EXPERIMENTAL AND NUMERICAL INVESTIGATIONS OF PROFILE AND SECONDARY AERODYNAMIC LOSSES IN HIGH PRESSURE AXIAL TURBINES

A Dissertation

by

HICHAM ALI CHIBLI

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Meinhard Schobeiri
Committee Members,	Je-Chin Han
	Alan Palazzolo
	Hamn-Ching Chen
Head of Department,	Andreas Polycarpou

August 2018

Major Subject: Mechanical Engineering

ABSTRACT

A series of experimental studies were conducted in order to investigate the generation and growth of aerodynamic losses in high pressure axial turbine stages. A two-dimensional linear blade cascade facility was utilized with two sets of blade designs to examine the effects of varying the flow incidence angle as well as the cascade stagger angle on the profile loss component and the blade loading. Experimental findings were compared to CFD results and empirical correlations from literature. A two-stage research axial turbine was assembled with a set of bowed stator and twisted rotor blades to study the three-dimensional aerodynamic losses at design and off-design operations. Extensive performance tests were carried out to create the full efficiency map of the turbine, while detailed interstage measurements resolved the complete flow field of the second stage. The radial distribution of the various flow parameters within the stator and rotor rows enabled accurately calculating the losses, which were compared to steady and transient CFD results. Findings from these experimental investigations were implemented into a generic streamline curvature method solver that inviscidly solves the radial equilibrium equations, and provides additional corrections for profile, secondary, trailing edge and seal leakage aerodynamic losses using empirical correlations to help predict the overall performance of axial turbines at various operating conditions.

DEDICATION

To my loving family!

ACKNOWLEDGEMENTS

I would like to sincerely thank my academic advisor Dr. M.T. Schobeiri for offering me the generous opportunity to join his research team at the Turbomachinery Laboratory at Texas A&M University. I am most grateful for his continued patience, guidance and support over the years. I would also like to thank Dr. J.C. Han, Dr. A.B. Palazzolo and Dr. H.C. Chen for kindly serving on my graduate committee.

Many thanks are due to all my good friends at the Turbomachinery Laboratory. Your comradeship and encouragement got me through the finish line.

And finally all my gratitude goes to my dear family whose endless love and support made this journey ever possible. Thank you!

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee consisting of Professor M.T. Schobeiri (committee chair and academic advisor), Professor J.C. Han and Professor A.B. Palazzolo of the Department of Mechanical Engineering and Professor H.C. Chen of the Department of Civil Engineering at Texas A&M University.

The turbomachinery loss formulation employed in this study and presented in Chapter 4 was provided by Prof. M.T. Schobeiri. The CFD simulations presented in Chapters 2, 3 and 4 were conducted by Dr. S.A. Abdelfattah of the Department of Mechanical Engineering, and published in 2012 and 2013 in two articles listed in the Biographical Sketch. The experimental measurements presented in Chapter 4 pertaining to the Westinghouse three-stage research turbine were gathered by Dr. J.L. Gilarranz and Dr. E.S. Johansen of the Turbomachinery Performance and Flow Research Laboratory, and published in 2004 in an article listed in the the Biographical Sketch.

All other work conducted for the dissertation was completed by the student independently.

Funding Sources

Professor M.T. Schobeiri secured financial funding for this study from Doosan Heavy Industries & Construction, South Korea under contract number: TEES–38760–32525. The Mechanical Engineering Department provided additional financial support through teaching assistantship positions in the Fall and Spring semesters of 2006-2007 and 2010-2011.

NOMENCLATURE

b	Blade trailing edge diameter	[mm]
c	Blade chord length	[mm]
C_p	Blade pressure coefficient	[ ]
d	Blade leading edge diameter	[mm]
f	Maximum blade camber height	[mm]
h	Blade height, Specific static enthalpy	[mm, j/kg]
H	Specific total enthalpy	[j/kg]
H_r	Relative specific total enthalpy	[j/kg]
i	Flow incidence angle	[deg]
$\dot{m}$	Mass flow rate	[kg/sec]
M	Mach number	[ ]
M_r	Relative Mach number	[ ]
n	Machine rotational speed	[rpm]
p	Static pressure	[Pa]
pr	Static pressure ratio	[ ]
P	Power, Total pressure	[W, Pa]
P_r	Relative total pressure	[Pa]
r^*	Immersion ratio	[ ]
r_c	Radius of curvature on a streamline	[mm]
Re	Reynolds number	[ ]

s	Blade spacing (pitch)	[mm]
T	Static temperature	[K]
$\mathrm{u/c_{\circ}}$	Dimensionless performance parameter	[ ]
U	Circumferential velocity	[m/s]
V	Absolute velocity	[m/s]
W	Relative velocity	[m/s]
Z	Loss coefficient	[ ]

Greek Symbols

α	Absolute velocity flow angle, Pitch angle	[deg]
β	Relative velocity flow angle, Yaw angle	[deg]
γ	Blade stagger, Meridional velocity flow angle	[deg]
δ	Boundary layer thickness, Blade deflection angle	[mm, deg]
δ_1	Boundary layer displacement thickness	[mm]
δ_2	Boundary layer momentum deficiency thickness	[mm]
ε	Blade lean angle	[deg]
ζ	Total pressure loss coefficient	[ ]
η_{ts}	Total–Static Efficiency	[ ]
λ	Stage load coefficient	[ ]
ρ	Density	[kg/m^3]
σ	Blade cascade solidity	[ ]
τ	Torque	[N.m]
φ	Stage flow coefficient	[ ]
Ψ	Zweifel number	[ ]

Other

RANS	Reynolds-averaged Navier-Stokes
SCM	Streamline Curvature Method
TPFL	Turbine Performance and Flow Laboratory
URANS	Unsteady Reynolds-averaged Navier-Stokes

Subscripts

$1-7$	Turbine station, Cascade axial plane location
c	Calibration, Corrected
d	Dynamic
$\mathrm{i} = 1 \ldots 40, 46$	Blade static pressure tap index
m	Meridional
n	Normalized
p	Profile
r	Radial, Rotor
ref	Reference
s	Isentropic, Secondary, Stator
x , z	Axial
y , θ	Tangential

Superscripts

—	Mass-averaged value

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION*

1.1 Profile Losses

The primary aerodynamic losses in axial turbines are the profile losses attributed to the viscous activity within the boundary layer growth on the blade suction and pressure surfaces, as the flow turns and expands through the stator and rotor rows as schematically illustrated in Fig. 1.1. The flow kinetic energy is dissipated into heat, leading to an irreversible increase in entropy and reduction in available total pressure. Since the blade geometry in a multi-stage turbine is optimized to match the flow angles at the design operating point, distortions to the optimum flow velocity triangles are induced during operations under off-design conditions. Consequently the blade row inlet flow velocity vectors deviate from their intended design incidence angles, leading to a cumulative increase in the stage profile losses. Off-design profile losses are commonly termed *incidence* losses.

Zweifel [1] introduced the aerodynamic loading coefficient that estimates the ratio of the actual to ideal tangential force acting on a blade cascade, which he empirically correlated to early measurements by Christiani [2] and Keller [3] to characterize an optimum blade cascade solidity for turbines and compressors. Through extensive experimental measurements, Pfeil [4, 5] investigated the optimum spacing/chord ratio for turbine and compressor blade cascades. He subsequently formulated an empirical correlation for the profile loss calculation as function of the Zweifel blade loading coefficient, cascade solidity the blade inlet and exit angles. Schobeiri [6] demonstrates

*Part of the data reported in this chapter is reprinted with permission from "An Experimental and Numerical Study of the Effects of Flow Incidence Angles on the Performance of a Stator Blade Cascade of a High Pressure Steam Turbine" by H.A. Chibli, S.A. Abdelfattah, M.T. Schobeiri, and C. Kang. ASME Turbo Expo: Power for Land, Sea, and Air, Volume 7: Turbomachinery, Parts A and B :821-830. doi:10.1115/GT2009-59131. Copyright 2009 by American Society of Mechanical Engineers.

the concept of an optimum cascade solidity as an equilibrium between the separation and friction loss components acting on the blade, as shown in Fig.1.2. For a densely packed blade cascade, the reduction in blade spacing at a constant chord length leads to increased frictional surface dissipation. On the other hand, separation losses are dominant in unguided flow in cascades with low solidity.

Ainley and Mathieson [8] pioneered the study of flow incidence on axial turbines in the early fifties. Their extensive measurements led to a widely adopted set of correlations that systematically cover all the aspects of the loss components incurred in an axial flow turbine at design and off-design conditions. In 1970, Dunham and Cane [9] published a refined model that better predicted the secondary losses in a turbine stage. In the following year, Craig and Cox [10] introduced a loss prediction system that, unlike that of Ainley and Mathieson, correlates positive and negative incidence independently. Reflecting the improved understanding of the flow physics inside turbomachines, Kacker and Okapuu [11] presented a modified correlation in 1982 that more accurately estimates the design point losses, which are used as the basis for extrapolating the off-design profile and secondary losses associated with

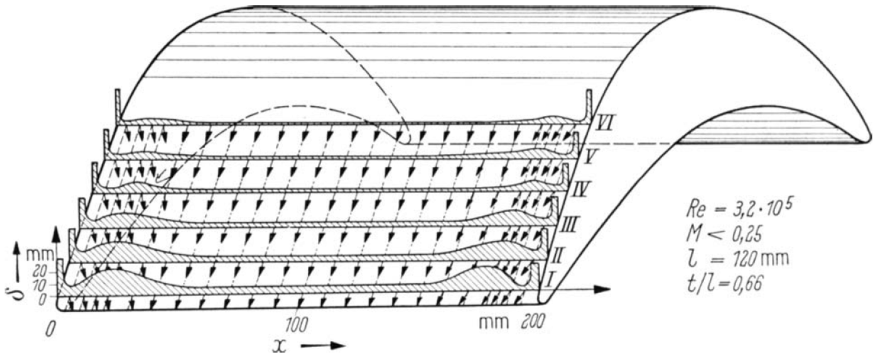


Figure 1.1: Growth of the boundary layer on the suction side of a high pressure turbine blade, obtained from Sauer et al. [7]

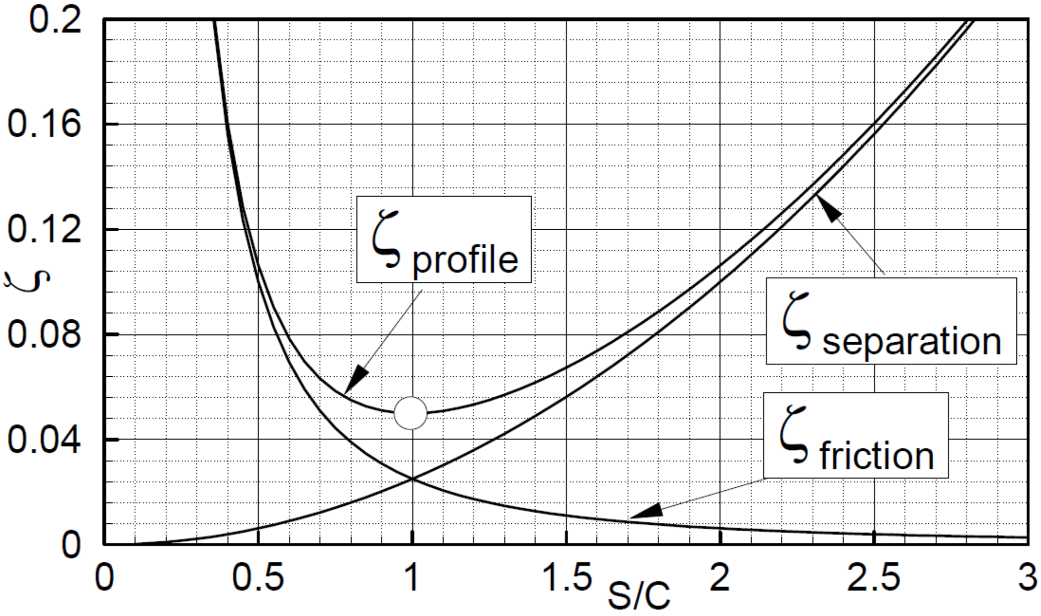


Figure 1.2: Optimum profile loss coefficient as a function of cascade spacing/chord ratio, obtained from Schobeiri [6]

different incidence conditions.

The effect of the blade leading edge diameter on the design and off-design loss calculations was suggested by Mukhtarov and Krichakin [12] in 1969. In 1987, Martelli and Boretti [13] incorporated the blade leading edge wedge angle to the Ainley and Mathieson correlation. A full review of the available correlations and a comprehensive comparison of the two most prominent studies by Ainley and Mathieson, and Mukhtarov and Krichakin, with a large array of existing experimental results from literature, was published by Moustapha et al. [14] in 1990. This study concluded that since all the existing empirical correlations were built around the first generation blades of the fifties, they lacked the capacity to accurately predict the performance of the modern turbine blade design. This study also formulated a correlation based on the Kacker and Okapuu model, which was subsequently extended by Benner et al. [15] in 1997.

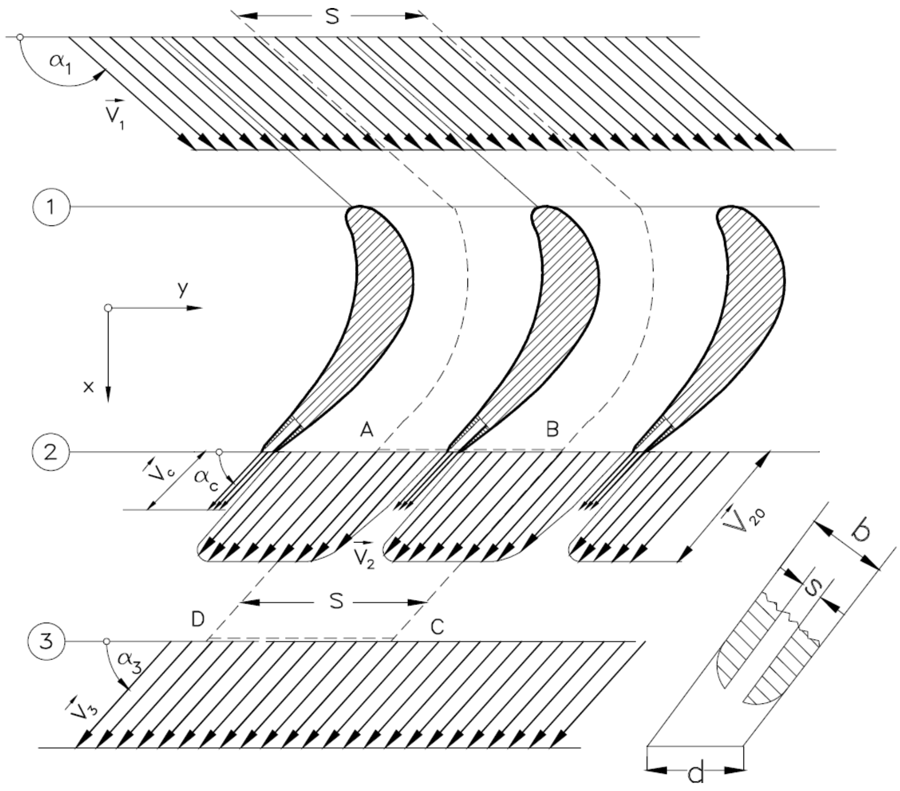


Figure 1.3: Trailing edge ejection and mixing downstream of a cooled gas turbine blade, obtained from Schobeiri [6]

In recent years, a number of studies have been conducted to address the effects of flow incidence on the development of losses in transonic blade cascades. Jouini et al. [16, 17] examined the effects of the Reynolds and Mach numbers on the profile losses. Li et al. [18] concluded that at transonic and low supersonic flows, the effect of Mach number becomes more prominent than that of off-design incidence in determining the cascade losses. They proposed an improved correlation that better predicted the shock wave related losses than the existing Kacker and Okapuu model.

Additionally, profile losses may incorporate blade trailing edge mixing losses as the boundary layer momentum deficiency propagates downstream of the blade rows. As demonstrated in Fig. 1.3, the wake structure with an exit velocity deficit under-goes a mixing process further downstream of the blade cascade causing irreversible total pressure losses. In transonic turbine stages, additional losses due to shockwave expansions are also applicable. Denton [19] estimated this loss as a function of the blade *base* pressure and geometry to contribute 1/3 of the total entropy generation in a typical transonic turbine blade cascade. By applying the conservation equations in a flow mixing control volume between the blade trailing edge and the mixing plane downstream of the cascade, Schobeiri [6] derived a loss model that incorporates the cascade solidity and discharge flow angle, the finite thickness at the blade trailing edge and the boundary layer thickness on the blade suction and pressure surfaces. Analytical and experimental studies by Schobeiri [20, 21] and Schobeiri and Pappu [22] expanded this model to include the effects of trailing edge ejection and mixing in cooled gas turbine blades and identify the optimum geometric ejection parameters as dictated by the heat transfer requirements.

1.2 Secondary Losses

Mapping the performance of multi-stage high pressure axial turbines requires sufficient understanding of the flow physics that drive the various loss mechanisms. The small geometric aspect ratios of blades used in these turbines create a highly three-dimensional flow field that is intrinsically unsteady. Consequently, complex secondary flow structures dominate the flow scene, especially in the vicinity of the endwall regions throughout the blade passage and contribute a major share of the overall aerodynamic losses generated in a typical stage.

Numerous experimental investigations have been carried out over the last five decades on linear and annular blade cascades in an effort to individually address the numerous aspects of the flow in axial turbines, that shape these losses. The classical inviscid theory of secondary flows in turbomachines which was laid out by Squire et al. [23], Smith [24] and Hawthorne [25] among others, postulated the development of the passage vortex as a result of the distortion to the inlet boundary layer vorticity through the curved blade channel. The detailed flow measurements and visualizations that followed enabled a more realistic understanding of the evolution process of the secondary flow structures. Langston et al. [26] performed extensive measurements of the flow within the endwall boundary layer of a linear cascade. In their study they described the formation of the blade leading edge horseshoe vortex, which comprises two counter rotating suction and pressure side legs. Sieverding et al. [27] used a colored smoke wire technique to visualize the entire stream surfaces in cascades and detailed the synchronous evolution of horseshoe and passage vortices through the blade passage [28]. Sharma et al. [29] introduced an effective method of predicting the secondary flow and endwall losses by estimating the spanwise extent of the secondary flow region at the trailing edge of a turbine cascade.

Schobeiri [6] developed an advanced system of correlations to empirically calculate the secondary losses in axial turbines with shrouded and unshrouded blading, using the comprehensive experimental results of earlier studies by Wolf [30], Berg [31] and Kirchberg and Pfeil [32]. In addition to evaluating the losses due to the boundary layer growth on the turbine endwalls, Schobeiri addresses the losses associated with the vortical activity in the blade passage and around the tip and hub clearance gaps. As shown in Fig. 1.4, endwall secondary flow vortices are formed in the hub and tip areas where the interactions between the low energetic viscous boundary layer and the pressure gradient that governs the flow field in the blade passage, causes flow circulation from the pressure to suction sides of the cascade channel. Tip and hub clearance vortices develop due to the pressure difference across the clearance gap between the blade and the turbine hub or casing. Fluid movement from the blade pressure to suction side across the clearnace gap induces a system

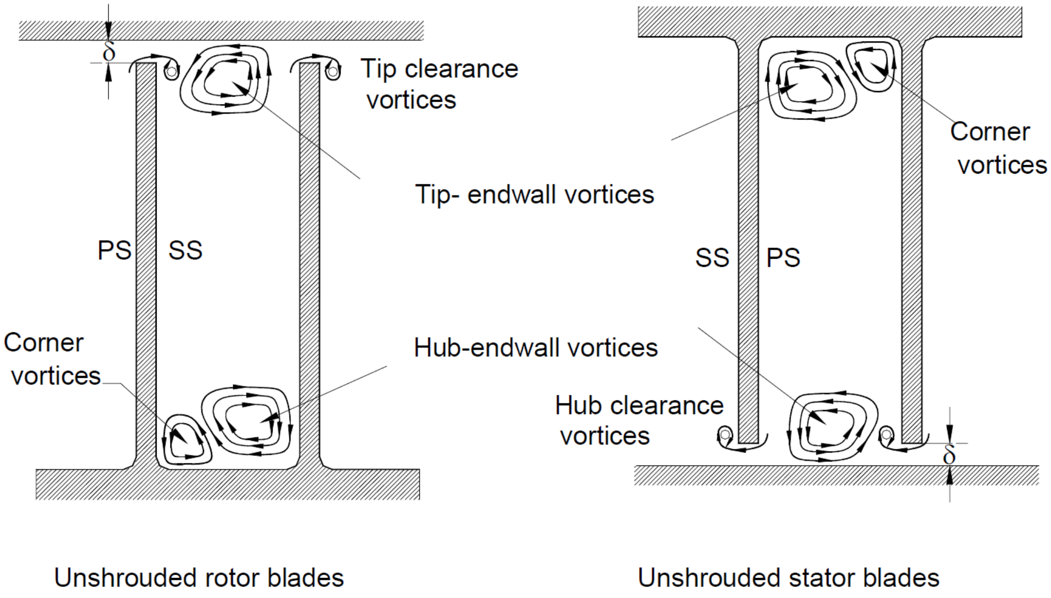


Figure 1.4: Development of endwall vortex activity in unshrouded rotor and stator blade passages, obtained from Schobeiri [6]

of bound vortices, which coalesce to form a free vortex system at the blade trailing edge as shown in Fig. 1.5. Employing numerical nonlinear dynamic simulations, Schobeiri and Abouelkheir [33] demonstrated a row-by-row loss calculation procedure to accurately predict the performance of a multi-stage axial turbine at design and off-design operating conditions.

Emunds et el. [34] conducted a computational study of the adjacent blade row effects in a 1.5-stage axial flow turbine with Traupel and VKI untwisted blade profiles. Two computational codes that employ the Baldwin-Lomax algebraic turbulence model [35] were used and results compared to the corresponding experimental results of measurements obtained downstream of each blade row. A qualitative agreement between the computations and measurements was established with the total pressure distribution, however a notable quantitative difference that stems from the numerical model inadequacies was observed for the same. Hodson et al. [36] experimentally

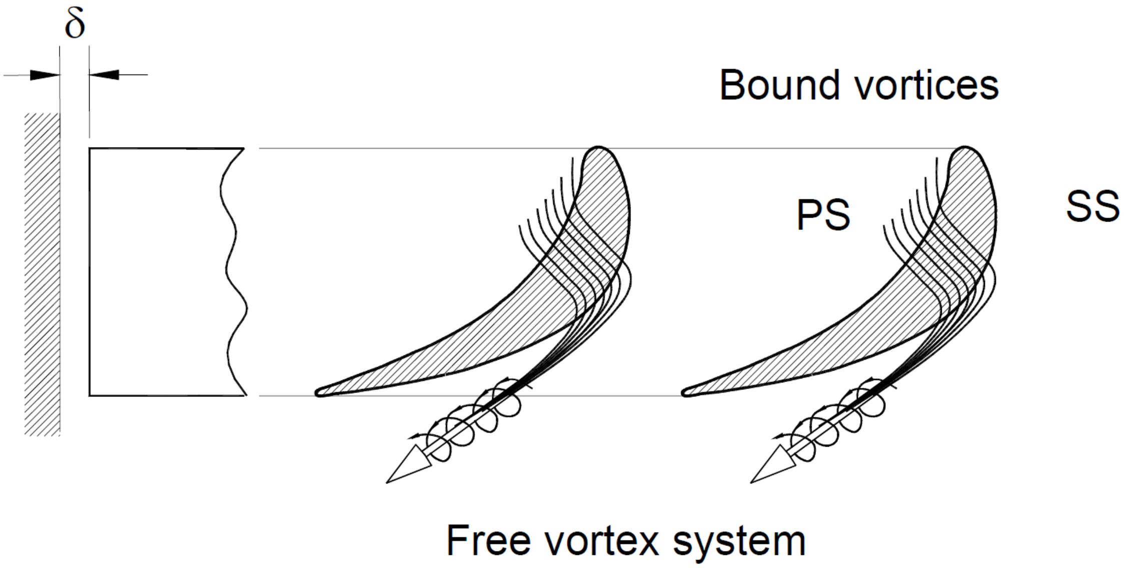


Figure 1.5: Tip clearance vortices for an unshrouded turbine blade, obtained from Schobeiri [6]

investigated the behavior of stator blade suction surface boundary layers in a four-stage low pressure turbine using surface mounted hot film anemometers to perform unsteady flow measurements. They concluded that passing wakes from upstream rotor rows periodically initiated transition at about 40% of the blade length, while the flow was entirely laminar otherwise, with a boundary layer separation occurring at about 75% of the blade length.

Halstead et al. [37, 38, 39, 40] presented a comprehensive four-part experimental and numerical investigation of boundary layer development in axial compressors and turbines. The experimental measurements were made using an array of densely packed surface mounted hot film gauges as well as traversing hot wire probes for surveying the boundary layer in a large low speed research facility. Steady and unsteady viscous codes were used to simulate the flow at a broad range of Reynolds numbers and pressure loading. Part 3 focused on the interaction of boundary layer growth and transition on pressure and suction surfaces of multi-stage low pressure turbines with periodically impinging wakes from upstream blade rows. Detailed analysis proved that the blade surface boundary layer was predominantly laminar contrary to existing assumptions. Transition to turbulence occurred later than previously postulated and was greatly influenced by the Reynolds number and wake frequency. Numerical codes were found to provide reasonable loss predictions relevant to overall stage performance, but did not adequately predict the onset and extent of boundary layer transition or separation in regions of adverse pressure gradients.

The wake-induced rotor-rotor and stator-rotor interactions in a five-stage low pressure turbine were investigated by Arndt [41] using hot film measurements. He found the interactions to result in strong amplitude modulated periodic and turbulent velocity fluctuations that had profound influence on the flow downstream of every rotor blade row. Schobeiri et al. [42, 43] established a comprehensive theoretical framework to investigate the phenomenon of steady and unsteady wake development and decay. This was followed by an experimental investigation of the effects of passing frequency of periodic unsteady wakes on the development and transition of boundary layers along the concave surface of a curved plate at zero pressure gradient [44] and a linear axial turbine cascade [45]. Employing the liquid crystal paint technique, Wright and Schobeiri [46] extended the investigation to heat transfer measurements under the same conditions. This enabled Chakka and Schobeiri [47] to develop an improved boundary layer transition model for unsteady flow applications in turbomachinery.

Camci et al. [48] experimentally investigated the aerodynamic characteristics of multiple configurations of full and partial length squealers in an axial turbine stage. Using a dynamic pressure transducer in a total pressure probe arrangement to generate a complete high resolution mapping of all the blade passages at the rotor exit plane, the authors quantified the performance impact of each squealer design arrangement. They reported that a suction side partial length squealer specifically matched to the rotor blade geometry is most effective at weakening the tip leakage vortex and reducing the stage total pressure defect. Krishnababu et al. [49] performed a numerical and experimental study of the effect of tip geometry on the tip leakage flow and heat transfer characteristics in an axial turbine. Three different tip configurations were included at two blade to casing clearance gaps: A base case of flat surface tip, a full length suction side squealer and a full tip cavity squealer. The latter was found to reduce the mass flow leakage most due to increased blockage caused by the emergence of separation bubbles on top of the pressure and suction sides of the squealer. The suction side squealer on the other hand was most effective at reducing the heat transfer coefficient to the blade tip compared to the base case.

Using the research axial turbine experimental facility at TPFL, Schobeiri et al. [50] performed detailed interstage and stage performance measurements on a three-stage high pressure turbine with 3D shrouded blades. Stator and rotor row total pressure loss coefficients as well as overall stage efficiency were calculated at design and off-design points. Rotational speed was varied from 75% to 116% of design speed, and a speed deviation of 10% was found to cause a 3% drop in stage efficiency relative to design point. In a follow-up investigation, Schobeiri and his coworkers [51] conducted a comparative study of two sets of shrouded high pressure turbine blades. The first turbine rotor incorporates a set of 3D convexly bowed blades whereas the second utilizes a set of fully cylindrical blade design. Detailed aerodynamic measurements clearly identified lower performance and particularly higher secondary losses generated closer to the endwalls at the tip and hub regions for the cylindrical blade design. The 3D bowed blade design on the other hand, induced higher axial velocities in those regions that stifled the activity of low energy vortices which lead to

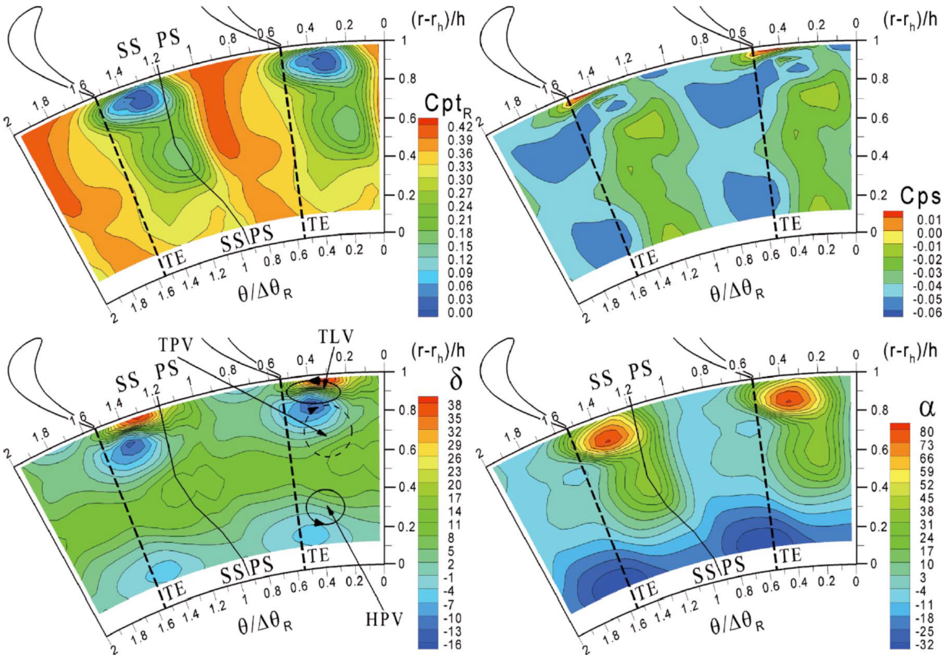


Figure 1.6: Unsteady interstage measuremnts of total pressure losses and flow deviation angles, obtained from Persico et al. [52]

increased secondary losses.

The advent of capable computational tools and advanced fast response probes in recent years, enabled detailed time-dependent investigations of the flow field in annular turbine cascades. Porreca et al. [53] conducted a comparative numerical and experimental study on a research axial turbine with full and partial shrouds in order to address the effect of the leakage flow on the interstage flow field and the overall turbine performance. Pullan [54] studied the three-dimensional stator and rotor interactions in a turbine using steady and transient experimental measurements and numerical simulations. Behr et al. [55] experimentally investigated the unsteady flow mechanisms of tip leakage across the rotor blade of an unshrouded axial turbine. Gaetani et al. [56, 57] and Persico et al. [52] experimentally investigated the three dimensional flow field in a high pressure turbine with two stator-rotor axial gaps. They conducted time-averaged as well as unsteady measurements with variable stator and rotor blade loadings, and showcased the time resolved evolution of the secondary flow structures, as illustrated in Fig.1.6. Schobeiri et al [58, 59] and Abdelfattah and Schobeiri [60] examined the deviation of computational results in comparison to comprehensive experimental measurements on multiple sets of axial turbine stages with varied blade designs. Both steady and unsteady Reynolds-averaged Navier-Stokes (RANS and URANS) simulations proved to poorly match the experimental measurements mainly due to the inability of the transition models available to accurately predict the different flow parameters in the highly complex flow field of low aspect ratio axial turbines.

1.3 Research Objectives

The scope of this research programme includes experimental, analytical and numerical investigations to better understand and model the various mechanisms of aerodynamic losses in axial turbine flows. The findings are used to improve empirical loss correlations that are implemented in a quasi-viscous flow code to predict performance of axial turbines. Measurements are also compared to CFD results to gauge the adequacy of the numerical solver and turbulence models implemented. These tasks are summarized as follows:

1.3.1 Linear Blade Cascade Facility (2D Flow Measurements)

- Modify an existing linear blade cascade test facility to accommodate flow incidence and stagger angle variation measurements for two distinct stator blade designs. Fabricate individual blades, assemble test facility and instrument with RTD, Prandtl and five-hole probes.

- Fully upgrade the five-hole calibration facility, instrument with required pressure and temperature taps and accurately align for axial flow. Calibrate multiple five-hole probes for different flow conditions (Mach Numbers) and generate calibration subroutines to be implemented in data analysis programs.

- Measure total temperature and total and static pressures upstream of the cascade inlet line, and use a linearly traversed five-hole probe at the exit plane of the cascade to measure and resolve the full flow field components. Also measure the pressure distribution around instrumented center blade at varying test conditions.

- Calculate the total pressure loss and the wake momentum deficiency thickness as function of test parameters, and use results to calibrate empirical correlations for design and off-design profile and mixing losses.

1.3.2 Two-Stage Axial Turbine Facility (3D Flow Measurements)

- Assemble an existing research turbine with two rows of newly designed bowed stator and twisted rotor blades. Instrument test rig with five-hole pressure probes and pressure and temperature rakes and setup seven-axis traversing system used in data acquisition.

- Conduct extensive performance tests at design and off-design conditions by varying the mass flow rate through the turbine while maintaining constant rpm speeds, and varying the rpm speeds of the turbine while maintaining constant mass flow rates. Data from all sets can be condensed to generate universal nondimensional operating parameter (u/c_o) that describes the full performance map of the turbine.

- Conduct interstage tests at design conditions using extensive data acquisition grid to fully describe the flow field of the second stage of the turbine. This data can be used to track the formation of secondary loss vortices and to visualize the effect of blade wakes on the flow patterns. Circumferential averaging of the measured data yields detailed hub to tip distributions of the various flow parameters.

- Compute the radial distribution of total pressure loss and efficiency for the stator and rotor rows of the second stage, and use results to calibrate empirical correlations for endwall secondary losses.

1.3.3 Streamline Curvature Method Analysis

- Use a quasi-viscous streamline curvature method code (SCM) that implements the standard radial equilibrium equation. Update profile, secondary, trailing-edge and tip leakage loss models with improved formulation calibrated by experimental findings. Provide additional loss corrections to account for variations in blade surface friction and flow Reynolds number.

- Numerically simulate using both existing and improved SCM code the flow through the current two-stage turbine as well as previously tested three-stage turbine with bowed blades. Compare outcome to experimental measurements and CFD results, and recommend further improvements to loss correlations to better capture the flow nature, and accurately predict the machine performance.

2. TWO-STAGE HIGH PRESSURE AXIAL TURBINE STUDY[*]

Momentum transfer in high pressure axial turbine stages occurs through flow expansion in stator and rotor rows of low aspect ratio blades. This leads to development of adverse secondary flows at endwall regions that contribute significantly to entropy generation in addition to the two-dimensional losses produced by the boundary layer development around the blade profile. The current study aims at experimentally quantifying these losses through detailed flow measurements conducted at the Turbine Performance and Flow Research Laboratory of a two-stage high pressure turbine with bowed stator blades and twisted rotor blades. The research turbine rig is equipped with an interstage five-hole probe traversing system and instrumented with a set of multiple rakes of pressure and temperature probes, a high precision torque meter and an integrated venturi section. The machine is operated at a wide range of design and off-design rotational speeds ranging from 1750 rpm to 3000 rpm and pressure ratios varied between 1.16 and 1.45. Global measurements enable the machine efficiency to be accurately calculated for each operational test condition, while detailed interstage flow measurements generate the total pressure loss radial distribution for the second stage stator and rotor blade rows. An examination of the detailed flow patterns establishes the development of endwall secondary flow structures that significantly contribute to the total measured stage losses. Computational results from a concurrent investigation comprised of steady and transient numerical simulations are also presented and shown to exhibit good qualitative agreement,

[*] Part of the data reported in this chapter is reprinted with permission from "Investigating the Cause of Computational Fluid Dynamics Deficiencies in Accurately Predicting the Efficiency and Performance of High Pressure Turbines: A Combined Experimental and Numerical Study" by M.T. Schobeiri, S.A. Abdelfattah and H.A. Chibli. ASME. J. Fluids Eng. 2012;134(10):101104-101104-12. doi:10.1115/1.4007679. Copyright 2012 by American Society of Mechanical Engineers.

yet significant quantitative deviation from experimental measurements. This discrepancy is attributed to the inadequacies of the currently used dissipation and transition models.

2.1 Experimental Facility

The experimental investigation was carried out on an existing research axial turbine facility at the Turbomachinery Performance and Flow Research Laboratory (TPFL) of Texas A&M University. This research facility (Fig. 2.1) which employs a modular platform was established by Schobeiri in 1999 to conduct aerodynamic and heat transfer flow measurements in modern axial gas and steam turbine designs. Up to three stages of low, medium or high pressure turbines with a wide range of blade numbers, designs and configurations can be experimentally accommodated. Early investigations of aerodynamic performance of leaned and cylindrical bladed three-stage steam turbines were conducted [51] using detailed five-hole measurements of interstage planes. These was followed by advanced heat transfer and film cooling

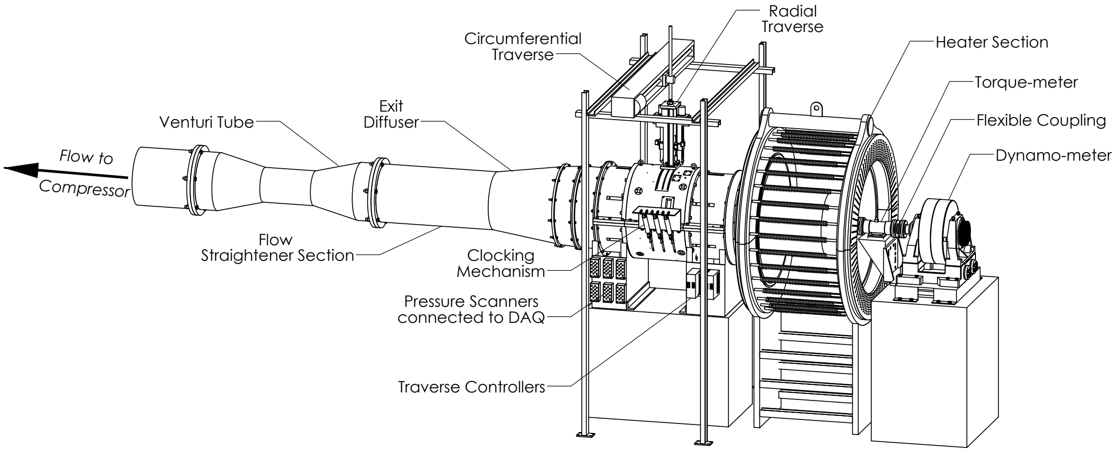


Figure 2.1: A schematic rendering of the research turbine facility at TPFL showing the main components, reproduced with permission from Schobeiri et al. [58]

measurements on a rotating gas turbine platform using optical pressure-sensitive and temperature-sensitive paint techniques [61, 62]. Most recently the facility has been modified to allow a detailed study of the effect of endwall contouring on the aerodynamic performance and the heat transfer and film cooling effectiveness of a variety of blade and platform cooling configurations [63, 64, 65].

Figure 2.2 shows the main components of the research facility: a frequency controlled 300 kW electric motor, a three-stage centrifugal compressor, a 150 kW eddy-current dynamometer, a 200 kW electric heater section and an extensively instrumented three-stage axial turbine platform devised with a high fidelity data acquisition system and automatically controlled multi-probe traversing system.

