO TO 3
      NERROR=1
       WRITE(6,416) NERROR,N
      STOP
    3 CONTINUE
      NLAST=N
      L1=L
    4 CONTINUE
    8 CONTINUE
       IF(IDEBUG.EQ.1000) WRITE(6,404) L,NLAST
      NOP(NLAST,L1)=-NOP(NLAST,L1)
      NLAST=0
   12 CONTINUE
       IF(IDEBUG.EQ.1000)  WRITE(6,408)
       IF(IDEBUG.EQ.1000)  WRITE(6,412)((N,(NOP(N,L),L=1,NBN),N=1,NET1)
C
C             ASSEMBLY
C
   14 CONTINUE
      LCOL=0
      KROW=0
      DO 16 I=1,NMAX
      DO 16 J=1,NMAX
      EQ(J,I)=0
   16 CONTINUE
   18 NELL=NELL+1
      READ(9) AA
      IF(NELL
      KC=0
      IF(NWGA.EQ.0)GO TO 21
      DO 20 I=1,NBN
      NK(I)=NOP(N,I)
   20 CONTINUE
      GO TO 23
   21 CONTINUE
      DO 22 J=1,NBN
      NN=NOP(N,J)
      M=IABS(NN)
      K=NOPP(M)
```

```
      IDF=MDF(M)
      DO 22 L=1,IDF
      KC=KC+1
      II=K+L-1
      IF(NN.LT.0)II=-II
      NK(KC)=II
   22 CONTINUE
   23 CONTINUE
C
CC              SET UP HEADING VECTORS
C
      DO 52 LK=1,NCN
      NODE=NK(LK)
      IF(LCOL.EQ.0)GO TO 28
      DO 24 L=1,LCOL
      LL=L
      IF(IABS(NODE).EQ.IABS(LHED(L)))GO TO 32
   24 CONTINUE
   28 LCOL=LCOL+1
      LDEST(LK)=LCOL
      LHED(LCOL)=NODE
      GO TO 36
   32 LDEST(LK)=LL
      LHED(LL)=NODE
   36 IF(KROW.EQ.0)GO TO 44
      DO 42 K=1,KROW
      KK=K
      IF(IABS(NODE).EQ.IABS(KHED(K)))GO TO 48
   42 CONTINUE
   44 KROW=KROW+1
      KDEST(LK)=KROW
      KHED(KROW)=NODE
      GO TO 52
   48 KDEST(LK)=KK
      KHED(KK)=NODE
   52 CONTINUE
      IF(IDEBUG.EQ.1000)  WRITE(6,420)KROW,LCOL
      IF(IDEBUG.EQ.1000) WRITE(6,424)
      IF(IDEBUG.EQ.1000)WRITE(6,428)(KHED(K),LHED(K),K=1,NMAX)
      IF(IDEBUG.EQ.1000)  WRITE(6,432)
      IF(IDEBUG.EQ.1000)WRITE(6,428)(KDEST(K),LDEST(K),K=1,NCN)

      IF(KROW.LE.NMAX.AND.LCOL.LE.NMAX)GO TO 54
      NERROR=2
      WRITE(6,417)NERROR
      STOP
   54 CONTINUE
      DO 56 L=1,NCN
      LL=LDEST(L)
      DO 56 K=1,NCN
      KK=KDEST(K)
      EQ(KK,LL)=EQ(KK,LL)+AA(K,L)
   56 CONTINUE
      IF(IDEBUG.EQ.1000)  WRITE(6,436)NELL
      IF(KROW.LT.NCRIT.AND.NELL.LT.NET1)GO TO 18
C
CC              FIND OUT WHICH MATRIX ELEMENTS ARE FULLY SUMMED
C
   60 LC=0
      DO 64 L=1,LCOL
      IF(LHED(L).GE.0)GO TO 64
      LC=LC+1
      LPIV(LC)=L
   64 CONTINUE
      IR=0
      KR=0
      DO 68 K=1,KROW
      KT=KHED(K)
      IF(KT.GE.0)GO TO 68
      KR=KR+1
      KPIV(KR)=K
      KRO=IABS(KT)
      IF(NCOD(KRO).NE.1)GO TO 68
      IR=IR+1
      JMOD(IR)=K
      NCOD(KRO)=2
      RI(KRO)=BC(KRO)
   68 CONTINUE
C
CC              MODIFY EQUATIONS WITH APPLIED BOUNDARY CONDITIONS
C
                IF(IDEBUG.EQ.1000)  WRITE(6,448)LC,KR
```

```
      IF(IDEBUG.EQ.1000)WRITE(6,428)(LPIV(K),KPIV(K),K=1,NMAX)
      IF(IR.EQ.0)GO TO 71
                        IF(IDEBUG.EQ.1000)   WRITE(6,450)
      DO 70 IRR=1,IR
      K=JMOD(IRR)
                        IF(IDEBUG.EQ.1000)   WRITE(6,428)K
      KH=IABS(KHED(K))
      DO 69 L=1,LCOL
      EQ(K,L)=0.0
      LH=IABS(LHED(L))
      IF(LH.EQ.KH)EQ(K,L)=1.0
   69 CONTINUE
   70 CONTINUE
   71 CONTINUE
      IF(KR.GT.0.AND.LC.GT.0)GO TO 72
      NERROR=3
      WRITE(6,418)NERROR
      STOP
   72 CONTINUE
C
C              SEARCH FOR ABSOLUTE PIVOT
C
      LPIVCO=LPIV(LC)
      KPIVRO=KPIV(KR)
      PIVOT=EQ(KPIVRO,LPIVCO)
C                NORMALIZE PIVOTAL ROW
C
      KRO=IABS(KHED(KPIVRO))
      LCO=IABS(LHED(LPIVCO))
       IF(IDEBUG.EQ.3) WRITE(6,452)KRO,LCO,PIVOT
      IF(ABS(PIVOT).LT.1E-08)WRITE(6,476)
      DO 80 L=1,LCOL
      QQ(L)=EQ(KPIVRO,L)/PIVOT
   80 CONTINUE
      RHS=R1(KRO)/PIVOT
      R1(KRO)=RHS
      PVKOL(KPIVRO)=PIVOT
C
C              ELIMINATE THEN DELETE PIVOTAL ROW AND COLUMN
C
      IF(KPIVRO.EQ.1)GO TO 104
      KPIVR=KPIVRO-1
      DO 100 K=1,KPIVR
      KRW=IABS(KHED(K))
      FAC=EQ(K,LPIVCO)
      PVKOL(K)=FAC
      IF(LPIVCO.EQ.1.OR.FAC.EQ.0.0)GO TO 88
      LPIVC=LPIVCO-1
      DO 84 L=1,LPIVC
      EQ(K,L)=EQ(K,L)-FAC*QQ(L)
   84 CONTINUE
   88 IF(LPIVCO.EQ.LCOL)GO TO 96
      LPIVC=LPIVCO+1
      DO 92 L=LPIVC,LCOL
      EQ(K,L-1)=EQ(K,L)-FAC*QQ(L)
   92 CONTINUE
   96 R1(KRW)=R1(KRW)-FAC*RHS
  100 CONTINUE
  104 IF(KPIVRO.EQ.NROW)GO TO 128
      KPIVR=KPIVRO+1
      DO 124 K=KPIVR,NROW
      KRW=IABS(KHED(K))
      FAC=EQ(K,LPIVCO)
      PVKOL(K)=FAC
      IF(LPIVCO.EQ.1)GO TO 112
      LPIVC=LPIVCO-1
      DO 108 L=1,LPIVC
      EQ(K-1,L)=EQ(K,L)-FAC*QQ(L)
  108 CONTINUE
  112 IF(LPIVCO.EQ.LCOL)GO TO 120
      LPIVC=LPIVCO+1
      DO 116 L=LPIVC,LCOL
      EQ(K-1,L-1)=EQ(K,L)-FAC*QQ(L)
  116 CONTINUE
  120 R1(KRW)=R1(KRW)-FAC*RHS
  124 CONTINUE
  128 CONTINUE
C
C              WRITE PIVOTAL EQUATION ON DISC
C
      WRITE(10)KRO,LCOL,LPIVCO,(LHED(L),QQ(L),L=1,LCOL),KROW,PIVOT,
     1KPIVRO,(PVKOL(K),KHED(K),K=1,KROW)
```