2.1.1 Compressor and Dynamometer

A three-stage centrifugal low pressure compressor that is housed in an external enclosure outside the test cell and connected via a straight pipe to the discharge section of the turbine test facility draws a maximum air volume flow rate of 4 m^3/sec

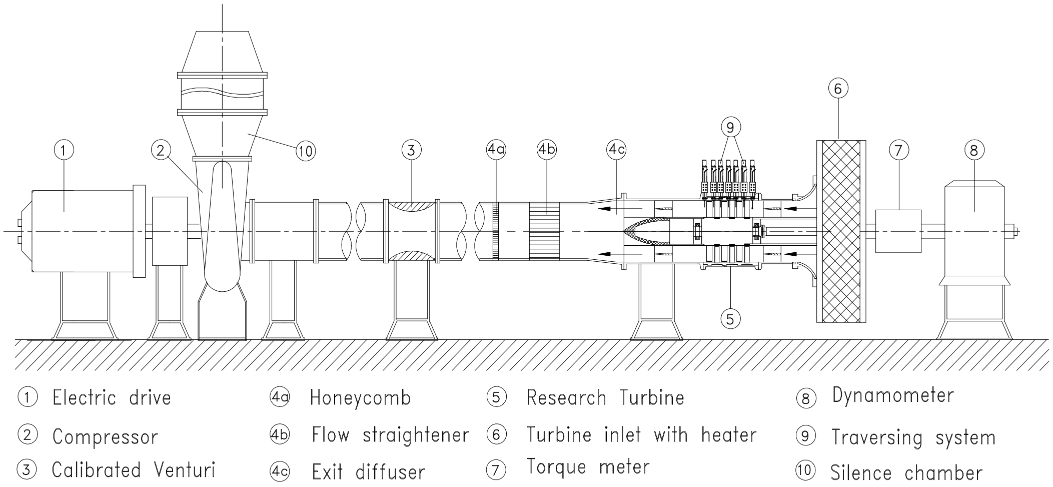


Figure 2.2: A cross section of the research turbine facility at TPFL with the main components labeled, obtained from Schobeiri et al. [50]

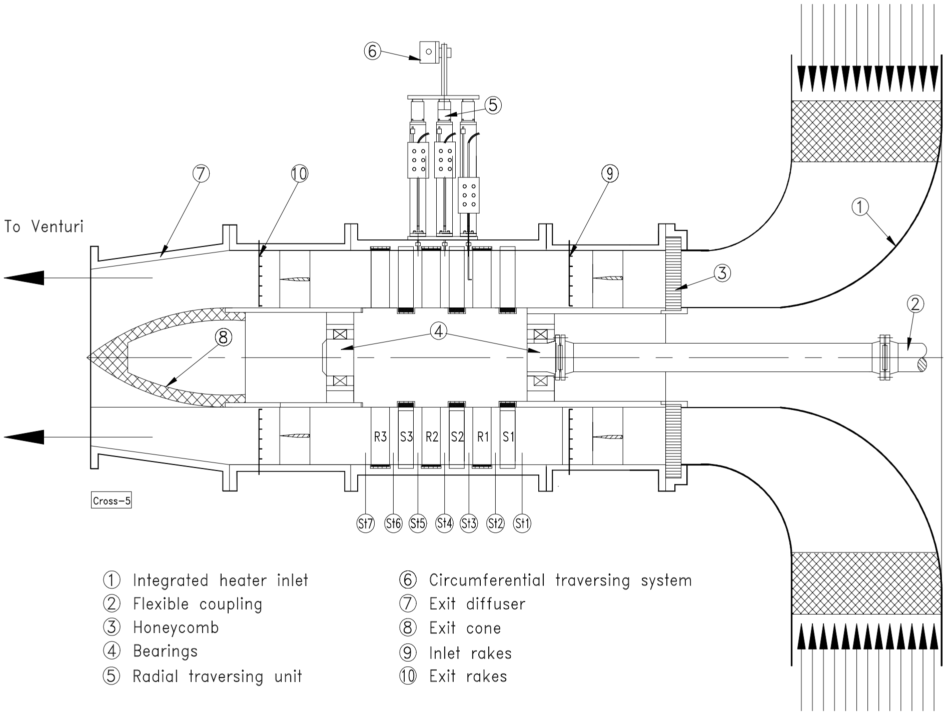


Figure 2.3: A schematic of the turbine test section, obtained from Schobeiri et al. [50]

at a maximum pressure drop of 55 kPa through the test section. In this configuration the inlet to the turbine is maintained at atmospheric pressure, while the pressure across the entire test section drops to sub-atmospheric levels as the flow expands through the turbine stages. A variable frequency control unit modulates a 300 kW electric motor that drives the compressor with a range of 0–65 Hz, needed to adjust the mass flow rate at the turbine suction.

The research turbine generates 0–110 kW that is offloaded by a 150 kW eddy-current low inertia dynamometer coupled to the engine shaft. The dynamometer which dissipates the resulting power through a water-circuit heat exchanger, has a maximum capacity of 500 N.m at a maximum speed of 8000 rpm. It is actively monitored via Texcel V4-EC controller, and can govern the turbine rotational speed with an accuracy of ±1 rpm.

2.1.2 Inlet Section Heater

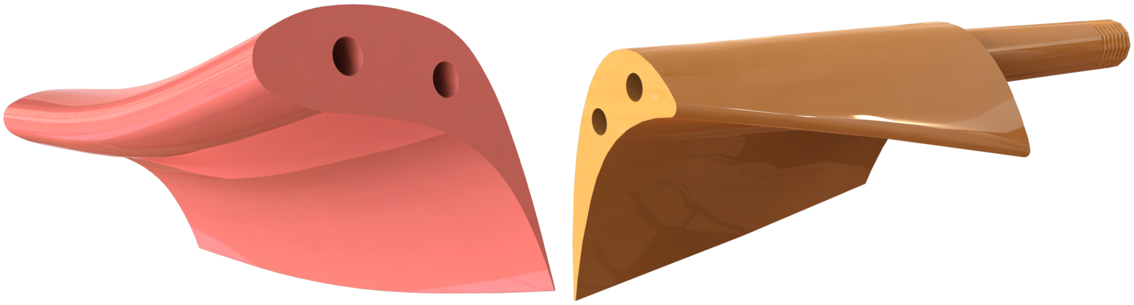
In order to avoid fluctuations of flow temperatures at the turbine suction owing to the the open circuit configuration, an electronically controlled 200 kW electric heater is integrated into the inlet section. The inlet section consists of an oversized radial intake nozzle that incorporates crossflow heating rods staggered circumferentially in the flow passage. Maintaining a constant $45 - 55°\text{C}$ inlet temperature establishes uniform turbine test conditions and prevents condensation of humid air and subsequent icing downstream of the turbine section. As the heated air accelerates through the inlet nozzle, it passes through an annular honeycomb section that acts as a flow straightener upstream of the first stator row of the turbine as shown in Fig. 2.3.

2.1.3 Turbine Component and Blading

At the core of this experimental facility (Fig. 2.1) is a modular high pressure turbine section with a versatile configuration that allows a setup of up to four test stages with a variety of blading options ranging from unshrouded zero degree reaction cylindrical designs to fully shrouded high deflection 3D designs. The turbine section incorporates stator rings that can be externally clocked using a set of built-in hydraulic pistons, allowing circumferentially offsetting the stator rings relative positions

Item	Specification	Item	Specification
Stage number	$\text{N} = 2$	Pressure ratio	$\text{pr} = 1.16 - 1.45$
Hub radius	$\text{r}_{\text{hub}} = 280.0$ mm	Speed range	$\text{n} = 1750 - 3000$ rpm
Tip radius	$\text{r}_{\text{tip}} = 343.5$ mm	Power	$\text{P} = 26 - 96$ kW
Blade height	$\text{h} = 63.5$ mm	Mass flow rate	$\dot{\text{m}} = 2.56 - 3.84$ kg/sec
Blade number	Stator 1, 2 = 66	Blade number	Rotor 1, 2 = 63

Table 2.1: Specifications of the research turbine section, reproduced with permission from Schobeiri et al. [58]



(a) Stator blade

(b) Rotor blade

Figure 2.4: Stator blade with compound lean and 3D twisted rotor blade, reproduced with permission from Schobeiri et al. [58]

to achieve higher performance. The rotor blades are attached to the rotor cylinder which is connected to the shaft via two locking mechanisms and is seated onto two heavy duty bearings that are housed inside the casing. For the current study, the turbine is configured with two shrouded high pressure steam turbine stages. Table 2.1 provides a summary of the dimensions and the operational range of the turbine. The turbine blades implemented in this rig were carefully designed for adverse off-design operating conditions without suffering major losses. Table 2.2 lists the geometric specifications of the bowed stator blade with symmetric compound lean (Fig. 2.4a) and the twisted 3D rotor blade with hub to tip taper and strong root deflection (Fig. 2.4b). Typical velocity triangles at the inlet and exit planes of the second stator row for the midspan section of the turbine are shown in Fig. 2.5 .

2.2 Turbine Instrumentation

2.2.1 Performance Instrumentation

The research turbine is instrumented with two sets of four probe rakes that are positioned upstream of the first stator row (station 1) and downstream of the last rotor row (station 7) to accurately measure and average the suction and discharge

Item	Stator	Rotor
Configuration	3D compounded lean	3D with sweep
Chord length	$c_{\text{hub}} = 37.3$ mm	$c_{\text{hub}} = 36.8$ mm
	$c_{\text{tip}} = 42.6$ mm	$c_{\text{tip}} = 38.6$ mm
Solidity	$\sigma_{\text{hub}} = 1.40$	$\sigma_{\text{hub}} = 1.32$
	$\sigma_{\text{tip}} = 1.31$	$\sigma_{\text{tip}} = 1.14$
Zweifel number	$\Psi_{\text{hub}} = 0.34$	$\Psi_{\text{hub}} = 0.72$
	$\Psi_{\text{tip}} = 0.42$	$\Psi_{\text{tip}} = 0.60$

Table 2.2: Specifications of the turbine blades, reproduced with permission from Schobeiri et al. [58]

flow conditions of the test section. The rakes are positioned symmetrically within the turbine annular space at each of the two stations, with a 10° circumferential offset to prevent any interference between any two corresponding inlet and exit rakes. Each rake consists of four Pitot tube total pressure probes spaced equidistantly in the radial direction and three total temperature J-type thermocouple sensors. The rakes are designed to have a streamlined airfoil shape with a rounded leading edge, and sharp trailing edge that helps reduce the wake thickness propagated downstream along the turbine passage. Wall pressure taps that measure the static pressure at every axial station are arranged at the top and bottom halves of the casing and on the three T-rings. All the pressure probes used are pneumatically connected to a set of calibrated pressure PSI (Pressure System Incorporated) scanners that report the differential pressures with an accuracy of 0.05% of full scale (2.5 – 5 psi). A dedicated PSI scanner simultaneously logs the barometric pressure that is used to establish the measurement reference conditions. All temperature sensors are connected to two multi-channel Fluke high accuracy scanners.

The mass flow rate through the test section is metered by a calibrated venturi tube section that is installed downstream of the axial turbine via an annular diffuser

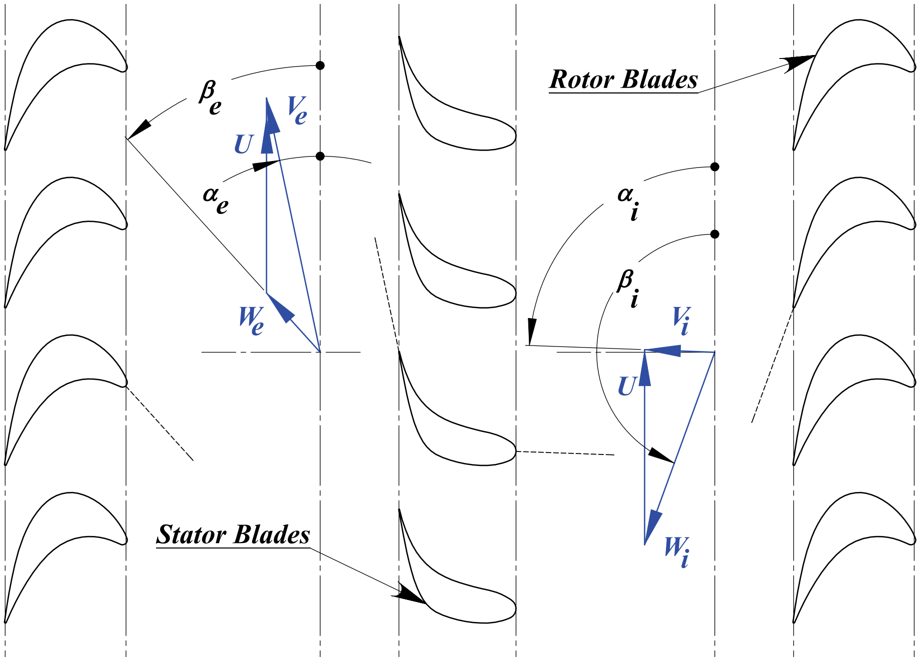


Figure 2.5: A schematic of velocity triangles at the midspan section of the second stage stator blade of the axial turbine, obtained from Sharma [66]

followed by a stainless steel honeycomb and flow straightener as shown in Fig. 2.1. The torque and the rotational speed of the turbine are constantly monitored by a high precision torque meter that is seated between the turbine and the dynamometer, and is connected to their shafts via two flexible couplings. This torque meter has a maximum rating of 8500 rpm and 678 N.m, and logs data with an accuracy of 0.02% of full scale.

2.2.2 Interstage Instrumentation

A seven-axis automatic traversing system mounted atop of the test facility, accurately positions and moves a set of three calibrated five-hole probes in the radial and circumferential directions at stations 3, 4 and 5 corresponding to the first rotor exit, the second stator exit and the second rotor exit planes, respectively. The system also allows additional radial traversing at stations 1, 2, 6 and 7. Three slots that provide

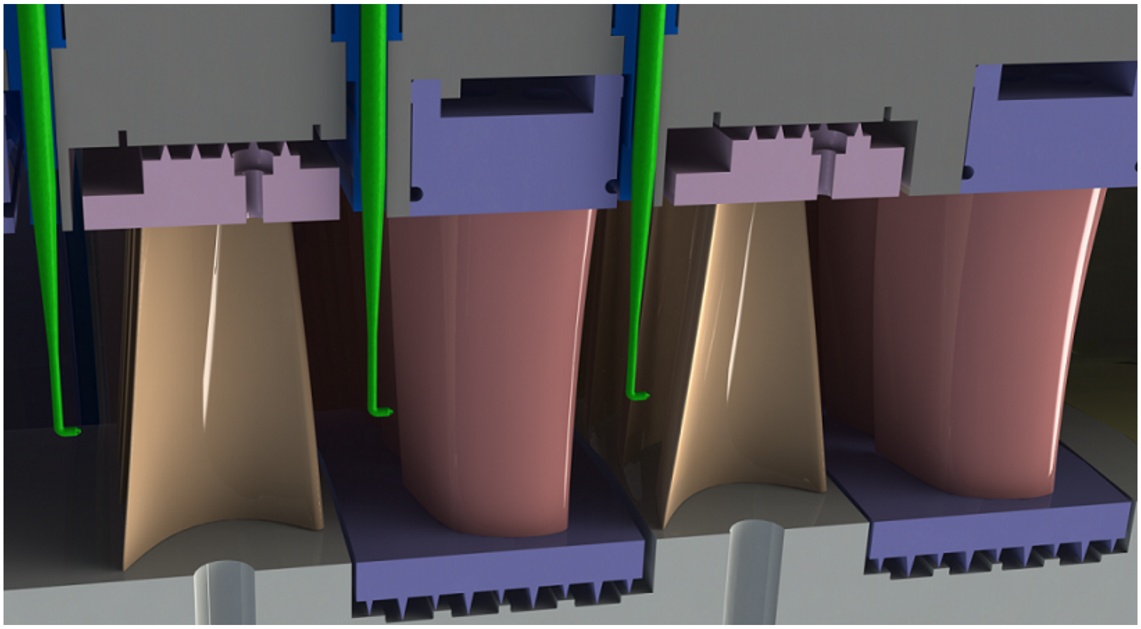


Figure 2.6: A cross section of the research turbine showing the measurement planes of the five-hole probes, reproduced with permission from Schobeiri et al. [58]

a measurement window of 90° are cut out of the casing top surface, and are sealed by a set of matching T-rings. These T-rings slide circumferentially inside the traversing slots and effectively prevent any adverse mass flow leakage. The data acquired from the five-hole probe measurements, is used to generate a detailed flow picture at each of the three stations and to estimate the spanwise distribution of the total pressure loss coefficients, and the efficiency for each blade row. Figure 2.6 shows a detailed view of the two turbine stages with the five-hole probes.

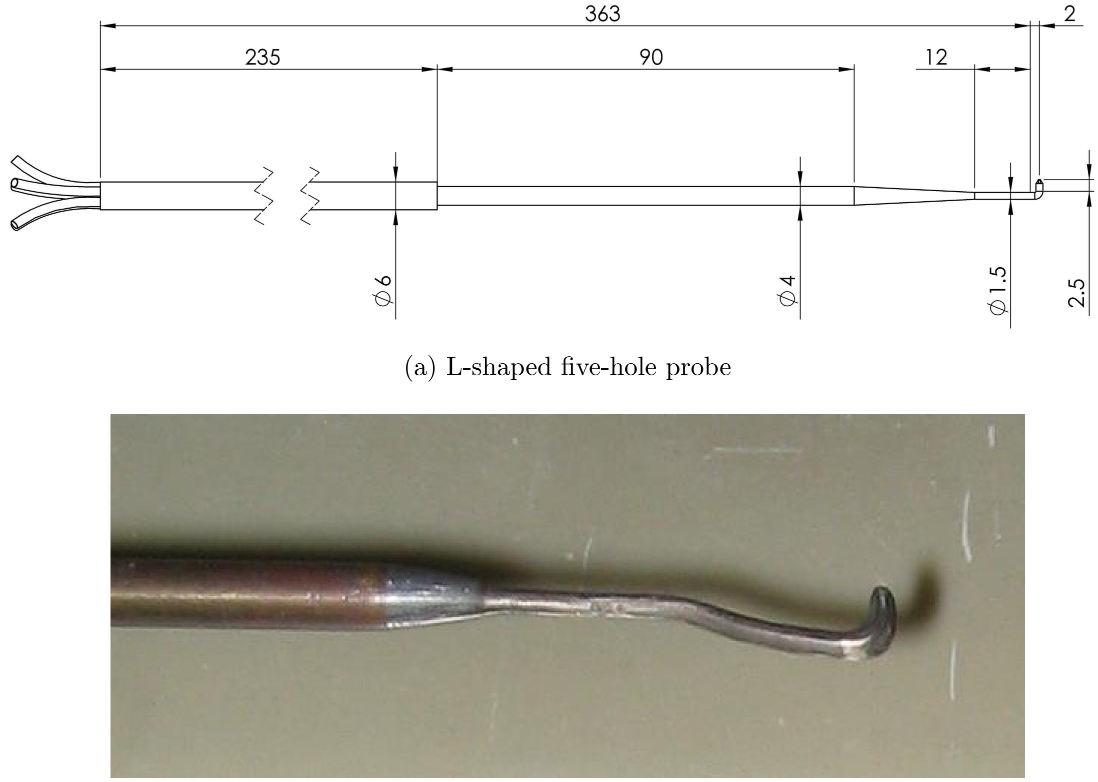
Three L-shaped five-hole probes with a tip length of 6.5 mm and a tip diameter of 1.5 mm were employed in this study. The probes were positioned at stations 3 and 5 were calibrated to low subsonic Mach numbers of $M = 0.1$ and $M = 0.15$ respectively, whereas the probe at station 4 was calibrated taking into account the slight compressibility effects at the moderate subsonic Mach number of $M = 0.3$. These probes were calibrated following the Bohn's non-nulling technique [67].

2.2.3 The Calibration of Five-Hole Probes

Five-hole probes, also known as five-axis probes are pneumatic instruments used to provide three-dimensional aerodynamic flow field measurements. When properly positioned, calibrated and coupled with accurate multi-channel pressure transducers and temperature scanners (typically using thermocouples) five-hole probe measurements can be resolved to determine the total and static pressures, velocity and flow angles. Since these instruments have low frequency response characteristics, they are best suited for steady state flow measurements. Fitting specially designed five-hole probes with built-in sensors and micro-processors has been demonstrated to extend the scope of their use into transient domains [53, 57] like unsteady turbomachinery interstage flow fields.

Five-hole probes are typically custom built of miniature tubes (brass or stainless-steel) that are fused together and machined to produce a symmetrically conical head, which is bent and positioned at a 90° angle from the circular body of the probe that houses all the tubes and extends outside the measurement domain. The head can have the standard L-shape design as shown in Fig. 2.7a, or can be axially offset from the probe body as with the cobra-shape design shown in Fig. 2.7b. Dimensions of the five-hole probes vary by application, where special attention is taken to properly scale the probe design to the measurement domain in order to reduce the interference with the flow stream to a minimum.

Two methods are employed in five-hole measurements, mainly the nulling and non-nulling techniques. The nulling technique is mostly suited for single point measurements or to applications where high flow variations are anticipated. The probe body is mounted on a frame and fitted with an automatic actuator that allows adjusting the probe angle in two planes such that the pressure is equalized on the



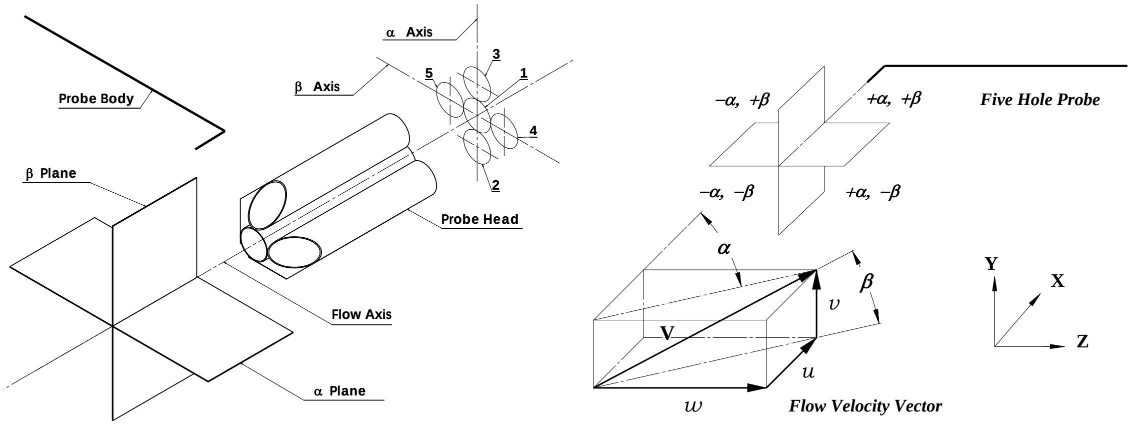
(a) L-shaped five-hole probe

(b) Cobra-shaped five-hole probe

Figure 2.7: Typical five-hole probe designs

symmetric holes at which point the probe is in line with the flow. This renders the nulling method effectively impractical for large flow domains where the setup is quite complex and the measurements are time consuming. The non-nulling technique on the other hand, requires the probe to be calibrated to account for flow pressure and direction variations before being placed in the measurement flow field at a fixed predetermined position. This allows for performing extensive multi-point flow measurements where the probe is typically radially or axially traversed across the flow domain without the need to continuously adjust the probe body or tip angles.

Numerous non-nulling calibration methodologies have been developed over the last three decades [68, 69, 70, 71], however the author will limit the following presentation to the Bohn's technique [67] which has been redeemed as the one of the most accurate and repeatable due to its iterative structure. For this purpose the probe is housed into an automatic calibration facility that utilizes an indexing mechanism to pitch and yaw the probe in one degree increments through a range of predetermined angles, while logging the five measured pressures at each increment. This angle range is typically kept to a maximum of ±40° to avoid flow separation around the probe head. Figure 2.8 shows the probe head and flow angle notations adopted. The incoming flow is kept axial by passing the same through a series of flow straighteners and screens before symmetrically accelerating the jet about the probe head using a converging nozzle. The flow conditions are maintained at steady state using an upstream flow regulating valve, and the temperature, static and total pressures of the flow are constantly monitored and logged. Five flow coefficients that relate the total and static pressures of the flow and the pitch (α) and yaw (β) angles to the five individual pressures of the probe are computed, then correlated via three-dimensional



(a) Probe hole number notations (b) Probe angle and axis notations

Figure 2.8: Five-hole probe coordinate system, obtained from Sharma [66]

curve fits to the flow angles. These coefficients are shown below:

$$Q_1 = \frac{P_4 - P_5}{P_1 - p} \quad \text{and} \quad Q_2 = \frac{P_3 - P_2}{P_1 - p} \quad \text{and} \quad Q_3 = \frac{P_1 - p}{P - p}$$
$$Q_4 = \frac{P_1 - P_4}{P - p} \quad \text{and} \quad Q_5 = \frac{P_1 - P_5}{P - p} \tag{2.1}$$

The calculated coefficients are then numerically correlated to the flow angles using high order nonlinear curve fitting as follows:

$$\alpha = f_1(Q_1, Q_2) \quad \text{and} \quad \beta = f_2(Q_1, Q_2) \quad \text{and} \quad Q_3 = f_3(\alpha, \beta)$$
$$Q_4 = f_4(\alpha, \beta) \quad \text{and} \quad Q_5 = f_5(\alpha, \beta) \tag{2.2}$$

Figure D.1 shows sample calibration curves for an L-shape five-hole probe that has been tested at a 0.2 Mach number. It is essential that the calibration takes place at a nominal flow Mach number that is close to that of the actual measurement conditions in order to minimize the error resulting from changes in flow density due to flow compressibility. The obtained numerical correlations (data fits) of the five flow coefficients are used to reduce the measured raw pressure data from the probe. This is accomplished using an iterative method that converges to the flow static and total pressures, as well as the flow actual angle of incidence in the pitch and yaw directions from the probe predetermined orientation within the test facility. Through a vectorial transformation, the absolute flow angles in a global stationary or rotating frame of reference can be calculated and the velocity components resolved accordingly. The data reduction algorithm is detailed below:

Step 1: Read five-hole probe raw data $P_1,\ P_2,\ P_3,\ P_4$ and P_5

Step 2: Calculate initial guess for $p_i = 0.8 \times P_1$

Step 3: Calculate Q_1 and Q_2 using Eq. 2.1

Step 4: Calculate α and β using Eq. 2.2

Step 5: Calculate Q_3, Q_4 and Q_5 using Eq. 2.1

Step 6: Calculate $p_{i+1} = P_1 - Q_3 \times \left[\frac{P_1-P_4}{2\times Q_4} + \frac{P_1-P_5}{2\times Q_5}\right]$

Step 7: Check if $|p_{i+1} - p_i| >$ error $\Rightarrow$ set $p_i = p_{i+1}$ & iterate to step 3

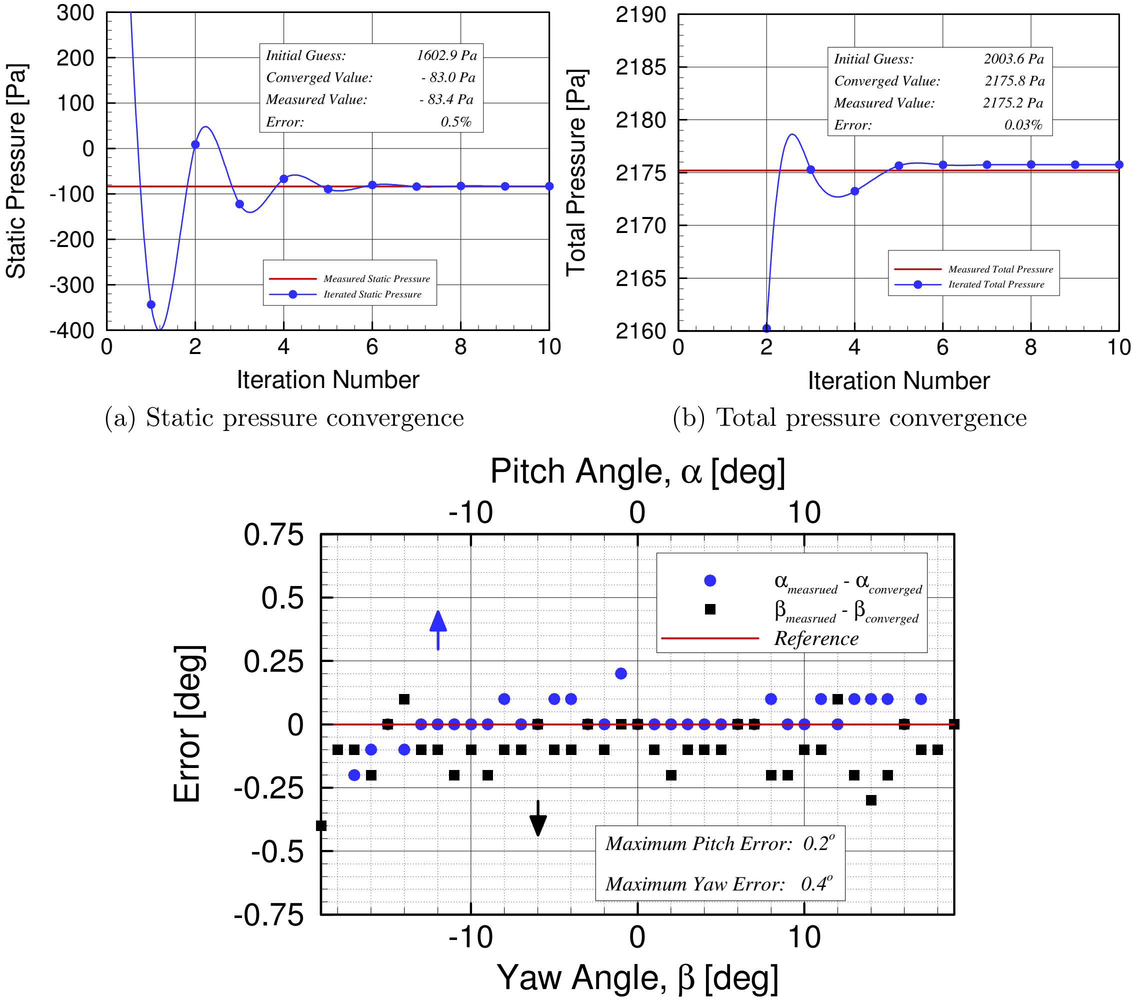
Step 8: Check if $|p_{i+1} - p_i| <$ error $\Rightarrow$ converge to solution

To demonstrate the Bohn's calibration procedure and evaluate the quality of the data reproduction, a numerical experiment was devised: Pressures logged during the calibration of a service five-hole probe at TPFL were fed as pseudo measurement pressures to the numerical subroutines, and the total and static pressures and flow angles were calculated and compared to the corresponding actual values assigned during the calibration process. Figures 2.9a and 2.9b depict a typical numerical convergence scheme of the static and total pressures, respectively. As shown, a small number of iterations leads to a solution that is in good agreement with the measured values. Note that the numerical error in this instance is within the measurement accuracy for the used pressure transducers. Figure 2.9c plots the error spread between the measured and resolved pitch and yaw angles for a calibration range of ±20°. A sample calibration subroutine source code is listed in Appendix J.1.

2.3 Data Acquisition and Analysis

As detailed earlier, measurements at three interstage planes that lie normal to turbine axis of rotation, were conducted using five-hole probes which were automatically traversed to cover the full flow field of the second stage blade rows. Figure 2.10 shows a sample measurement matrix which has a total of $N_c \times N_r$ elements representing the circumferential and radial data points, respectively. In the current investigation, the measurement grid has a total of $N_c = 61$ and $N_r = 37$, with a higher density of point distribution maintained close to the blade hub and tip regions in order to get a better resolution of the secondary flows inherent to these areas.

After inserting the probe radially as illustrated in Fig.2.11, the probe head must be rotated to achieve *nominal* alignment with the flow. Since the probe calibration algorithm numerically correlates the pressure data to pitch and yaw angles relative to the probe position (Fig.2.8), reducing the flow incidence on the probe head ensures the measured data stays within the calibration range and contributes to better con-



(a) Static pressure convergence

(b) Total pressure convergence

(c) Flow angles convergence

Figure 2.9: Sample probe data resolution using Bohn's calibration technique [67]

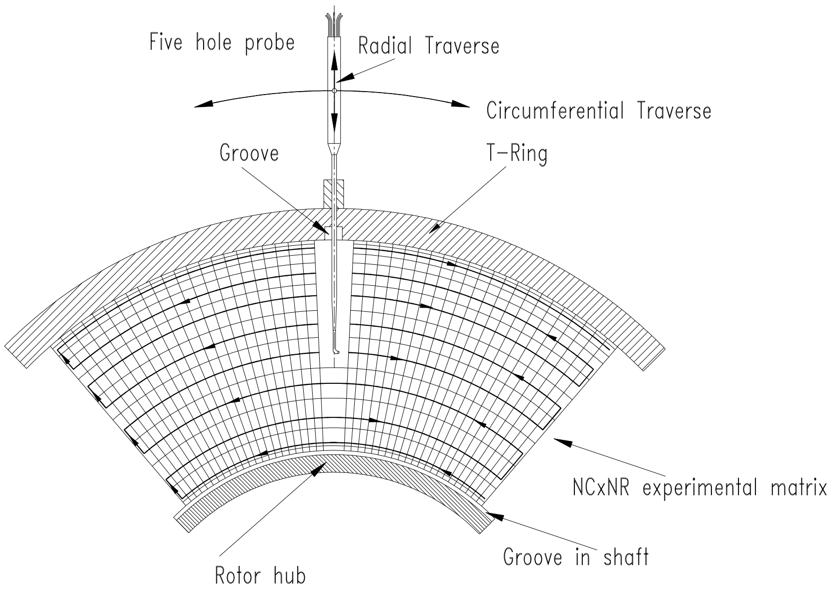


Figure 2.10: A schematic of a typical experimental measurement grid, obtained from Schobeiri et al. [50]

vergence. To account for the highly directional nature of the flow field resulting from the flow turning of stationary and rotating blade rows with exit metal angles varying from hub to tip, the probe position was adjusted at three radial locations (hub, mid and tip) for every interstage measurement station. Once the initial midspan flow alignment angle ($\beta_\circ$) is fixed, the probe is relatively offset from this reference position close to the hub and tip regions using a stepper motor that is mounted on the traversing system. The velocity components in the turbine reference coordinate system shown in Fig. 2.11 are calculated using the following spatial transformation:

$$\begin{aligned} V_r &= -V_c \cos\beta_c \sin\alpha_c \\ V_z &= V_c \cos\beta_c \cos\alpha_c \cos\beta_\circ + V_c \sin\beta_c \sin\beta_\circ \\ V_\theta &= V_c \cos\beta_c \cos\alpha_c \sin\beta_\circ - V_c \sin\beta_c \cos\beta_\circ \end{aligned} \tag{2.3}$$

where $\beta_\circ$ denotes the probe initial flow alignment angle. V_c, α_c and β_c refer to the velocity, pitch and yaw angles resolved in the probe frame of reference shown in Fig.2.8. The flow absolute, relative and meridional angles (α, β and γ respectively) in the turbine $r - \theta - z$ coordinate system can then be determined accordingly.

The use of five-hole probes is suitable for the steady state aerodynamic measurements required in the scope of this study. These probes are intrinsically characterized by a low frequency response, which *averages out* the local unsteadiness of the data, fluctuating at the basic passing frequency of the rotor which varies between 2500 and 3000 Hz. A set of 40 samples were acquired at each data point that falls within 20% of the endwall regions (blade hub and tip) and 20 samples at data points located elsewhere, over a duration of 10 seconds per set. Time averaging of the raw data is required before the spatial data analysis can take place. The LabVIEW scripts used to control the traversing system and automate the data acquisition for this test procedure are shown in Figs. H.5 and H.6.

Due to the large number of circumferential and radial data points in the inter-

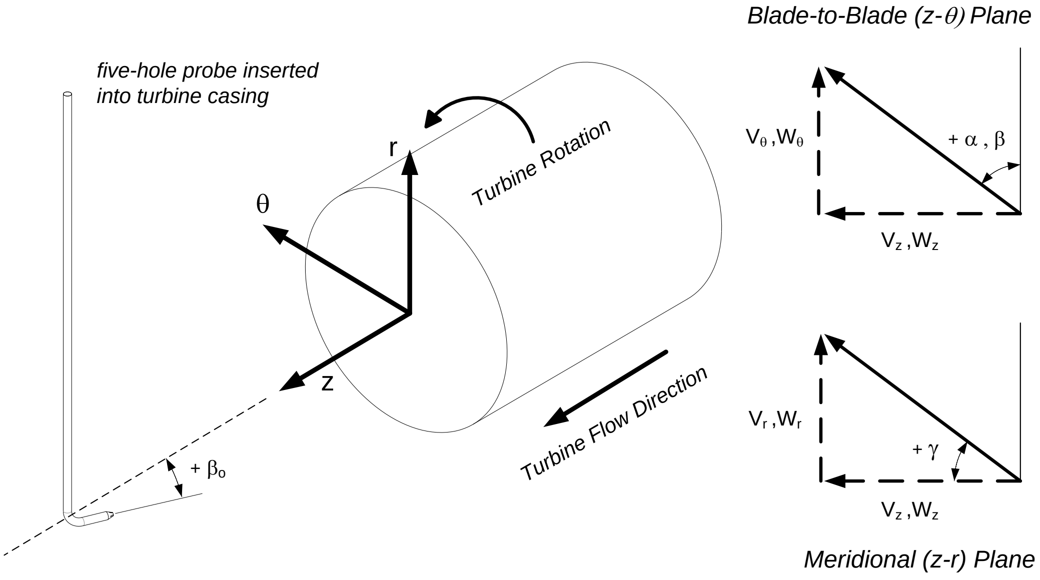


Figure 2.11: The reference coordinate system of the research turbine

stage probe traversing matrix, and allowing the flow to stabilize at each measurement point, a full turbine aerodynamic test duration is rather time intensive. Variations in atmospheric conditions consequently affect the acquired data as the test facility utilizes an open loop air intake system. As such all data acquired needs to be locally corrected to a standard pressure reference. Barometric pressure and relative humidity values are continuously logged throughout the test and used to correct the air density and viscosity. Analysis of the corrected raw data follows using the calibration subroutines for each of the three five-hole probes to resolve absolute and relative pressures, velocities and flow angles at their corresponding stations. Upstream and downstream pressure and temperature data is averaged for the duration of the test and the mass flow rate is calculated using the venturi tube calibration curves. The measured torque and rotational speed determine the turbine net output power after accounting for the bearings and windage losses.

In order to produce the radial distribution from hub to tip of the various flow parameters at every measurement station, the data is arithmetically averaged in the circumferential direction. Traupel [72] extensively reviewed the significance of appropriate averaging techniques in turbomachinery, and Dzung [73] introduced the the consistent averaging technique that yields inherently concordant results at a plane under inhomogeneous flow conditions. This technique was subsequently enhanced by Schobeiri et al. [50] to obtain consistently averaged data for distributed quantities in the radial direction.

The mass flow rate at every measurement station is calculated using the density and axial velocity radial profiles from hub to tip, and matched to the global mass flow rate determined from the venturi tube. This enables *fine-tuning* the initial estimate of the probe flow alignment angle ($\beta_\circ$) through an iterative procedure. The flow streamlines through the second turbine stage are then numerically mapped using mass flow rate conservation in the meridional plane and used to evaluate the performance of the stator and rotor rows. Entropy generation along a streamline leads to a drop in the measured total pressure in a stationary blade row, and a drop in relative total pressure for a rotating blade row. The total pressure loss coefficient profiles for the stator and the rotor can be calculated as follows:

$$\zeta_s = \frac{P_3 - P_4}{P_3 - p_4} \text{ (stator)} \quad \text{and} \quad \zeta_r = \frac{P_{r_4} - P_{r_5}}{P_{r_4} - p_5} \text{ (rotor)} \tag{2.4}$$

Table A.1 lists the uncertainty bandwidth in all the measured and calculated values that are presented in this study. The analysis method of Kline and McKlintock [74] has been adopted in these calculations. Note that to establish repeatability of all experimental data, three measurements of each data set were collected and analyzed. The FORTRAN 77 source code of the data analysis program is listed in Appendix J.2.

2.4 Numerical Treatment

A concurrent numerical investigation of the two-stage turbine flow was completed and presented by Schobeiri et al. [58]. Simulations using the commercially available ANSYS-CFX finite volume solver were run on the Texas A&M supercomputing facility. Both steady and unsteady Reynolds-averaged Navier-Stokes (RANS and URANS) simulations were performed in an effort to get a detailed picture of the flow field. The shear stress transport (SST) turbulence model developed by Menter [75] was used with a built-in proprietary transition model. Boundary conditions that match the experimental measurements at the suction (total pressure) and discharge (static pressure) at different rotational speeds were imposed. An ideal gas model was assumed for the working medium fluid. A mesh sensitivity study was conducted and a compromise between resources and required accuracy resolution was deemed

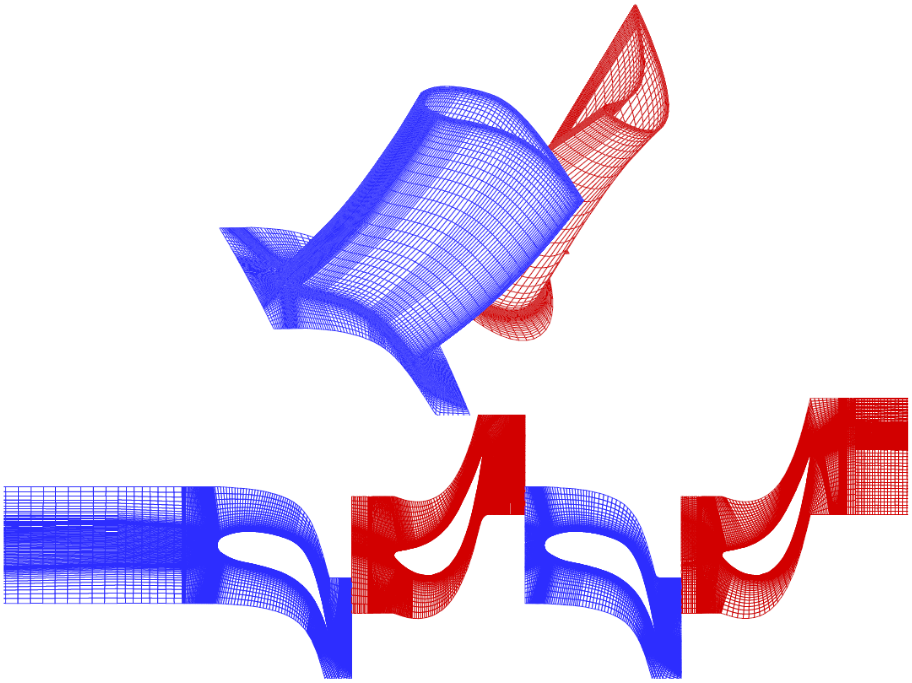


Figure 2.12: The numerical grid of the flow domain, reproduced with permission from Schobeiri et al. [58]

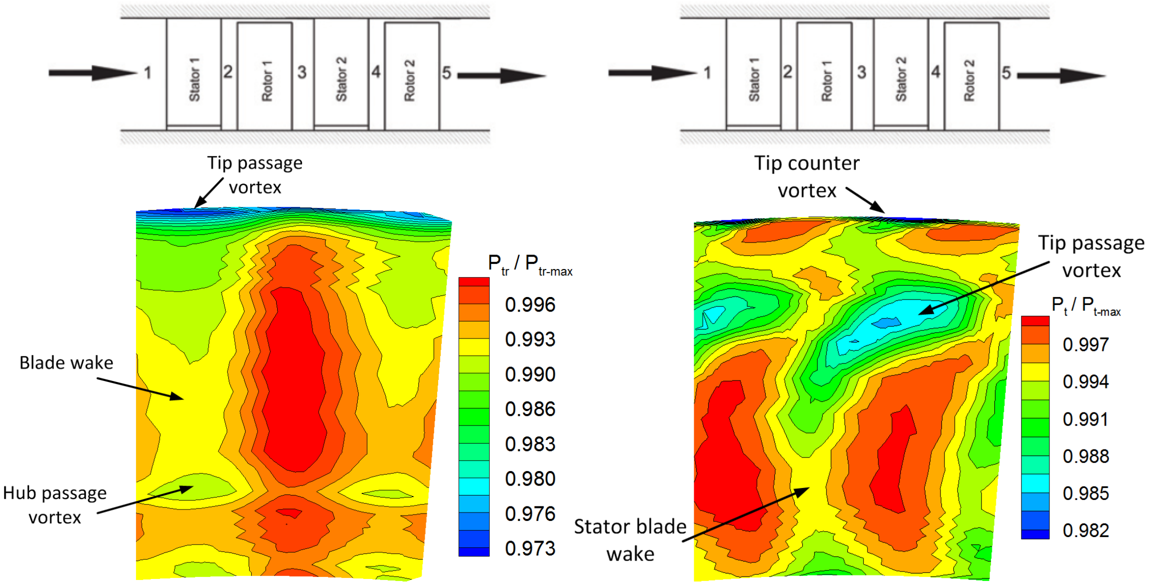
best at a mesh size of 4.6×10^6 elements. Figure 2.12 shows the computational domain with the required numerical interfaces for the two turbine stages analyzed. For the steady simulations each turbine row was modeled as a separate domain with a stage interface (mixing plane) imposed between the stationary and rotating subdomains. For the URANS simulations on the other hand, a fully unsteady sliding mesh interface was imposed between the stationary and rotating grids.

2.5 Results and Discussion

2.5.1 Detailed Blade Row Measurements

In the following, the experimental results downstream of the stator and rotor blades are presented for the turbine operational point of $n = 3000$ rpm and $pr = 1.44$. A close examination of the interstage steady state flow patterns is essential to characterize the aerodynamic performance of the current blade designs, and qualify the role of the stator-rotor interactions as well as the secondary flow structures in the measured loss generation.

Figure 2.13a reports the dimensionless total relative pressure distribution at station 3, locally normalized by the maximum pressure value measured in that plane. Four distinct flow regions are visible in this figure. A freestream domain is observed in the center region which extends radially from the hub surface to about 80% of the blade height, and circumferentially from about 25% to 75% of the blade channel. A region of gradually reduced total relative pressure develops around the freestream domain in the circumferential direction due to the stator blade wake which is convected through the rotor channel. At the tip section of the wake region, a steep drop in total relative pressure is evident, indicating the activity of a tip passage vortex which interacts with the low momentum boundary layer stream at the shroud end-



(a) Relative total pressure at station 3

(b) Total pressure at station 4

Figure 2.13: Normalized total pressure contours at blade row exit planes

wall. This structure seems to compressed radially against the shroud surface under the effect of the outward radial velocity acting close to the tip region of station 3. A fourth flow domain characterized by a local drop of total relative pressure is evident at the the bottom of the wake region within about 30% of the hub surface. This is indicative of the presence of the secondary flow hub passage vortex in this region. Although the intensity of this structure is about one third of that of the tip passage vortex, it exhibits a rather strong circumferential penetration that divides the freestream into two radial subregions. At the bottom of the stator wake region, another area of decreased total relative pressure develops in direct proximity to the hub endwall where a low energy boundary layer dominates the flow.

Dimensionless total pressure contours at station 4 are shown in Fig. 2.13b, with four regions that define the flow scene. A freestream lossless domain is found to occupy 80% of the blade passage width from the hub surface up to 60% of the blade height. A thin moderately reduced momentum flow region develops on the perimeter of the freestream and extends radially from the hub surface to the tip

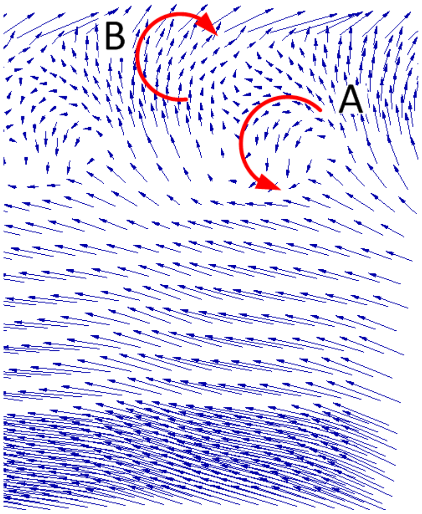


Figure 2.14: Secondary flow vector plot at station 4

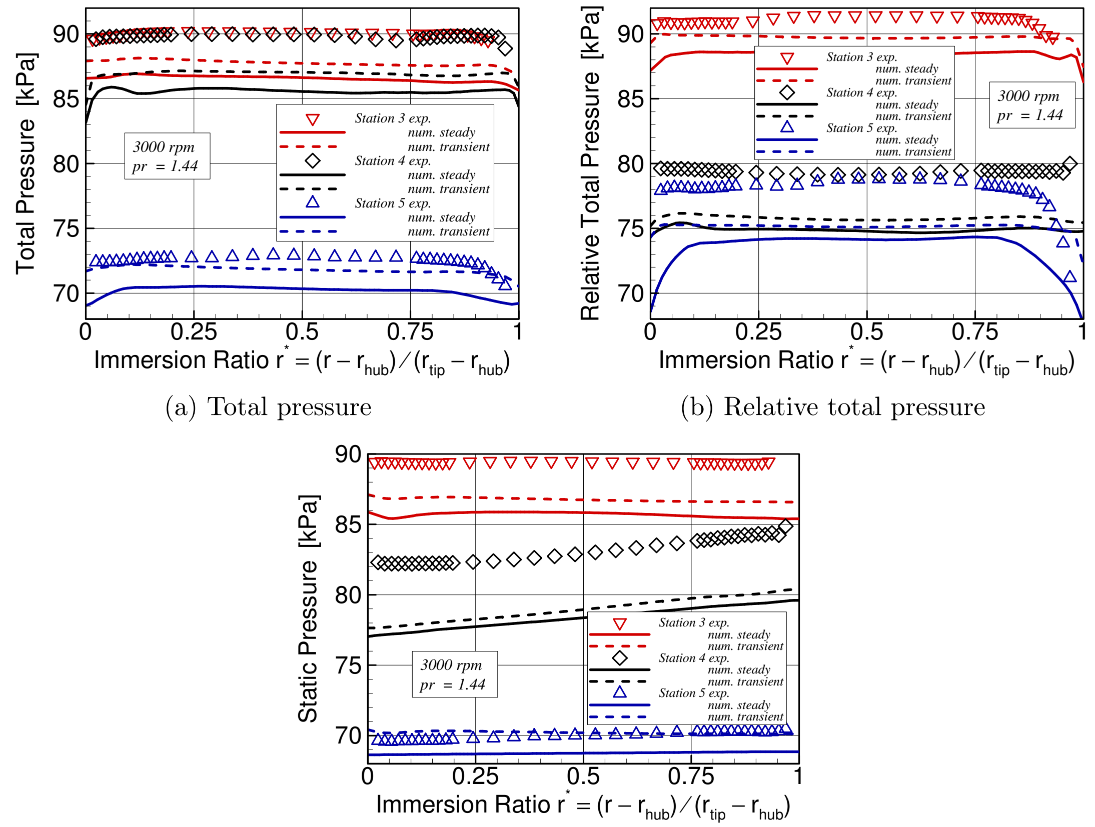
endwall following the wake of the stator blade trailing edge. Between 60% and 80% of the blade height, a region of abrupt total pressure drop dominates the full width of the blade channel. Inspecting the vortical secondary flow patterns presented in Fig. 2.14 at location (A), a tip passage vortex structure can be clearly identified. The placement of this structure close to the upper end of the blade midspan, is consistent with the strong radial pressure gradients inside stator row channels. Furthermore, Fig. 2.14 shows a tip vortex structure that is active at location (B), and that rotates in a counter direction to the tip passage vortex. This is evidenced by the thin area of steep total pressure drop that can be observed at the tip of the passage in Fig. 2.13b. This vortex seems to be radially compressed against the tip endwall by the accelerated outward radial velocity above 80% of the blade span. A full presentation of the turbine contour plots is provided in Appendix B.

2.5.2 Averaged Blade Row Measurements

2.5.3 Absolute and Relative Pressures

Figure 2.15 shows the radial distribution of the absolute and relative total pressures and the static pressure at the three interstage measurement stations. The extent of the agreement between the total pressure profiles at stations 3 and 4 in Fig. 2.15a shows the positive effect of the circumferential lean of stator blades on reducing the secondary losses at the hub and tip regions. Reviewing the results of the steady and transient numerical simulations reveals that both approaches fail to match the total pressure measurements across the stator blade row. There is a quantitative under-prediction of total pressure by 2-5% as well as a qualitative mismatch of the entropic flow behavior characterized by a magnified drop in total pressure between stations 3 and 4.

The secondary losses caused by the excessive vortical flow activity in the endwall



(a) Total pressure

(b) Relative total pressure

(c) Static pressure

Figure 2.15: Radial distribution of measured pressures at three interstage stations with steady and transient numerical simulation results, reproduced with permission from Schobeiri et al. [58]

regions are concentrated close to the rotor tip as shown in Fig. 2.15b with a pronounced drop in the measured relative total pressure between stations 4 and 5 above 75% of the blade height. Both steady and transient simulations under-predict the magnitude of the relative total pressure at the exit plane of the rotor blade by 3-5%, with the transient simulation results matching the experimental profile trend in the hub region and the steady results showing better agreement close the blade tip.

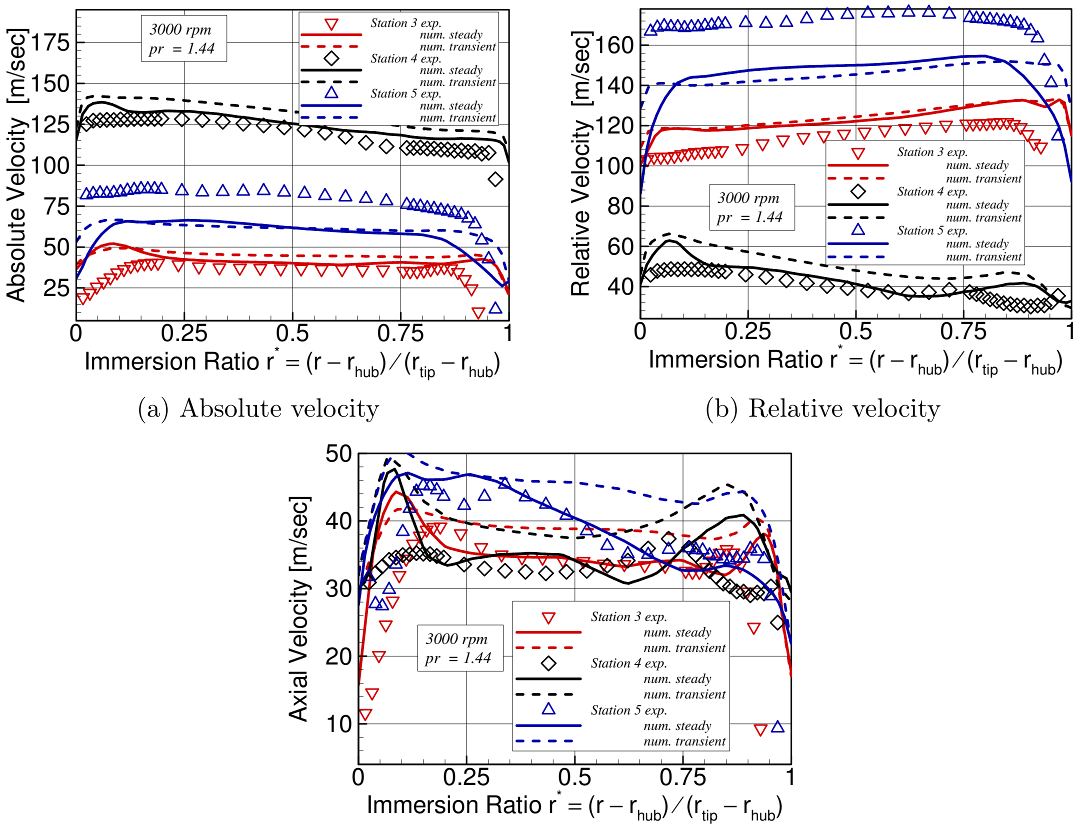
The static pressure radial profiles shown in Fig. 2.15c exhibit similar discrepancy between measurements and computations, with a significant deviation at the stator exit plane. This is attributed to the failure of the numerical solver to accurately predict the boundary layer development and transition in the turbine flow path, and the subsequent mismatch in the dynamic pressure profiles.

2.5.4 Absolute and Relative Velocities

The radial distributions of the absolute, relative and axial velocity vector components are shown in Fig. 2.16. A significant under-prediction of 5-10% is observed between numerical results and experimental measurements at station 5 for both the absolute velocity (Fig. 2.16a) and the relative velocity (Fig. 2.16b) profiles; in contrast, stations 3 and 4 exhibit a moderate over-prediction of the computational results. Furthermore, it is noted that the steady simulations yield a better agreement with measurements in comparison to transient results, especially in the endwall regions as evidently seen in Fig. 2.16c where the mismatch in the axial velocity profiles indicates the failure of the numerical solver to accurately predict the mass flow rate at the experimentally prescribed pressure ratio.

2.5.5 Absolute and Relative Flow Angles

Figure 2.17 displays the radial distributions of the absolute, relative and meridional flow angles resolved in the turbine coordinate system explained in Fig. 2.11. As with the absolute and relative velocity vectors, substantial differences between measurements and numerical results are observed, particularly in the endwall regions of the flow regime. It is also noted that the transient simulations do not offer significant quantitative or qualitative improvements over the results of the steady computations. Figure 2.17c emphasizes the deviations in the numerically computed meridional flow angles relative to the measured values, which highlights the corresponding mismatch in the axial velocity profiles observed earlier. Deviations in the calculations of flow angles could lead to positive or negative flow incidence on stationary and rotating



(a) Absolute velocity

(b) Relative velocity

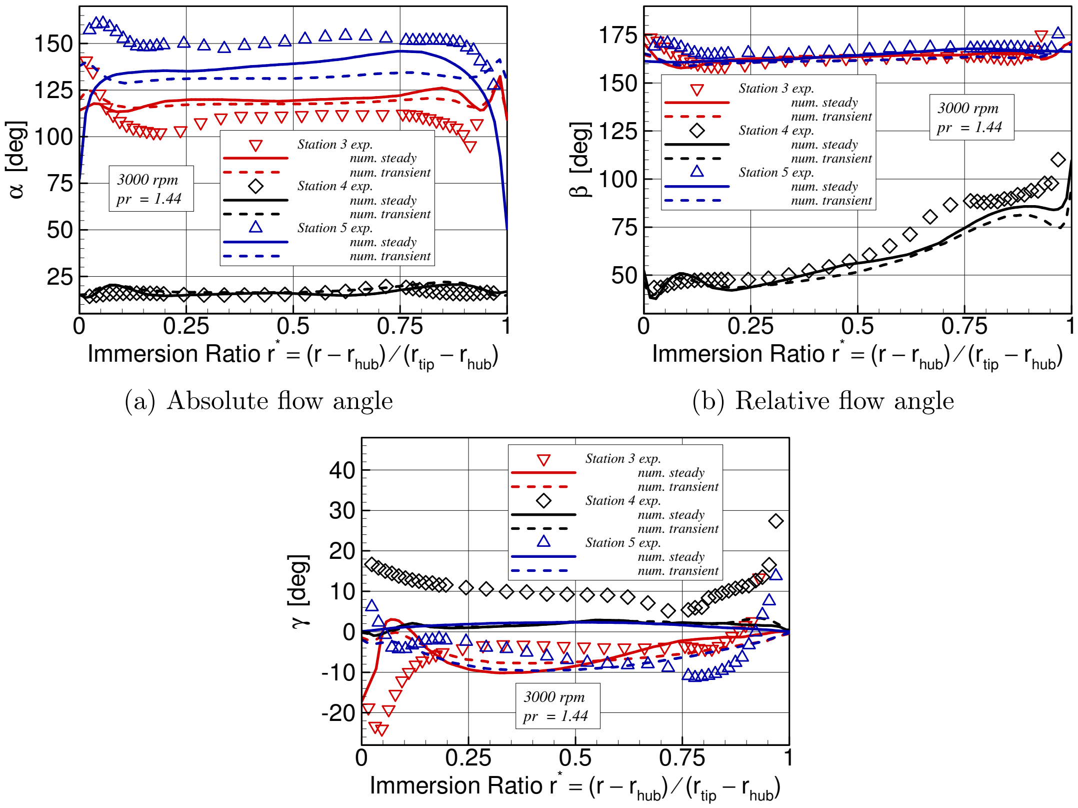
(c) Axial velocity

Figure 2.16: Radial distribution of measured velocities at three interstage stations with steady and transient numerical simulation results, reproduced with permission from Schobeiri et al. [58]

blade rows, and cause a bias in the evaluation of blade profile and mixing losses.

2.5.6 Total Pressure Loss Coefficients

The total pressure loss coefficients for the stator and rotor rows are plotted in Fig. 2.18 as a function of the immersion ratio. The stator blade row exhibits a mostly uniform loss distribution between 5% and 95% of the blade height. This pattern is only interrupted between 65% and 80% of the blade height, where an area of locally elevated losses is detected. This increase in loss generation is attributed to the tip passage vortex that was discussed earlier at the same radial position. At the hub,



(a) Absolute flow angle

(b) Relative flow angle

(c) Meridional flow angle

Figure 2.17: Radial distribution of measured flow angles at three interstage stations with steady and transient numerical simulation results, reproduced with permission from Schobeiri et al. [58]

the endwall boundary layer contributes to the rise of the calculated losses mainly due to viscous dissipation. A tip region of substantially larger level of loss generation is driven by the strong counter tip vortex active in the low momentum boundary layer flow close to the shroud endwall. The rotor blade calculations reveal a U-shape loss distribution where a minimum reading is registered at the midspan location of the passage. The loss generation increases significantly above 75% of the blade height, and below 30% of the same, where the complex secondary flow structures noted in the relative total pressure contours, dominate the flow.

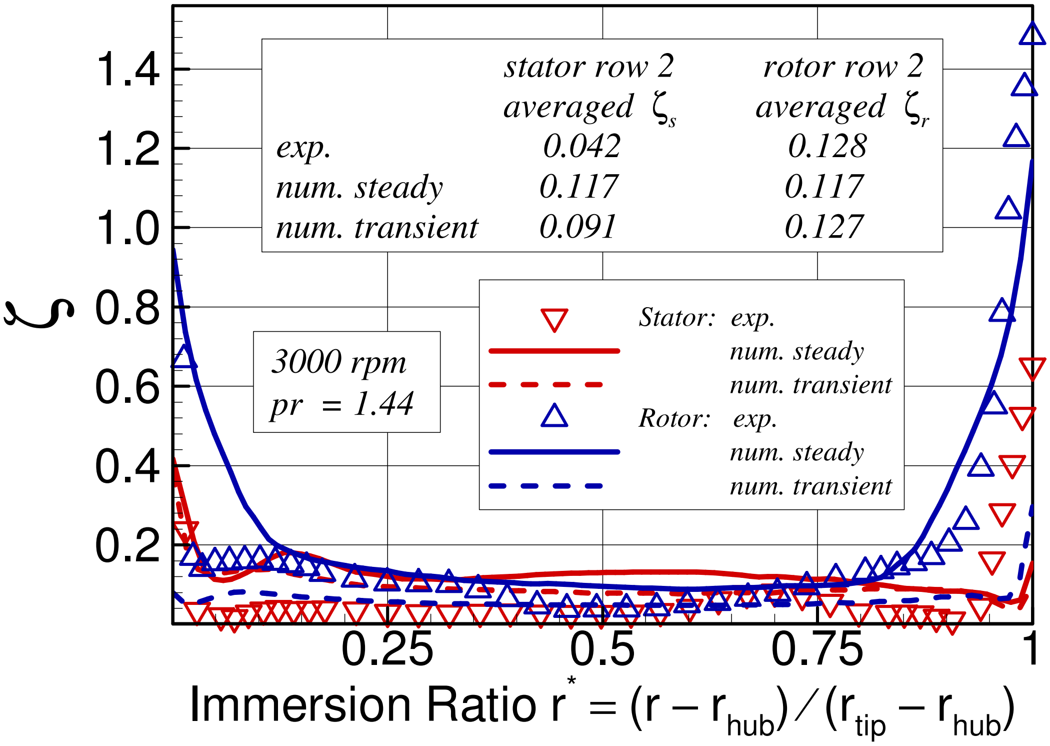


Figure 2.18: Radial distribution of measured total pressure loss coefficients for stator and rotor rows with steady and transient numerical simulation results, reproduced with permission from Schobeiri et al. [58]

The numerical results show significant discrepancies for both the steady and transient simulations. For the stator row, the computed losses are over-predicted by a factor of 200-300%, whereas for the rotor row the losses are slightly under-predicted, with the URANS yielding acceptable agreement with measurements. Neither the steady nor the transient simulations are able to match the trend of the radial distribution of the losses for the stationary or rotating blade rows. In particular, the endwall regions show substantial deviations from the measured values indicating the systemic failure of the numerical solvers to predict the location and intensity of the vortical structures responsible for the loss mechanisms along the flow path.

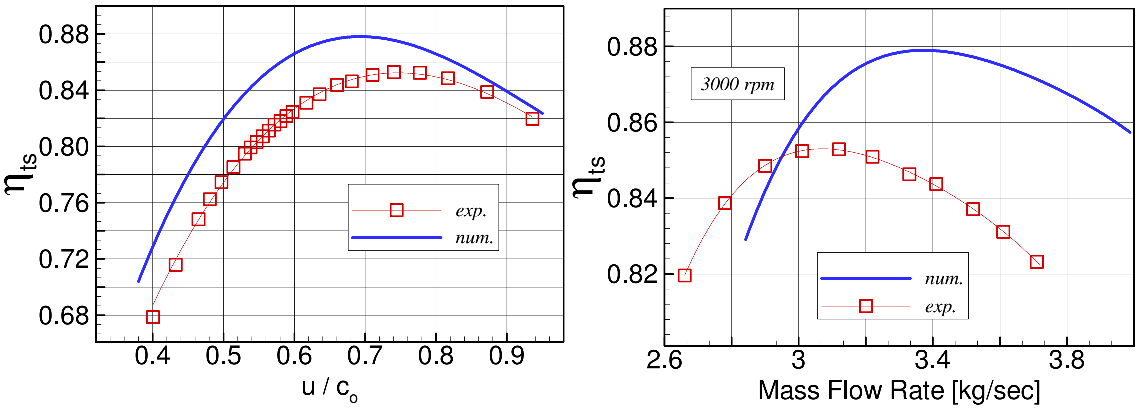
Considering the detailed experimental and numerical data presented in this study, Schobeiri et al.[58] stipulated that the documented inability of steady and transient numerical solvers to closely match the experimental measurement results is rooted in the turbulence and transition models currently employed. The deficient modeling of the intermittency function and Reynolds stress tensor directly affect the boundary layer growth and transition as well as the dissipation equation. The resulting under/over-prediction of wall shear stresses leads to biased determination of velocity and pressure profiles required for accurate loss calculations.

2.5.7 Turbine Efficiency Measurements

The turbine total to static efficiency was calculated and presented in Fig. 2.19 as a function of the dimensionless performance parameter $u/c_\circ$ and the mass flow rate. The turbine was operated in two modes during the course of this investigation. A full set of measurements was completed with variable machine rotational speeds ranging from $n = 1750$ rpm to 3000 rpm, in increments of 50 rpm, at a fixed pressure ratios of $pr = 1.44$. An additional set was obtained with variable pressure ratios, ranging from $pr = 1.16$ to 1.45 at the fixed design rotational speed of $n = 3000$ rpm.

Examining the experimental measurements, a maximum efficiency of $\eta_{ts} = 85.3\%$ is noted at $u/c_\circ = 0.74$. The deviation from the design condition point has a pronounced adverse effect on the efficiency of the unit, mainly due to the distortions of the velocity triangles through the various blade rows. At the upper operational point of $u/c_\circ = 0.94$ a drop of 3.3% in efficiency is detected, whereas a drop of 17.4% is noted at the lower end of the test range for $u/c_\circ = 0.4$.

As observed in Fig. 2.19a the steady numerical simulations results exhibit a significant deviation of 2.5% from experimental measurements at the peak operating point. Additionally, the inability of the steady numerical simulations to match the mass flow rates through the turbine stages is evident in Fig. 2.19b where a 10%



(a) Efficiency versus $u/c_\circ$

(b) Efficiency versus mass flow rate

Figure 2.19: Measured turbine performance for full operating range with steady numerical simulation results, reproduced with permission from Schobeiri et al. [58]

difference at the peak operating point is noted. These observations are consistent with numerical deviations from key measured flow parameters shown earlier in interstage results and can be attributed to the deficiency of the dissipation and transition models used. They further demonstrate the need for numerical code calibration for use on *tested* existing designs, and the inadequacy of current steady and transient numerical simulations alike to provide reliable a priori predictions where performance guarantees are issued on new stage designs.

2.6 Conclusions

A comprehensive set of steady state aerodynamic tests were conducted on a two-stage research axial turbine housed at the Turbomachinery Performance and Flow Research Laboratory *(TPFL)* of Texas A&M University. Interstage measurements at three planes located at the exit of the first rotor, second stator and second rotor were conducted using a set of calibrated five-hole probes which were traversed radially and circumferentially via an automatic seven-axis traversing system. Detailed flow pictures at these locations were obtained and analyzed to study the evolution of the secondary structures close to the hub and tip endwalls and their interaction with the mainstream flow. Row performance calculations were performed, and the primary and secondary stage losses were evaluated.

Performance tests were also performed for a wide operational range of rotational speeds varied between 1750 – 3000 rpm at a fixed pressure ratios of $pr = 1.44$, as well as a variable pressure ratio between 1.16 – 1.45 at fixed machine rotational speeds of $n = 3000$ rpm. The turbine efficiency was calculated using the readings of a high precision torque meter and taking into account the additional windage and bearing mechanical losses, and presented as function of the dimensionless performance parameter $u/c_{\circ}$ and mass flow rate. The calculated efficiency peaked at the design condition point of $u/c_{\circ} = 0.74$, with pronounced drops as the operational point was moved away in either direction.

A detailed comparison of experimental measurements with steady and transient simulations from a concurrent numerical investigation was made. Results from steady simulations (RANS) exhibited significant quantitative discrepancies with measurements, while transient computations (URANS) brought a modest overall improvement at the expense of time and computational resources. Based on the nature and magnitude of the differences in the various flow parameters observed at interstage stations, it was concluded that the numerical deficiencies stem from the inadequacies of dissipation and transition models currently adopted. These shortcomings effectively diminish the capability of RANS-based Navier-Stokes solvers to produce reliable a priori design and off-design performance predictions in turbomachinery applications where complex unsteady flow conditions lead to mostly transitional boundary layer development on blade surfaces and endwalls.

3. LINEAR CASCADE PROFILE LOSS MEASUREMENTS*

Profile losses are attributed to viscous dissipation mechanisms of the boundary layer growth and development along the surfaces of stator and rotor blades of an axial turbine stage. In addition to the blade row wall shear losses, the momentum deficiency at the trailing edge propagates additional wake mixing losses in the form of lowered available total pressure before the flow turns in the following blade row. Off-design operation of axial turbines further exacerbates this phenomenon, by causing distortions to the optimum flow velocity triangles at multiple blade rows. As a result, inlet flow velocity vectors become mismatched with their intended blade camber lines, contributing to a cumulative increase in turbine profile losses. The current numerical and experimental study is aimed at quantifying profile losses in a simulated axial turbine blade row at both design and off-design conditions. These findings are subsequently used to calibrate existing empirical profile loss correlations that can be used as predictive modules in computational turbine simulations. The first part of this study investigates the effects of varying the flow incidence on a linear blade cascade that was built, scaled and positioned to replicate the midspan passage through the stator blade row of a large scale steam turbine. The second part investigates the effects of the stagger angle on the overall performance and flow characteristics of a high pressure stator blade cascade. While the deflection angle of the blade is unchanged, varying the stagger angle of the cascade at a fixed solidity alters the inlet and exit flow angles and the throat area of the blade passage.

*Part of the data reported in this chapter is reprinted with permission from "An Experimental and Numerical Study of the Effects of Flow Incidence Angles on the Performance of a Stator Blade Cascade of a High Pressure Steam Turbine" by H.A. Chibli, S.A. Abdelfattah, M.T. Schobeiri, and C. Kang. ASME Turbo Expo: Power for Land, Sea, and Air, Volume 7: Turbomachinery, Parts A and B :821-830. doi:10.1115/GT2009-59131. Copyright 2009 by American Society of Mechanical Engineers.

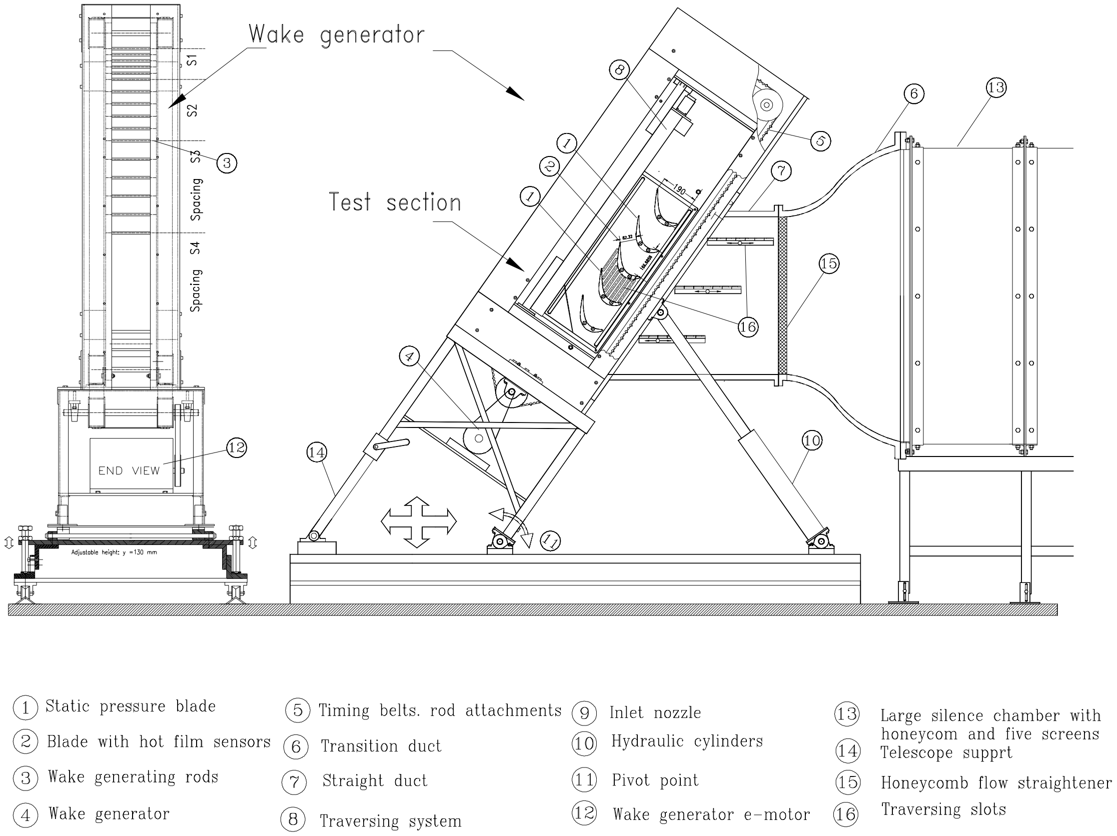


Figure 3.1: Turbine cascade research facility with the components and the adjustable test section, obtained from Schobeiri et al. [76]

3.1 Experimental Facility

To simulate the effects of stagger and flow incidence angles on the performance of a high pressure axial turbine linear blade cascade, a multi-purpose large scale, subsonic research facility at TPFL was utilized. This facility shown in Fig. 3.1 was designed and has been in extensive operation since 1993 [78, 79, 76, 80] mainly to investigate the effects of unsteady wake flow on turbine cascade aerodynamics and heat transfer, particularly on unsteady boundary layer transition. The research facility consists of a large centrifugal blower, a diffuser, a settling chamber, a nozzle,

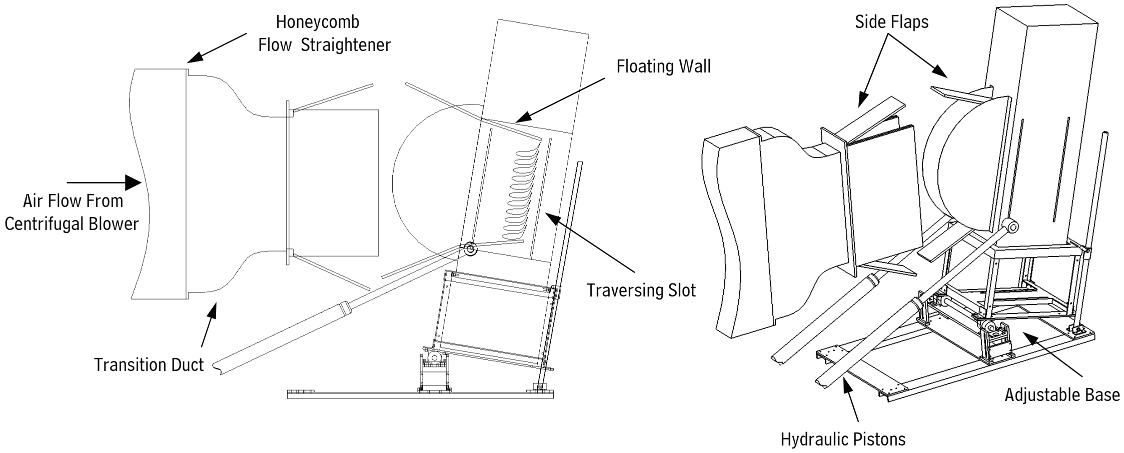
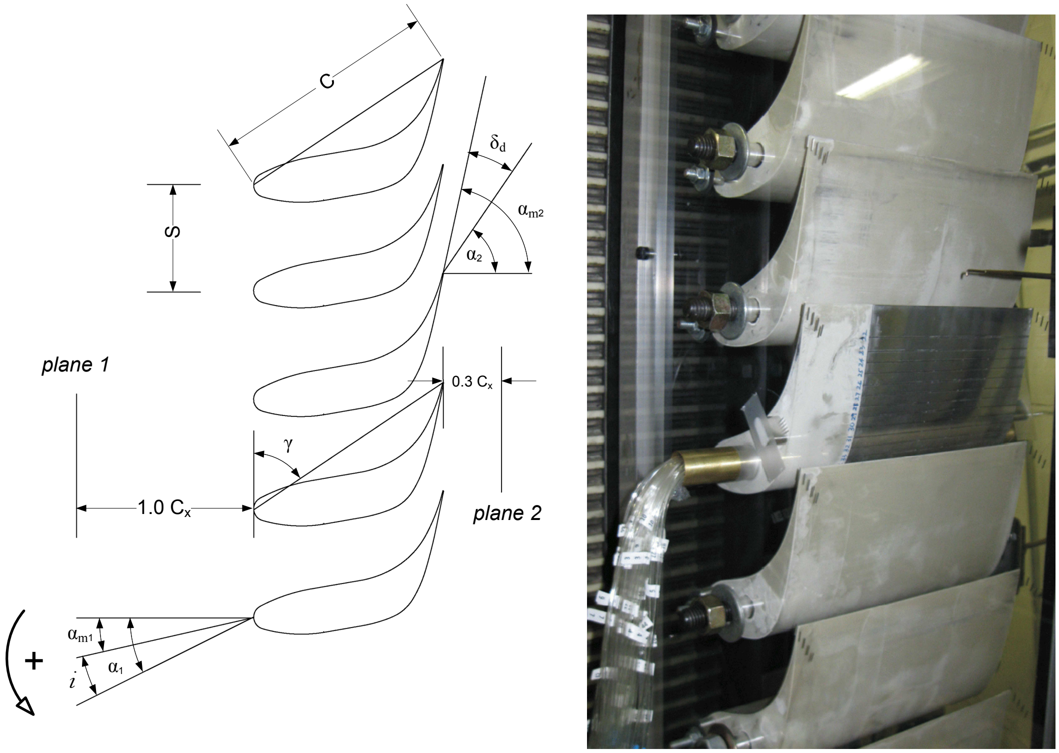


Figure 3.2: A schematic rendering of the linear blade cascade facility at TPFL with the main components labeled, reproduced with permission from Chibli et al. [77]

an unsteady wake generator, and a turbine cascade test section. The blower with a nominal volumetric flow rate of 15 m^3/s is capable of generating a maximum velocity of 100 m/s at the test section inlet. The settling chamber consists of five screens and one honeycomb flow straightener to maintain the uniformity of the flow. A detailed schematic of the cascade test facility is shown in Fig. 3.2.

Figure 3.3a shows the layout of the linear cascade under investigation in the first part of this study. The blades used belong to the second stage stator of a high pressure steam turbine currently under development by Doosan Heavy Industries & Construction of South Korea. A total number of 12 blades have been used to establish the periodicity of the flow. Since this study is solely devoted to the midspan measurements, a blade aspect ratio of 4/3 was implemented to maintain the two-dimensional nature of the flow in the cascade, as demonstrated by Roger et al. [81]. One of the blades in the center of the cascade was especially manufactured with 40 built-in pressure taps to enable the static pressure measurement around the blade suction and pressure surfaces. Table 3.1 summarizes the different parameters of the



(a) Blade cascade side view with geometric and angular notations shown

(b) Instrumented blade with five-hole probe positioned in measurement plane

Figure 3.3: Linear blade cascade geometry layout, reproduced with permission from Chibli et al. [77]

cascade under investigation.