```
      DO 120 K=1,KROW
      EQ(K,LCOL)=0.0
  129 CONTINUE
      DO 130 L=1,LCOL
      EQ(KROW,L)=0.0
  130 CONTINUE
          IF(IDEBUG.EQ.1000)  WRITE(6,436)NELL
C
C           REARRANGE HEADING VECTORS
C
      LCOL=LCOL-1
      IF(LPIVCO.EQ.LCOL+1)GO TO 136
      DO 132 L=LPIVCO,LCOL
      LHED(L)=LHED(L+1)
  132 CONTINUE
  136 KROW=KROW-1
      IF(KPIVRO.EQ.KROW+1)GO TO 144
      DO 140 K=KPIVRO,KROW
      KHED(K)=KHED(K+1)
  140 CONTINUE
  144 CONTINUE
                  IF(IDEBUG.EQ.1000)  WRITE(6,420)KROW,LCOL
                  IF(IDEBUG.EQ.1000) WRITE(6,424)
      IF(IDEBUG.EQ.1000)WRITE(6,428)(KHED(K),LHED(K),K=1,NMAX)
C
C           DETERMINE WHETHER TO ASSEMBLE, ELIMINATE OR BACKSUBSTITUTE
C
      IF(KROW.GT.NCRIT)GO TO 60
      IF(NELL.LT.NET1)GO TO 18
      IF(KROW.GT.1)GO TO 60
      LCO=IABS(LHED(1))
      KPIVRO=1
      PIVOT=EQ(1,1)
      KRO=IABS(KHED(1))
      LPIVCO=1
      DO 1 I=1,0
      WRITE(6,452)LCO,KRO,PIVOT

      IF(ABS(PIVOT).LT.1E-08)WRITE(6,476)
      R1(KRO)=R1(KRO)/PIVOT
      WRITE(10)KRO,LCOL,LPIVCO,LHED(1),DO(1),KROW,PIVOT,KPIVRO,
     1PIVKOL(1),KHED(1)
      CALL BACSUB
  400 FORMAT(1X,'NODE',1X,'NLAST')
  404 FORMAT(1X,2I5)
  408 FORMAT(//,'NODAL NUMBERING',/)
  412 FORMAT(9I5)
  416 FORMAT(/,' ERROR=',I5,//,I5,'THE ELEMENT HAS MORE THAN ONE NODE,
     1,WITH THE SAME NODAL NUMBER',/)
  417 FORMAT(//,' ERROR=',I5,//,' THE DIFFERENCE NMAX-NCRIT IS NOT,
     1,SUFFICIENTLY LARGE TO PERMIT THE ASSEMBLY OF THE NEXT ELEMENT--,
     2,EITHER INCREASE NMAX OR LOWER NCRIT.',/)
  418 FORMAT(//,' ERROR=',I5,//,' THERE ARE NO MORE ROWS FULLY SUMMED,
     1,'THIS MAY BE DUE TO -- (1) INCORRECT CODING OF NOP OR NK ARRAYS,
     2,  (2) INCORRECT VALUE OF NCRIT. INCREASE NCRIT TO PERMIT WHOLE,
     3,TO BE ASSEMBLED',/)
  420 FORMAT(//,'KROW=',I5,'LCOL=',I5,/)
  424 FORMAT(//,'KHED','LHED',/)
  428 FORMAT(2I6)
  432 FORMAT(' KDEST',',LDEST',/)
  436 FORMAT(//,' EQ MATRIX FOR ELEMENT NO.',I3,/)
  440 FORMAT(20F5.2)
  448 FORMAT(1X,'LC=',I3,//,'KR=',I3,'  LPIV','  KPIV',/)
  452 FORMAT(1X,'PIVOTAL ROW=',I4,'PIVOTAL COLUMN=',I4,'PIVOT=',E20.10)
  456 FORMAT(//,' JMOD',/)
  460 FORMAT(//,' RIGHT HAND VECTOR',/)
  464 FORMAT(I5,E20.10)
  468 FORMAT(//,' PIVOTAL ROW',/)
  476 FORMAT(' WARNING-MATRIX SINGULAR OR ILL CONDITIONED')
  480 FORMAT(' FAC=',E20.10)
      RETURN
      END
```

```
      SUBROUTINE BACSUB
      PARAMETER NX1=3,NX2=12,NX3=15,NE2=8
      PARAMETER NPO=45
      PARAMETER NP=NPO*(2*NX3+1)
      PARAMETER NE=NE2*NX3,NME=NP,NBN=27,NCN=27
      PARAMETER NMAX=3*NPO*NCRIT=3*NPO-3,MWGA=1,NTRA=1,IPRINT=0
C
C             BACK SUBSTITUTION FOR FULL PIVOTING
C
      COMMON/FROM1/NOP(NE,NBN),NOPP(NN),MDF(NN),NCOD(NP),BC(NP),P1(NP)
      COMMON/FROM2/AA(NCN,NCN),EQ(NMAX,NMAX)
      DIMENSION LDEST(NCN),KDEST(NCN),SK(NCN),LHED(NMAX),KHED(NMAX),
     1KPIV(NMAX),LPIV(NMAX),JMOD(NMAX),QQ(NMAX),PVKOL(NMAX),
     2SK(NMAX*NMAX)
      COMMON/PARAM/NDT,NOR,NE1,NP1,NP2,NPOIN
      EQUIVALENCE (SK(1),EQ(1,1))
C
C             BACK SUBSTITUTION
C
      DO 4 I=1,NPOIN
    4 SK(I)=BC(I)
      DO 32 IV=1,NPOIN
      BACKSPACE 10
      READ(10)KRO,LCOL,LPIVCO,(LHED(L),QQ(L),L=1,LCOL),KROW,PIVOT,
     1KPIVRO,(PVKOL(K),KHED(K),K=1,KROW)
      BACKSPACE 10

      IF(IPRINT.EQ.1)WRITE(6,408)(LHED(L),L=1,LCOL)
      IF(IPRINT.EQ.1)WRITE(6,412)(QQ(L),L=1,LCOL)
      IF(NCOD(KRO).GT.0)GO TO 24
      SUM=0.0
      QQ(LPIVCO)=0.0
      DO 16 L=1,LCOL
      SUM=SUM-QQ(L)*SK(IABS(LHED(L)))
   16 CONTINUE
      SK(KRO)=R1(KRO)+SUM
      GO TO 32
   24 CONTINUE
      NCOD(KRO)=1
   32 CONTINUE
      DO 36 L=1,NPOIN
      J=NOPP(L)-1
      IDF=MDF(L)
      R1(L)=SK(J+1)
   36 CONTINUE
      WRITE(8) (I,R1(I),I=1,NPOIN)
      WRITE(8) NOP
  404 FORMAT(1X,,TAPE CONTENTS,)
  408 FORMAT(10I5)
  412 FORMAT(5E20.10)
  416 FORMAT(/,,RESULTS,/)
  420 FORMAT(I5,6E10.10)