For the second part of this study, a linear cascade of a total of eight blades has been used, as shown in Fig. 3.3b. Since the stagger angle change had to be investigated, the blades were fabricated and installed such that they are allowed to be accurately rotated about a fixed axis, with their positions calibrated to a set of markings itched on the inner wall of the test facility used. The middle blade was instrumented with 46 static pressure measuring taps, that are connected to the pressure scanners for digital readout of the measurements. Table 3.2 summarizes the geometric properties of the blade profile used in the stagger angle study. The

Parameter	Value	Parameter	Value
Blade chord	c = 150 mm	Number of blades	N = 12
Blade height	h = 200 mm	Solidity	$\sigma = 2.2$
Blade inlet angle	$\alpha_1 = 5.5°$	Stagger angle	$\gamma = 56.1°$
Blade exit angle	$\alpha_2 = 77.1°$	Zweifel number	$\Psi = 0.23$

Table 3.1: Specifications of the flow incidence study blade cascade

precision aluminum molds used to fabricate the test blades in this study are shown in Figs. F.1 and F.2.

To enable the precise rotation and positioning of the blade cascade with respect to the incoming air flow, the facility had to be modified to include a fully adjustable base. With the help of a sliding floor and two hydraulic pistons installed symmetrically on the sides, the operator is capable of changing the angle that the entire test section makes with the vertical from $-20°$ to $+20°$ within an accuracy of $0.1°$. Since the angular alignment of the cascade for each incidence angle setting will alter the periodic nature of the flow, floating walls were installed at the top and bottom of the cascade. These walls help regulate the flow through the cascade by enforcing a uniform mass flow rate through each individual blade channel. Special attention was devoted to maintain a zero angle of incidence for the incoming flow onto the blade cascade used in the second part of this study, in order to be able to isolate the effects of the stagger angle variations on the blade performance.

3.2 Instrumentation and Data Acquisition

Three types of measurements were conducted during the course of this series of experiments: Upstream pressure measurements at *plane–1* shown in Fig. 3.3a using an array of four Prandtl/Pitot tubes (flow incidence study) and a cobra-style five-hole probe (stagger angle study), downstream measurements at *plane–2* using

Parameter	Value	Case	Stagger Angle	Zweifel number
Blade chord	c = 152.7 mm	1	$\gamma = 42.9°$	$\Psi = 0.26$
Blade height	h = 200 mm	2	$\gamma = 45.6°$	$\Psi = 0.33$
Number of blades	N = 8	3	$\gamma = 48.3°$	$\Psi = 0.39$
Deflection angle	$\theta = 107.7^o$	4	$\gamma = 51.0°$	$\Psi = 0.44$
Solidity	$\sigma = 1.6$	5	$\gamma = 53.6°$	$\Psi = 0.49$

Table 3.2: Specifications of the stagger angle study blade cascade

a cobra-style five-hole probe (flow incidence study) and an L-style five-hole probe (stagger angle study), and blade surface pressure distribution measurements using static pressure taps. Total temperature was measured at several stations throughout the test section with the use of J-type thermocouple probes with an accuracy of 0.1°C.

The five-hole probes used were carefully calibrated in a flow field nominal Mach number $M = 0.2$, and yaw and pitch angles ranging from $-20°$ to $-20°$ in increments of 1.0° using Bohn's calibration method. In the first part of this study, all pneumatic pressure probes and pressure taps used were connected to a 48 channel Scani-Valve system and logged by a high precision differential pressure transducer (MKS Baratron - 100 Torr series) calibrated with a dead weight tester device to an overall accuracy of 0.15% of actual reading (Fig. E.1). Data points were sampled at 2 KHz for a duration of 1 second per reading using a 16 channel, 12 bit analoge-digital (A/D) board (NI-PCI-MIO-16E-1) controlled by a personal computer using LabVIEW. All pressure ports in the second part of this study were connected to a high speed pressure PSI (Pressure System Incorporated) scanner system with a total of 3 modules with 16 channels each. This system, enabled an instantaneous read of all pressure measurements with an minimum established accuracy of 0.01% of full scale. Figure G.1b shows a view of the pressure scanning system used.

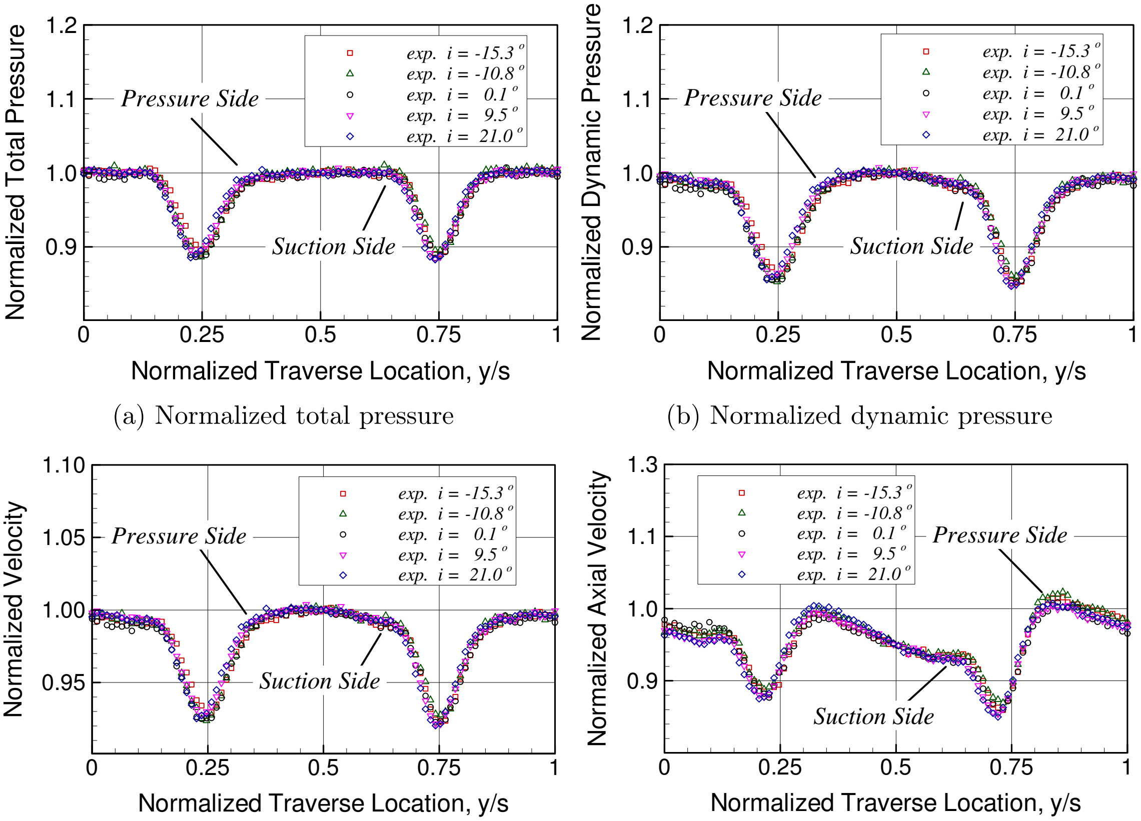
A traversing system was installed at the side of the test facility that allows the five-hole probe to travel tangentially through a vertical sealed slot (Fig. G.1) alongside the exit plane of the blade cascade. The traversing system consists of a slider and lead screw that is connected to a DC stepper motor with an encoder and decoder controlled by the LabVIEW data acquisition program. The optical encoder provides a continuous feedback to the stepper motor for accurate positioning of the probe. A traversing step size of 1.5 mm was determined to adequately describe the flow behavior inside the wake generated at the trailing edge of the blades studied. A total number of 4 blade spans were traversed for every incidence/stagger angle reported. Local ambient pressure and humidity conditions were recorded at intervals of 20 minutes throughout the experimental procedure. Complete ambient conditions profiles were subsequently generated with the use of shape-preserving cubical spline interpolation, and used to correct the air density and viscosity. Figure 3.3b shows a side view of the cascade with the five-hole probe mounted on the linear traversing system as well as the instrumented static pressure blade. The LabVIEW scripts used to control the traversing system and automate the data acquisition are shown in Figs. H.4, H.1, H.3 and H.2.

3.3 Numerical Treatment

To gain further insight into the physics of the cascade flow field, an in-house numerical study using ANSYS-CFX package was carried out at TPFL. A total of 165000 quad elements were generated for the multi-block structured mesh used. The numerical iterations were run on a dedicated workstation powered by two parallel Xeon Quad Core CPU's and utilizing a total of 16 Gb of memory (RAM). The grid employs two periodic boundary conditions at the upper and lower ends of the computational domain, and fixed total and static pressures at the inlet and outlet sides, respectively. An incompressible flow model was adopted. The turbulence model used in this study was the Shear Stress Transport $k - \omega$ model (SST) which uses a blending function at the wall. Therefore, great care was taken to create a highly refined discretization at the wall boundaries.

3.4 Flow Incidence Study Results and Discussion

Figures 3.4a and 3.4b show the exit total and dynamic pressure profiles for the three limit cases of $i = -15.3°$, $+0.1°$ and $+21.0°$ along the span of the two center blades of the test section. The average discharge freestream values were used to



(a) Normalized total pressure

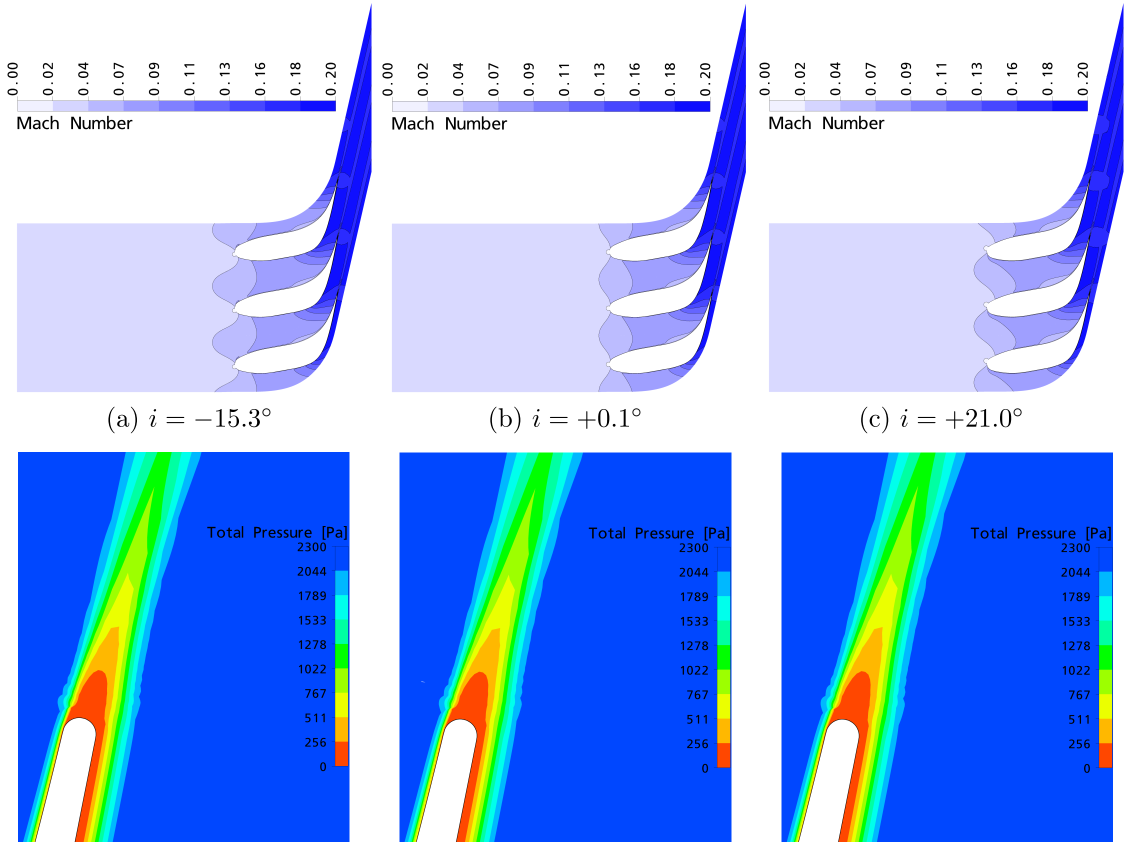
(b) Normalized dynamic pressure

(c) Normalized velocity

(d) Normalized axial velocity

Figure 3.4: Normalized experimental cascade exit pressure and velocity profiles for flow incidence angle range of $-15.3°$ to $+21.0°$, reproduced with permission from Chibli et al. [77]

normalize the results, which illustrate a satisfactory level of periodicity through the cascade. The wake effect is evident in all three profiles as the normalized total and dynamic pressures dip to their lowest values of about 0.89 and 0.86 respectively, at the trailing edge of each blade. No visible difference in flow pattern is detected as incidence is increased in the negative direction. Nevertheless, a limited trend of increased losses can be noticed as incidence is increased in the positive sense. This behavior can be further observed in the invariant numerically computed wake structure at the trailing edge of the blade shown in Fig. 3.5, for each of the three



(a) $i = -15.3°$ (b) $i = +0.1°$ (c) $i = +21.0°$

(d) $i = -15.3°$ (e) $i = +0.1°$ (f) $i = +21.0°$

Figure 3.5: Numerical Mach number and total pressure trailing edge contours for flow incidence angle range of $-15.3°$ to $+21.0°$, reproduced with permission from Chibli et al. [77]

incidence angles considered.

The exit velocity profiles shown in Fig. 3.4c were normalized by the average discharge freestream velocity. For all three cases of incidence, a higher exit axial velocity component is evident when moving closer to the pressure side of the blade channel, as shown in Fig. 3.4d. This in part is due to the higher deviation from the blade exit metal angle on the suction side of the cascade passage causing a notable flow over-turning. Detailed experimental and numerical cascade exit pressure and velocity profiles are shown in Figs. C.1 and C.2 for negative and positive flow incidence angles, respectively.

To quantify the losses within the blade cascade, the mass-averaged total and dynamic pressures at the exit plane were calculated as follows:

$$\overline{P}_2 = \frac{\displaystyle\int_{y-\frac{s}{2}}^{y+\frac{s}{2}} \rho_2 V_{x_2} P_2 \, \mathrm{d}y}{\displaystyle\int_{y-\frac{s}{2}}^{y+\frac{s}{2}} \rho_2 V_{x_2} \, \mathrm{d}y} \quad \text{and} \quad \overline{p}_{d_2} = \frac{\displaystyle\int_{y-\frac{s}{2}}^{y+\frac{s}{2}} \rho_2 V_{x_2} p_{d_2} \, \mathrm{d}y}{\displaystyle\int_{y-\frac{s}{2}}^{y+\frac{s}{2}} \rho_2 V_{x_2} \, \mathrm{d}y} \tag{3.1}$$

The cascade total pressure profile loss coefficient can then be evaluated as:

$$\zeta_p = \frac{P_1 - \overline{P}_2}{\overline{p}_{d_2}} \tag{3.2}$$

As seen in Fig. 3.6, no significant variation in the loss coefficient can be registered for the experimental data as incidence is varied from −15.3° to +21.0°. Numerical data shows a consistent flat pattern, where off-design operation within ±30° range of flow incidence projects no notable influence on the overall cascade aerodynamic performance. To better understand this behavior, some key qualities of the cascade geometry should be properly identified.

The large blade inlet diameter to spacing ratio and blade nose wedge angle greatly

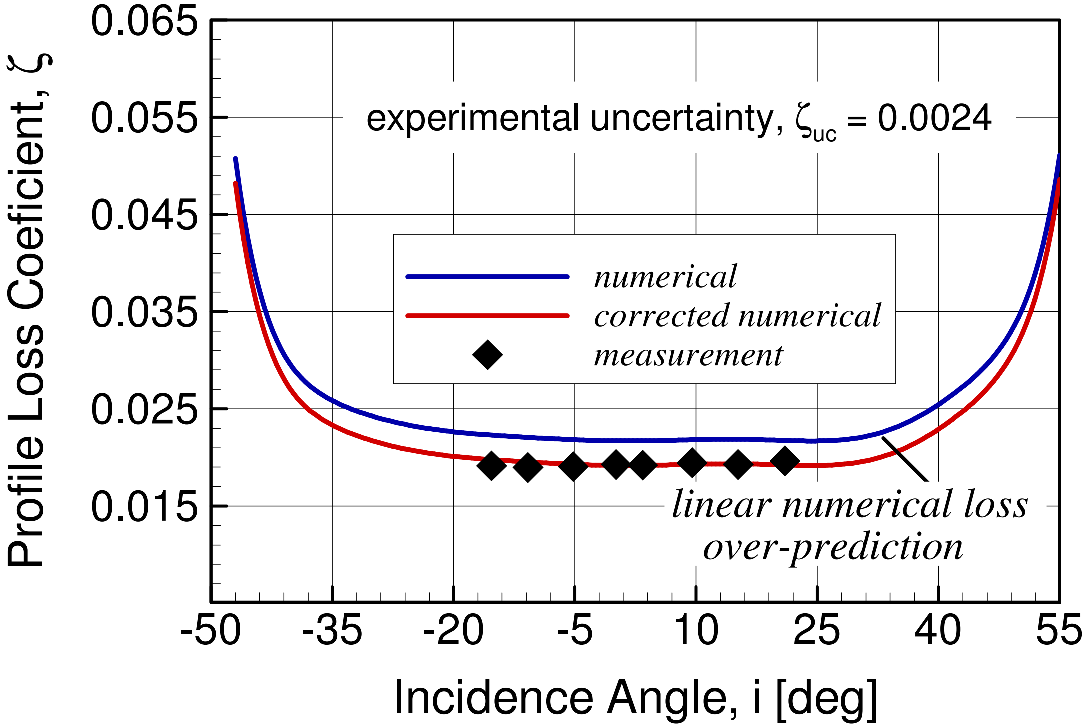
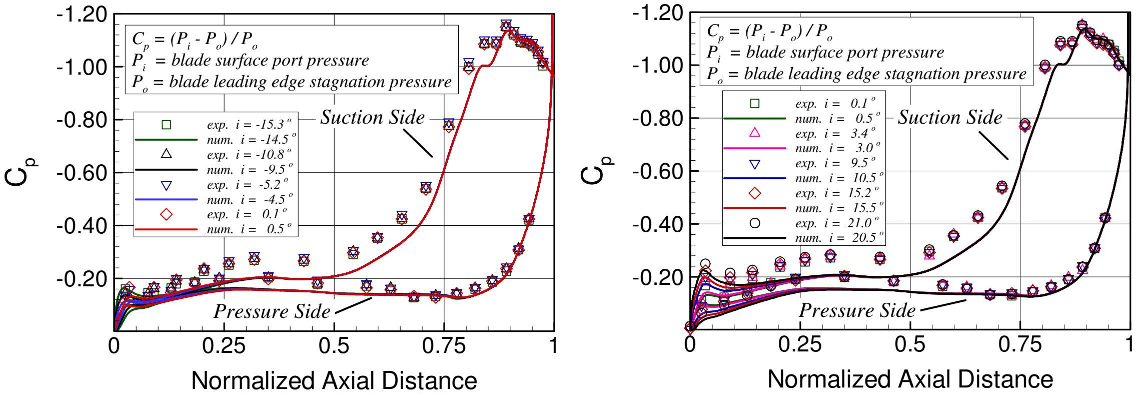


Figure 3.6: Numerical and experimental total pressure loss coefficients for an extended range of flow incidence angles

contribute to the insensitivity of the current cascade to flow incidence, as empirically demonstrated by Benner et al. [15]. A smaller wedge angle generally indicates a more pronounced profile curvature discontinuity at the point where the blade nose blends with the blade body, which in turn promotes the growth of local separation bubbles. Coupled with high flow incidence, these bubbles can dramatically grow in magnitude resulting in full flow separation and significant profile loss coefficient. A large blade leading edge diameter contributes to an overall increase in the profile loss at design condition. However the thick blade nose better accommodates the incoming incident flow, and smoothly guides it into the blade passage, thus making the cascade more tolerant to off-design operation.

Furthermore, a close examination of the blade cascade in Fig. 3.3a reveals that



(a) Negative flow incidence angles

(b) Positive flow incidence angles

Figure 3.7: Numerical and experimental blade C_p distributions for flow incidence angle range of $-15.3°$ to $+21.0°$, reproduced with permission from Chibli et al. [77]

an inlet blade channel of relatively unchanged cross sectional area, stretches about $0.5 \times C_x$ into the blade passage. As such, the flow expansion is delayed until the incoming flow has reached the second half of the channel, as clearly demonstrated in Fig. 3.5. This creates a *buffer* inlet zone that isolates the convergent nozzle section of the blade passage, thus making the cascade performance immune to the detrimental effects of off-design flow incidence.

The C_p distribution around the blade suction and pressure surfaces was calculated and plotted for the cases of negative and positive flow incidence, in Fig. 3.7a and Fig. 3.7b respectively. As flow incidence is increased in the positive sense, both the experimental and numerical results show a limited pattern of flow under-expansion on the pressure side, and over-expansion on the suction side of the blade in the leading edge area of the cascade channel. The opposite trend is observed for the cases of negative incidence. Nevertheless, no significant change in flow pattern is observed in the second half of the channel as incidence angles are varied from extreme negative to positive values. This further illustrates the point discussed earlier where the straight

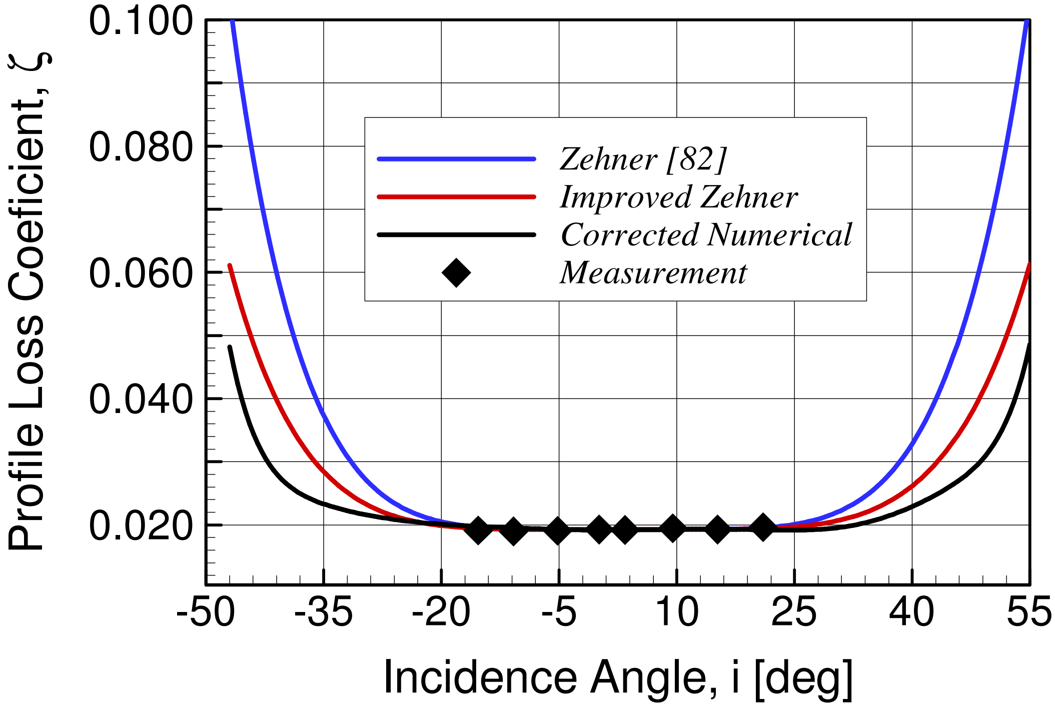


Figure 3.8: Numerical, experimental and empirical total pressure loss coefficients for an extended range of flow incidence angles

inlet blade passage isolates the converging rear section of the nozzle from the inlet flow off-design fluctuations, leading to a fairly invariant profile loss behavior.

3.5 Off-Design Profile Loss Correlation

To obtain an empirical off-design profile loss correlation using the experimental and numerical findings of this investigation, a blade cascade incidence loss model previously introduced by Zehner [82] was adopted. Since the numerical simulations were observed to over-predict the profile loss by a fixed error that falls within the experimental uncertainty of $\zeta_{uc} = 0.0024$ as demonstrated in Fig. 3.6, a linear correction was applied to the calculated loss to match the measurements within the base incidence band of $-15.3°$ to $+21.0°$. This effectively extended the benchmark profile loss coefficient for a broad incidence angle range of $-45°$ to $+55°$.

Zehner estimates the incidence profile loss coefficient as:

$$\begin{aligned} \zeta_p &= 1 - \left(1 - \zeta_{p_N}\right) \times e^{-a(\Delta\beta^*)^b} \\ \Delta\beta^* &= \frac{\Delta\beta}{180 - \beta_{1_N}} \quad \text{for} \quad \Delta\beta > 0 \\ \Delta\beta^* &= \frac{\Delta\beta}{-\beta_{1_N}} \quad \text{for} \quad \Delta\beta < 0 \end{aligned} \tag{3.3}$$

with $\Delta\beta = \beta_1 - \beta_{1_N}$, where β_{1_N} and ζ_{p_N} are defined as nominal inlet flow angle and profile loss coefficient at design condition. Note that in this formulation, β is measured tangentially from blade suction to pressure side. The geometric coefficients a and b are empirically correlated at a reference $Re = 4 \times 10^5$, and summarized in Table 3.3 as follows:

$$a, b = \sum_{i=0}^{2} c_i {\gamma_s}^i \quad \text{and} \quad \gamma_s = \frac{f}{s}\sqrt{\hat{\gamma} \cdot \hat{\delta}} \tag{3.4}$$

where $\hat{\gamma}$ and $\hat{\delta}$ are the cascade stagger and blade deflection angles in radians, f is the maximum blade camber height and s is the cascade spacing.

As evident in Fig. 3.8, the empirically calculated profile loss using Zehner's correlation is in agreement with the measured data within an incidence angle band of ±20°. However as flow incidence increases beyond this limit, the deviation from the corrected numerical predictions significantly grows. This is attributed to the lack of a

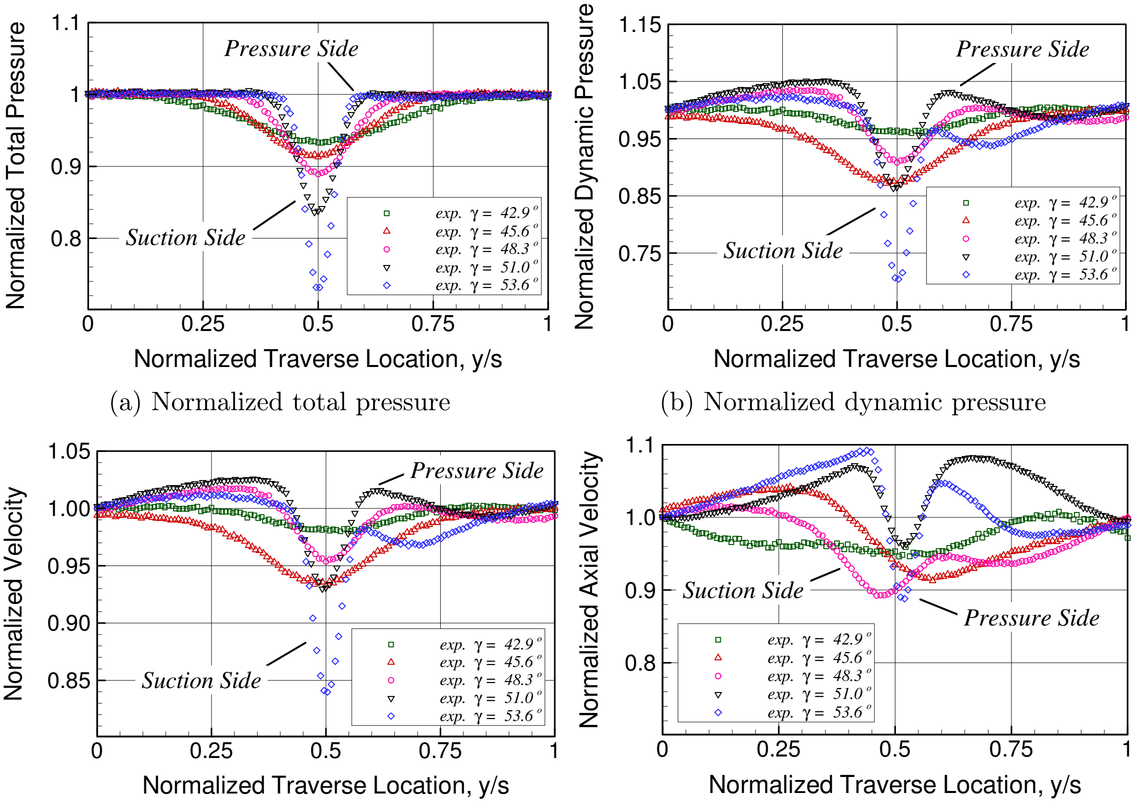
	$\Delta\beta > 0$		$\Delta\beta < 0$	
i	a	b	a	b
0	2.587	4.175	0.446	2.413
1	-0.426	10.802	3.820	10.380
2	-1.216	-13.881	-2.899	-10.116

Table 3.3: Empirical geometric coefficient c_i, obtained from Zehner [82]

corrective factor in the original formulation that accounts for the blade leading edge diameter effect on the cascade sensitivity to flow incidence. Therefore, the following improved correlation is recommended:

$$\zeta_p = 1 - \left(1 - \zeta_{p_N}\right) \times e^{-a\left(1-(d/s)^{0.5}\right)(\Delta\beta^*)^b} \tag{3.5}$$

where d is the blade leading edge diameter. For the blade cascade used in this study $d/s = 0.240$.



(a) Normalized total pressure

(b) Normalized dynamic pressure

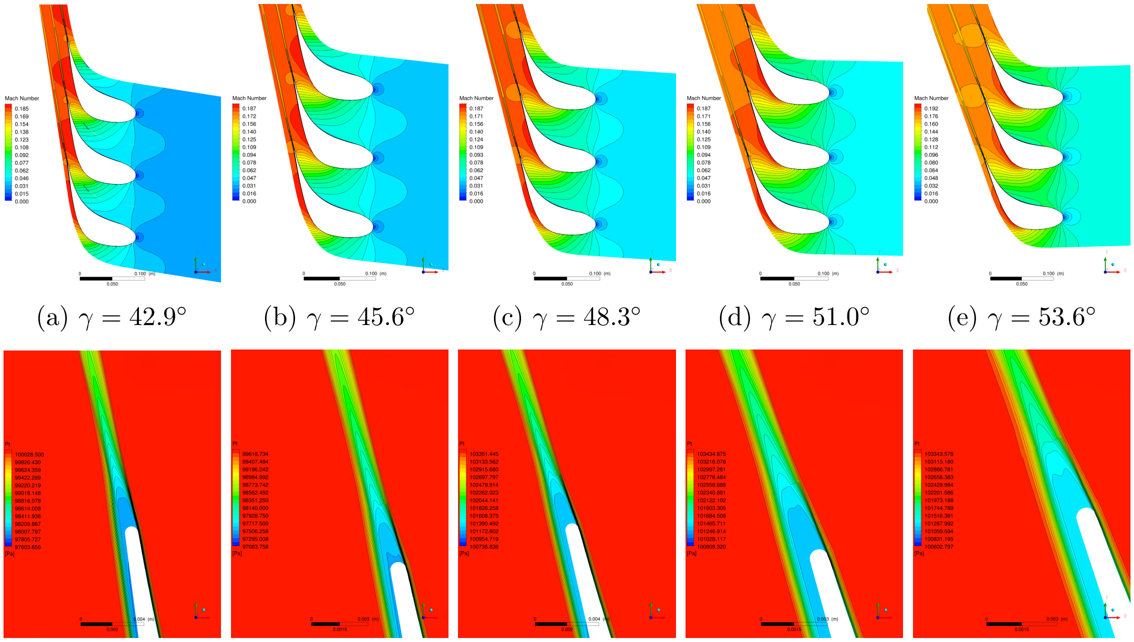
(c) Normalized velocity

(d) Normalized axial velocity

Figure 3.9: Normalized experimental cascade exit pressure and velocity profiles for five cascade stagger angles $\gamma = 42.9°$, $45.6°$, $48.3°$, $51.0°$ and $53.6°$

3.6 Stagger Angle Study Results and Discussion

Figure 3.9 shows the normalized measured pressure and velocity profiles at the blade cascade exit plane for the five stagger angles of $\gamma = 42.9°$, $45.6°$, $48.3°$, $51.0°$ and $53.6°$. The total pressure wake structure shown in Fig. 3.9a varies significantly with the stagger angle, as the wake width increases with the lower stagger angle setting, while the wake depth increases with the higher stagger angle setting. This is attributed to the smaller cascade channel throat area that enhances the wake mixing for the lower stagger angle cases. Increasing the stagger angle limits the entropic activity to the immediate vicinity of the blade wall and does not extend the total pressure drop towards the freestream section of the channel. The detailed experimental and numerical results for the individual cascade stagger angles presented in



(a) $\gamma = 42.9°$ (b) $\gamma = 45.6°$ (c) $\gamma = 48.3°$ (d) $\gamma = 51.0°$ (e) $\gamma = 53.6°$

(f) $\gamma = 42.9°$ (g) $\gamma = 45.6°$ (h) $\gamma = 48.3°$ (i) $\gamma = 51.0°$ (j) $\gamma = 53.6°$

Figure 3.10: Numerical Mach number and total pressure trailing edge contours for five cascade stagger angles $\gamma = 42.9°$, $45.6°$, $48.3°$, $51.0°$ and $53.6°$

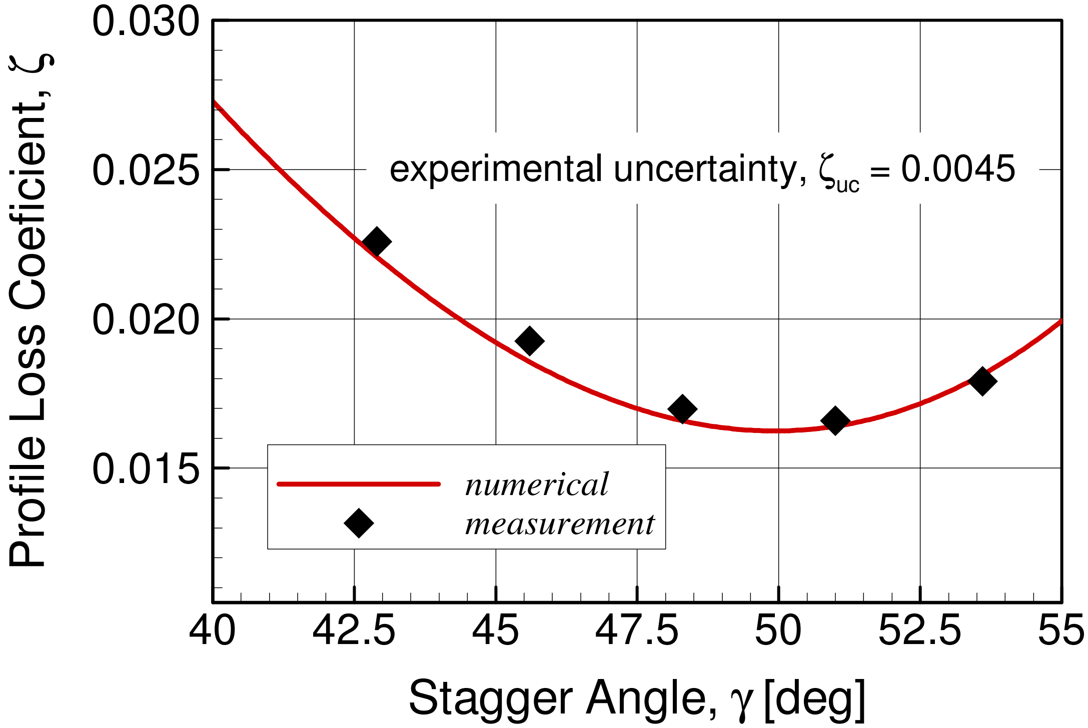


Figure 3.11: Numerical and experimental total pressure loss coefficients for an extended range of cascade stagger angles

Fig. C.3 are mass-averaged using the procedure provided in Eq. 3.1 to calculate the total pressure loss coefficients for the five stagger angle cases. Figure 3.11 shows acceptable agreement between the simulation loss calculation results and measurements, with the error contained within the experimental uncertainty of $\zeta_{uc} = 0.0045$.

The pressure distribution about the blade surface was measured for all five cases, and plotted in Fig. 3.12 for the respective stagger angles considered. Examining the experimental and numerical results for the blade pressure side, it is noticed that for all the stagger angle cases, the C_p distribution remains mostly flat and unaltered for 75% of the cascade channel axial length, with a steep increase towards the cascade exit line. It is also evident that the change in C_p magnitude is gradual and proportional to the change in stagger angle value, on the pressure side of the blade.

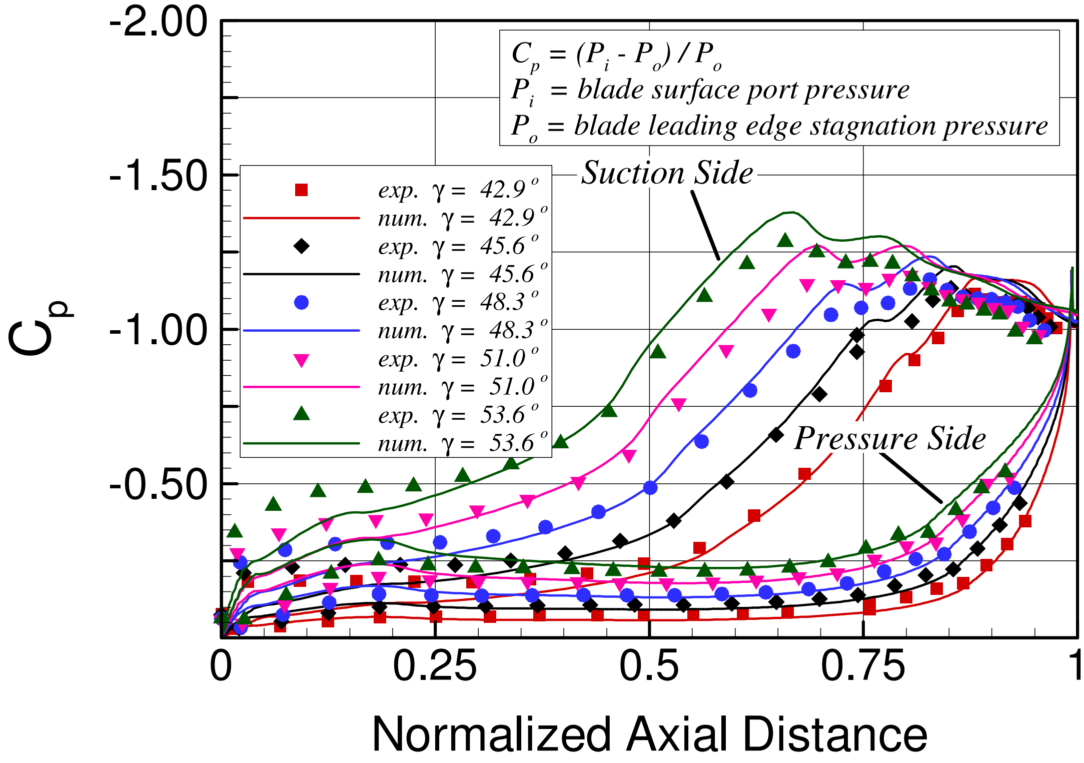


Figure 3.12: Numerical and experimental blade C_p distributions for five cascade stagger angles $\gamma = 42.9°$, $45.6°$, $48.3°$, $51.0°$ and $53.6°$

On the other hand, the behavior of the C_p distribution varies widely with the stagger angle value on the suction side of the blade. For the minimum stagger angle value of 42.89°, the C_p assumes a nearly fixed value of approximately -0.15 for about 50% of the cascade channel axial length, before it steeply climbs to a maximum value of approximately -1.2 towards an axial location of $0.85 \times C_x$. This behavior indicates that the first half of the channel provides a nearly constant cross sectional area, whereas the bulk of the flow expansion takes place in the second half of the cascade nozzle, where the cross sectional area steeply drops to 23% of the inlet area. Nonetheless, as the stagger angle increases, the throat area of the cascade increases, and the inlet side of the channel nozzle evidently becomes more convergent. This reflects itself with a significant shift in the C_p distribution whose magnitude grows rapidly closer to the leading edge of the blade. At the maximum stagger angle value of 53.57°, it is observed that the C_p magnitude climbs to a value of approximately -0.5 at a channel position of $0.15 \times C_x$. Therefore, by increasing the stagger angle, a more quadratic behavior of the blade with a more uniform pressure distribution throughout the cascade channel is achieved. The numerical Mach number contours depicted in Fig. 3.10 show similar expansion trends with higher velocities in the blade channel for the cases with increased stagger angle.

The blade profile loss coefficient ζ_p and the averaged pressure distribution coefficient C_p from the numerical results shown in Figs. 3.11 and 3.12 respectively, can be correlated to the cascade stagger angle using the following relations:

$$\zeta_p = \sum_{n=0}^{3} A_n \gamma^n \quad \text{and} \quad \overline{C}_p = \sum_{n=0}^{3} B_n \gamma^n \tag{3.6}$$

where the empirical parameters A_n and B_n are listed in Table 3.4. It should be noted that the correlations presented in Eq. 3.6 are specific to the blade geometry, cascade solidity and flow conditions investigated in this study, and additional empirical corrections are required to expand their application to different configurations.

n	$\mathrm{A_n}$	$\mathrm{B_n}$
0	9.12549×10^{-2}	2.16523×10^{-1}
1	2.10210×10^{-3}	1.83311×10^{-2}
2	-1.74495×10^{-4}	-1.15252×10^{-3}
3	2.04901×10^{-6}	1.02927×10^{-5}

Table 3.4: Correlation parameters for the empirical relations shown in Eq. 3.6

3.7 Design Profile Loss Correlation

The Pfeil [4, 5] design profile loss correlation for a turbine blade cascade at reference $Re = 3.5 \times 10^5$ can be calculated as:

$$\zeta_p = \left(\varepsilon_{opt}\right)_{single} \times \left[1 + K\left(\frac{c}{s}\right)^3_{opt}\right] \times C_L \frac{c}{s}\frac{1}{sin\alpha_\infty} \tag{3.7}$$

where the cascade lift coefficient for inviscid flow is defined by Schobeiri [6] as:

$$\begin{aligned} C_L &= 2\frac{sin^2\alpha_2}{sin\alpha_\infty}\left(cot\alpha_2 - cot\alpha_1\right)\frac{s}{c} \\ cot\alpha_\infty &= \frac{1}{2}\left(cot\alpha_2 + cot\alpha_1\right) \end{aligned} \tag{3.8}$$

For a turbine blade cascade, Zweifel [1] empirically determined the optimum lift coefficient range $\left(C_L\right)_{opt} = 0.8 - 1.05$. Applying Zweifel's criteria into Eq. 3.7 yields an approximate value for the cascade profile loss at optimum solidity. As illustrated in Fig. 1.2, the point of optimum solidity represents the cascade chord to spacing ratio at which the summation of the friction and separation loss components acting on the blade reaches a minimum value.

In order to calibrate the empirical parameters in Eq. 3.7 using the measurements conducted on the five blade cascades in this study, the respective geometric and flow parameters were input into Eqs. 3.7 and 3.8. Additionally, a Reynolds number correction was applied to the correlated loss coefficient using the following relation:

$$\frac{\zeta}{\zeta_{ref}} = \left(\frac{Re_{ref}}{Re}\right)^{\frac{1}{5}} \tag{3.9}$$

As shown in Fig. 3.13, it was determined that the empirical parameters $K = 0.062$ and $\left(\varepsilon_{opt}\right)_{single} = 0.0084$ provide an acceptable match between the correlated and measured profile loss coefficients. For a given blade cascade, these improved parameters predict a lower profile loss coefficient compared to Pfeil's original empirical values, reflecting the enhancements in turbine blade performance achieved in the last five decades.

3.8 Uncertainty Analysis

Incorporating the nominal uncertainties of the individual measured parameters, and following the analysis method of Kline and McKlintock [74], the experimental uncertainties for the various calculated flow parameters were estimated and summarized in Table A.1. Three measurements of each data set were collected and analyzed to establish measurement repeatability. The FORTRAN 77 source code of the data

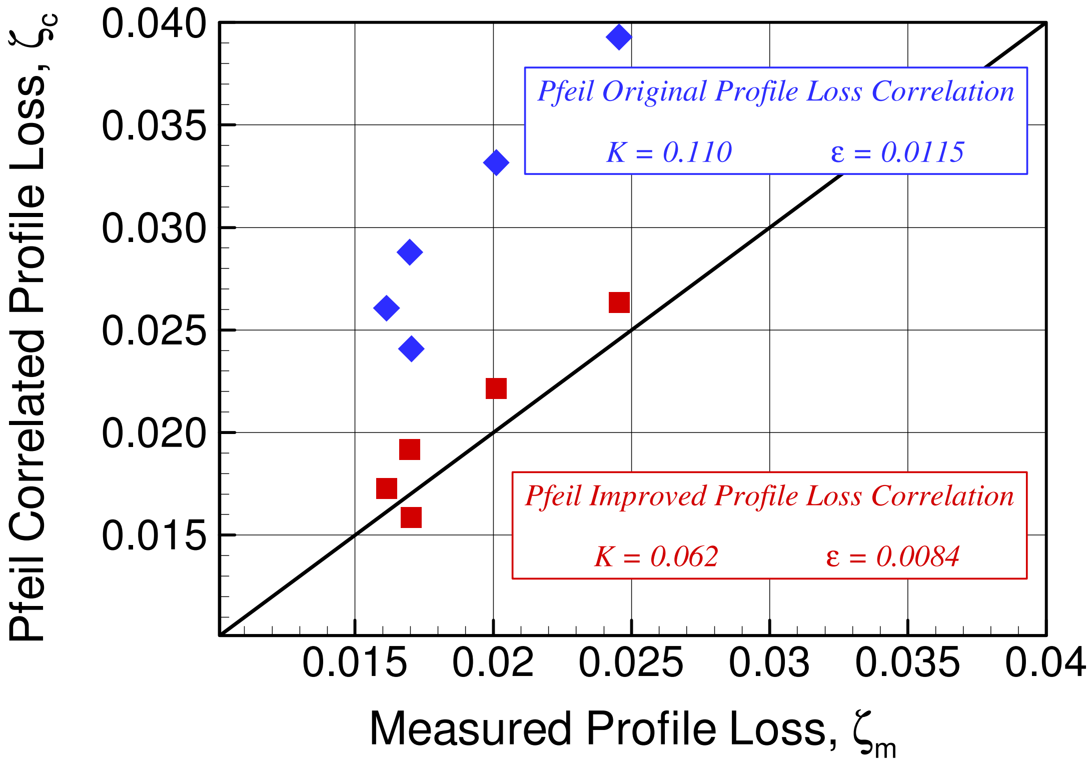


Figure 3.13: Pfeil profile loss correlation at optimum blade cascade solidity with original and improved empirical parameters

analysis programs is listed in Appendix J.3 and Appendix J.4 for the flow incidence and stagger angle studies, respectively.

3.9 Conclusions

A comprehensive experimental and numerical study in two parts was carried out to quantify the profile losses in a simulated axial turbine blade row at design and off-design conditions. The first part of this study investigates the effects of off-design operation on the performance of a linear stator blade cascade. The inlet incidence angle was varied between $-15.3°$ and $+21.0°$ for a total of eight cases with detailed measurements of the flow field upstream and downstream of the test section. An instrumented blade was equipped with pressure taps, and used to measure the pressure profile around the pressure and suctions sides, enabling a better understanding of the flow physics in the blade passage. The large leading edge nose diameter of the blade, and the fairly invariant cross sectional area of the inlet blade passage, contributed to a blade cascade design that is forgiving of considerable off-design operational conditions. The second part of this study examines the effects of the stagger angle on the overall performance and flow characteristics of a high pressure linear blade cascade. Blades were positioned and rotated at a fixed solidity to simulate the change in cascade stagger angle for five different cases of $\gamma = 42.9°$, 45.6°, 48.3°, 51.0° and 53.6°. Comprehensive upstream, downstream and blade surface pressure measurements were conducted to evaluate the midspan inlet and exit flow field and accurately compute the profile loss coefficient for each of the five unique cascade configurations. The numerical and experimental findings of this study were used to calibrate two existing design and off-design profile loss correlations that would subsequently be employed as reliable empirical predictive modules.

4. STREAMLINE CURVATURE METHOD NUMERICAL SIMULATIONS

The design of turbomachinery stages involves an iterative process that requires rapid computational analysis tools that the aerodynamicist can effectively utilize to arrive at a preliminary design concept that satisfies process, performance and structural criteria. Low fidelity analysis and design tools are the workhorse of the turbomachinery industry where resource and time limitations restrict the role of RANS and URANS numerical solvers until a mature design candidate has evolved. The streamline curvature method (SCM) is a numerically stable technique that implements the radial equilibrium equation, typically coupled with explicit formulation empirically calibrated to address the major aerodynamic losses that are inherent to turbomachinery flows. In the current investigation, a generic SCM solver was used to perform numerical simulations of the flow through two axial research turbines that were experimentally tested at TPFL at design and off-design conditions. In addition to running the SCM solver with the existing loss prediction formulation, an improved set of empirical correlations were implemented and shown to provide significant improvements in matching the measurement results over a wide range of operating conditions. Detailed interstage and performance comparisons were also made with RANS computational results and were found to offer an equivalent overall *predictive* quality.

4.1 Derivation of the Radial Equilibrium Equation

The detailed derivation of the radial equilibrium equation for the steady state axisymmetric flow in axial turbomachinery is presented in this section closely following the work of Wennerström [83]. The multi-stage turbomachine is simulated by a rotating *bladeless* annulus, where the flow is deflected from inlet to exit in a circumferentially uniform fashion at different streamlines stacked radially from hub to shroud. An *artificial* field force is applied to the flow to compensate for the absence of blade and endwall forces required for momentum transfer. Starting with the Cauchy equation of motion, the local time dependency is dropped ($\partial/\partial t = 0$) and the flow is assumed circumferentially symmetric ($\partial/\partial\theta = 0$). The momentum equation is resolved in an intrinsic coordinate system fixed along a streamline in the meridional plane of the turbomachine, as illustrated in Fig. 4.1. Coordinates directions, vectors and angle notations are detailed in Fig.4.2. The momentum equation tangential component in the $r - \theta - z$ coordinate system can be expressed as:

$$V_r \frac{\partial V_\theta}{\partial r} + V_z \frac{\partial V_\theta}{\partial z} + \frac{V_r V_\theta}{r} = F_\theta \tag{4.1}$$

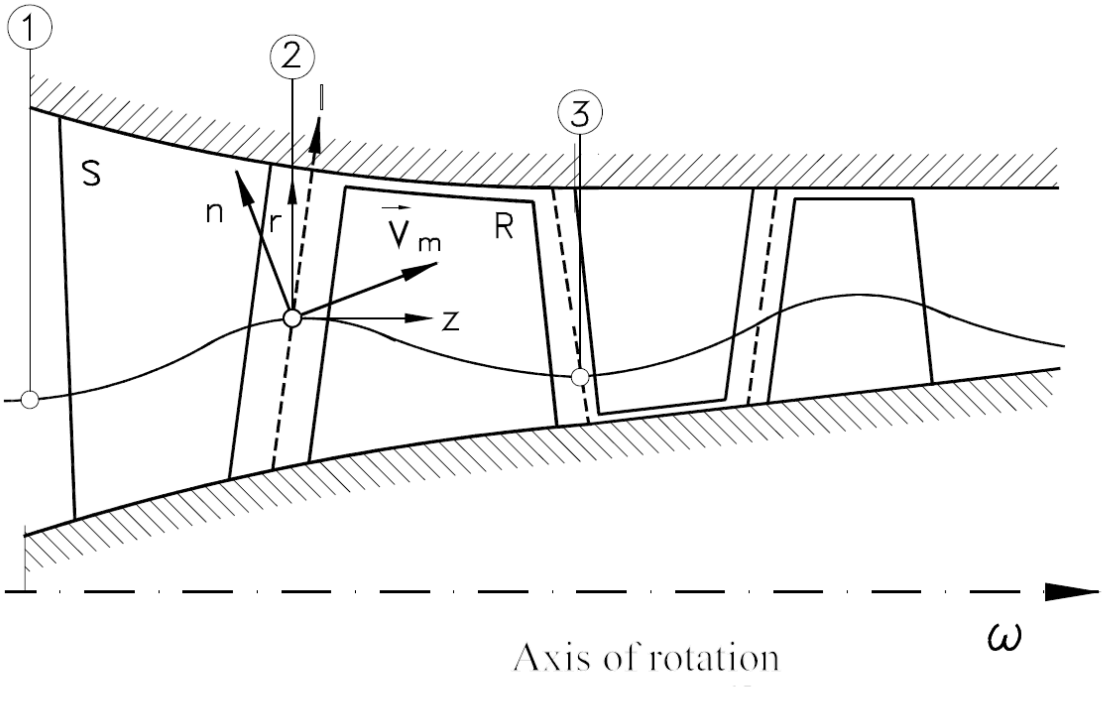
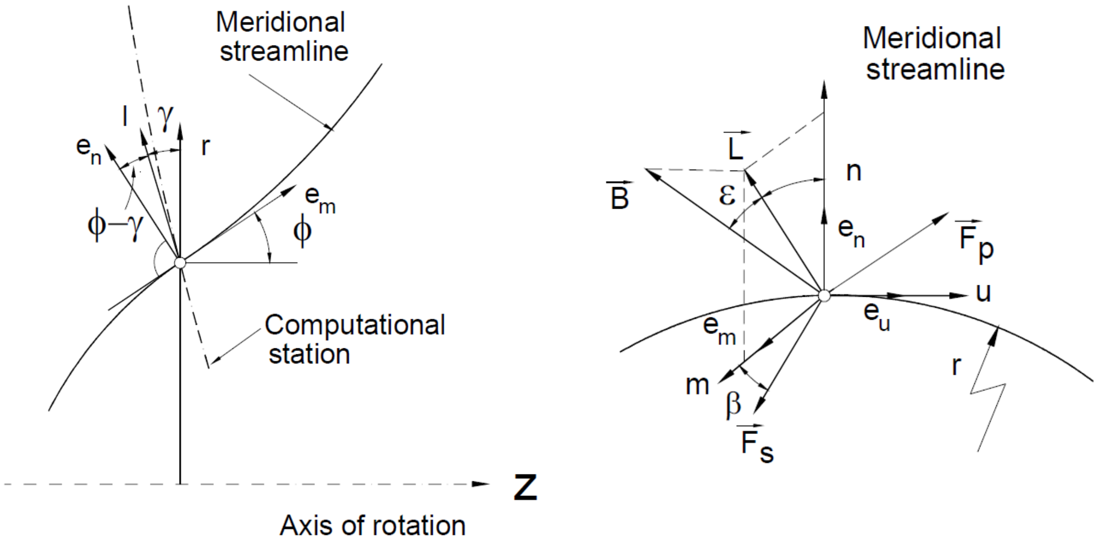


Figure 4.1: Meridional view of an axial turbomachine showing streamline directions: n = normal, m = meridional, r = radial, z = axial and l = computing station, obtained from Schobeiri [6]



(a) (b)

Figure 4.2: (a) Coordinate directions in the meridional plane, and (b) orientation of vectors with respect to $m - \theta - n$ orthogonal coordinate system, obtained from Schobeiri [6]

where F_θ represents the body forces in the θ-direction. V_r and V_z can be resolved in the $m - \theta - n$ orthogonal coordinate system shown in Fig. 4.2 as:

$$\begin{aligned} V_r &= V_m sin\phi + V_n cos\phi = V_m sin\phi \\ V_z &= V_m cos\phi - V_n sin\phi = V_m cos\phi \end{aligned} \tag{4.2}$$

since $V_n = 0$, being normal to the streamline direction. Using the coordinate relationships: $\partial m/\partial r = sin\phi$, $\partial m/\partial z = cos\phi$, $\partial n/\partial r = cos\phi$ and $\partial n/\partial z = -sin\phi$, and applying the chain rule, we can write:

$$\begin{aligned} V_m sin\phi \left(\frac{\partial V_\theta}{\partial m}\frac{\partial m}{\partial r} + \frac{\partial V_\theta}{\partial n}\frac{\partial n}{\partial r}\right) + V_m cos\phi \left(\frac{\partial V_\theta}{\partial m}\frac{\partial m}{\partial z} + \frac{\partial V_\theta}{\partial n}\frac{\partial n}{\partial z}\right) \\ + \frac{V_\theta V_m}{r} sin\phi = F_\theta \end{aligned} \tag{4.3}$$

which can be further expanded as:

$$\begin{aligned} V_m sin\phi \left(sin\phi \frac{\partial V_\theta}{\partial m} + cos\phi \frac{\partial V_\theta}{\partial n} \right) + V_m cos\phi \left(cos\phi \frac{\partial V_\theta}{\partial m} - sin\phi \frac{\partial V_\theta}{\partial n} \right) \\ + \frac{V_\theta V_m}{r} sin\phi = F_\theta \end{aligned} \tag{4.4}$$

Rearranging and applying the identity $sin^2\phi + cos^2\phi = 1$, we arrive at:

$$V_m \frac{\partial V_\theta}{\partial m} + \frac{V_\theta V_m}{r} sin\phi = F_\theta \tag{4.5}$$

Similarly the momentum equation axial component in the general $r-\theta-z$ coordinate system can be expressed as:

$$V_r \frac{\partial V_z}{\partial r} + V_z \frac{\partial V_z}{\partial z} = -\frac{1}{\rho} \frac{\partial p}{\partial z} + F_z \tag{4.6}$$

Using the chain rule and expanding we get:

$$\begin{aligned} V_m sin\phi \left(cos\phi \, sin\phi \frac{\partial V_m}{\partial m} - sin^2\phi V_m \frac{d\phi}{dm} + cos^2\phi \frac{\partial V_m}{\partial n} \right) \\ V_m cos\phi \left(cos^2\phi \frac{\partial V_m}{\partial m} - sin\phi \, cos\phi \, V_m \frac{d\phi}{dm} - sin\phi \, cos\phi \frac{\partial V_m}{\partial n} \right) \\ = -\frac{1}{\rho} \left(cos\phi \frac{\partial p}{\partial m} - sin\phi \frac{\partial p}{\partial n} \right) + F_z \end{aligned} \tag{4.7}$$

Rearranging and using the identity $d\phi = dm/r_c$, where r_c is the streamline radius of curvature, we arrive at:

$$\begin{aligned} V_m \frac{\partial V_m}{\partial m} \left(sin^2\phi \, cos\phi + cos^3\phi \right) + V_m \frac{\partial V_m}{\partial n} \left(cos^2\phi \, sin\phi + cos^2\phi \, sin\phi \right) \\ - \frac{V_m^2}{r_c} \left(sin^3\phi + cos^2\phi \, sin\phi \right) = -\frac{1}{\rho} \left(cos\phi \frac{\partial p}{\partial m} - sin\phi \frac{\partial p}{\partial n} \right) + F_z \end{aligned} \tag{4.8}$$

Using the identities $sin^2\phi\, cos\phi = cos\phi - cos^3\phi$ and $sin\phi\, cos^2\phi = sin\phi - sin^3\phi$, Eq. 4.8 can be simplified as:

$$cos\phi V_m \frac{\partial V_m}{\partial m} - sin\phi \frac{V_m^2}{r_c} = -\frac{1}{\rho}\left(cos\phi \frac{\partial p}{\partial m} - sin\phi \frac{\partial p}{\partial n}\right) + F_z \tag{4.9}$$

Finally the momentum equation radial component in the general $r - \theta - z$ coordinate system can be expressed as:

$$V_r \frac{\partial V_r}{\partial r} + V_z \frac{\partial V_r}{\partial z} - \frac{V_\theta^2}{r} = -\frac{1}{\rho}\frac{\partial p}{\partial r} + F_r \tag{4.10}$$

Expanding and applying the chain rule we get:

$$\begin{aligned} V_m sin\phi \left(sin^2\phi \frac{\partial V_m}{\partial m} + sin\phi\, cos\phi V_m \frac{d\phi}{dm} + cos\phi\, sin\phi \frac{\partial V_m}{\partial n}\right) - \frac{V_\theta^2}{r} \\ + V_m cos\phi \left(sin\phi\, cos\phi \frac{\partial V_m}{\partial m} + cos^2\phi V_m \frac{d\phi}{dm} - sin^2\phi \frac{\partial V_m}{\partial n}\right) \\ = -\frac{1}{\rho}\left(sin\phi \frac{\partial p}{\partial m} + cos\phi \frac{\partial p}{\partial n}\right) + F_r \end{aligned} \tag{4.11}$$

Rearranging Eq.4.11 yields the following:

$$\begin{aligned} V_m \frac{\partial V_m}{\partial m}\left(sin^3\phi + cos^2\phi\, sin\phi\right) + \frac{V_m^2}{r_c}\left(sin^2\phi\, cos\phi + cos^3\phi\right) \\ + V_m \frac{\partial V_m}{\partial n}\left(sin^2\phi\, cos\phi - sin^2\phi\, cos\phi\right) - \frac{V_\theta^2}{r} \\ = -\frac{1}{\rho}\left(sin\phi \frac{\partial p}{\partial m} + cos\phi \frac{\partial p}{\partial n}\right) + F_r \end{aligned} \tag{4.12}$$

Using the identities $sin^2\phi\, cos\phi = cos\phi - cos^3\phi$ and $sin\phi\, cos^2\phi = sin\phi - sin^3\phi$, Eq. 4.12 can be simplified as:

$$sin\phi V_m \frac{\partial V_m}{\partial m} + cos\phi \frac{V_m^2}{r_c} - \frac{V_\theta^2}{r} = -\frac{1}{\rho}\left(sin\phi \frac{\partial p}{\partial m} + cos\phi \frac{\partial p}{\partial n}\right) + F_r \tag{4.13}$$

Multiplying Eq. 4.9 by $cos\phi$ and Eq. 4.13 by $sin\phi$ and adding the two entities yields:

$$V_m \frac{\partial V_m}{\partial m} - \frac{V_\theta^2}{r} sin\phi = -\frac{1}{\rho}\frac{\partial p}{\partial m} + F_z cos\phi + F_r sin\phi \tag{4.14}$$

Multiplying Eq. 4.9 by $sin\phi$ and Eq. 4.13 by $cos\phi$ and subtracting the first entity from the second leads to:

$$\frac{V_m^2}{r_c} - \frac{V_\theta^2}{r} cos\phi = -\frac{1}{\rho}\frac{\partial p}{\partial n} - F_z sin\phi + F_r cos\phi \tag{4.15}$$

Define the body forces in the $m - \theta - n$ system as follows:

$$F_m = F_z cos\phi + F_r sin\phi \quad \text{and} \quad F_n = F_r cos\phi - F_z sin\phi \tag{4.16}$$

Substituting Eq. 4.16 into Eqs. 4.14 and 4.15, respectively yields:

$$\begin{aligned} V_m \frac{\partial V_m}{\partial m} - \frac{V_\theta^2}{r} sin\phi &= -\frac{1}{\rho}\frac{\partial p}{\partial m} + F_m \\ \frac{V_m^2}{r_c} - \frac{V_\theta^2}{r} cos\phi &= -\frac{1}{\rho}\frac{\partial p}{\partial n} + F_n \end{aligned} \tag{4.17}$$

Substituting $\partial r/\partial m = sin\phi$ into Eq. 4.5 and rearranging leads to:

$$\frac{V_m}{r}\left(r\frac{\partial V_\theta}{\partial m} + V_\theta \frac{\partial r}{\partial m}\right) = \frac{V_m}{r}\frac{\partial\left(rV_\theta\right)}{\partial m} = F_\theta \tag{4.18}$$

Static and total enthalpies are defined as:

$$h = pv + u \quad \text{and} \quad H = h + \frac{1}{2}V^2 \tag{4.19}$$

Differentiating the identities in Eq. 4.19 leads to the following:

$$dh = pdv + du + vdp = d\left(H - \frac{1}{2}V^2\right) = \frac{1}{\rho}dp + \delta Q = \frac{1}{\rho}dp + Tds \tag{4.20}$$

Substituting $V^2 = V_m^2 + V_n^2 + V_\theta^2 = V_m^2 + V_\theta^2$ and rearranging leads to:

$$\frac{1}{\rho}dp = dH - Tds - V_m dV_m - V_\theta dV_\theta \tag{4.21}$$

Since the computations inside a multi-stage turbomachine are carried out at a fixed grid of geometric points that belong on the meridional space between rotor or stator blades, or the meridional duct space between stator and rotor blade rows, a convenient direction is introduced as the l direction. The l station is a curve that connects a group of points on the streamlines that extend from the hub to the shroud. The l curve is often defined to trace the blade profile projection onto the meridional plane. It is generally not normal to the z-axis or the m direction, but can never be parallel to the latter. Therefore, it is commonly referred to as the quasi-orthogonal computational station. As shown in Figs. 4.1 and 4.2 the n and m directions can be projected onto the l direction using:

$$\frac{d}{dl} = \frac{dn}{dl}\frac{\partial}{\partial n} + \frac{dm}{dl}\frac{\partial}{\partial m} = cos\left(\phi - \gamma\right)\frac{\partial}{\partial n} + sin\left(\phi - \gamma\right)\frac{\partial}{\partial m} \tag{4.22}$$

Equation 4.22 can be rearranged to eliminate the n direction dependency:

$$\frac{\partial}{\partial n} = \frac{1}{cos\left(\phi - \gamma\right)}\frac{d}{dl} - tan\left(\phi - \gamma\right)\frac{\partial}{\partial m} \tag{4.23}$$

Equation 4.21 can be partially differentiated to yield:

$$\begin{aligned}\frac{1}{\rho}\frac{\partial p}{\partial n} &= \frac{\partial H}{\partial n} - T\frac{\partial s}{\partial n} - V_m\frac{\partial V_m}{\partial n} - V_\theta\frac{\partial V_\theta}{\partial n} \\ \frac{1}{\rho}\frac{\partial p}{\partial m} &= \frac{\partial H}{\partial m} - T\frac{\partial s}{\partial m} - V_m\frac{\partial V_m}{\partial m} - V_\theta\frac{\partial V_\theta}{\partial m}\end{aligned} \tag{4.24}$$

Substituting Eq. 4.24 in Eq. 4.17 we get:

$$\begin{aligned}V_m\frac{\partial V_m}{\partial m} - \frac{V_\theta^2}{r}sin\phi &= -\frac{\partial H}{\partial m} + T\frac{\partial s}{\partial m} + V_m\frac{\partial V_m}{\partial m} + V_\theta\frac{\partial V_\theta}{\partial m} + F_m \\ \frac{V_m^2}{r_c} - \frac{V_\theta^2}{r}cos\phi &= -\frac{\partial H}{\partial n} + T\frac{\partial s}{\partial n} + V_m\frac{\partial V_m}{\partial n} + V_\theta\frac{\partial V_\theta}{\partial n} + F_n\end{aligned} \tag{4.25}$$

The n component of Eq. 4.25 can expanded using Eq. 4.23:

$$\begin{aligned}\frac{V_m^2}{r_c} - \frac{V_\theta^2}{r}cos\phi = &-\frac{1}{cos\left(\phi-\gamma\right)}\frac{dH}{dl} + tan\left(\phi-\gamma\right)\frac{\partial H}{\partial m} + \frac{T}{cos\left(\phi-\gamma\right)}\frac{ds}{dl} \\ &-tan\left(\phi-\gamma\right)T\frac{\partial s}{\partial m} + \frac{V_m}{cos\left(\phi-\gamma\right)}\frac{dV_m}{dl} - tan\left(\phi-\gamma\right)V_m\frac{\partial V_m}{\partial m} \\ &+\frac{V_\theta}{cos\left(\phi-\gamma\right)}\frac{dV_\theta}{dl} - tan\left(\phi-\gamma\right)V_\theta\frac{\partial V_\theta}{\partial m} + F_n\end{aligned} \tag{4.26}$$

Multiply the m component of Eq. 4.25 by $tan\left(\phi-\gamma\right)$ and add to Eq.4.26:

$$\begin{aligned}&\frac{V_m^2}{r_c} - \frac{V_\theta^2}{r}cos\phi + tan\left(\phi-\gamma\right)V_m\frac{\partial V_m}{\partial m} - \frac{V_\theta^2}{r}sin\phi\, tan\left(\phi-\gamma\right) \\ &= -\frac{1}{cos\left(\phi-\gamma\right)}\frac{dH}{dl} + \frac{T}{cos\left(\phi-\gamma\right)}\frac{ds}{dl} + \frac{V_m}{cos\left(\phi-\gamma\right)}\frac{dV_m}{dl} \\ &\quad+\frac{V_\theta}{cos\left(\phi-\gamma\right)}\frac{dV_\theta}{dl} + tan\left(\phi-\gamma\right)F_m + F_n\end{aligned} \tag{4.27}$$

Multiply Eq. 4.27 by $cos\left(\phi-\gamma\right)$ and rearrange:

$$\begin{aligned}V_m\frac{dV_m}{dl} = &\,sin\left(\phi-\gamma\right)V_m\frac{\partial V_m}{\partial m} + cos\left(\phi-\gamma\right)\frac{V_m^2}{r_c}\frac{dH}{dl} - T\frac{ds}{dl} \\ &-sin\left(\phi-\gamma\right)F_m - cos\left(\phi-\gamma\right)F_n + \mathbf{A}\end{aligned} \tag{4.28}$$

where **A** is given as:

$$\mathbf{A} = -\frac{V_\theta^2}{r}\left(cos\left(\phi-\gamma\right)\, cos\phi + sin\left(\phi-\gamma\right)\, sin\phi\right) - V_\theta\frac{dV_\theta}{dl} \tag{4.29}$$

Using the following identities:

$$\begin{aligned} dr &= \frac{\partial r}{\partial m}dm + \frac{\partial r}{\partial n}dn = sin\phi\frac{dm}{dl}dl + cos\phi\frac{dn}{dl}dl \\ \frac{dr}{dl} &= cos\left(\phi-\gamma\right)\, cos\phi + sin\left(\phi-\gamma\right)\, sin\phi \end{aligned} \tag{4.30}$$

The expression **A** can be written as:

$$\mathbf{A} = -\frac{V_\theta^2}{r}\frac{dr}{dl} - V_\theta\frac{dV_\theta}{dl} = -\frac{V_\theta}{r}\left(V_\theta\frac{dr}{dl} + r\frac{dV_\theta}{dl}\right) = -\frac{V_\theta}{r}\frac{d}{dl}\left(V_\theta r\right) \tag{4.31}$$

Substituting Eq. 4.31 in Eq. 4.28, we arrive at:

$$\begin{aligned} V_m\frac{dV_m}{dl} = sin\left(\phi-\gamma\right)V_m\frac{\partial V_m}{\partial m} + cos\left(\phi-\gamma\right)\frac{V_m^2}{r_c} + \frac{dH}{dl} - T\frac{ds}{dl} \\ -\frac{V_\theta}{r}\frac{d}{dl}\left(V_\theta\, r\right) - sin\left(\phi-\gamma\right)F_m - cos\left(\phi-\gamma\right)F_n \end{aligned} \tag{4.32}$$

In order to transform the radial equilibrium equation into the rotating frame of reference, we introduce the relative speed and relative total enthalpy (*rothalpy*) as shown:

$$V_\theta = W_\theta + \omega\, r \quad \text{and} \quad H_r = h + \frac{W^2}{2} - \frac{U^2}{2} \tag{4.33}$$

The total enthalpy can be rewritten in terms of the relative total enthalpy as follows:

$$\begin{aligned} H = h + \frac{V_m^2}{2} + \frac{V_\theta^2}{2} &= h + \frac{W_m^2}{2} + \frac{W_\theta^2}{2} + \frac{\omega^2 r^2}{2} + \omega\, r\, W_\theta \\ H = H_r + \omega^2 r^2 + \omega\, r\, W_\theta &= H_r + \omega\, r\left(\omega\, r + W_\theta\right) \\ \frac{dH}{dl} = \frac{dH_r}{dl} + \frac{d}{dl}\left(\omega\, r\left(\omega\, r + W_\theta\right)\right) &= \frac{dH_r}{dl} + 2\,\omega^2 r\frac{dr}{dl} + \omega\frac{d}{dl}\left(r\, W_\theta\right) \end{aligned} \tag{4.34}$$

Substituting Eq. 4.34 into Eq. 4.32, and applying the entity $dr/dl = cos\gamma$ yields the radial equilibrium equation in a rotating frame of reference :

$$\begin{aligned} V_m \frac{dV_m}{dl} &= sin\left(\phi - \gamma\right) V_m \frac{\partial V_m}{\partial m} + cos\left(\phi - \gamma\right) \frac{V_m^2}{r_c} + \frac{dH_r}{dl} - T\frac{ds}{dl} \\ -2\,\omega\, W_\theta cos\gamma &- \frac{W_\theta}{r}\frac{d}{dl}\left(W_\theta\, r\right) - sin\left(\phi - \gamma\right) F_m - cos\left(\phi - \gamma\right) F_n \end{aligned} \tag{4.35}$$

Rearrange the m component of Eq. 4.25 to get:

$$T\frac{\partial s}{\partial m} = \frac{\partial H}{\partial m} - \frac{V_\theta}{r}\left(r\frac{\partial V_\theta}{\partial m} + V_\theta \frac{\partial r}{\partial m}\right) - F_m = \frac{\partial H}{\partial m} - \frac{V_\theta}{r}\frac{\partial}{\partial m}\left(V_\theta\, r\right) - F_m \tag{4.36}$$

The Euler equation of motion in a turbomachinery stage defines the total enthalpy drop at a fixed rotational speed as:

$$\begin{gathered} H_2 - H_1 = U_2 V_{\theta 2} - U_1 V_{\theta 1} = \omega\left(r_2 V_{\theta 2} - r_1 V_{\theta 1}\right) \\ \Delta H = \omega \Delta\left(r V_\theta\right) \Rightarrow dH = \omega\, d\left(r V_\theta\right) \end{gathered} \tag{4.37}$$

Further expanding Eq. 4.37 yields:

$$\begin{aligned} dH &= \frac{\partial H}{\partial m} dm + \frac{\partial H}{\partial n} dn \\ \frac{\partial H}{\partial m} &= \omega \frac{\partial}{\partial m}\left(r\, V_\theta\right) = \frac{U}{r}\frac{\partial}{\partial m}\left(r\, V_\theta\right) \end{aligned} \tag{4.38}$$

Substituting Eqs.4.18 and 4.38 into Eq. 4.36 and rearranging one can write:

$$T\frac{\partial s}{\partial m} = -\frac{W_\theta}{V_m} F_\theta - F_m \tag{4.39}$$

Noting from the velocity triangle that $tan\beta = W_\theta / V_m$ Eq. 4.39 now reads as:

$$F_m = -T\frac{\partial s}{\partial m} - F_\theta\, tan\beta \tag{4.40}$$

Since the body forces simulate the forces exerted by the blades on the fluid as it turns through a turbomachinery stage, it is beneficial to express all the forces in terms of the flow angles and blading geometry for the stator and rotor rows. Two forces can be defined to act on a streamline that is tangent to the blade camber surface, the pressure force vector $\boldsymbol{F_p}$ and the viscous force vector $\boldsymbol{F_s}$. In the right handed streamline coordinate system shown in Fig. 4.2b, m, θ and n are the principal orthogonal axes, with the corresponding unit vectors $\boldsymbol{e_m}$, $\boldsymbol{e_\theta}$ and $\boldsymbol{e_n}$ respectively. In this system, $\boldsymbol{F_s}$ represents the body force tangent to the streamline and acts to oppose the fluid motion. It is responsible for the irreversible increase in entropy along a streamline. $\boldsymbol{F_s}$ lies in the $m - \theta$ plane at an angle β from the meridional direction. The vector $\boldsymbol{L}$ coincides with the l direction described earlier and lies in the $n - m$ plane at an angle $\phi - \gamma$ from the n direction. $\boldsymbol{B}$ is a unit vector that is tangent to the blade camber surface at the point of analysis, and lies in the $l - \theta$ plane at an angle ε (lean) from the vector $\boldsymbol{L}$. It follows that $\boldsymbol{F_p}$ is normal to both $\boldsymbol{B}$ and $\boldsymbol{F_s}$ which together define the local plane of the blade camber surface at any point. $\boldsymbol{B}$ and $\boldsymbol{F_s}$ are at angle of $\pi/2 - (\phi - \gamma)$. The unit vector $\boldsymbol{B}$ can be expressed as a function of the lean angle:

$$\begin{gathered} \boldsymbol{B} = cos\,\varepsilon\,\boldsymbol{e_l} - sin\,\varepsilon\,\boldsymbol{e_\theta} \\ \boldsymbol{B} = cos\,\varepsilon\,sin\,(\phi - \gamma)\,\boldsymbol{e_m} - sin\,\varepsilon\,\boldsymbol{e_\theta} + cos\,\varepsilon\,cos\,(\phi - \gamma)\,\boldsymbol{e_n} \end{gathered} \tag{4.41}$$

where $\boldsymbol{e_l}$ is the unit vector of $\boldsymbol{L}$. The unit vector for $\boldsymbol{F_s}$ is defined as:

$$\frac{\boldsymbol{F_s}}{F_s} = cos\beta\,\boldsymbol{e_m} + sin\beta\,\boldsymbol{e_\theta} \tag{4.42}$$

Note that $\boldsymbol{F_s}$ by definition acts tangent to a streamline, and therefore has no component in the n direction. The magnitude of the vector $\boldsymbol{B} \times \boldsymbol{F_s}/F_s$ is $sin\,(\pi/2 - (\phi - \gamma))$, which can also be written as $cos\,(\phi - \gamma)$. Since $\boldsymbol{F_p}$ is normal to both $\boldsymbol{B}$ and $\boldsymbol{F_s}$, the unit vector for $\boldsymbol{F_p}$ can be expressed as:

$$\begin{aligned}\frac{\boldsymbol{F_p}}{F_p} = \frac{1}{cos\,(\phi-\gamma)}\left[\boldsymbol{B}\times\frac{\boldsymbol{F_s}}{F_s}\right] &= -sin\beta\, cos\,\varepsilon\, \boldsymbol{e_m} + cos\beta\, cos\,\varepsilon\, \boldsymbol{e_\theta} \\ &+ \left[\frac{cos\beta\, sin\,\varepsilon}{cos\,(\phi-\gamma)} + sin\beta\, cos\,\varepsilon\, tan\,(\phi-\gamma)\right]\boldsymbol{e_n}\end{aligned} \tag{4.43}$$

The body forces in the m, n and θ directions can be now decomposed along the directions of the unit vectors of $\boldsymbol{F_s}$ and $\boldsymbol{F_p}$ determined in the Eqs. 4.42 and 4.43 respectively, as shown below:

$$\begin{aligned}F_m &= F_s cos\beta - F_p sin\beta\, cos\,\varepsilon \\ F_\theta &= F_s sin\beta + F_p cos\beta\, cos\,\varepsilon \\ F_n &= F_p\left[\frac{cos\beta\, sin\,\varepsilon}{cos\,(\phi-\gamma)} + sin\beta\, cos\,\varepsilon\, tan\,(\phi-\gamma)\right]\end{aligned} \tag{4.44}$$

Inserting Eq. 4.44 into Eq. 4.40, the blade pressure and viscous forces can now be written as follows:

$$\begin{aligned}F_s &= -cos\beta\, T\frac{\partial s}{\partial m} \\ F_p &= \frac{F_\theta}{cos\beta\, cos\,\varepsilon} + \frac{sin\beta}{cos\,\varepsilon}T\frac{\partial s}{\partial m}\end{aligned} \tag{4.45}$$

Substituting Eqs. 4.44 and 4.45 into Eq. 4.35 yields the radial equilibrium equation in the final and most useful form:

$$\begin{aligned}V_m\frac{dV_m}{dl} &= V_m\, sin\,(\phi-\gamma)\frac{\partial V_m}{\partial m} + cos\,(\phi-\gamma)\frac{V_m^2}{r_c} - \frac{W_\theta}{r}\frac{d}{dl}(rW_\theta) \\ &+ \left[sin\,(\phi-\gamma)\; cos^2\beta - tan\,\varepsilon\, sin\beta\, cos\beta\right]T\frac{\partial s}{\partial m} + \frac{dH_r}{dl} \\ &-T\frac{ds}{dl} - 2\,\omega\, W_\theta\, cos\gamma - tan\,\varepsilon\, F_\theta\end{aligned} \tag{4.46}$$

Using Eq. 4.18 the circumferential force F_θ can be written as:

$$F_\theta = \frac{W_m}{r}\frac{\partial}{\partial m}\left(rW_\theta + \omega\, r^2\right) = \frac{W_m}{r}\left(W_\theta \frac{\partial r}{\partial m} + r\frac{\partial W_\theta}{\partial m} + 2\,\omega\, r\frac{\partial r}{\partial m}\right) \tag{4.47}$$

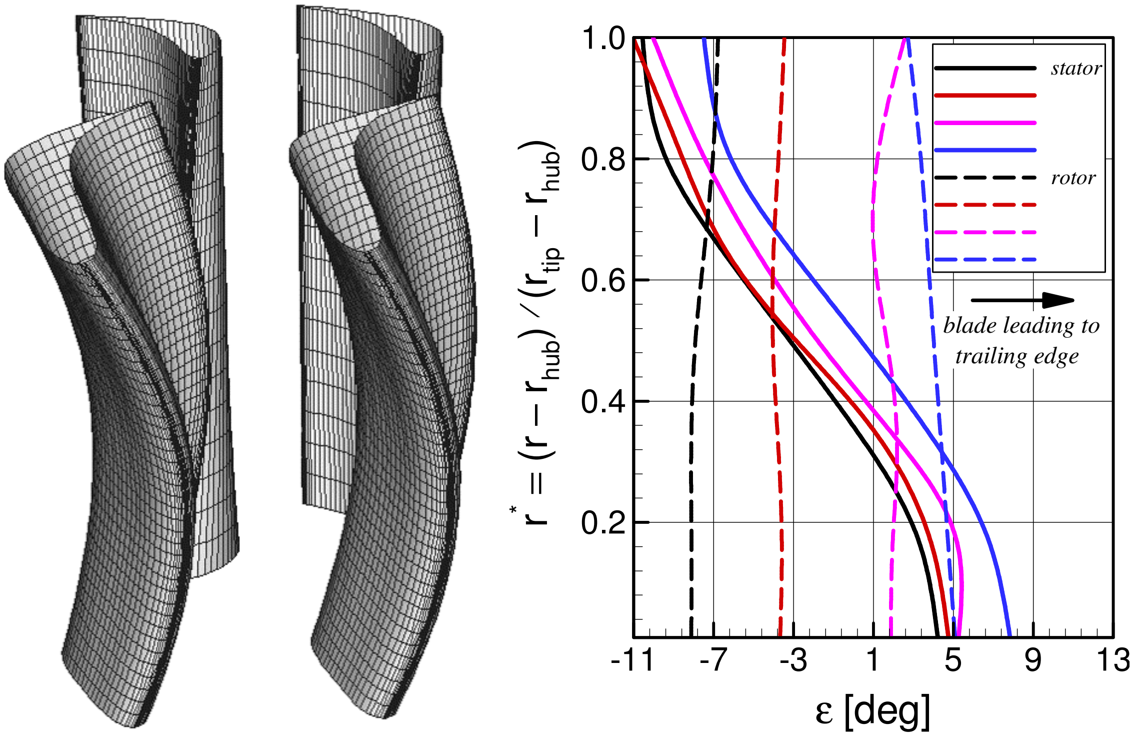
which can be further simplified to look like:

$$F_\theta = \frac{V_m}{r}\frac{\partial}{\partial m}\left(rW_\theta\right) + 2\,\omega\, V_m\, sin\phi \tag{4.48}$$

There are a number of approaches to utilize the radial equilibrium equation as a preliminary turbomachinery design or analysis tool, and the most appropriate form of the equation to be adopted depends on the specific application, the numerical scheme used as well as the geometry available for these calculations. Schobeiri [6] provides a comprehensive step by step procedure to numerically implement the radial equilibrium equation in a streamline curvature method solver, with design and analysis case studies presented. In the current investigation, a generic throughflow streamline curvature solver that implements the radial equilibrium equation in a multi-stage axial turbine was employed to simulate the flow and predict the performance of two research turbines that were experimentally tested at TPFL as design and off-design conditions.