      RETURN
      END
```

```
      SUBROUTINE OUTPUT
C
C
C        THIS SUBROUTINE PRINTS THE NODAL VALUES
C        OF THE POTENTIAL
C
C
      PARAMETER NX1=3,NX2=12,NX3=15
      PARAMETER NPO=45
      PARAMETER NE=NPO*(2*NX3+1)
      PARAMETER NE2=8
      PARAMETER NET=NE2*NX3
      REAL*8
      COMMON/FRON1/NOP(NET,27),NOPP(N),MDF(N),NCOD(N),BC(N),R(N)
      COMMON/CMPRSS/POT(N),MO,GAMA
      COMMON/ITER/ITER,CREL
      COMMON/PARAM/NDT,NDR,NE1,NP1,NP2,NPOIN
      DIMENSION RELRR(N)
      WRITE(6,1)
      WRITE(6,2)
      WRITE(6,3) (I,R(I),I=1,NPOIN)
      IF(ITER.GT.0) THEN
      PRINT *,    RESULTS OF THE PERIVIOUS ITERATION WERE ,
      WRITE(6,2)
      WRITE(6,3) (I,POT(I),I=1,NPOIN)

      NPTS=0
      RELMIN=0.0
      RELMAX=0.0
      DO 4 I=1,NPOIN
      RELRR(I)=(POT(I)-R(I))/POT(I)*100.
      RELRR(I)=ABS(RELRR(I))
      IF(RELRR(I).EQ.0.0) GO TO 4
      RELMAX=AMAX0(RELMAX,RELRR(I))
      RELMIN=AMIN0(RELMIN,RELRR(I))
      IF(RELRR(I).LE.CREL) NPTS=NPTS+1
    4 CONTINUE
      PRINT *,  ,    SUMMARY  OF THE RESULTS  ,
      WRITE(6,5)
      WRITE(6,3) (I,RELRR(I),I=1,NPOIN)
      PRINT *,, RELATIVE ERROR TOLERANCE (PERCENT) =,,CREL
      PRINT *,,    NUMBER OF NODES CONVERGED , ,,NPTS
      PRINT *,,    MAXIMUM  RELATIVE CHANGE (PERCENT ) =,,RELMAX
      PRINT *,,    MINIMUM  RELATIVE CHANGE (PERCENT ) =,,RELMIN
      END IF
      RETURN
    1 FORMAT(1H1,20X,, SOLUTION IS OBTAINED ,,//)
    2 FORMAT(8(2X,,NODE,,3X,,VALUE,))
    3 FORMAT(8(2X,I3,1X,E9.3))
    5 FORMAT(8(2X,,NODE,,3X,,RELRR,))
      RETURN
      END
```

```
      SUBROUTINE VELOCY
C
C
C         THIS SUBROUTINE IS USED TO CALCULATE VELOCITIES
C
C
      PARAMETER NX1=3,NX2=12,NX3=15
      PARAMETER NPO=45
      PARAMETER NE2=8 ,NCR=NX3+1
      PARAMETER NET=NE2*NX3
      PARAMETER NE=NPO*(2*NX3+1)
      REAL MO,LAMBDA,LAMBD1,LAMBD2
      COMMON/INPU2/THBLH(2,NCR),ZBLH(NCR),THBLT(2,NCR),ZBLT(NCR),
     1 NGB1(NE2,9),NGB2(NE2,9),RH(NCR),RT(NCR)
      COMMON/FROM1/NOP(NET,27),NOPP(N),MDF(N),NCOD(N),BC(N),R(N)
      COMMON/VELO/XVP(100),NVELP,NVR,NVT
      COMMON/PROV1/XV,YV,ZV,IV,I,J,IP,NELMP,IELMP,ITET
      COMMON/PARAM/NDT,NDR,NE1,NP1,NP2,NPOIN
      COMMON/DIM/IDIM,INTG
      DO 6 IV=1,NVELP
      XV=XVP(IV)
      IP=0
      DO 1 I=1,NX3+1
      IF(XV.LT.ZBLH(I)) THEN
      IP=I-1
      GO TO 2
      END IF
    1 CONTINUE
    2 CONTINUE
      IF(XV.EQ.ZBLH(NX3+1)) IP=NX3
      IF(IP.EQ.0.OR.XV.GT.ZBLH(NX3+1)) THEN
      WRITE(6,100)
      STOP
      END IF
      DXP=(ZBLH(IP+1)-ZBLH(IP))
      DXV=XV-ZBLH(IP)
      AZ=DXV/DXP
      RVT=RT(IP)*(1.-AZ)+RT(IP+1)*AZ
      RVH=RH(IP)*(1.-AZ)+RH(IP+1)*AZ
      DRV=(RVT-RVH)/NVR
      TBL1=THBLH(1,IP)*(1.-AZ)+THBLH(1,IP+1)*AZ
      TBL2=THBLH(2,IP)*(1.-AZ)+THBLH(2,IP+1)*AZ
      DTV=(TBL2-TBL1)/NVT
      DRP=(RVT-RVH)/NDR
      DTP=(TBL2-TBL1)/NDT
      DO 3 I=1,NVT+1
      YV=TBL1+(I-1)*DTV
      IF(INTG.GT.2.AND.IP.GT.NX1.AND.IP.LE.NX2) THEN
      IF(YV.LT.(TBL1+DTP)) YV=TBL1
      IF(YV.GT.(TBL2-DTP)) YV=TBL2
      END IF
      DO 3 J=1,NVR+1
      ZV=RVH+(J-1)*DRV
      KR=0
      KT=0
      DO 4 K=1,NDR
      ZPR=RVH+K*DRP
      IF(ZV.LE.ZPR) THEN
      KR=K
      GO TO 7
      END IF
    4 CONTINUE
    7 CONTINUE
      DO 5 K=1,NDT
      YPT=TBL1+K*DTP
      IF(YV.LT.YPT) THEN
      KT=K
      GO TO 8
      END IF
    5 CONTINUE
      KT=NDT
    8 CONTINUE
      NELMP=(KT-1)*NDR+KR
      IELMP=(IP-1)*NE1+NELMP
      CALL PROP
      CALL PRINTV
    3 CONTINUE
    6 CONTINUE
      RETURN
  100 FORMAT(///,20X,'PLANE X-LOCATION OUT OF SOLUTION DOMAIN,,
     1  //,20X,,CHECK XVP ARRAY,)
      END
```