The turbine and blade geometries were discretized at a number of axial and radial calculation stations that cover the space from hub to shroud and from inlet to exit. The average blade camber line and the blade lean angles were estimated using a 3D CAD program along each quasi-orthogonal station, as shown in Fig 4.3. To correct for the acceleration in the blade channel, the passage blockage in the circumferential direction was measured to be used by the solver to numerically correct the meridional velocity via the continuity equation. Additional blade profile measurements were made at every computational radial station including blade stagger and deflection angles and leading and trailing edge circle radii.

In order to resolve the entropy generation term in the radial equilibrium equation, the solver employs a set of empirical loss correlations that address the various loss mechanism in the cascade including the blade base profile loss (design) and incidence profile loss (off-design), the mixing loss at the blade trailing edge, secondary losses due to endwall vortex activity as well as tip leakage losses through the labyrinth seals. The entropy generation across each blade row is calculated along each streamline and applied at the trailing edge station. Two sets of loss correlations were considered



(a) Discretized stator and rotor blades

(b) Blade radial lean angle distribution

Figure 4.3: Discretized stator and rotor blades showing the lean angle distribution along the quasi-orthogonal stations, obtained from Doosan Heavy Industries & Construction [84]

in this study: Existing formulation (summarized in Appendix I) using the classic methods of Ainley and Mathieson [8] and Baljé and Binsley [85, 86], and improved formulation theorized by Schobeiri [6] and fine-tuned through extensive experimental measurements conducted at TPFL over the last three decades for multiple families of turbine blade designs [50, 62, 78, 76, 51, 65].

4.2 Improved Turbomachinery Loss Models

The mechanisms responsible for loss generation in axial flow turbines are presented in this section, with calibrated empirical correlations to estimate the individual contributions of each loss component on the entropy generation in a turbomachinery stage. This improved formulation is subsequently programmed in a streamline curvature method solver and is used to run numerical simulations to predict flow and performance in two research axial air turbines.

4.2.1 Profile Losses

Profile losses are caused by the viscous dissipation activity associated with the growth and development of the boundary layer along the pressure and suction surfaces of the stator and rotor blades. As shown in Fig. 1.2 two major mechanisms are responsible for the irreversible loss of total pressure, as friction forces are highly dominant in densely packed blade cascades whereas losses attributed to the separation of the boundary layer under an adverse pressure gradient are commonly observed in unguided flows in low solidity blade cascades.

The total pressure loss coefficient in a blade cascade is defined in Eq. 3.2. Note that the total pressures, velocities and flow angles used are the absolute values for stationary blade cascades, and the relative counterparts for rotating blade cascades. The first step in this process is to use Eqs. 3.7 and 3.8 to estimate the design total pressure loss coefficient for the optimum blade cascade solidity at the given design

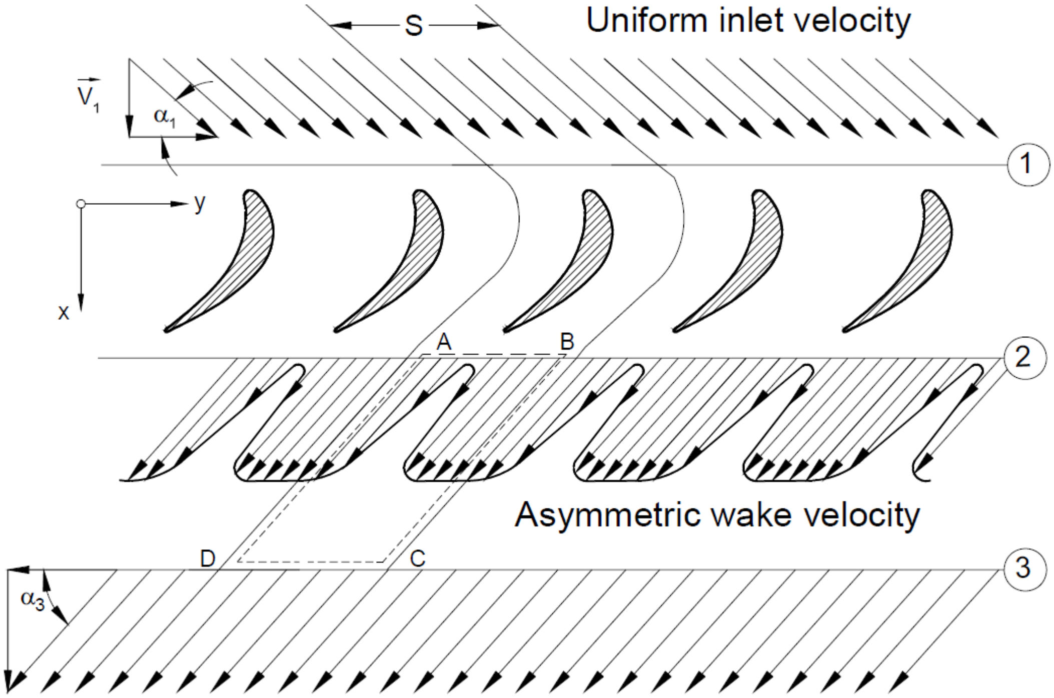


Figure 4.4: Blade cascade trailing edge wake mixing, obtained from Schobeiri [6]

flow angles. The design loss coefficient is then geometrically corrected to the actual space to chord ratio of the blade cascade. Since these correlations are based on the boundary layer momentum deficiency thickness, a Reynolds number correction is required to scale the design profile loss to the actual flow conditions in the blade cascade using Eq. 3.9, Schobeiri [87]. The final step in this procedure is to account for additional losses due to the flow incidence at the blade leading edge using Eq. 3.5, as the flow at the cascade inlet typically deviates from the inlet design flow angle, and is dictated by the row velocity triangles.

4.2.2 Trailing Edge Thickness Mixing Losses

The blade trailing edge thickness causes an exit velocity deficit that is illustrated by the wake structure shown downstream of the blade cascade in Fig. 4.4. Additional losses are caused by the mixing process that yields a uniform circumferential flow profile further downstream of the blade cascade. Schobeiri [20, 6] defines this loss as a function of the boundary layer thickness as well as the blade geometry as shown below:

$$\zeta = \frac{G_1^2 - 2RG_2 + R}{G_1^2} - cos^2\alpha_3 \left[\frac{2G_1^2 - 2G_2 + R}{G_1^2} - \frac{1}{R}\frac{G_1^2}{G_2^2}\right] \tag{4.49}$$

where the auxiliary functions G_1, R and G_2 are defined as:

$$\begin{gathered} G_1 = 1 - D - \Delta_1 \\ G_2 = 1 - D - \Delta_1 - \Delta_2 \\ \Delta_2 = \frac{\Delta_1}{H_{12}} \quad \text{and} \quad R = \frac{\rho_3}{\rho_2} \end{gathered} \tag{4.50}$$

The nondimensional identities D, Δ_1 and Δ_2 are defined below, using the angle notations in Fig. 4.4 as follows:

$$\begin{gathered} D = \frac{d}{s} = \frac{b}{s\,sin\alpha_2} \quad \text{and} \quad \Delta_1 = \frac{\delta_{1y}}{s} \quad \text{and} \quad \Delta_2 = \frac{\delta_{2y}}{s} \\ \delta_{1y} = \frac{\delta_{1s} + \delta_{1p}}{sin\alpha_2} \quad \text{and} \quad \delta_{2y} = \frac{\delta_{2s} + \delta_{2p}}{sin\alpha_2} \end{gathered} \tag{4.51}$$

The geometric dimensions d and b are detailed in Fig. 4.5. In order to estimate the displacement and momentum thicknesses, we use the skin friction coefficient for a zero pressure gradient flat plate, Schlichting [88]:

$$c_{f_{FP}} = 2\left(\frac{\delta_2}{c}\right)_{FP} = 0.074\,Re_c^{-\frac{1}{5}} \tag{4.52}$$

with c as the blade chord length, and Re_c as the Reynolds number based on the chord length and exit cascade velocity. To account for acceleration or deceleration of the flow in the curved blade channel, the skin factor must be corrected as shown

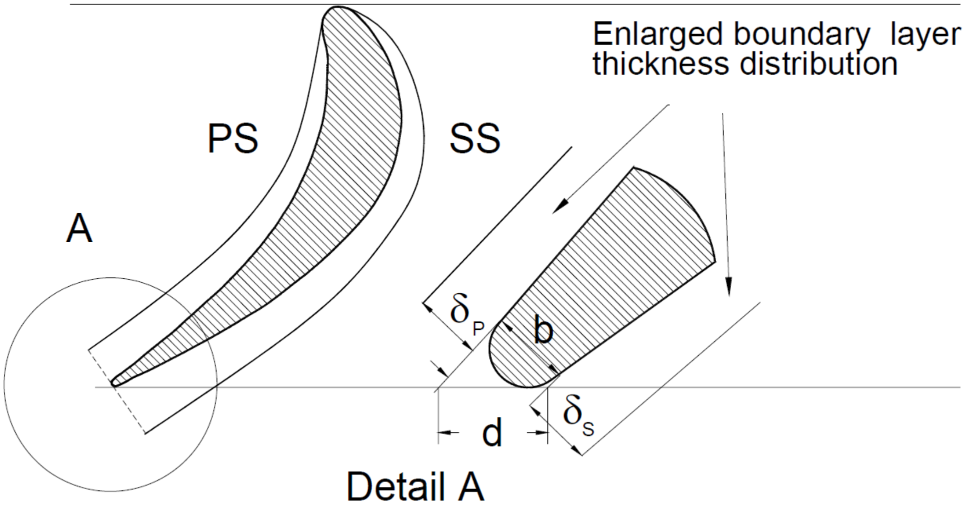


Figure 4.5: Blade trailing edge geometry detail, obtained from Schobeiri [6]

by Schobeiri [6] using the velocity ratio suggested by Kirchberg and Pfeil [32]:

$$c_f = c_{f_{FP}}\, f^{0.8} \quad \text{and} \quad \frac{\delta_2}{c} = \left(\frac{\delta_2}{c}\right)_{FP} f^{0.8} \quad \text{and} \quad f = 0.2 \sum_{n=0}^{4} \left(\frac{V_1}{V_2}\right)^n \tag{4.53}$$

where V_1 and V_2 are the cascade inlet and exit velocities for a stator blade cascade, treated as W_2 and W_3 for a rotor blade cascade. The pressure and suction momentum thicknesses can be averaged as $\delta_{2s} + \delta_{2p} \approx 2\,\delta_2$. The displacement thickness can be calculated as $\delta_1 = \delta_2\, H_{12}$, where the boundary layer form parameter varies from 2.59 for a typical laminar flow (Blasius solution) to 1.3-1.4 for typical turbulent flows. An average value $H_{12} \approx 2.0$ is an appropriate estimate for turbomachinery applications that exhibit significant laminar and transitional flow regimes.

4.2.3 Secondary Flow Losses

Complex vortex systems inherent to a high pressure turbomachinery stage are responsible for the secondary flow loss generation. Tip and hub clearance vortices which are generated by the pressure gradient across the blade pressure and suction

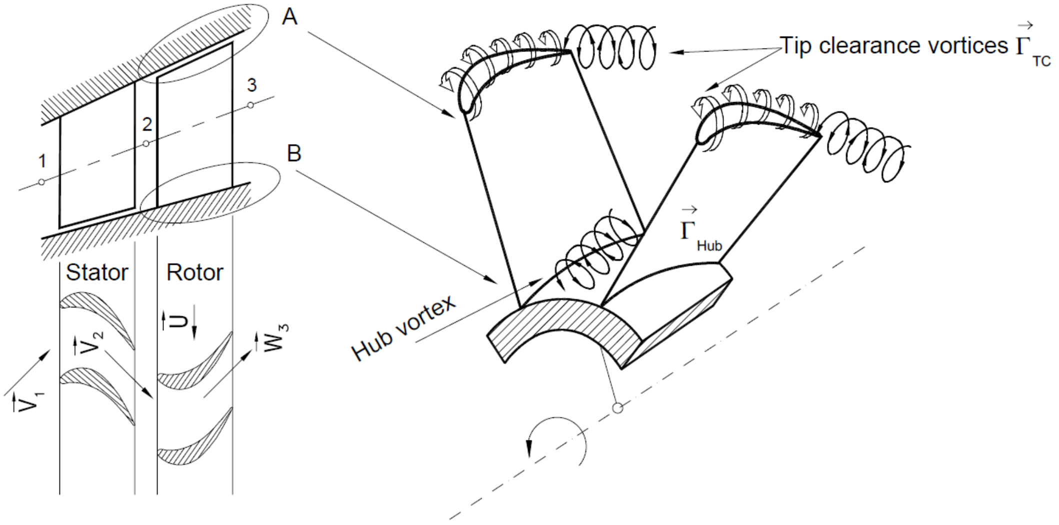


Figure 4.6: Secondary flow vortices in an unshrouded turbine stage, obtained from Schobeiri [6]

sides dominate the passage flow inside an unshrouded turbine stage as shown in Fig. 4.6. Boundary layer growth on the endwalls of a shrouded turbine stage shown in Fig. 4.7 induces tip and hub passage vortex structures that adversely influence the blade channel flows and contribute the majority of the secondary losses. Schobeiri [6] provides the total pressure loss coefficients for the stator and rotor blade cascades in an unshrouded turbine stage as introduced by Berg [31]:

$$\begin{aligned} \zeta_{stator} &= 0.676\,(cot\alpha_2 - cot\alpha_1)^2\,\frac{sin^2\alpha_2}{sin\alpha_\infty}\left[\frac{\delta-\delta_\circ}{c}\right]^{0.6} \\ \zeta_{rotor} &= 0.676\,(cot\beta_3 - cot\beta_2)^2\,\frac{sin^2\beta_3}{sin\beta_\infty}\left[\frac{\delta-\delta_\circ}{c}\right]^{0.6} \end{aligned} \tag{4.54}$$

where δ is the actual blade clearance and $\delta_\circ$ the fictive blade clearance at which the clearance loss becomes zero, and the average cascade flow angle is defined by Zweifel [1] as $cot\alpha_\infty = 1/2\,(cot\alpha_2 + cot\alpha_1)$. For shrouded turbine stages, Schobeiri [6]

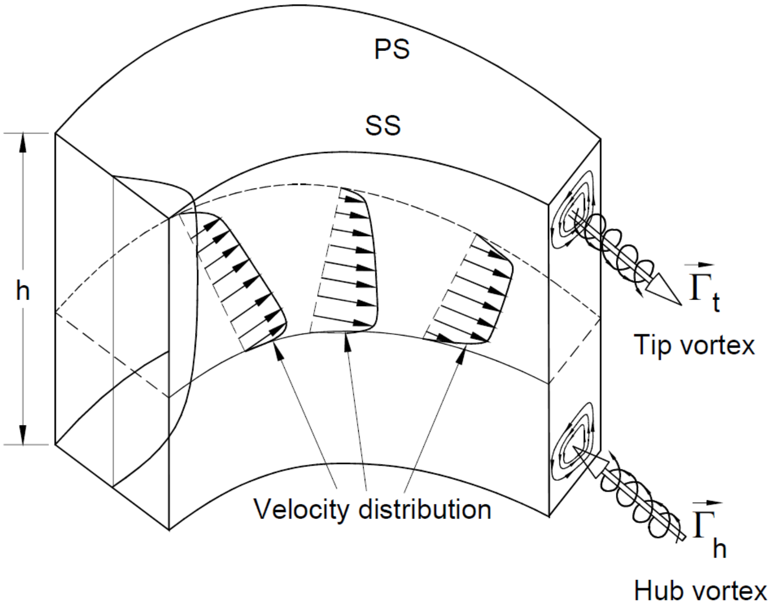


Figure 4.7: Schematic of endwall secondary vortex activity in a shrouded turbine stage, obtained from Schobeiri [6]

estimates the total pressure loss coefficient as follows:

$$\zeta = 2\frac{c}{h}c_f\left[K_1 + K_2\left(C_L\frac{c}{s}\right)^2\frac{sin\alpha_\infty}{sin^2\alpha_2}\right] \tag{4.55}$$

where the empirical coefficients are given as $K_1 = 4.65$ and $K_2 = 0.675$ from experiments by Wolf [30], and c_f is calculated using Eqs. 4.52 and 4.53.

4.2.4 Leakage Flow Losses

Flow leakage across the seals in shrouded turbine stages occurs due to the low total pressure drop across the clearance space between the stator shroud and turbine hub and rotor shroud and turbine casing. High performance labyrinth seals use a series of staggered seal teeth that dissipate the kinetic energy of the leaked flow and increase the leakage *resistance* across the path, as shown in Fig. 4.8. Seal leakage contributes to the secondary losses induced by the mixing process that takes place between the

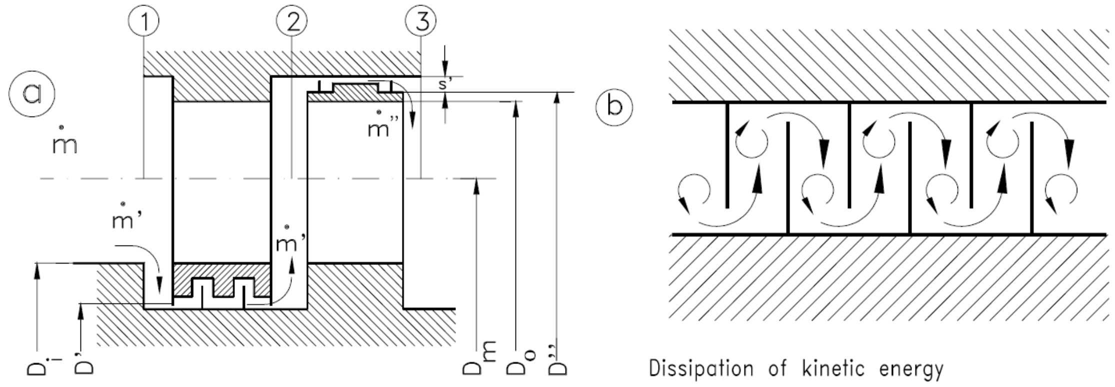


Figure 4.8: Flow leakage through the labyrinth seals of stator and rotor shrouds with a seal detail showing the reduction of pressure across the passage by means of dissipation of kinetic energy, obtained from Schobeiri [6]

low momentum high entropic leaked flow and the flow exiting the stator or rotor blade rows. Additionally the leaked flow around the rotor row does not contribute to the momentum transfer to the rotating blades, and results in a lower power output of the stage. Following the approach introduced by Pfeil [89], Schobeiri [6] provides an empirical correlation for estimating the leakage flows around the stator and rotor shrouds as follows:

$$
\begin{aligned}
\frac{\dot{m}'}{\dot{m}} &= \frac{\alpha' D' C'}{D_m h}\sqrt{\frac{2(1-r)\lambda}{n'\varphi^2}} \quad (Stator)\\
\frac{\dot{m}''}{\dot{m}} &= \frac{\alpha'' D'' C''}{D_m h}\sqrt{\frac{2r\lambda}{n''\varphi^2}} \quad (Rotor)
\end{aligned}
\tag{4.56}
$$

The values of contraction coefficients α' and α'' depend on the seal design (typically 0.8-0.93) and n' and n'' represent the number of seal teeth for the stator and rotor shrouds respectively. The stage flow and load coefficients φ and λ and degree of reaction r in a turbine stage are defined as:

$$\varphi = \frac{V_{m3}}{U_3} \quad \text{and} \quad \lambda = \frac{l_m}{U_3^2}$$
$$[5pt] l_m = U_2 V_{u2} + U_3 V_{u3} \tag{4.57}$$
$$r = \frac{\Delta h''}{\Delta h'' + \Delta h'}$$

The total pressure loss in a stator and rotor row can be calculated as:

$$\begin{aligned} \Delta P' &= 2\frac{\dot{m}'}{\dot{m}}\frac{\rho}{2}V_2^2 \\ \Delta P'' &= 2\frac{\dot{m}''}{\dot{m}}\frac{\rho}{2}W_3^2 \end{aligned} \tag{4.58}$$

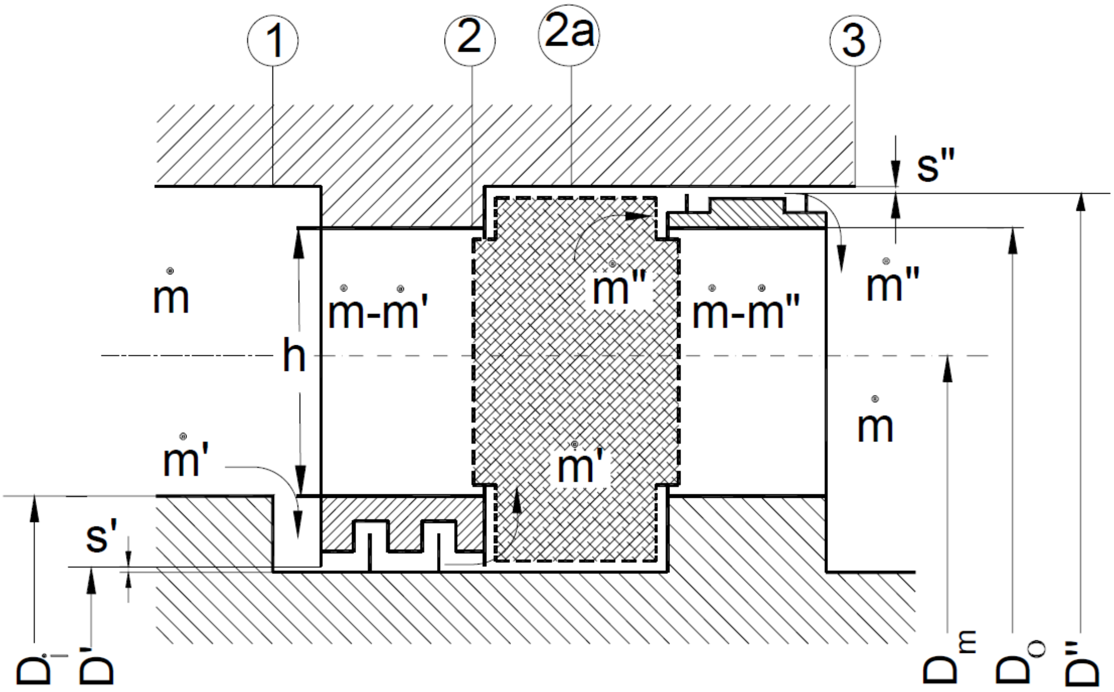


Figure 4.9: A detailed view of the stator and rotor blades and the labyrinth seal geometries, obtained from Schobeiri [6]

The total pressure loss coefficient in a turbine blade row due to flow leakage in labyrinth seals can be expressed as:

$$\zeta_{stator} = 2\,\frac{\dot{m}'}{\dot{m}} \quad \text{and} \quad \zeta_{rotor} = 2\,\frac{\dot{m}''}{\dot{m}} \tag{4.59}$$

4.2.5 Surface Roughness Correction

Surface roughness can have a detrimental effect on the aerodynamic performance of a turbomachinery stage as viscous dissipation at the blade and endwalls increases proportionally with the degradation of the surface finish. This is especially applicable to modern high temperature gas turbine stages where special protective coatings are required to shield the blade surfaces from the excessive temperatures in the hot-gas-path. Hummel et al. [90] introduced a correction to the blade row efficiency as a function of surface roughness. The model was extensively correlated to experimental measurements as shown in Fig. 4.10. The row isentropic efficiency drop can be written as a function of the surface roughness as:

$$\Delta\eta_s = \sum_{n=1}^{5} C_n \left(\frac{R_a}{c}\right)^n \tag{4.60}$$

where R_a is the centerline averaged surface roughness, and c is the blade chord length. The empirical coefficients C_n are listed in Table 4.1. $\Delta\eta_s$ is reported in percentage points. Using the stage performance relations described by Schobeiri [6] the row isentropic efficiency drop can be related to the row total pressure loss coefficient as follows:

$$Z = \frac{A + Z^*}{1 - A} \quad \text{and} \quad A = 1 - \frac{\eta_s}{\eta_s^*} = (1 + Z^*)\,\Delta\eta_s \tag{4.61}$$

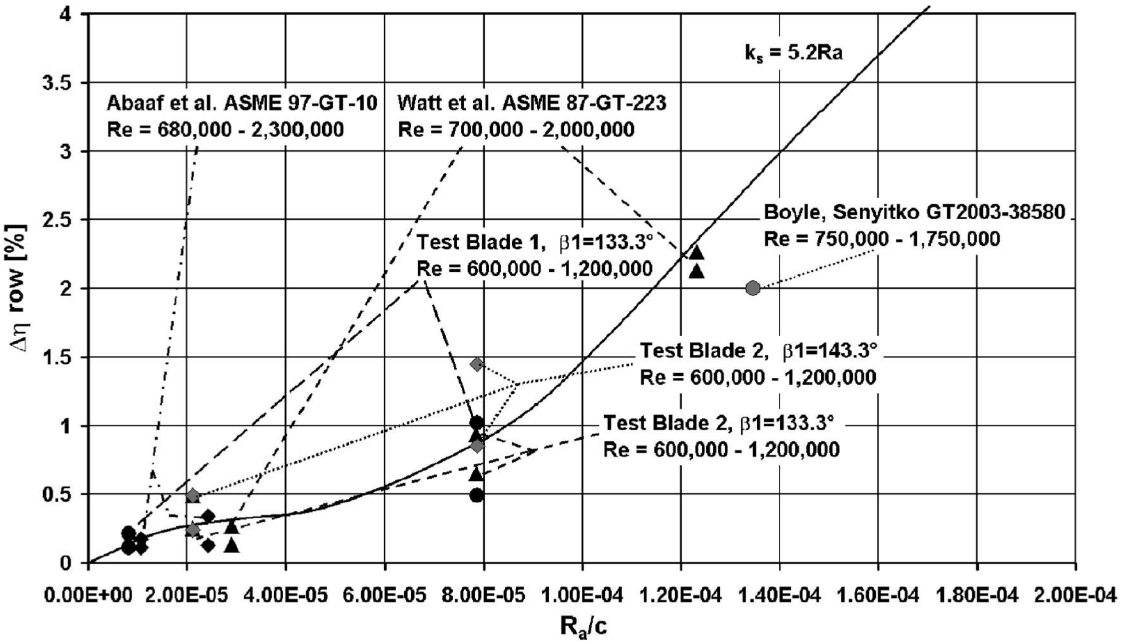


Figure 4.10: Row efficiency change due to surface roughness with experimental data correlated, obtained from Hummel et al. [90]

where Z^* represents the reference loss at zero surface roughness. The row loss coefficient is defined as:

$$Z_{stator} = \zeta_{stator} \frac{V_2^2}{2l_m} \quad \text{and} \quad Z_{rotor} = \zeta_{rotor} \frac{W_3^2}{2l_m} \tag{4.62}$$

n	$\mathrm{C_n}$
1	25065
2	-738024159
3	10725020200704
4	$-515602259271 \times 10^5$
5	$808346042568362 \times 10^5$

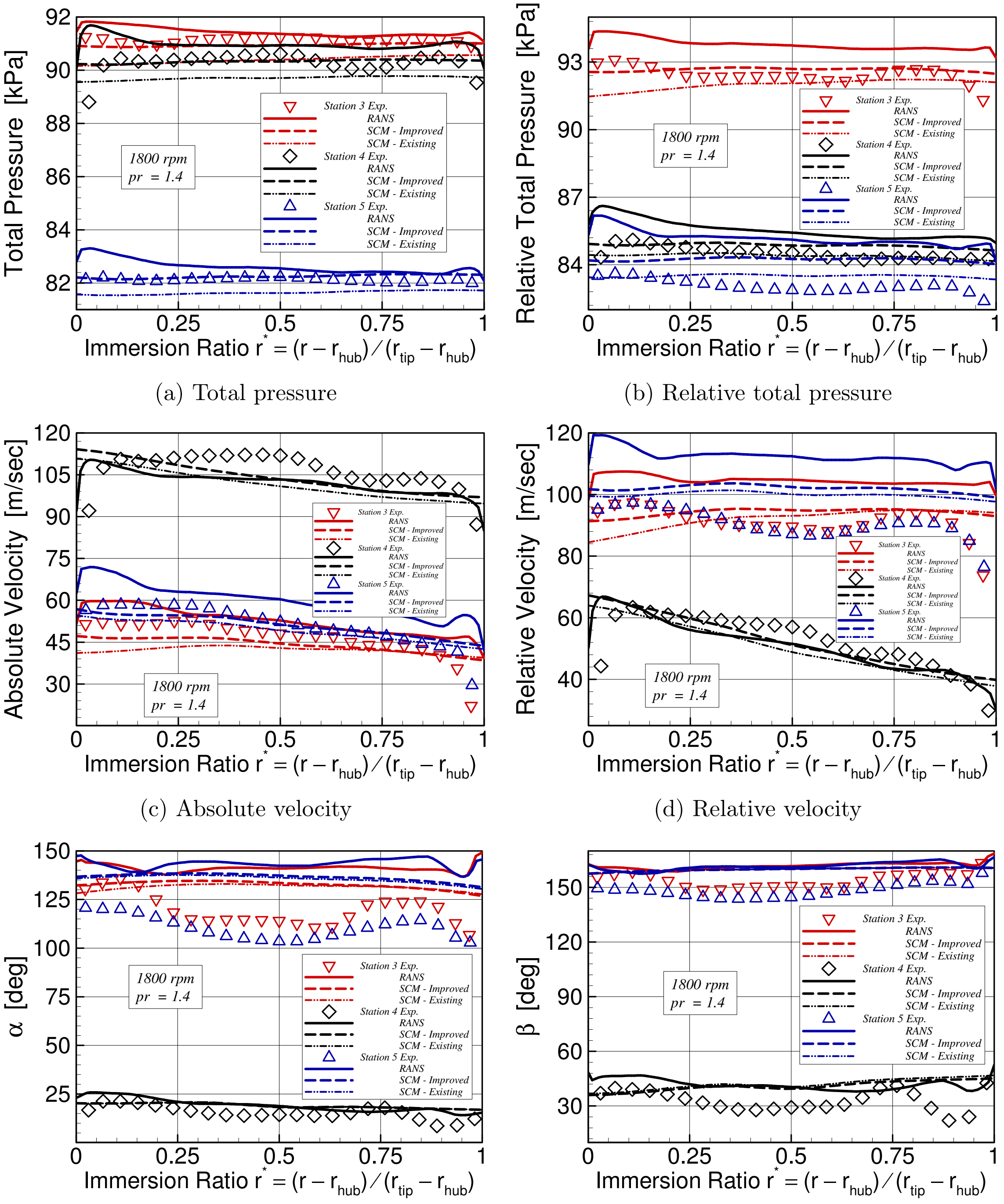
Table 4.1: Empirical coefficients for the surface roughness correction by Hummel et al. [90] presented in Eq. 4.60

4.3 Results and Discussion

Two research turbines were considered in this computational study. A three-stage axial turbine [50, 60] and a two-stage axial turbine [58, 59] that were comprehensively air tested at TPFL with extensive interstage and performance measurements conducted at design and off-design operating conditions. The experimental measurements are compared to results from RANS and streamline curvature method numerical simulations. The SCM simulations were conducted using two sets of empirical loss correlations, existing and improved, at rotational speeds and boundary conditions that were matched to the experimental measurements. A total inlet pressure and static discharge pressure target solution was adopted with ideal gas (air) as the fluid medium. Blade geometric setup data was prepared at 7 radial and 7 axial computational stations per blade creating a 49 point numerical grid per row. Additional calculation stations were positioned in the duct space between the bladed sections as well. Interstage results are presented for three rotational speeds for the three-stage axial turbine (Westinghouse 9600 blade series) at 1800, 2400 and 2600 rpm, and for a single rotational speed of 3000 rpm for the two-stage axial turbine (Doosan blades). Performance maps for both machines were created over a full range of operational conditions.