```
      SUBROUTINE SUBVEL
C
C
C         THIS SUBROUTINE IS USED TO CALCULATE
C              SURFACE VELOCITIES
C
      PARAMETER NX1=3,NX2=12,NX3=15
      PARAMETER NPO=45
      PARAMETER NE2=8 ,NCR=NX3+1
      PARAMETER NET=NE2*NX3
      PARAMETER NE=NPO*(2*NX3+1)
      REAL MO,LAMBDA,LAMBD1,LAMBD2
      COMMON/INPU2/THBLH(2,NCR),ZBLH(NCR),THBLT(2,NCR),ZBLT(NCR),
     1 NGB1(NE2,9),NGB2(NE2,9),RH(NCR),RT(NCR)
      COMMON/FROM1/NOP(NET,27),NOPP(N),MDF(N),NCOD(N),RC(N),R(N)
      COMMON/VELO/XVP(100),NVELP,NVP,NVT
      COMMON/PROV1/XV,YV,ZV,IV,I,J,IP,NELMP,IELMP,ITET
      COMMON/PROV2/DR,DT,DZ,DTH,PHI
      COMMON/PARAM/NDT,NDR,NE1,NP1,NP2,NPOIN
      COMMON/SURF/RVV(51,3,2),TV(51,3,2),AV(51,3,2),VEL(51,3,2)
     1     ,TRV(51,3,2),VELR(51,3,2),RV(51,3),TB(51,2),XB(51),
     2     XMACH(51,3,2),CP(51,3,2)
      COMMON/PRINT/IDEBUG,IVEL,ICOMP,NPP,IMESH
      COMMON/DIM/IDIM,INTG
      CHORD=ZBLH(NX2+1)
      IF(IDIM.EQ.2) CHORD=1.0
      DX=CHORD/50.
      SIGMA=37.5/180.*ACOS(-1)
      CS=COS(SIGMA)
      DO 6 IV=1,51
      XX=DX*(IV-1)
      XV=XX
      IF(IDIM.EQ.2) XV=XX*CS
      IP=0
      DO 1 I=1,NX2+2
      IF(XV.LT.ZBLH(I)) THEN
      IP=I-1
      GO TO 2
      END IF
    1 CONTINUE
    2 CONTINUE
      IF(XV.EQ.CHORD) IP=NX2
      DXP=(ZBLH(IP+1)-ZBLH(IP))
      DXV=XV-ZBLH(IP)
      AZ=DXV/DXP
      RVT=RT(IP)*(1.-AZ)+RT(IP+1)*AZ
      RVH=RH(IP)*(1.-AZ)+RH(IP+1)*AZ
      RVM=(RVH+RVT)/2
      TBL1=THBLH(1,IP)*(1.-AZ)+THBLH(1,IP+1)*AZ
      TBL2=THBLH(2,IP)*(1.-AZ)+THBLH(2,IP+1)*AZ
      DRP=(RVT-RVH)/NDR
      DTP=(TBL2-TBL1)/NDT
      RV(IV,1)=RVH
      RV(IV,2)=RVM
      RV(IV,3)=RVT
      XB(IV)=XV
      IF(IDIM.EQ.2) XB(IV)=XX
      TB(IV,1)=TBL1
      TB(IV,2)=TBL2
      DO 10 IP=1,3
      DO 10 IS=1,2
      YV=TB(IV,IS)
      ZV=RV(IV,IP)
      KR=0
      KT=0
      DO 4 K=1,NDR
      ZPR=RVH+K*DRP
      IF(ZV.LE.ZPR) THEN
      KR=K
      GO TO 7
      END IF
    4 CONTINUE
    7 CONTINUE
      IF(IS.EQ.1) KT=1
      IF(IS.EQ.2) KT=NDT
      NELMP=(KT-1)*NDR+KR
      IELMP=(IP-1)*NE1+NELMP
      CALL PROP
      RVV(IV,IP,IS)=DR
      TV(IV,IP,IS)=DT
```

```
      TRV(IV,IR,IS)=DTR
      AV(IV,IR,IS)=DZ
      VEL(IV,IR,IS)=SQRT(DR**2+DT**2+DZ**2)
      VELR(IV,IR,IS)=SQRT(DR**2+DTR**2+DZ**2)
      IF(IDIM.EQ.2) CP(IV,IR,IS)=1-VEL(IV,IR,IS)**2
   10 CONTINUE
    6 CONTINUE
      DO 11 IR=1,3
      IF(IR.EQ.1) WRITE(6,101)
      IF(IR.EQ.2) WRITE(6,102)
      IF(IR.EQ.3) WRITE(6,103)
      IF(IDIM.EQ.2) THEN
      WRITE(6,105)
      ELSE
      WRITE(6,100)
      END IF
      DO 12 IV=1,5
      IF(IDIM.EQ.2) THEN
      WRITE(6,104) IV,XB(IV),RV(IV,IR),TB(IV,1),VEL(IV,IR,1),
     1 CP(IV,IR,1),TB(IV,2),VEL(IV,IR,2),CP(IV,IR,2)
      ELSE
      WRITE(6,104) IV,XB(IV),RV(IV,IR),TB(IV,1),VEL(IV,IR,1),
     1 VELR(IV,IR,1),TB(IV,2),VEL(IV,IR,2),VELR(IV,IR,2)
      END IF
   12 CONTINUE
   11 CONTINUE
      RETURN
  100 FORMAT(///,2X,,J,,9X,,X,,9X,,R,,7X,,TH1,,7X,,VEL,,6X,,RVEL,
     1   ,9X,,TH2,,7X,,VEL,,6X,,RVEL,,//)
  105 FORMAT(///,2X,,J,,9X,,X,,9X,,R,,7X,,TH1,,7X,,VEL,,6X,,CP1 ,
     1   ,9X,,TH2,,7X,,VEL,,6X,,CP2 ,,//)
  101 FORMAT(1H1,//,20X,,SURFACE VELOCITIES AT THE HUB,,//)
  102 FORMAT(1H1,//,20X,,SURFACE VELOCITIES AT THE MEAN SECTION,,//)
  103 FORMAT(1H1,//,20X,,SURFACE VELOCITIES AT THE TIP,,//)
  104 FORMAT(1X,I2,4X,5E10.4,2X,3E10.4)
      END
```

```
      SUBROUTINE PRINTV
C
CC        THIS SUBROUTINE PRINTS VELOCITIES
C
      PARAMETER NE2=8,NX1=3,NX2=12,NX3=15,NCR=NX3+1
      PARAMETER NPO=45
      PARAMETER N=NPO*(2*NX3+1)
      REAL MO,MACHT,LAMBDA,LAMBD1,LAMBD2
      COMMON/CMPRSS/POT(N),MO,GAMA,LAMBDA
      COMMON/INPU1/LAMBD1,LAMBD2,OMEGA,NRL,CHORD,PITCH
      COMMON/VELO/XVP(100),NVELP,NVP,NVT
      COMMON/PROV1/XV,YV,ZV,IV,I,J,IP,IELMP,IELMP,ITET
      COMMON/PROV2/DR,DT,DZ,DTR,PHI
      COMMON/INPU2/THBLH(2,NCR),ZBLH(NCR),THBLT(2,NCR),ZBLT(NCR),
     1 NGB1(NE2,9),NGB2(NE2,9),RH(NCR),RT(NCR)
      COMMON/PARAM/NDT,NDR,NE1,NP1,NP2,NPOIN
      IF(I.EQ.1.AND.J.EQ.1) THEN
      WRITE(6,10) IV,XV,IP
      IF(XV.EQ.ZBLH(1)) WRITE(6,100)
      IF(XV.LT.0) WRITE(6,20)
      IF(XV.EQ.0) WRITE(6,30)
      IF(XV.GT.0.AND.XV.LT.ZBLH(NX2+1)) WRITE(6,40)
      IF(XV.EQ.ZBLH(NX2+1)) WRITE(6,50)
      IF(XV.GT.ZBLH(NX2+1)) WRITE(6,60)
      IF(XV.EQ.ZBLH(NX3+1)) WRITE(6,70)
      END IF
      IF(J.EQ.1) WRITE(6,80) YV
      TVEL=SQRT(DR**2+DT**2+DZ**2)
      TVELR=SQRT(DR**2+DTR**2+DZ**2)
      TTO=1+(GAMA-1)*MO**2*((1-TVEL**2)/2.
     1     +OMEGA*(DT*ZV-LAMBD1/PITCH))
      MACHT=MO*TVEL/SQRT(TTO)
      DENSI=(TTO)**(1/(GAMA-1))
      WRITE(6,90) J,ZV,DR,DT,DTR,DZ,TVEL,TVELR,PHI,DENST,MACHT,IELMP
     1  ,ITET
      RETURN
   10 FORMAT(///,20X,,CALCULATED VELOCITIES AT RADIAL,,
     1  , CROSS-SECTIONAL PLANE NO.,,I4,//,20X,
     2  ,PLANE AXIAL COORDINATE = ,,F10.5,//,20X,
     3  ,VELOCITIY PLANE IS WITHIN THE,,I3,,TH HEXAHEDRAL,,
     4  , PRISM ROW,,///)
   20 FORMAT(//,20X,,VELOCITY PLANE IS IN THE UPSTREAM SIDE OF,,
     1  , THE BLADES,,/)
   30 FORMAT(//,20X,,VELOCITY PLANE IS AT THE INLET OF,,
     1  , THE BLADE ROW,,/)
   40 FORMAT(//,20X,,VELOCITY PLANE IS WITHIN THE BLADE PASSAGE,,/)
   50 FORMAT(//,20X,,VELOCITY PLANE IS AT THE EXIT OF,,
     1  , THE BLADE PASSAGE,,/)
   60 FORMAT(//,20X,,VELOCITY PLANE IS ON THE DOWNSTREAM SIDE,,
     1  , OF THE BLADES,,/)
   70 FORMAT(//,20X,,VELOCITY PLANE IS THE EXIT PLANE OF THE,,
     1  , SOLUTION DOMAIN,,/)
   80 FORMAT(//,20X,,THETA=,,F10.5,//,2X,,J,,9X,,R,,9X,,DR,,
     1 9X,,DT,,8X,,DTR,,9X,,DZ,,9X,,VEL,,8X,,REL,,8X,,PHI,,6X,,DENSTY,,
     2    6X,,MACH,,4X
     2 ,,PR NO.,,2X,,ITET,,/,2X,,-,,9X,,-,,3(9X,,--,),3(8X,,--,),
     3 8X,,---,,6X,,------,,9X,,-----,,4X,,------,,2X,,----,,/)
   90 FORMAT(1X,I2,4X,E10.4,9E11.4,2(2X,I4))
  100 FORMAT(//,20X,,VELOCITY PLANE IS THE INLET PLANE,,
     1  ,OF THE SOLUTION DOMAIN,,/)
      END
```

```
      SUBROUTINE PROP
C
C
C     THIS SUBROUTINE CALCULATES THE VELOCITIES
C     AT A GIVEN LOCATION (XV,YV,ZV) : (ZV,TV,WV)
C
C
C     FIELD VARIABLE(POTENTIAL) MODEL USED IS
C     PHI=N(I)*Q(I)
C     N(I)=SHAPE FUNCTION    ...I=1,10
C     N(1)=L(1)*(2*L(1)-1)
C     N(2)=L(2)*(2*L(2)-1)
C     N(3)=L(3)*(2*L(3)-1)
C     N(4)=L(4)*(2*L(4)-1)
C     N(5)=4*L(1)*L(2)
C     N(6)=4*L(2)*L(3)
C     N(7)=4*L(1)*L(3)
C     N(8)=4*L(1)*L(4)
C     N(9)=4*L(2)*L(4)
C     N(10)=4*L(3)*L(4)
C     L(I)=NATURAL COORDINATE    ...I=1,4
C     Q(I)=NODAL VALUE OF THE POTENTIAL
C     U(I)=DPHIDX=D(N(I))/DX * Q(I)    ...I=1,10
C     V(I)=DPHIDY=D(N(I))/DY * Q(I)    ...I=1,10
C     W(I)=DPHIDZ=D(N(I))/DZ * Q(I)    ...I=1,10
C
      PARAMETER NX1=3,NX2=12,NX3=15
      PARAMETER NPO=45
      PARAMETER NE2=8   ,  NCR=NX3+1
      PARAMETER NN=NPO*(2*NX3+1)
      PARAMETER NET=NE2*NX3
      REAL MO,LAMBDA,LAMBD1,LAMBD2
      DIMENSION NG(27),X(27),Y(27),Z(27)
      DIMENSION RP(27),XP(8),YP(8),ZP(8),D(4)
      COMMON/FROM1/NOP(NET,27),NOPP(NN),MDF(NN),NCOD(NN),BC(NN),P(NN)
      COMMON/INPUT/LAMBD1,LAMBD2,OMEGA,CHORD,NBL,PITCH
      COMMON/INPU2/THBLH(2,NCR),ZBLH(NCR),THBLT(2,NCR),ZBLT(NCR),
     ,NGB1(NE2,9),NGB2(NE2,9),RH(NCR),RTT(NCR)
      COMMON/COORD/RBL(2,NPO),TBL(2,NPO),ZBL(2,NPO)
      COMMON/BNDR/DFIDX,BETAIN,BETOUT
      COMMON/PRINT/ IDEBUG,IVEL,ICOMP,NP,IMFSH
      COMMON/CMPRSS/ POT(NN),MO,GAMA,LAMBDA
      COMMON/PERE/NCODT(6,10),NPER(6),LE,RC
      DIMENSION A(4),B(4),C(4)
      COMMON/PROV1/XV,YV,ZV,IV,IIP,JJP,IP,NELMP,ILLMP,ITET
      COMMON/PROV2/AR,DT,DZ,DTR,PHI
      COMMON/PARAM/NDT,NDR,NE1,NP1,NP2,NPOIN
      COMMON/DIM/IDIM,INTG
      COMMON/ISBL/THETAH(2,NX2-NX1),THETAT(2,NX2-NX1)
      COMMON/VNUMER/CL(4),CNN(10),CN1(10),CN2(10),CN3(10)
      COMMON/VELJAC/ZZ(10),YY(10),XX(10),AA(3),BB(3),CC(3),DETJ,PT(10)
      DATA NCODT/27,27,27,1,1,1,7,9,25,19,19,21,1,1,7,25,27,3,9,3,
     1 27,27,21,14,17,18,13,10,10,11,4,5,16,22,23,24,14,14,4,13,14,15,
     2 18,15,14,14,1,2,8,6,26,23,20,12,5,2,17,26,24/
      PI=ACOS(-1)
      NE=NE1
      NP=0
      J=NELMP
      CALL GENER(IP)
      DO 30 K=1,9
      Z(K)=RBL(1,NGB2(J,K))
      Z(K+10)=RBL(2,NGB2(J,K))
      Z(K+9)=(Z(K)+Z(K+10))/2.
      Y(K)=TBL(1,NGB2(J,K))
      Y(K+10)=TBL(2,NGB2(J,K))
      Y(K+9)=(Y(K)+Y(K+10))/2.
      X(K)=ZBL(1,NGB2(J,K))
      X(K+10)=ZBL(2,NGB2(J,K))
      X(K+9)=(X(K)+X(K+10))/2.
   30 CONTINUE
      IF(INTG.GT.2.AND.IP.GT.NX1.AND.IP.LE.NX2) THEN
      R1=(RBL(1,1)+RBL(2,1))/2.
      R2=(RBL(1,2*NDR+1)+RBL(2,2*NDR+1))/2.
      DR=R2-R1
      IF(J.LE.NDR) THEN
      IF(IMFSH.EQ.1) THEN
      AZ1=(Z(10)-R1)/DR
      AZ2=(Z(11)-R1)/DR
      AZ3=(Z(12)-R1)/DR
      Y(10)=THETAT(1,(IP-NX1))*AZ1+THETAH(1,(IP-NX1))*(1.-AZ1)
```