4.3.1 Detailed Interstage Experimental and Numerical Comparisons

The detailed interstage results for the Westinghouse 9600 three-stage turbine at rotational speeds of 1800, 2400 and 2600 rpm are shown in Figs. 4.11, 4.12 and 4.13 respectively. The Doosan two-stage turbine interstage data is shown in Fig. 4.14 at the design speed of 3000 rpm. Observing the total pressure trends for the 9600 turbine, RANS results are noted to be significantly over-predicted compared to the SCM improved model results for the three rotational speeds considered. The existing



(a) Total pressure

(b) Relative total pressure

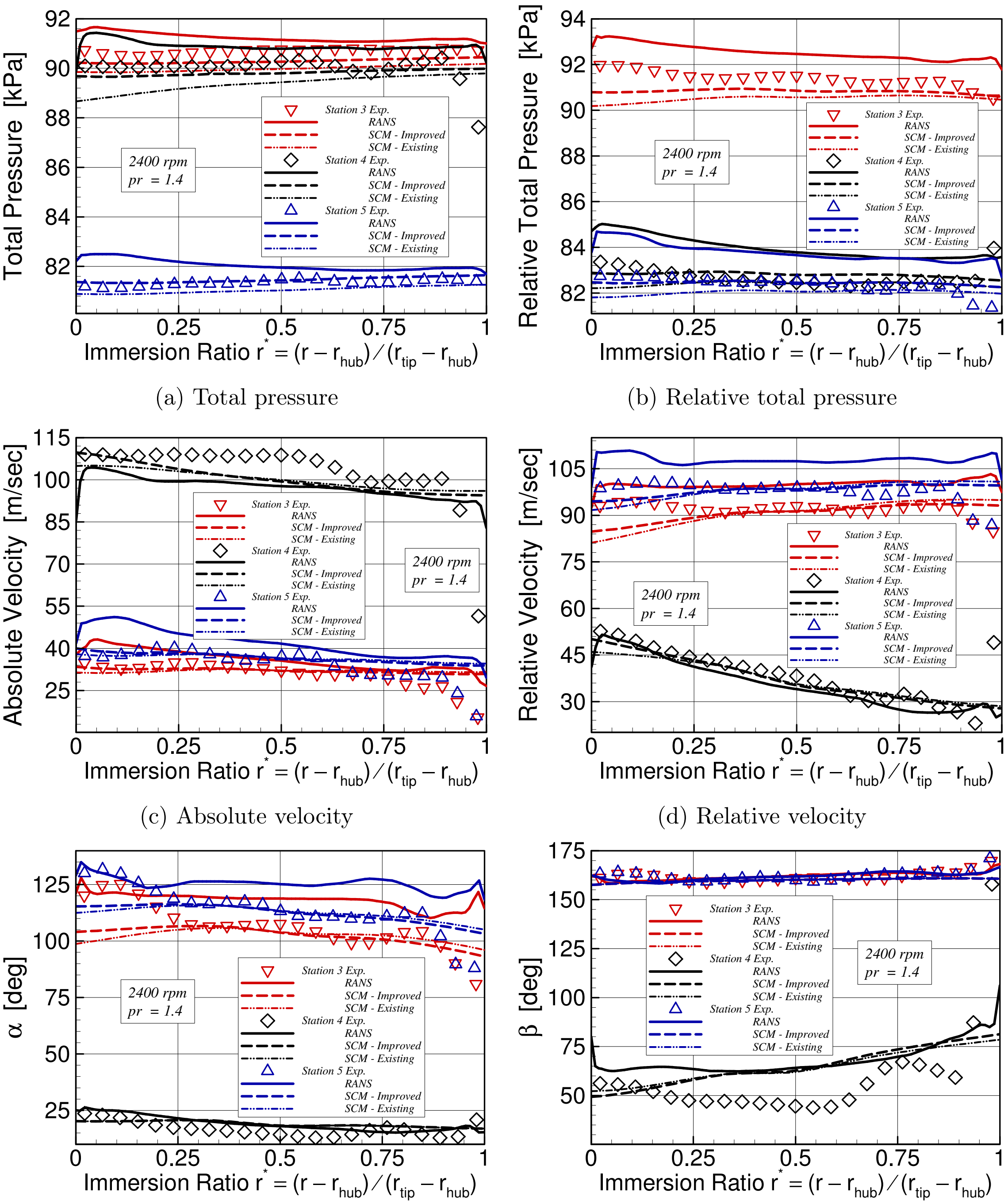
(c) Absolute velocity

(d) Relative velocity

(e) Absolute flow angle

(f) Relative flow angle

Figure 4.11: Radial distribution of measured pressures, velocities and flow angles at three interstage stations with RANS and SCM computational results for the Westinghouse 9600 three-stage turbine at 1800 rpm



(a) Total pressure

(b) Relative total pressure

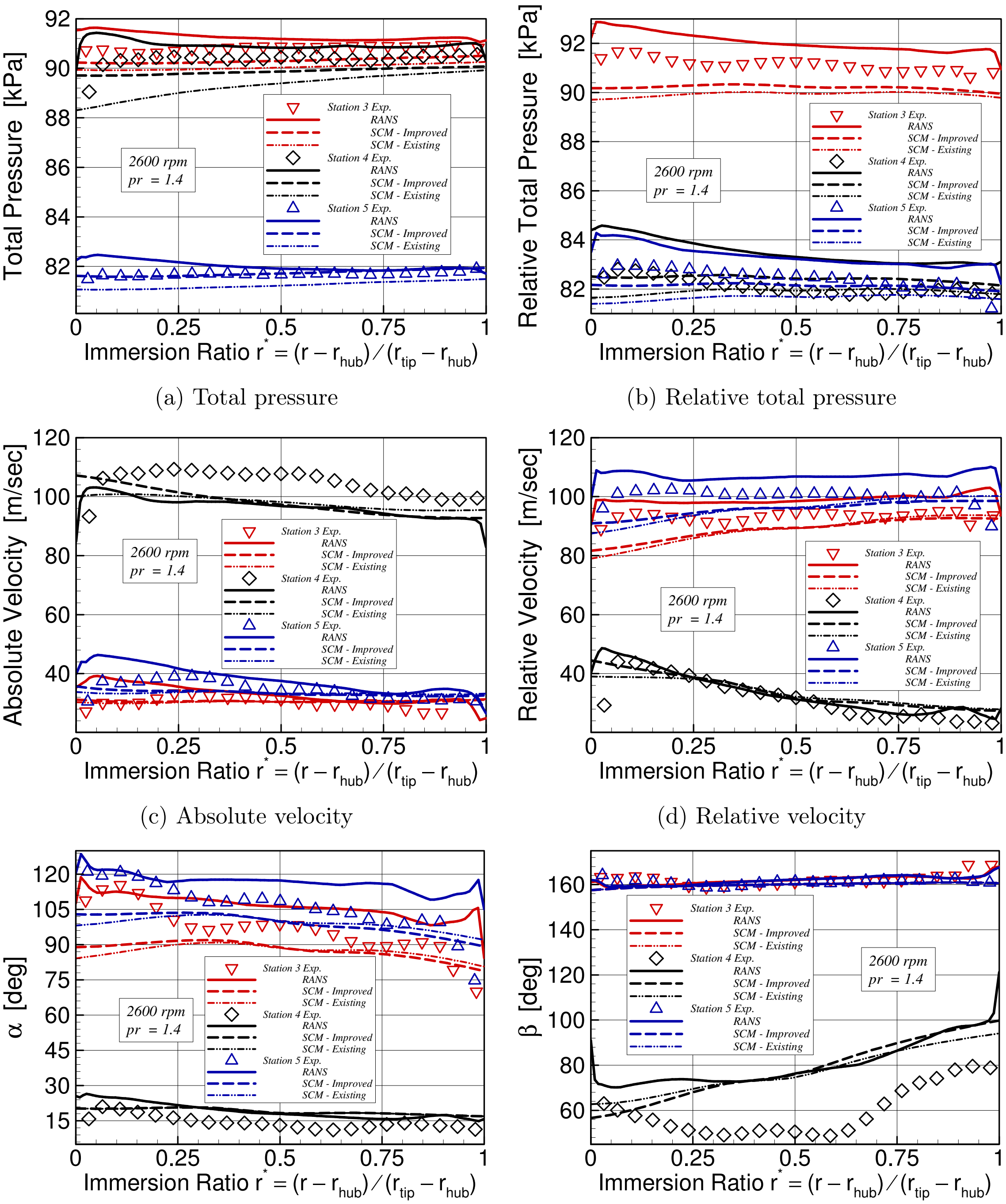
(c) Absolute velocity

(d) Relative velocity

(e) Absolute flow angle

(f) Relative flow angle

Figure 4.12: Radial distribution of measured pressures, velocities and flow angles at three interstage stations with RANS and SCM computational results for the Westinghouse 9600 three-stage turbine at 2400 rpm



(a) Total pressure

(b) Relative total pressure

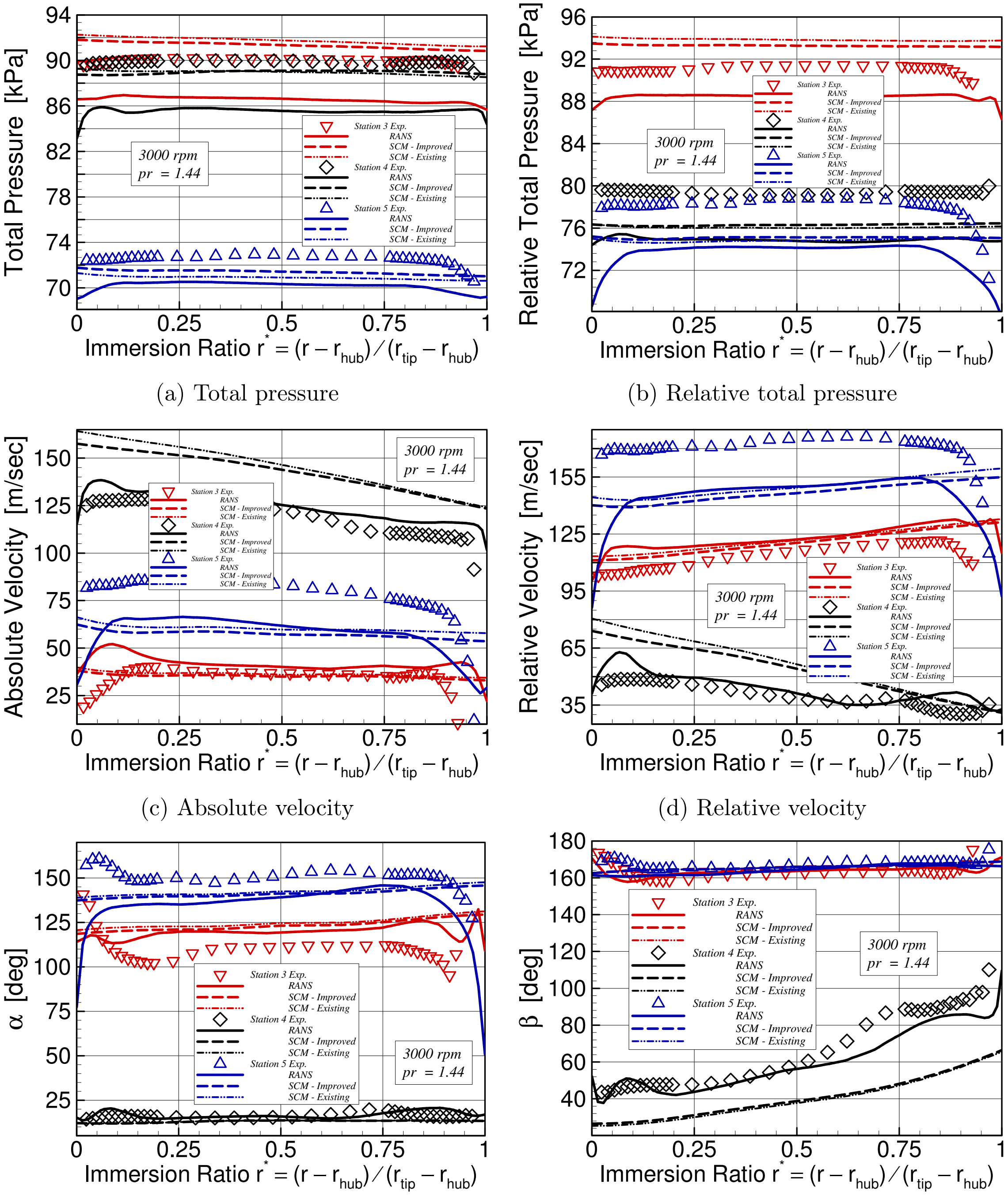
(c) Absolute velocity

(d) Relative velocity

(e) Absolute flow angle

(f) Relative flow angle

Figure 4.13: Radial distribution of measured pressures, velocities and flow angles at three interstage stations with RANS and SCM computational results for the Westinghouse 9600 three-stage turbine at 2600 rpm



(a) Total pressure

(b) Relative total pressure

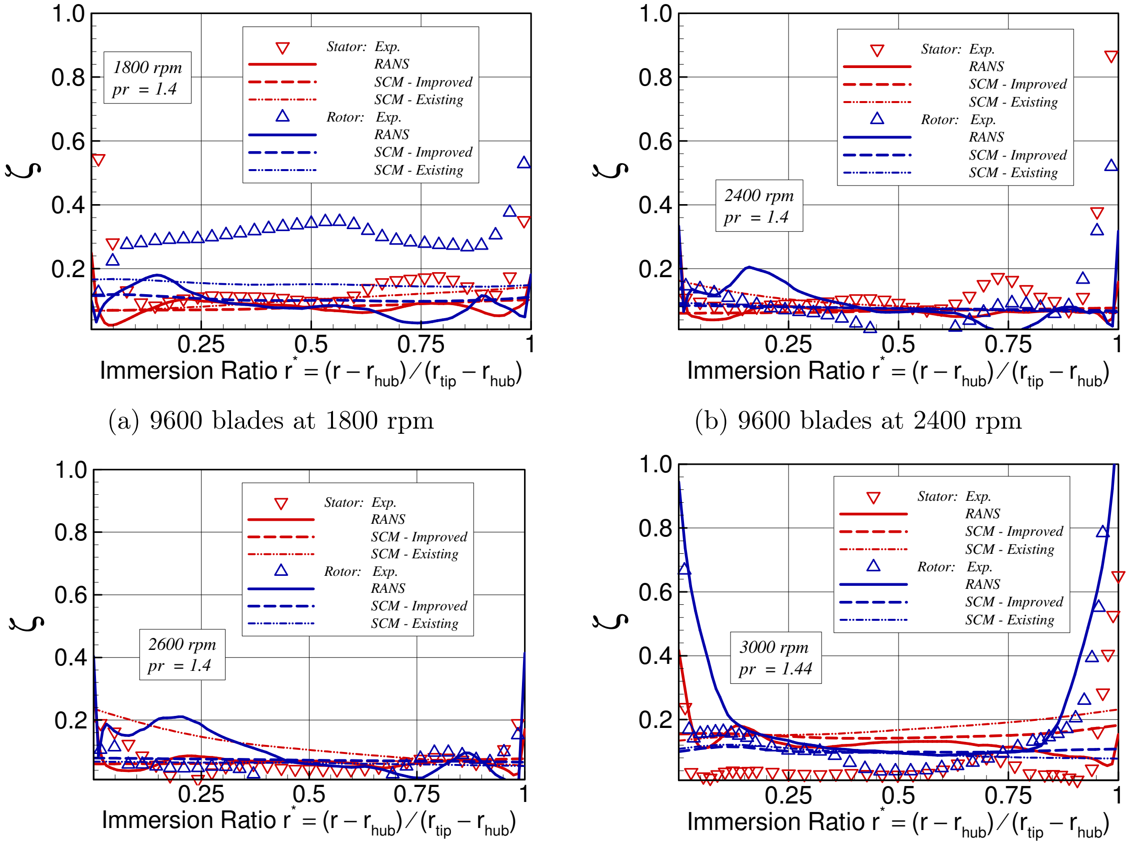
(c) Absolute velocity

(d) Relative velocity

(e) Absolute flow angle

(f) Relative flow angle

Figure 4.14: Radial distribution of measured pressures, velocities and flow angles at three interstage stations with RANS and SCM computational results for the Doosan two-stage turbine at 3000 rpm



(a) 9600 blades at 1800 rpm

(b) 9600 blades at 2400 rpm

(c) 9600 blades at 2600 rpm

(d) Doosan blades at 3000 rpm

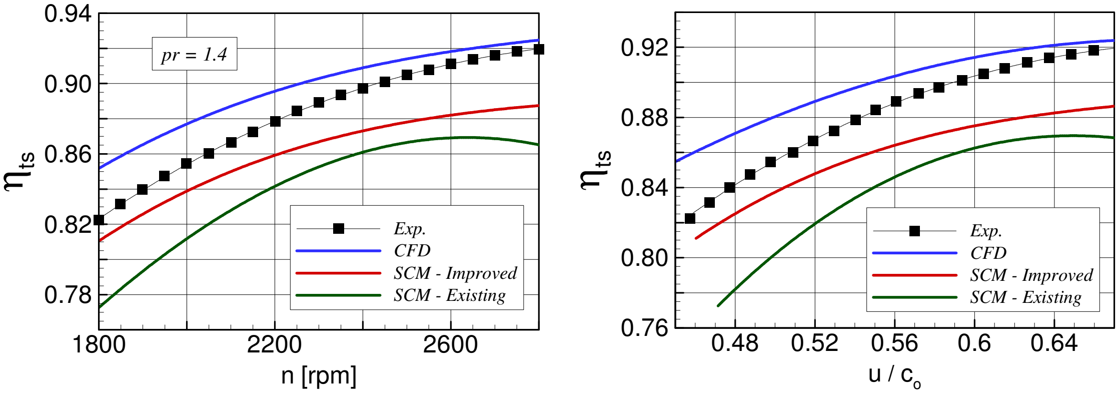
Figure 4.15: Radial distribution of measured total pressure loss coefficients for stator and rotor rows with RANS and SCM computational results for the Westinghouse 9600 three-stage turbine at 1800, 2400 and 2600 rpm and the Doosan two-stage turbine at 3000 rpm

SCM model results on the other hand are under-predicted, especially at the hub region. This trend is reversed when observing the corresponding results for the Doosan turbine. The relative total pressure trends exhibit a similar behavior, although the deviation of the RANS results is more pronounced especially at stations 4 and 5 (second stator and rotor exit planes). For all cases considered, the SCM improved model achieves a closer match with the total and relative total pressure measurements.

The absolute velocity trends in the 9600 turbine data at the three operating speeds indicate comparable results of the RANS and SCM (existing and improved) simulations. There is a modest over-prediction at stations 3 and 5 (rotor exit planes) and under-prediction for the stator exit plane at station 4. The relative velocity trends are similar, with a slightly more pronounced over-prediction of RANS results at the rotor exit planes compared to SCM (existing and improved). The Doosan turbine results on the other hand, show good agreement of RANS and SCM simulations with measurements at station 3 (first rotor exit plane) and a severe under-prediction at station 5 (second rotor exit plane). The RANS results show better agreement with the measurements at station 4 (stator exit plane) where the SCM simulations are severely over-predicted. The improved SCM model results only offer limited advantages over the existing formulation for absolute and relative velocity calculations.

The flow angle trends in the 9600 turbine data at the three operating speeds are in line with the observations made about the absolute and relative velocities. Acceptable agreement between RANS, SCM and measurements is noted at station 4, and a modest mismatch at stations 3 and 5. The Doosan turbine data shows good agreement between RANS, SCM and measurements at station 4 for the absolute flow angle and stations 3 and 5 for the relative flow angle. The absolute flow angle for both RANS and SCM is significantly mismatched with the measurements at stations 3 and 5. The relative flow angle at station 4 shows a significant under-prediction by the SCM simulations compared to acceptable agreement between RANS and measurements.

Figure 4.15 shows the radial distributions of the total pressure loss coefficient for the second stator and rotor rows for both turbines. The lack of a proper wall treatment in the SCM solver leads to a loss profile that is mostly flat and inadequate at providing any insight into the placement and intensity of the wake and vortex structures in the flow field. The SCM results are thus only useful to furnish integral



(a) Efficiency versus rotational speed

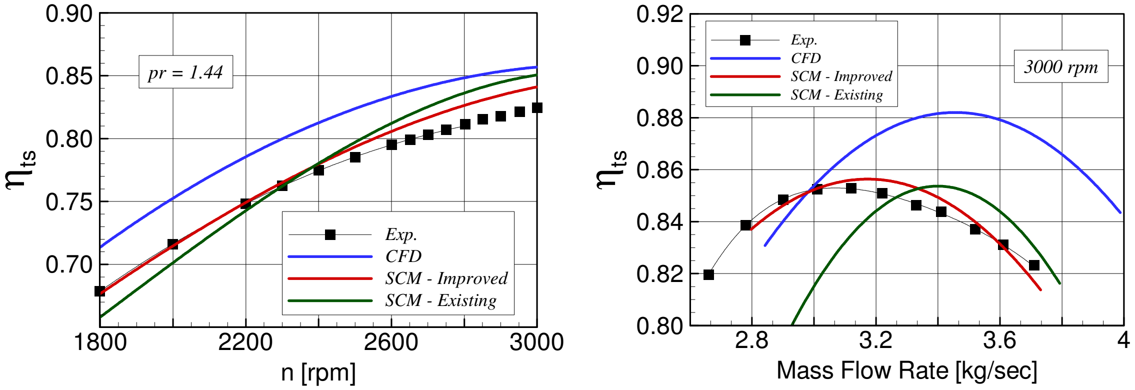
(b) Efficiency versus $u/c_\circ$

Figure 4.16: Measured turbine performance for full operating range with RANS and SCM computational results for the Westinghouse 9600 three-stage air turbine

loss values for the entire blade rows. Table 4.2 shows the averaged total pressure loss coefficients of the stator and rotor blade rows for the four interstage cases considered. A careful review of the averaged values reveals that the improved SCM results offer an equivalent row performance predictive *quality* when compared with RANS simulations.

	9600		9600		9600		Doosan	
Stage 2	1800 rpm		2400 rpm		2600 rpm		3000 rpm	
	ζ_s	ζ_r	ζ_s	ζ_r	ζ_s	ζ_r	ζ_s	ζ_r
RANS	0.0780	0.0905	0.0655	0.0893	0.0650	0.0994	0.1174	0.1169
SCM - Imp.	0.0812	0.1044	0.0677	0.0759	0.0666	0.0721	0.1512	0.1023
SCM - Ext.	0.1013	0.1517	0.0911	0.0755	0.1169	0.0613	0.1735	0.0963

Table 4.2: Averaged total pressures loss coefficients for second stator and rotor rows showing RANS and SCM computational results for the Westinghouse 9600 three-stage turbine and Doosan two-stage turbine



(a) Efficiency versus rotational speed

(b) Efficiency versus mass flow rate

Figure 4.17: Measured turbine performance for full operating range with RANS and SCM computational results for the Doosan two-stage air turbine

4.3.2 Experimental and Numerical Performance Comparisons

Figure 4.16 shows the isentropic efficiency of the Westinghouse 9600 three-stage turbine plotted against a full operating range of 1800 to 2800 rpm at a fixed pressure ratio of 1.4. The RANS results show an over-prediction of the turbine performance compared to measurements, whereas the the improved SCM simulations under-predict the efficiency. A notable trend shows a better agreement between improved SCM and measurements at lower rotational speeds compared to a better matching of RANS results to experimental data at higher rotational speeds.

Figure 4.17 shows the full performance map for the Doosan two-stage turbine as a function of rotational speed at a fixed pressure ratio, and the mass flow rate at a fixed rotational speed. Reviewing the data in Fig. 4.17a reveals that the RANS calculations over-predict the isentropic efficiency by about 2.5%. Figure 4.17b further demonstrates that the RANS results over-predict the mass flow rate by about 10%. For the same conditions, the improved SCM simulations achieve an acceptable match to the experimental measurements at design and off-design speeds and mass flow rates.

For both the Westinghouse and Doosan turbines, the performance predictions of the improved SCM results are significantly in better agreement with the measurements compared to the existing SCM simulations. These findings highlight the role of a properly calibrated set of empirical loss correlations in enabling a low fidelity preliminary design and analysis tool to provide reliable and accurate performance predictions that match or exceed the quality of those afforded by advanced RANS and URANS solvers that require significantly more time and computational resources.

4.4 Conclusions

A generic streamline curvature method solver was utilized to run flow simulations in two axial research turbines that were extensively air tested at TPFL for a wide range of design and off-design operating conditions. Existing and improved versions of the throughflow solver were used to generate full performance maps as well detailed interstage calculations of flow parameters for a total of four case studies. Results from experimental measurements were compared to the SCM calculations in order to validate the improved loss formulation that was implemented to cover the major aerodynamic loss mechanisms in axial turbines. Comparisons to RANS results were made to quantify and qualify the advantage of employing fully viscous Navier-Stokes solvers that require significantly more time and computational resources. A properly calibrated SCM solver was found capable of providing multi-stage as well as row performance predictions that are of equivalent or higher *quality* as RANS results. However, SCM simulations failed at capturing the detailed flow structure inside the turbine blade channel due to the lack of proper wall treatment and boundary layer resolution.

5. CONCLUSION

High pressure axial turbines exhibit inherently complex three-dimensional, viscous, compressible and transient flow fields. Steady and unsteady Reynolds-averaged Navier-Stokes simulations have recently become prominent in the turbomachinery design and analysis process. However, the effective implementation of RANS and URANS simulations requires substantial time and computational resources and the need to actively calibrate the solvers to produce a posteriori reliable design and off-design performance predictions. As a result, aerodynamicists continue to rely on traditional low fidelity design and analysis computational tools for predicting the performance of multi-stage axial turbine stages.

The current investigation provides a systemic approach to properly characterize the attributes of the individual loss mechanisms that are responsible for irreversible entropy generation in stator and rotor blade rows. This is accomplished through a series of experimental measurements of the growth of aerodynamic losses in two-dimensional linear blade cascades and three-dimensional annular turbine cascades. Findings from these studies were used to calibrate empirical loss correlations that were subsequently implemented in a generic streamline curvature method solver to predict the performance of axial turbines at various operating conditions.

The first part of this study involved conducting a comprehensive set of steady state aerodynamic tests on a two-stage research axial turbine at the Turbomachinery Performance and Flow Research Laboratory of Texas A&M University. This was comprised of detailed interstage measurements at three blade exit planes using a set of automatically traversed calibrated five-hole probes, as well as intensive turbine performance measurements for a wide operational range of rotational speeds and pressure ratios. Steady and transient simulation results from a concurrent computational investigation revealed significant quantitative discrepancies with the experimental measurements, which were attributed to the inadequacies of the turbulence and transition models employed by the numerical solvers.

In the second part of this study two linear blade cascades were experimentally and numerically investigated to evaluate the profile losses in a simulated axial turbine blade row at design and off-design conditions, by varying the cascade stagger and flow incidence angles respectively. Detailed measurements of the flow field upstream and downstream of the test section were conducted using linearly traversed five-hole probes. An instrumented blade was equipped with pressure taps, and used to measure the pressure profile around the pressure and suctions sides, enabling a better understanding of the flow physics in the blade passage. The numerical and experimental findings of this study were used to calibrate two existing design and off-design profile loss correlations that would subsequently be employed as reliable empirical predictive modules.

In the final part of this study a streamline curvature method solver was utilized to run flow simulations in two axial research turbines for a wide range of design and off-design operating conditions. Experimentally tuned empirical correlations that address profile, secondary, blade trailing edge mixing and seal leakage loss components were embedded in the throughflow solver. Full performance maps and detailed interstage calculations of flow parameters were generated and compared to experimental and RANS results. SCM simulations demonstrated acceptable agreement with experimental measurements, and exceeded the performance predictive *quality* of RANS at a fraction of the time and computational resources.

REFERENCES

[1] Zweifel, O., 1945. "The Spacing of Turbo-machine Blading Especially with Large Angular Deflection". *The Brown Boveri Review,* **32**, pp. 436–444.

[2] Christiani, K., 1928. "Experimentelle Untersuchungen eines Tragflächenprofils bei Gitteranordnung". *Luftfahrtforschung,* **2**.

[3] Keller, C., 1923. "Axialgebläse vom Standpunk der Tragflächentheorie". Dissertation, ETH Zürich, Zürich, Germany.

[4] Pfeil, H., 1964. "Beitrag zur Optimalen Auslegung von Axial-Schaufelgittern". Dissertation D17, Technischen Hochschule Darmstadt, Darmstadt, Germany.

[5] Pfeil, H., 1969. "Optimale Primärverluste in Axialgittern und Axialstufenvon Strömungsmaschinen". *VD-Forschungsheft,* **535**.

[6] Schobeiri, M. T., 2012. *Turbomachinery Flow Physics and Dynamic Performance, Second and Enhanced Edition.* Springer-Verlag, Berlin, Germany.

[7] Sauer, H., Schmidt, R., and Vogeler, K., 2012. "Influence of Chord Length and Inlet Boundary Layer on the Secondary Losses of Turbine Blades". *Journal of Turbomachinery,* **134**, pp. 011015–1–011015–9.

[8] Ainley, D. G., and Mathieson, G. C. R., 1951. "A Method of Performance Estimation for Axial Flow Turbines". *British Aeronautical Research Council, Reports and Memoranda,* **2974**.

[9] Dunham, J., and Came, P. M., 1970. "Improvements to the Ainley/Mathieson Method of Turbine Performance Prediction". *Journal of Engineering for Power,* **92**, pp. 252–256.

[10] Craig, H. R. M., and Cox, H. J. A., 1971. "Performance Estimate of Axial Flow Turbines". *Proc. Institution of Mechnical Engineers,* **185**(32), pp. 407–424.

[11] Kacker, S. C., and Okapuu, U., 1982. "A Mean-Line Prediction Method for Axial Flow Turbine Efficiency". *Journal of Engineering for Power,* **104**, pp. 111–119.

[12] Mukhtarov, M. K., and Krichakin, V. I., 1969. "Procedure of Estimating Flow Section Losses in Axial Flow Turbines When Calculating Their Characteristics". *Teploenergetika,* **16**, pp. 76–79.

[13] Martelli, F., and Boretti, A. A., 1987. "Development of an Experimental Correlation for Transonic Turbine Flow". *Journal of Turbomachinery,* **109**, pp. 246–250.

[14] Moustapha, S. H., Kacker, S. C., and Tremblay, B., 1990. "An Improved Incidence Losses Prediction Method for Turbine Airfoils". *Journal of Turbomachinery,* **112**, pp. 267–276.

[15] Benner, M. W., Sjolander, S. A., and Moustapha, S. H., 1997. "Influence of Leading-Edge Geometry on Profile Losses in Turbines at Off-Design Incidence: Experimental Results and an Improved Correlation". *Journal of Turbomachinery,* **119**, pp. 193–200.

[16] Jouini, D. B. M., Sjolander, S. A., and Moustapha, S. H., 2001. "Aerodynamic Performance of a Transonic Turbine Cascade at Off-Design Conditions". *Journal of Turbomachinery,* **123**, pp. 510–518.

[17] Jouini, D. B. M., Sjolander, S. A., and Moustapha, S. H., 2002. "Midspan Flow-Field Measurements for Two Transonic Linear Turbine Cascades at Off-Design Conditions". *Journal of Turbomachinery,* **124**, pp. 176–186.

[18] Li, S. M., Chu, T. L., Yoo, Y. S., and Ng, W. F., 2004. “Transonic and Low Supersonic Flow Losses of Two Steam Turbine Blades at Large Incidences”. *Journal of Fluids Engineering,* **126**, pp. 966–975.

[19] Denton, J. D., 1993. “The 1993 IGTI Scholar Lecture: Loss Mechanisms in Turbomachines”. *Journal of Turbomachinery,* **115**, pp. 621–656.

[20] Schobeiri, M. T., 1985. “Einfluß der Hinterkantenausblasung auf die Hinter den Gekühlten Schaufeln Entstehenden Mischungsverluste”. *Forschung im Ingenieurwesen,* **51**, pp. 25–28.

[21] Schobeiri, M. T., 1989. “Optimum Trailing Edge Ejection for Cooled Gas Turbine Blades”. *Journal of Turbomachinery,* **111**, pp. 510–514.

[22] Schobeiri, M. T., and Pappu, K., 1999. “Optimization of Trailing Edge Ejection Mixing Losses Downstream of Cooled Turbine Blades: A Theoretical and Experimental Study”. *Journal of Fluids Engineering,* **121**, pp. 118–125.

[23] Squire, H. B., and Winter, K. G., 1951. “The Secondary Flow in a Cascade of Airfoils in a Non-uniform Stream”. *Journal of the Aeronautical Sciences,* **18**.

[24] Smith, L. H., 1955. “Secondary Flow in Axial-Flow Turbomachinery”. *ASME Transactions*, pp. 1065–1076.

[25] Hawthorne, W. R., 1955. “Rotational Flow Through Cascades”. *The Quarterly Journal of Mechanics and Applied Mathematics,* **3**, pp. 266–279.

[26] Langston, L., Nice, M. L., and Hooper, R. M., 1977. “Three-Dimensional Flow Within a Turbine Cascade Passage”. *Journal of Engineering for Power,* **99**, pp. 21–28.

[27] Sieverding, C., and den Bosch, P. V., 1983. “The Use of Colored Smoke to Visualize Secondary Flows in a Turbine Blade Cascade”. *Journal of Fluid Mechanics,* **134**, pp. 85–89.

[28] Sieverding, C., 1985. “Recent Progress in the Understanding of Basic Aspects of Secondary Flows in Turbine Blade Passages”. *Journal of Turbomachinery,* **107**, pp. 248–257.

[29] Sharma, O. P., and Butler, T. L., 1987. “Predictions of Endwall Losses and secondary Flows in Axial Flow Turbine Cascades”. *Journal of Turbomachinery,* **109**, pp. 229–236.

[30] Wolf, H., 1960. “Die Randverluste in Geraden Schaufelgittern”. Dissertation, Technische Universität Dresden, Dresden, Germany.

[31] Berg, H., 1973. “Untersuchungen über den Einfluß der Leistungszahl auf Verluste in Axialturbinen”. Dissertation D16, Technischen Hochschule Darmstadt, Darmstadt, Germany.

[32] Kirchberg, G., and Pfeil, H., 1971. “Einfluß der Stufenkenngrössen auf die Auslegung von HD-Turbinen”. *Zeitschrift Konstruktion,* **23**.

[33] Schobeiri, T., and Abouelkheir, M., 1992. “Row-by-Row Off-Design Performance Calculation Method for Turbines”. *Journal of Propulsion and Power,* **8**, pp. 823–828.

[34] Emunds, R., Jennions, I. K., Bohn, D., and Gier, J., 1999. “The Computation of Adjacent Blade-Row Effects in a 1.5-Stage Axial FlowTurbine”. *Journal of Turbomachinery,* **121**, pp. 1–10.

[35] Baldwin, B. S., and Lomax, H., 1978. “Thin Layer Approximation and Algebraic Model for Separated Turbulent Flow”. *AIAA Paper 78-0257.*

[36] Hodson, H. P., Huntsman, I. I., and Steele, A. B., 1994. “An Investigation of Boundary Layer Development in a Multistage LP Turbine”. *Journal of Turbomachinery,* **116**, pp. 375–383.

[37] Halstead, D. E., Wisler, D. C., Okiishi, T. H., Walker, G. J., Hodson, H. P., and Shin, H. W., 1997. “Boundary Layer Development in Axial Compressors and Turbines: Part 1 of 4 – Composite Picture”. *Journal of Turbomachinery,* **119**, pp. 114–127.

[38] Halstead, D. E., Wisler, D. C., Okiishi, T. H., Walker, G. J., Hodson, H. P., and Shin, H. W., 1997. “Boundary Layer Development in Axial Compressors and Turbines: Part 2 of 4 – Compressors”. *Journal of Turbomachinery,* **119**, pp. 426–444.

[39] Halstead, D. E., Wisler, D. C., Okiishi, T. H., Walker, G. J., Hodson, H. P., and Shin, H. W., 1997. “Boundary Layer Development in Axial Compressors and Turbines: Part 3 of 4 – LP Turbines”. *Journal of Turbomachinery,* **119**, pp. 225–237.

[40] Halstead, D. E., Wisler, D. C., Okiishi, T. H., Walker, G. J., Hodson, H. P., and Shin, H. W., 1997. “Boundary Layer Development in Axial Compressors and Turbines: Part 4 of 4 – Computations and Analyses”. *Journal of Turbomachinery,* **119**, pp. 128–139.

[41] Arndt, N., 1993. “Blade Row Interaction in a Multistage Low-Pressure Turbine”. *Journal of Turbomachinery,* **115**, pp. 137–146.

[42] Schobeiri, M. T., John, J., and Pappu, K., 1996. “Development of Two-Dimensional WakesWithin Curved Channels: Theoretical Framework and Experimental Investigation”. *Journal of Turbomachinery,* **118**, pp. 506–518.

[43] Schobeiri, M. T., Pappu, K., and John, J., 1995. “Theoretical and Experimental Study of Development of Two-Dimensional Steady and Unsteady Wakes Within Curved Channels”. *Journal of Fluids Engineering,* **117**, pp. 593–598.

[44] Schobeiri, M. T., Read, K., and Lewalle, J., 1995. “Effect of Unsteady Wake Passing Frequency on Boundary Layer Transition, Experimental Investigation and Wavelet Analysis”. ASME 95-GT-437, presented at the International Gas Turbine and Aero-Engine Congress and Exposition, Houston, Texas, June 5-8, 1995.

[45] Schobeiri, M. T., Chakka, P., and Pappu, K., 1998. “Unsteady Wake Effects on Boundary Layer Transition and Heat Transfer Characteristics of a Turbine Blade”. ASME paper: 98-GT-291, presented at the IGTI Gas Turbine and Aeroengine Congress and Exhibition, Stockholm, June 2-5, 1998.

[46] Wright, L., and Schobeiri, M. T., 1999. “The Effect of Periodic Unsteady Flow on Aerodynamics and Heat Transfer on a Curved Surface”. *Journal of Heat Transfer,* **121**, pp. 22–33.

[47] Chakka, P., and Schobeiri, M. T., 1999. “Modeling Unsteady Boundary Layer Transition on a Curved Plate Under Periodic Unsteady Flow Conditions: Aerodynamic and Heat Transfer Investigations”. *Journal of Turbomachinery,* **121**, pp. 88–97.

[48] Camci, C., Dey, D., and Kavurmacioglu, L., 2005. “Aerodynamics of Tip Leakage Flows Near Partial Squealer Rims in an Axial Flow Turbine Stage”. *Journal of Turbomachinery,* **127**, pp. 14–24.

[49] Krishnababu, S. K., Newton, P. J., Dawes, W. N., Lock, G. D., Hodson, H. P., Hannis, J., and Whitney, C., 2009. “Aerothermal Investigations of Tip Leakage

Flow in Axial Flow Turbines – Part I: Effect of Tip Geometry and Tip Clearance Gap". *Journal of Turbomachinery,* **131**, pp. 011006–1–011006–14.

[50] Schobeiri, M. T., Gilarranz, J. L., and Johansen, E. S., 2004. "Aerodynamic and Performance Behavior of a Three-Stage High Efficiency Turbine at Design and Off-Design Operating Points". *International Journal of Rotating Machinery,* **10**, pp. 33–44.

[51] Schobeiri, M. T., Suryanarayanan, A., Jermann, C., and Neuenschwander, T., 2004. "A Comparative Aerodynamic and Performance Study of a Three-Stage High-Pressure Turbine with 3-D Bowed Blades and Cylindrical Blades". GT2004-53650, presented at ASME TURBO EXPO 2004, Vienna, Austria, June 14-17, 2004.

[52] Persico, G., Gaetani, P., and Osnaghi, C., 2009. "A Parametric Study of the Blade Row Interaction in a High Pressure Turbine Stage". *Journal of Turbomachinery,* **131**, pp. 031006–1–031006–13.

[53] Porreca, L., Behr, T., Schlienger, J., Kalfas, A. I., Abhari, R. S., Ehrhard, J., and Janke, E., 2005. "Fluid Dynamics and Performance of Partially and Fully Shrouded Axial Turbines". *Journal of Turbomachinery,* **127**, pp. 668–678.

[54] Pullan, G., 2006. "Secondary Flows and Loss Caused by Blade Row Interaction in a Turbine Stage". *Journal of Turbomachinery,* **128**, pp. 484–491.

[55] Behr, T., Kalfas, A. I., and Abhari, R. S., 2007. "Unsteady Flow Physics and Performance of a One-and-1/2-Stage Unshrouded High Work Turbine". *Journal of Turbomachinery,* **129**, pp. 348–359.

[56] Gaetani, P., Persico, G., Dossena, V., and Osnaghi, C., 2007. "Investigation of the Flow Field in a High-Pressure Turbine Stage for Two Stator-Rotor Ax-

ial Gaps – Part I: Three-Dimensional Time-Averaged Flow Field". *Journal of Turbomachinery,* **129**, pp. 572–579.

[57] Gaetani, P., Persico, G., Dossena, V., and Osnaghi, C., 2007. "Investigation of the Flow Field in a High-Pressure Turbine Stage for Two Stator-Rotor Axial Gaps – Part II: Unsteady Flow Field". *Journal of Turbomachinery,* **129**, pp. 580–590.

[58] Schobeiri, M. T., Abdelfattah, S., and Chibli, H., 2012. "Investigating the Cause of Computational Fluid Dynamics Deficiencies in Accurately Predicting the Efficiency and Performance of High Pressure Turbines: A Combined Experimental and Numerical Study". *Journal of Fluids Engineering,* **134**, pp. 101104–1–101104–12.

[59] Schobeiri, M. T., and Abdelfattah, S., 2013. "On the Reliability of RANS and URANS Numerical Results for High-Pressure Turbine Simulations: A Benchmark Experimental and Numerical Study on Performance and Interstage Flow behavior of high-pressure turbines at design and off-design conditions using two different turbine designs". *Journal of Turbomachinery,* **135**, pp. 061012–1–061012–12.

[60] Abdelfattah, S., and Schobeiri, M. T., 2012. "Experimental and Numerical Investigations of Aerodynamic Behavior of a Three-Stage High-Pressure Turbine at Different Operation Conditions". *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science,* **226**, pp. 1535–1549.

[61] Ahn, J., Schobeiri, M. T., Han, J. C., and Moon, H. K., 2007. "Effect of Rotation on Leading Edge Region Film Cooling of a Gas Turbine Blade with Three Rows

of Film Cooling Holes”. *International Journal of Heat and Mass Transfer,* **50**, pp. 15–25.

[62] Suryanarayanan, A., Mhetras, S. P., Schobeiri, M. T., and Han, J. C., 2009. “Film-Cooling Effectiveness on a Rotating Blade Platform”. *Journal of Turbomachinery,* **131**, pp. 1–12.

[63] Rezasoltani, M., Lu, K., Schobeiri, M. T., and Han, J.-C., 2015. “A Combined Experimental and Numerical Study of the Turbine Blade Tip Film Cooling Effectiveness Under Rotation Condition”. *Journal of Turbomachinery,* **137**, pp. 051009–1–051009–12.

[64] Rezasoltani, M., Schobeiri, M. T., and Han, J. C., 2014. “Experimental Investigation of the Effect of Purge Flow on Film Cooling Effectiveness on a Rotating Turbine With Nonaxisymmetric End Wall Contouring”. *Journal of Turbomachinery,* **136**, pp. 091009–1–091009–10.

[65] Lu, K., Rezasoltani, M., Schobeiri, M., and Han, J., 2014. “A Combined Numerical and Experimental Study of the Effect of Non-Axisymmetric Contouring on Performance and Film Cooling Behavior of a Rotating Turbine Endwall”. GT-2014-25659, presented at ASME TURBO EXPO 2014, Dusseldorf, Germany, June 16-30, 2014.

[66] Sharma, K., 2011. “Inter-Stage and Performance Test of a Two-Stage High Pressure Turbine”. Master’s Thesis, Texas A&M University, College Station, TX.

[67] Rubner, K., and Bohn, D., 1972. “Verfahren für die Auswertung der Messergebnisse von Strömungssonden”. *Flugwissenschaft,* **20**, pp. 35–42.

[68] Treaster, A. L., and Yocum, A. M., 1978. The Calibration and Application of Five-Hole Probes. Tech. rep., Navy Department, Naval Sea Systems Command.

[69] Ligrani, P. M., Singer, B. A., and Baun, L. R., 1989. "Miniature Five-Hole Pressure Probe for Measurement of Three Mean Velocity Components in Low-Speed Flows". *Journal of Physics E: Scientific Instruments,* **22**, pp. 868–876.

[70] Reichert, B. A., and Wendt, B. J., 1994. A New Algorithm for Five-Hole Probe Calibration, Data Reduction, and Uncertainty Analysis. Tech. rep., NASA.

[71] Morrison, G. L., Schobeiri, M. T., and Pappu, K. R., 1998. "Five-Hole Pressure Probe Analysis Technique". *Flow Measurement and Instrumentation,* **9**, pp. 153–158.

[72] Traupel, W., 1977. *Thermische Turbomaschinen.* Springer-Verlag, Berlin, Germany.