```
      Y(11)=THETAT(1,(IP-NX1))*AZ2+THETAH(1,(IP-NX1))*(1.-AZ2)
      Y(12)=THETAT(1,(IP-NX1))*AZ3+THETAH(1,(IP-NX1))*(1.-AZ3)
      ELSE
      AZ1=(Z(16)-R1)/DR
      AZ2=(Z(17)-R1)/DR
      AZ3=(Z(18)-R1)/DR
      Y(16)=THETAT(1,(IP-NX1))*AZ1+THETAH(1,(IP-NX1))*(1.-AZ1)
      Y(17)=THETAT(1,(IP-NX1))*AZ2+THETAH(1,(IP-NX1))*(1.-AZ2)
      Y(18)=THETAT(1,(IP-NX1))*AZ3+THETAH(1,(IP-NX1))*(1.-AZ3)
      END IF
      END IF
      IF(J.GT.(NDT-1)*NDR) THEN
      IF(IMESH.EQ.1) THEN
      AZ1=(Z(16)-R1)/DR
      AZ2=(Z(17)-R1)/DR
      AZ3=(Z(18)-R1)/DR
      Y(16)=THETAT(2,(IP-NX1))*AZ1+THETAH(2,(IP-NX1))*(1-AZ1)
      Y(17)=THETAT(2,(IP-NX1))*AZ2+THETAH(2,(IP-NX1))*(1-AZ2)
      Y(18)=THETAT(2,(IP-NX1))*AZ3+THETAH(2,(IP-NX1))*(1-AZ3)
      ELSE
      AZ1=(Z(10)-R1)/DR
      AZ2=(Z(11)-R1)/DR
      AZ3=(Z(12)-R1)/DR
      Y(10)=THETAT(2,(IP-NX1))*AZ1+THETAH(2,(IP-NX1))*(1.-AZ1)
      Y(11)=THETAT(2,(IP-NX1))*AZ2+THETAH(2,(IP-NX1))*(1.-AZ2)
      Y(12)=THETAT(2,(IP-NX1))*AZ3+THETAH(2,(IP-NX1))*(1.-AZ3)
      END IF
      END IF
      END IF
      DO 6 LL=1,27
      NG(LL)=NOP(IELMP,LL)
      IF(NG(LL).LT.0) NG(LL)=-NG(LL)
      IF(NG(LL).NE.0) THEN
      RP(LL)=R(NG(LL))
      IF(IP.LE.NX1.OR.IP.GT.NX2) THEN
      LAMBDA=LAMBD1
      IF(IP.GT.NX2) LAMBDA=LAMBD2
      IF(NELMP.GT.(NDT-1)*NDR) THEN
      DO 160 JPP=1,9
      IF(LL.EQ.NPER(JPP)) THEN
      RP(LL)=RP(LL)-LAMBDA
      END IF
  160 CONTINUE
      END IF
      END IF
      ELSE
      RP(LL)=0.0
      END IF
    6 CONTINUE
      N1=NPER(1)
      N2=NPER(2)
      N3=NPER(3)
      N7=NPER(7)
      N8=NPER(8)
      N9=NPER(9)
      IF(IP.EQ.NX1+1.AND.NELMP.GT.(NDT-1)*NDR) THEN
      RP(N1)=RP(N1)-LAMBD1
      RP(N2)=RP(N2)-LAMBD1
      RP(N3)=RP(N3)-LAMBD1
      END IF
      IF(IP.EQ.NX2.AND.NELMP.GT.(NDT-1)*NDR) THEN
      RP(N7)=RP(N7)-LAMBD2
      RP(N8)=RP(N8)-LAMBD2
      RP(N9)=RP(N9)-LAMBD2
      END IF
      LLL=0
      DO 9 LL=1,9,2
      IF(LL.EQ.5 ) GO TO 9
      LLL=LLL+1
      XP(LLL)=X(LL)
      YP(LLL)=Y(LL)
      ZP(LLL)=Z(LL)
      XP(LLL+4)=X(LL+18)
      YP(LLL+4)=Y(LL+18)
      ZP(LLL+4)=Z(LL+18)
    9 CONTINUE
      A1=XP(8)-XP(1)
      B1=YP(8)-YP(1)
      C1=ZP(8)-ZP(1)
      KO=1
      DO 70 K=1,3
      A2=XP(KO+K)-XP(1)
```

```
      B2=YP(KO+K)-YP(1)
      C2=ZP(KO+K)-ZP(1)
      AN=B1*C2-B2*C1
      BN=C1*A2-A1*C2
      CN=B2*A1-B1*A2
      IF(AN.EQ.0.) THEN
      IF(BN.EQ.0) GO TO 100
      IF(CN.EQ.0.) GO TO 110
      XPP=XV
      CB=ABS(C1/B1)
      YPP=(ZV-ZP(1)+1/CB*(YV-YP(1)))/(CB+1/CB)+YP(1)
      ZPP=CB*(YPP-YP(1))+ZP(1)
      GO TO 120
      ELSE
      IF(BN.EQ.0.) GO TO 130
      IF(CN.EQ.0.) GO TO 140
      R1=AN*XP(1)+BN*YP(1)+CN*ZP(1)
      R2=(BN/(AN**2+BN**2))*(R1-XV*AN+YV*AN**2/BN)
      R3=R2+ZV*BN/CN-YV
      ZPP=R3/(BN/CN+BN*CN/(AN**2+BN**2))
      YPP=R2-ZPP*CN*BN/(AN**2+BN**2)
      XPP=(R1-YPP*BN-ZPP*CN)/AN
      GO TO 120
      END IF
  100 IF(CN.EQ.0.) THEN
      PRINT *,'  ERROR IN SUBROUTINE PROP '
      STOP
      END IF
      XPP=XV
      YPP=YV
      ZPP=ZP(1)
      GO TO 120
  110 XPP=XV
      ZPP=ZV
      YPP=YP(1)
      GO TO 120
  130 IF(CN.EQ.0.) GO TO 150
      YPP=YV
      CA=ABS(C1/A1)
      XPP=(ZV-ZP(1)+1/CA*(XV-XP(1)))/(CA+1/CA)+XP(1)
      ZPP=CA*(XPP-XP(1))+ZP(1)
      GO TO 120
  140 ZPP=ZV
      BA=ABS(B1/A1)
      XPP=(YV-YP(1)+1/BA*(XV-XP(1)))/(BA+1/BA)+XP(1)
      YPP=BA*(XPP-XP(1))+YP(1)
      GO TO 120
  150 YPP=YV
      ZPP=ZV
      XPP=XP(1)
  120 CONTINUE
      IF(K.EQ.1) THEN
      IF(XV.GE.XPP) KO=3
      GO TO 70
      END IF
      IF(ZV.LT.ZPP) GO TO 80
   70 CONTINUE
      IF(IMESH.EQ.1) THEN
      IF(KO.EQ.3) ITET=1
      IF(KO.EQ.1) ITET=3
      END IF
      IF(IMESH.EQ.2) THEN
      IF(KO.EQ.3) ITET=4
      IF(KO.EQ.1) ITET=5
      END IF
      GO TO 90
   80 CONTINUE
      IF(IMESH.EQ.1) THEN
      IF(KO.EQ.1) THEN
      IF(K.EQ.2) ITET=6
      IF(K.EQ.3) ITET=2
      ELSE
      IF(K.EQ.2) ITET=5
      IF(K.EQ.3) ITET=6
      END IF
      END IF
      IF(IMESH.EQ.2) THEN
      IF(KO.EQ.1) THEN
      IF(K.EQ.2) ITET=1
      IF(K.EQ.3) ITET=6
      ELSE
      IF(K.EQ.2) ITET=3
```

```
      IF(K.EQ.3) ITET=2
      END IF
      END IF
   90 CONTINUE
      L=ITET
      DO 3 I=1,10
      J=NCONT(L,I)
      RT(I)=RP(J)
    3 CONTINUE
      DO 11 I=1,10
      J=NCONT(L,I)
      XX(I)=X(J)
      YY(I)=Y(J)
   11 ZZ(I)=Z(J)
      RC=(ZZ(1)+ZZ(2)+ZZ(3)+ZZ(4))/4.
      IF(IDTM.EQ.2) RC=1.0
      IF(ITET.EQ.3.OR.ITET.EQ.5) GO TO 190
      IF(IP.LE.NX1.OR.IP.GT.NX2) GO TO 190
      IF(INTG.GT.2.AND.NELMP.LE.NDR) THEN
      IF(IMESH.EQ.1) THEN
      IF(ITET.EQ.1.OR.ITET.EQ.6) GO TO 170
      ELSE
      IF(ITET.EQ.2.OR.ITET.EQ.4) GO TO 170
      END IF
      END IF
      IF(INTG.GT.2.AND.NELMP.GT.(NDT-1)*NDR) THEN
      IF(IMESH.EQ.1) THEN
      IF(ITET.EQ.2.OR.ITET.EQ.4) GO TO 170
      ELSE
      IF(ITET.EQ.1.OR.ITET.EQ.6) GO TO 170
      END IF
      END IF
  190 CONTINUE
      DET=0.0
      DO 4 I=1,4
      CL(I)=0.0
      J=2
      IF(I.GT.1) J=1
      K=3
      IF(I.GT.2) K=2
      LL=4
      IF(I.GT.3) LL=3
      A(I)=(-1)**I*((YY(K)*ZZ(LL)-YY(LL)*ZZ(K))+(YY(LL)*ZZ(J)-
     $YY(J)*ZZ(LL))+(YY(J)*ZZ(K)-YY(K)*ZZ(J)))
      B(I)=(-1)**(I+1)*((XX(K)*ZZ(LL)-XX(LL)*ZZ(K))+(XX(LL)*ZZ(J)-
     $XX(J)*ZZ(LL))+(XX(J)*ZZ(K)-XX(K)*ZZ(J)))
      C(I)=(-1)**I*((XX(K)*YY(LL)-XX(LL)*YY(K))+(XX(LL)*YY(J)-
     $XX(J)*YY(LL))+(XX(J)*YY(K)-XX(K)*YY(J)))
      D(I)=(-1)**(I+1)*((XX(K)*YY(LL)-XX(LL)*YY(K))*ZZ(J)
     1 +(XX(LL)*YY(J)-XX(J)*YY(LL))*ZZ(K)
     2 +(XX(J)*YY(K)-XX(K)*YY(J))*ZZ(LL))
      DET=DET-XX(I)*A(I)
    4 CONTINUE
      DO 20 I=1,4
      A(I)=A(I)/DET
      B(I)=B(I)/DET
      C(I)=C(I)/DET
      D(I)=D(I)/DET
      CL(I)=-( A(I)*XV+B(I)*YV+C(I)*ZV+D(I))
   20 CONTINUE
      DZ=VEL(CL,A,RT)
      DT=VEL(CL,B,RT)
      IF(IDTM.EQ.3.AND.ZV.NE.0.) DT=DT/ZV
      DR=VEL(CL,C,RT)
      GO TO 180
  170 CONTINUE
      IF(ITET.EQ.1.OR.ITET.EQ.4) THEN
      CL(1)=0.0
      Z34=XX(3)-XX(4)
      Z24=XX(2)-XX(4)
      R34=ZZ(3)-ZZ(4)
      R24=ZZ(2)-ZZ(4)
      RM4=ZV-ZZ(4)
      ZM4=XV-XX(4)
      DETA=R24*Z34-R34*Z24
      CL(2)=(Z34*RM4-R34*ZM4)/DETA
      CL(3)=(R24*ZM4-RM4*Z24)/DETA
      CL(4)=1.-CL(2)-CL(3)
      CALL TSOVEL
      END IF
      IF(ITET.EQ.2.OR.ITET.EQ.6)THEN
      CL(3)=0.0
```

```
      R24=ZZ(2)-ZZ(4)
      R14=ZZ(1)-ZZ(4)
      Z24=XX(2)-XX(4)
      Z14=XX(1)-XX(4)
      RM4=ZV-ZZ(4)
      ZM4=XV-XX(4)
      DETA=R14*Z24-R24*Z14
      CL(1)=(Z24*RM4-R24*ZM4)/DETA
      CL(2)=(R14*ZM4-RM4*Z14)/DETA
      CL(4)=1.-CL(1)-CL(2)
      CALL ISOVEL
      END IF
      IF(IOTM.EQ.3.AND.ZV.NE.0.) DT=DT/ZV
  180 CONTINUE
      DTR=DT-OMEGA*ZV
      PHI=RT(1)*CL(1)*(2*CL(1)-1)+RT(2)*CL(2)*(2*CL(2)-1)+
     $ RT(3)*CL(3)*(2*CL(3)-1)+RT(4)*CL(4)*(2*CL(4)-1)+
     $ RT(5)*4.*CL(1)*CL(2)+RT(6)*4.*CL(2)*CL(3)+
     $ RT(7)*4.*CL(1)*CL(3)+RT(8)*4.*CL(1)*CL(4)+
     $ RT(9)*4.*CL(2)*CL(4)+RT(10)*4.*CL(3)*CL(4)
    8 FORMAT(5X,I4,6(E10.4))
    2 FORMAT(4(4X,'MODE',I2,1X,'X=',F5.2,'Y=',F5.2,'Z=',F5.2))
      RETURN
      END
```

REFERENCES

1. Lorenz, H., "Theorie und Berechnung der Vollturbinen Kreisel-Pumpem", V.D.I. Zeitschr., Bd. 49, No. 41, 1905.

2. Bauersfeld, W., "Zuschrift an die Redaktion", V.D.I. Zeitschr. fur Math. und Phys., Bd. 49, No. 49, 1905.

3. V. Mises, R., "Theorie der Wasserrader", Zeitschr. fur Math. und Phys., Bd. 57, 1909.

4. Stodola, A., "Steam and Gas Turbines", Mc Graw Hill Book Co., Inc., 1927.

5. Dreyfus, L.A., "A Three-Dimensional Theory of Turbine Flow and its Application to the Design of Wheel Vanes for Francis and Propeller Turbines", Acta Polytechnica, Mech. Eng. Ser., Vol. 1, No. 1, 1946.

6. Reissner, H., "Blade Systems of Circular Arrangement in Steady Compressible Flow", R. Courant Anniversary Volume, Interscience Pub. Inc., 1948.

7. Howell, A.R., "A Theory of Arbitrary Aerofoils in Cascade", Phil. Mag., Vol. 39, No. 299, December 1948, pp. 913-927.

8. Pollard, D. and Wordsworth, J., "A Comparison of Two Methods for Predicting the Potential Flow Around Arbitrary Aerofoils in Cascade", A.R.C. C.P. 618, June 1962.

9. Garrick, J.R., "On the Plane Potential Flow Past a Lattice of Arbitrary Airfoils", N.A.C.A. Report 788, 1944.

10. Hall, W.S. and Thwaites, B., "On the Calculation of Cascade Flows", A.R.C. C.P. 806, November 1963.

11. Schlichting, H., "Berechnung der Reibungslosen Incompressiblen Stromung fur ein vorgegebenes ebenes Schaufelgitter", V.D.K. Forschungshaft 447, 1955.

12. Mellor, G.L., "An Analysis of Axial Compressor Cascade Aerodynamics" Jl of Basic Engrg., September 1959.

13. Martensen, E., "Archieve for Rotational Mechanics Analysis", Vol. 3, No. 3, 1959.

14. Merchant, W. and Collar, A.R., "Flow of an Ideal Fluid Past a Cascad of Blades (Part II)", A.R.C. R.& M. No. 1893, May 1941.

15. Gostelow, J.P., "Potential Flow Through Cascades-A Comparison Betwee Exact and Approximate Solutions", A.R.C. C.P. No. 807, 1965.

16. Lamb, H., "Hydrodynamics", C.U.P. 5th Ed., 1932, p. 68.

17. Wu, C.H., "A General Theory of Three-Dimensional Flow in Subsonic and Supersonic Turbomachines of Axial-, Radial-, and Mixed-Flow Types", N.A.C.A. TN 2604, 1952.

18. Marsh, H., "A Digital Computer Program for the Through Flow Fluid Mechanics in an Arbitrary Turbomachine Using a Matrix Method", National Gas Turbine Establishment, August 1966.

19. Katsanis, T., "Fortran Program for Calculating Transonic Velocities on a Blade-to-Blade Stream Surface of a Turbomachine", N.A.S.A. Report TN D-5427, September 1969.

20. Bosman, C. and El-Shaarawi, M.A.I., "Quasi-Three-Dimensional Numeric Solution of Flow in Turbomachines", Trans. ASME, Journal of Fluid Engineering, March 1977, pp. 132-140.

21. Farrel, C. and Adamczyk, J., "Full Potential Solution of Transonic Quasi-3-D-Flow Through Cascade Using Artificial Compressibility", ASME Paper, 81-GT-70, 1981.

22. Smith, L.H., "The Radial Equilibrium Equation of Turbomachinery", Trans. ASME, Journal of Engineering for Power, Series A, Vol. 88, 1966, pp. 1-12.

23. Senoo, Y. and Nakase, Y., "An Analysis of Flow Through a Mixed Flow Impeller", Trans. ASME, Journal of Engineering for Power, January 1972, pp. 43-50.

24. Katsanis, T. and McNally, W.D., "Fortran Program for Calculating Velocities and Streamlines on the Hub-Shroud Mid-Channel Flow Surface of an Axial- or Mixed-Flow Turbomachine", N.A.S.A. Report, No. TN D-7343, July 1973.

25. Novak, R.A., "The Mean Streamsheet Momentum/Continuity Solutions for Turbomachinery", Lecture 30, Symposium on Fluid Dynamics of Turbomachinery, Iowa State University, 1975.

26. Novak, R.A. and Hearsey, R.M., "A Nearly Three-Dimensional Computing System for Turbomachinery", Trans. ASME, Journal of Fluids Engineering March 1977, pp. 154-166.

27. Kundig, A., "A Fast Numerical Procedure to Solve the Meridional Equations of Motion in a Multistage Axial Flow Turbomachines", ASME Paper, No. 81-GT-133.

28. Adler, D. and Krimerman, Y., "The Numerical Calculation of the Meridional Flow Field in Turbomachines Using The Finite Element Method", Israel Journal of Technology, 1974, 12 (No. 3/4).

29. Hirsch, Ch. and Warzee, G., "A Finite-Element Method for Through Flow Calculations in Turbomachines", Trans. ASME, Journal of Fluids Engineering, September 1976, pp. 403-421.

30. Adler, D. and Krimerman, Y., "Calculation of the Blade-to-Blade Compressible Flow Field in Turbo Impellers Using the Finite-Element Method", Journal Mechanical Engineering Science, Vol. 19, No. 3, 1977, pp. 108-112.

31. Habashi, W.G., Dueck, E.G. and Kenny, D.P., "Finite-Element Approach to Compressor Blade-to-Blade Cascade Analysis", AIAA Journal, Vol.17, No. 7, July 1979, pp. 693-698.

32. Deconinc, H. and Hirsch, Ch., "Finite-Element Methods for Transonic Blade-to-Blade Calculation in Turbomachines", Trans. ASME, Journal of Engineering for Power, Vol. 103, October 1981, pp. 665-677.

33. Haas, W. and Keck, H., "Three-Dimensional Flow in Turbomachines - A Finite Element Solution on Minicomputers", Computers and Structures, Vol. 12, 1980, pp. 435-443.

34. Baskharone, E. and Hamed, A., "A New Approach in Cascade Flow Analysis Using the Finite Element Method", AIAA Journal, Vol. 19, No. 1, January 1981, pp. 65-68.

35. Ecer, A. and Akay, H.U., "Investigation of Transonic Flow in a Cascade Using an Adaptive Mesh", AIAA Paper 80-1430; also, AIAA Journal, Vol. 19, 1981, pp. 1174-1182.

36. Akay, H.U. and Ecer, A., "Transonic Flow Computations in Cascades Using Finite Element Method", ASME 26th International Gas Turbine Conference, Houston, Tex., March 1981; also, ASME Journal of Engineering for Power, Vol. 103, No. 4, 1981, pp. 657-664.

37. Ecer, A. and Akay, H.U., "Finite-Element Analysis of Transonic Flows in Cascades-Importance of Computational Grids in Improving Accuracy and Convergence", N.A.S.A. C.R. 3446, July 1981.

38. Akay, H.U. and Ecer, A., "Finite-Element Analysis of Transonic Flows in Highly Staggered Cascades", AIAA Journal, Vol. 20, No. 3, March 1982, pp. 410-415.

39. Krimerman, Y. and Adler, D., "The Complete Three-Dimensional Calculation of the Compressible Flow Field in Turbo Impellers", Journal of Mechanical Engineering Science, Vol. 20, No. 3, 1978, pp. 149-158.

40. Hirsch, Ch. and Warzee, G., "An Integrated Quasi-3D Finite Element Calculation Program for Turbomachinery Flows", Trans. ASME, Journal of Engineering for Power, Vol. 101, January 1979.

41. Laskaris, T.E., "Finite-Element Analysis of Three-Dimensional Potential Flow in Turbomachines", AIAA Journal, Vol. 16, No. 7, July 1978, pp. 717-722.

42. Whirlow, D.K., Farn, C.L.S. and Goldschmied, F.R., "Analysis of the Three-Dimensional Potential Flow of Centrifugal Lift Fans", AIAA Journal of Hydronautics-Paper H-528, 1980.

43. Sarathy, K.P., "Computation of Three-Dimensional Flow Fields Through Rotating Blade Rows and Comparison with Experiment", ASME Paper, No. 81-GT-121, March 1981.

44. Denton, J.D., "Extension of the Finite Area Time Marching Method to Three-Dimensions", VKI Lecture Series 84, Transonic Flows in Axial Turbomachinery, February 1976.

45. Whitney, W.J., Szanca, E.M., Moffitt, T.P. and Monroe, D.E., "Cold-Air Investigation of a Turbine for High-Temperature-Engine Application. I. Turbine Design and Overall Stator Performance", NASA TN D-3751, 1967.

46. Mengütürk, M. and Sverdrup, E.F., "Calculated Tolerance of a Large Electric Utility Gas Turbine to Erosion Damage by Coal Gas Ash Particles", Erosion: Prevention and Useful Applications", ASTM STP 664, Adler, W.F. (Ed.), 1979, pp. 193-224.

47. Katsanis, T. and McNally, W.D., "Fortran Program for Calculating Velocities in a Magnified Region on a Blade-to-Blade Stream Surface of a Turbomachine", NASA TN D5091, April 1969.

48. Güneş, D. and Mengütürk, M., "Improved Particle Trajectory Calculation Around Blade Leading Edge", To be presented at the Sixth International Conference on Erosion by Liquid and Solid Impact (ELSI VI), Cambridge, 5-8 September, 1983.

49. Hammer, P.C., Marlowe, O.P. and Stroud, A.H., "Numerical Integration Over Simplexes and Cones", Math. Tables Aids Comp., Vol. 10, 1956, pp. 130-137.