[73] Dzung, L. S., 1971. "Konsistente Mittewerte in der Theorie der Turbomaschinen für Kompressible Medien". *BBC-Mitt.,* **58**, pp. 485–492.

[74] Kline, S. J., and Mcklintock, F. A., 1953. "Describing Uncertainties in Single-Sample Experiments". *Mech. Eng. ASME,* **75**, pp. 3–8.

[75] Menter, F. R., 1993. "Zonal Two-Equation $\kappa - \omega$ Turbulence Models for Aerodynamic Flows". *AIAA Paper 93-2906.*

[76] Schobeiri, M. T., Öztürk, B., and Ashpis, D., 2005. "On the Physics of Flow Separation Along a Low Pressure Turbine Blade Under Unsteady Flow Conditions". *Journal of Fluids Engineering,* **127**, pp. 503–513.

[77] Chibli, H. A., Abdelfattah, S. A., Schobeiri, M. T., and Kang, C., 2009. "An Experimental and Numerical Study of the Effects of Flow Incidence Angles on the Performance of a Stator Blade Cascade of a High Pressure Steam Turbine".

GT-2009-59131, presented at ASME TURBO EXPO 2009: Power for Land, Sea, and Air, Orlando, Florida, June 8-12, 2009.

[78] Schobeiri, M. T., John, J., and Pappu, K., 1997. "Experimental Study on the Effect of Unsteadiness on Boundary Layer Development on a Linear Turbine Cascade". *Journal of Experiments in Fluids,* **23**, pp. 303–316.

[79] Schobeiri, M. T., and Chakka, P., 2002. "Prediction of Turbine Blade Heat Transfer and Aerodynamics Using Unsteady Boundary Layer Transition Model". *International Journal of Heat Transfer,* **45**, pp. 815–829.

[80] Schobeiri, M. T., Pappu, K., and Wright, L., 1995. "Experimental Study of the Unsteady Boundary Layer Behavior on a Turbine Cascade". 95-GT-2435, presented at the International Gas Turbine and Aero-Engine Congress and Exposition, Houston, TX, June 5-8, 1995.

[81] Roger, P., Sjolander, S., and Moustapha, S., 1992. "Establishing Two-Dimensional Flow in a Large-Scale Planar Turbine Cascade". AIAA-92-3066, presented at AIAA/SAE/ASEE 28th Joint Propulsion Conference and Exhibit, Nashville, TN, July 6-8, 1992.

[82] Zehner, P., 1980. "Vier-Quadranten Charakteristiken Mehrstufiger Axialer Turbinen". *VDI-Forsch Berlin,* **75**.

[83] Wennerström, A., 1974. *On the Treatment of Body Forces in the Radial Equilibrium Equation of Turbomachinery.* Traupel-Festschrift, Juris-Verlag, Zürich, Germany.

[84] Schobeiri, M., Kim, D., Hwang, H., and Chibli, H., 2008. Development, Design and Experimental Performance Evaluation and Analysis of Three-Dimensional, High Efficiency Blades for High Pressure Steam Turbines of Doosan Heavy In-

dustries & Construction. Tech. rep., Turbomachinery Performance and Flow Research Laboratory, Texas A&M University.

[85] Baljé, O. E., and Binsley, R. L., 1968. "Axial Turbine Performance Evaluation. Part A–Loss-Geometry Relationships". *Journal of Engineering for Power,* **90**, pp. 341–348.

[86] Baljé, O. E., and Binsley, R. L., 1968. "Axial Turbine Performance Evaluation. Part B–Optimization With and Without Constraints". *Journal of Engineering for Power,* **90**, pp. 349–359.

[87] Schobeiri, M. T., 2017. *Gas Turbine Design, Components and System Design Integration.* Springer International Publishing, Cham, Switzerland.

[88] Schlichting, H., 1979. *Boundary Layer Theory.* McGraw-Hill, New York, NY.

[89] Pfeil, H., 1971. "Zur Frage der Spaltverluste in Labyrinthgedichteten Hochdruckstufen von Dampfturbinen". *Zeitschrift Konstruktion,* **23**.

[90] Hummel, F., Lötzerich, M., Cardamone, P., and Fottner, L., 2005. "Surface Roughness Effects on Turbine Blade Aerodynamics". *Journal of Turbomachinery,* **127**, pp. 453–461.

APPENDIX A

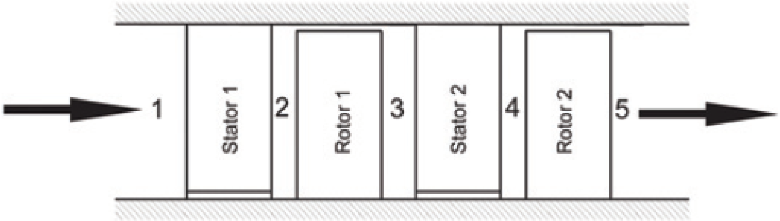
MEASUREMENT UNCERTAINTY SUMMARY

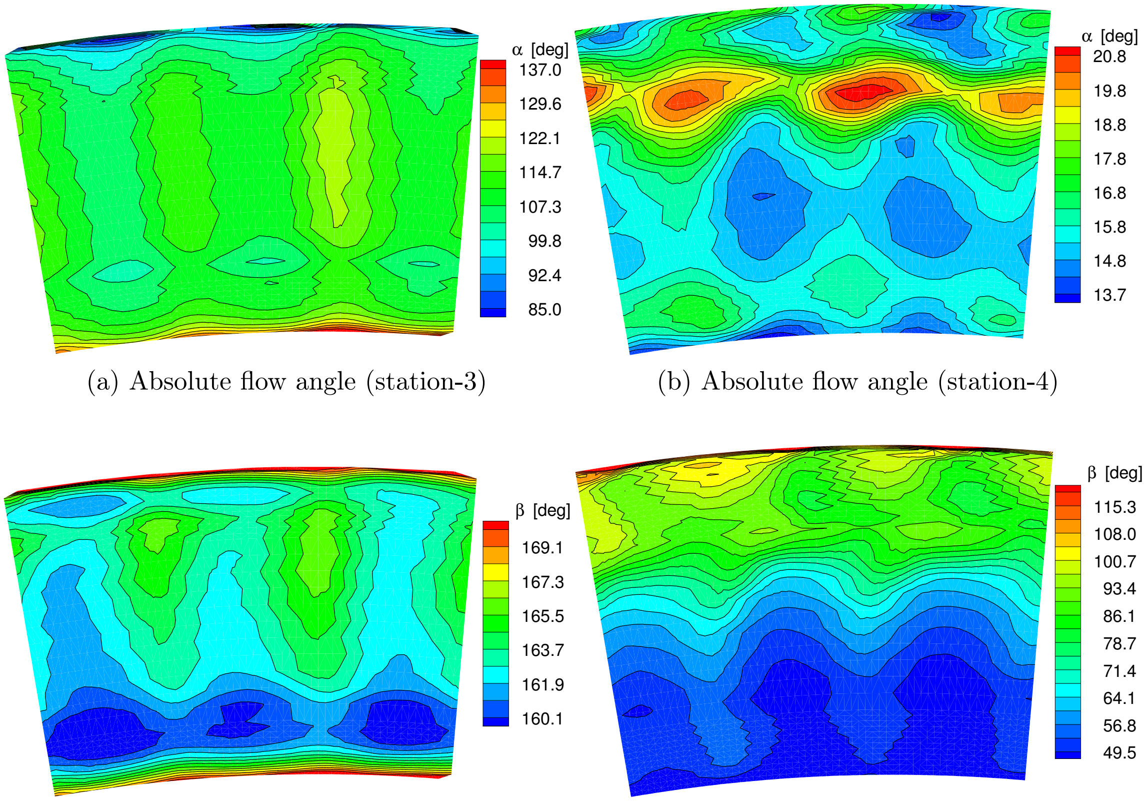
		Variable	**Error**	
Two-Stage Axial Turbine	*venturi tube*	$\dot{m}$	0.012	[kg/sec]
	torque meter	τ	0.14	[N.m]
		n	1.7	[rpm]
		P	68.9	[W]
	rakes	p	17.0	[Pa]
		P	17.0	[Pa]
		T	0.1	[°C]
	five-hole probes	p	11.7	[Pa]
		P	35.1	[Pa]
		P_r	80.9	[Pa]
		V	0.53	[m/sec]
		W	0.56	[m/sec]
		α	0.30	[deg]
		β	0.30	[deg]
		γ	0.36	[deg]
		ζ_s	0.005	
		ζ_r	0.009	
		η_{ts}	0.004	
Linear Blade Cascade I		p	1.3	[Pa]
Flow Incidence Study		P	4.2	[Pa]
		ζ_p	0.0024	
		C_p	0.0018	
Linear Blade Cascade II		p	2.5	[Pa]
Stagger Angle Study		P	7.5	[Pa]
		ζ_p	0.0045	
		C_p	0.0018	

Table A.1: Calculated and measured uncertainty values

APPENDIX B

TURBINE INTERSTAGE CONTOUR PLOTS





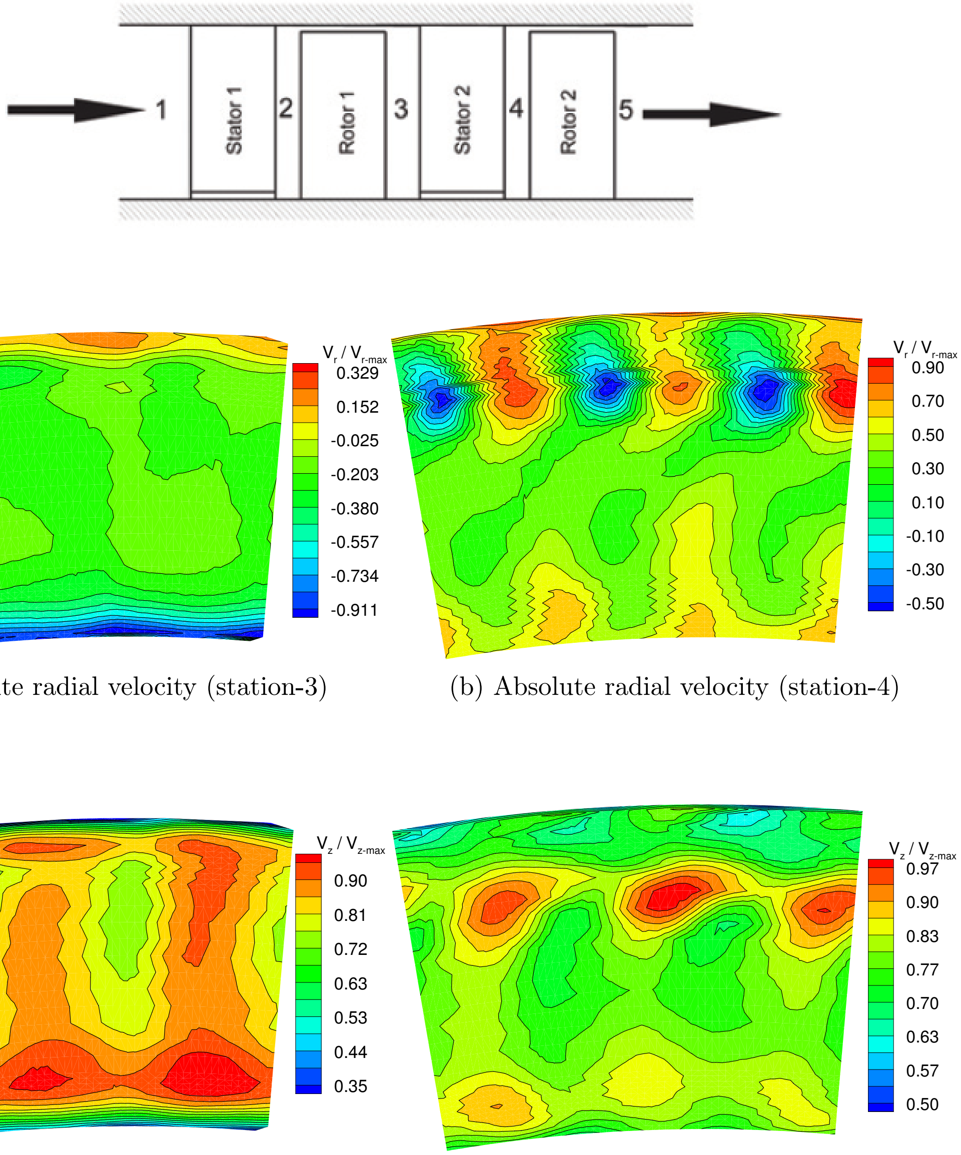
(a) Absolute flow angle (station-3)

(b) Absolute flow angle (station-4)

(c) Relative flow angle (station-3)

(d) Relative flow angle (station-4)

Figure B.1: Contour plots for interstage absolute and relative flow angles



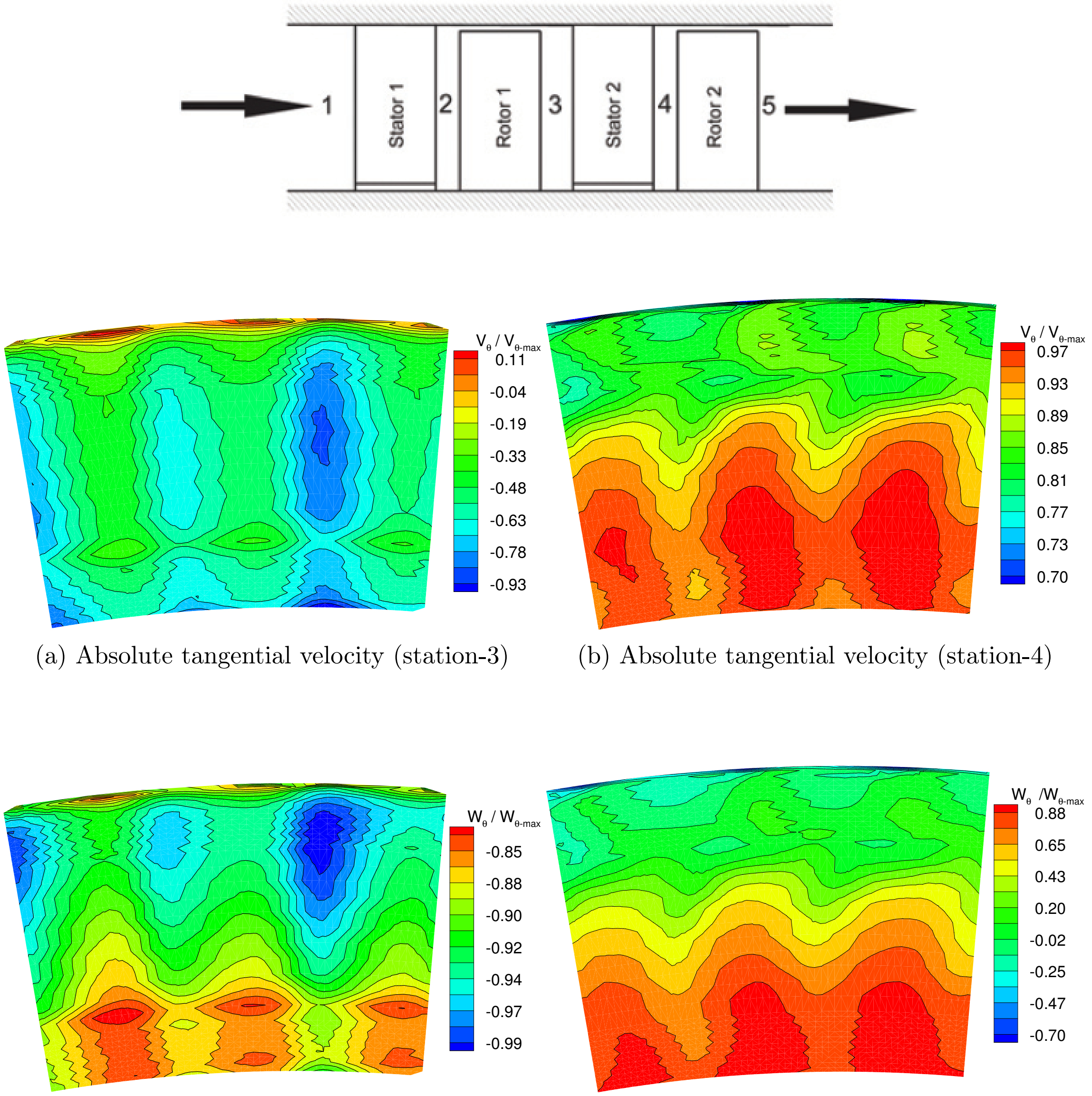
(a) Absolute radial velocity (station-3)

(b) Absolute radial velocity (station-4)

(c) Absolute axial velocity (station-3)

(d) Absolute axial velocity (station-4)

Figure B.2: Contour plots for normalized interstage absolute radial and axial velocities



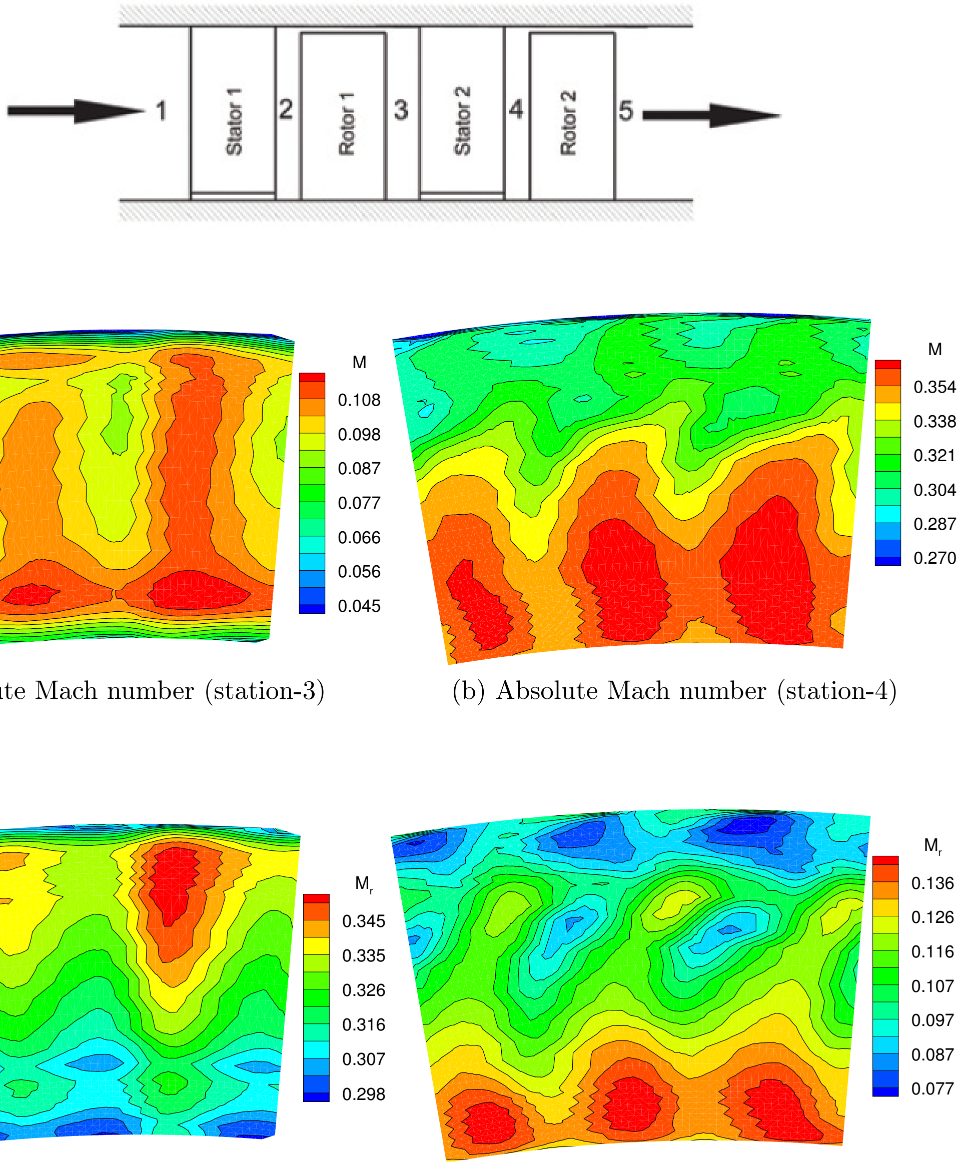
(a) Absolute tangential velocity (station-3)

(b) Absolute tangential velocity (station-4)

(c) Relative tangential velocity (station-3)

(d) Relative tangential velocity (station-4)

Figure B.3: Contour plots for normalized interstage absolute and relative tangential velocities



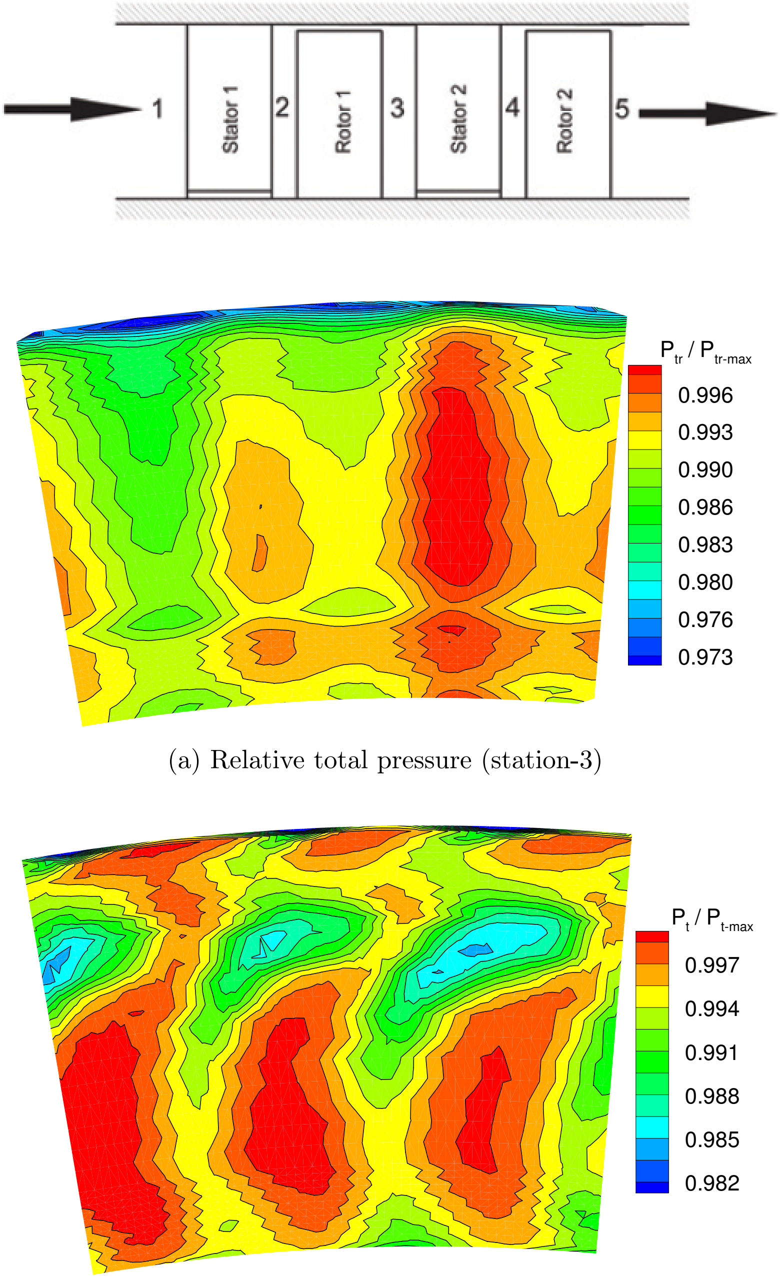
(a) Absolute Mach number (station-3)

(b) Absolute Mach number (station-4)

(c) Relative Mach number (station-3)

(d) Relative Mach number (station-4)

Figure B.4: Contour plots for interstage absolute and relative Mach numbers



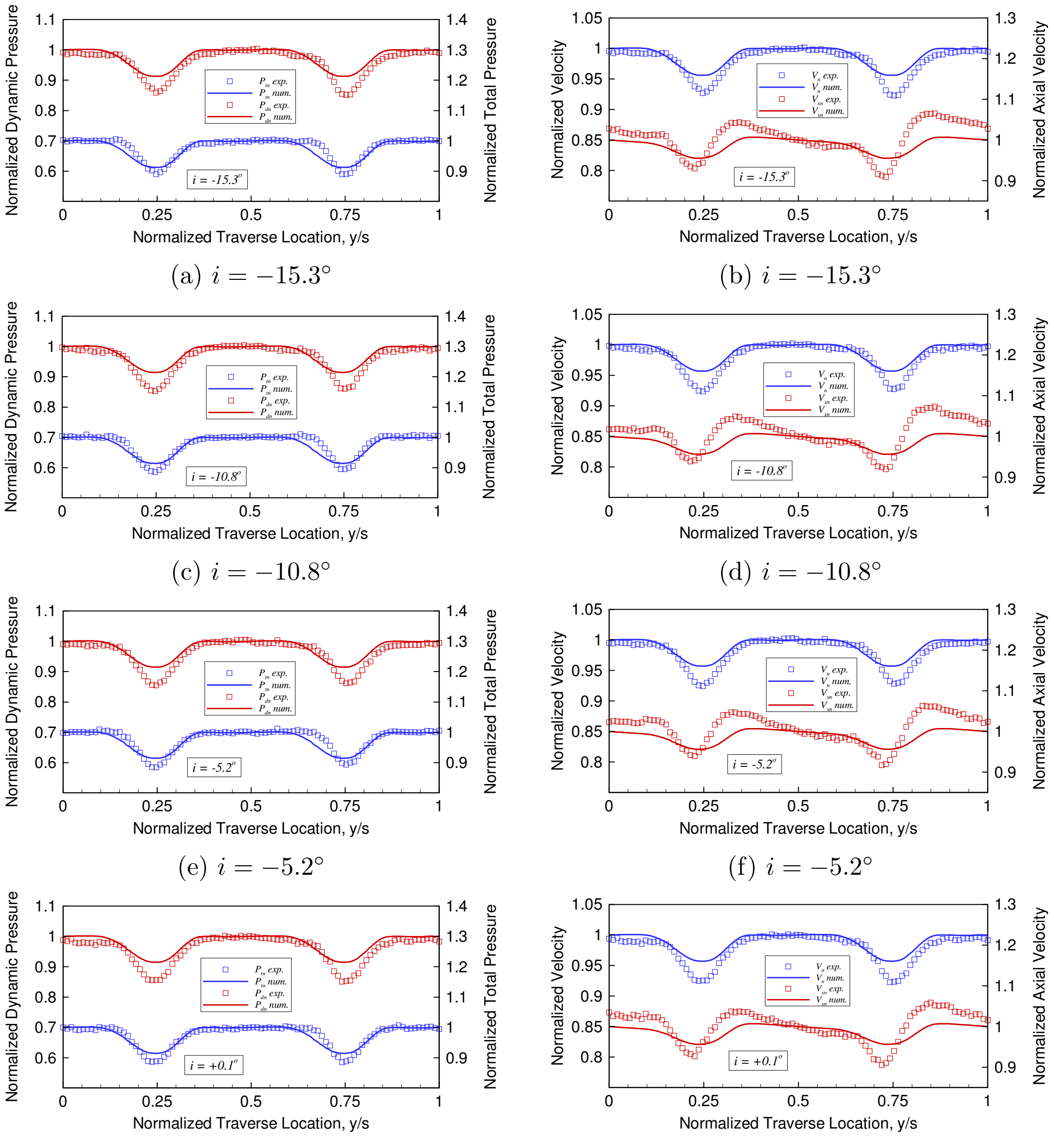
(a) Relative total pressure (station-3)

(b) Total pressure (station-4)

Figure B.5: Contour plots for normalized interstage relative and absolute total pressures

APPENDIX C

LINEAR BLADE CASCADE EXIT PRESSURE AND VELOCITY PROFILES



(a) $i = -15.3°$

(b) $i = -15.3°$

(c) $i = -10.8°$

(d) $i = -10.8°$

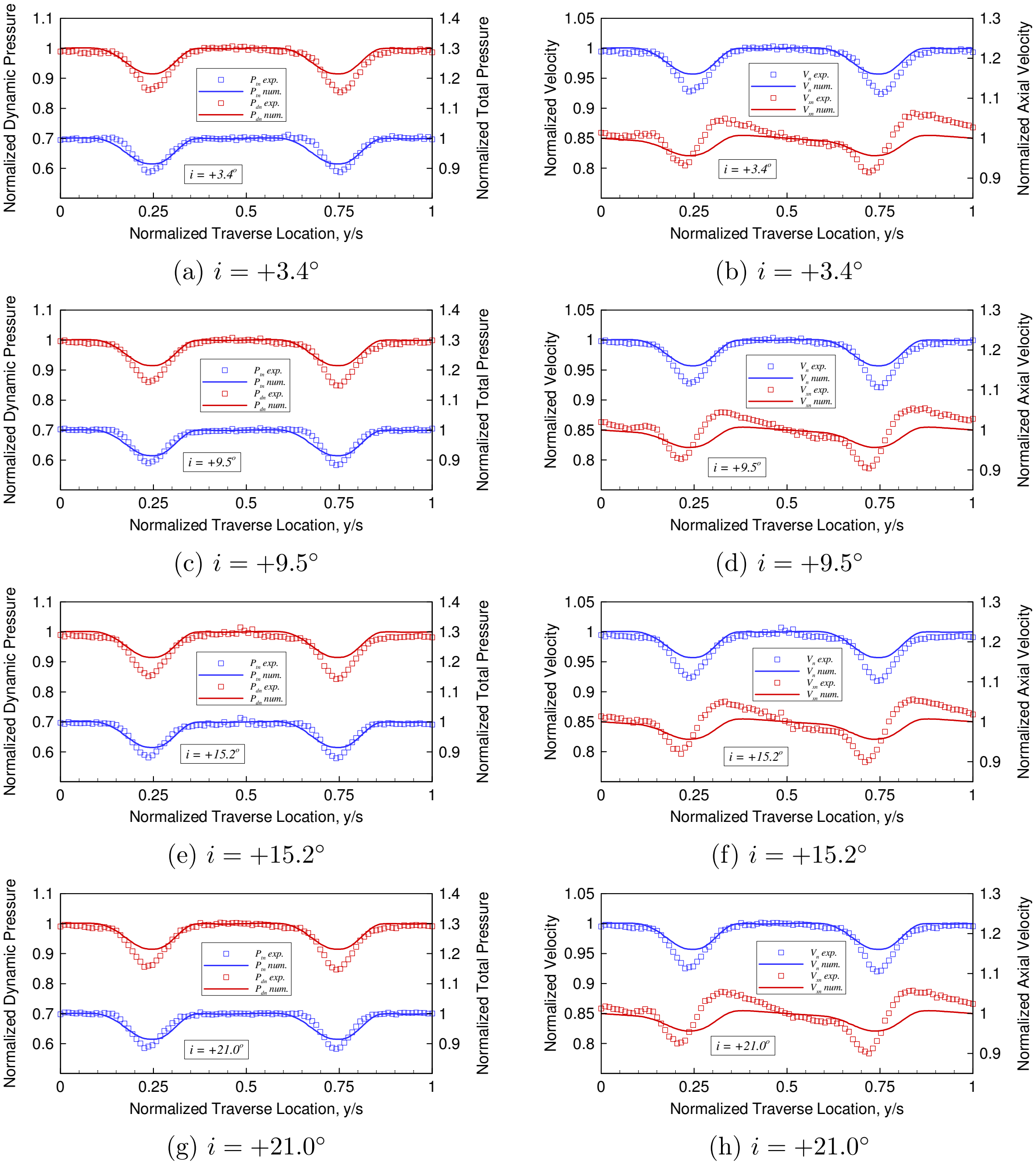
(e) $i = -5.2°$

(f) $i = -5.2°$

(g) $i = +0.1°$

(h) $i = +0.1°$

Figure C.1: Normalized experimental and numerical cascade exit pressure and velocity profiles for negative flow incidence angles



(a) $i = +3.4°$

(b) $i = +3.4°$

(c) $i = +9.5°$

(d) $i = +9.5°$

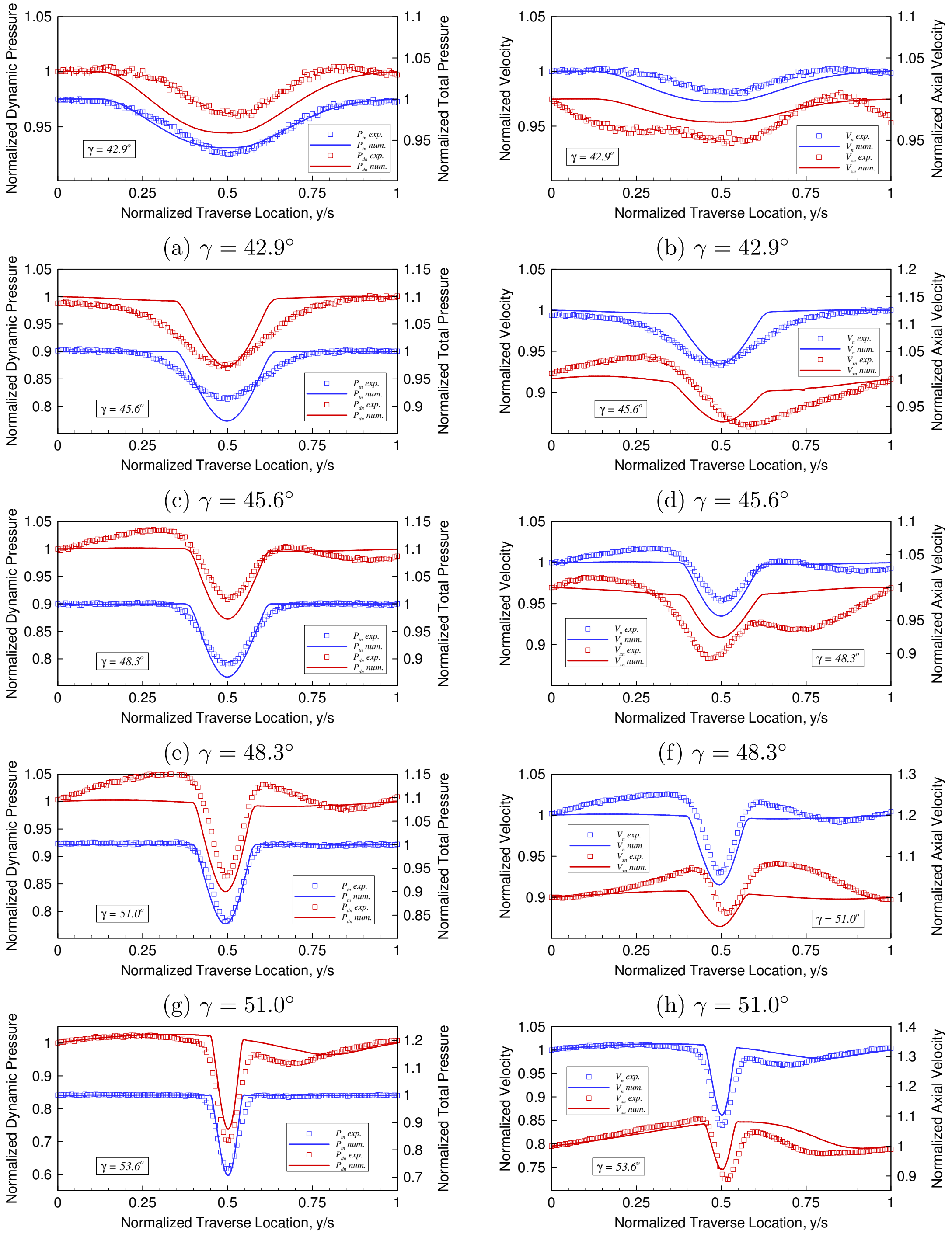
(e) $i = +15.2°$

(f) $i = +15.2°$

(g) $i = +21.0°$

(h) $i = +21.0°$

Figure C.2: Normalized experimental and numerical cascade exit pressure and velocity profiles for positive flow incidence angles



(a) $\gamma = 42.9°$

(b) $\gamma = 42.9°$

(c) $\gamma = 45.6°$

(d) $\gamma = 45.6°$

(e) $\gamma = 48.3°$

(f) $\gamma = 48.3°$

(g) $\gamma = 51.0°$

(h) $\gamma = 51.0°$

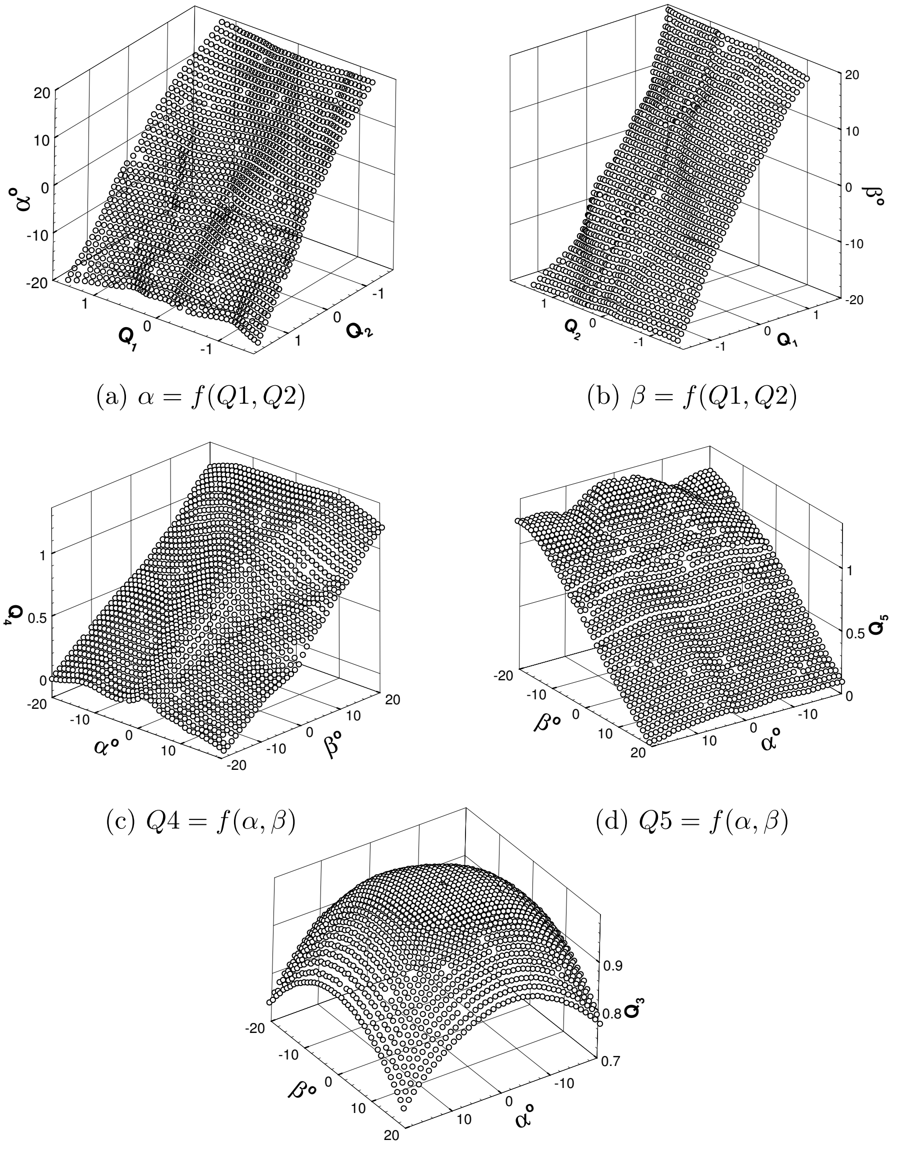
(i) $\gamma = 53.6°$

(j) $\gamma = 53.6°$

Figure C.3: Normalized experimental and numerical cascade exit pressure and velocity profiles for five cascade stagger angles $\gamma = 42.9°$, $45.6°$, $48.3°$, $51.0°$ and $53.6°$

APPENDIX D

SAMPLE FIVE-HOLE PROBE CALIBRATION CURVES



(a) $\alpha = f(Q1, Q2)$

(b) $\beta = f(Q1, Q2)$

(c) $Q4 = f(\alpha, \beta)$

(d) $Q5 = f(\alpha, \beta)$

(e) $Q3 = f(\alpha, \beta)$

Figure D.1: Five-hole probe calibration curves

APPENDIX E

MKS BARATRON PRESSURE TRANSDUCER CALIBRATION

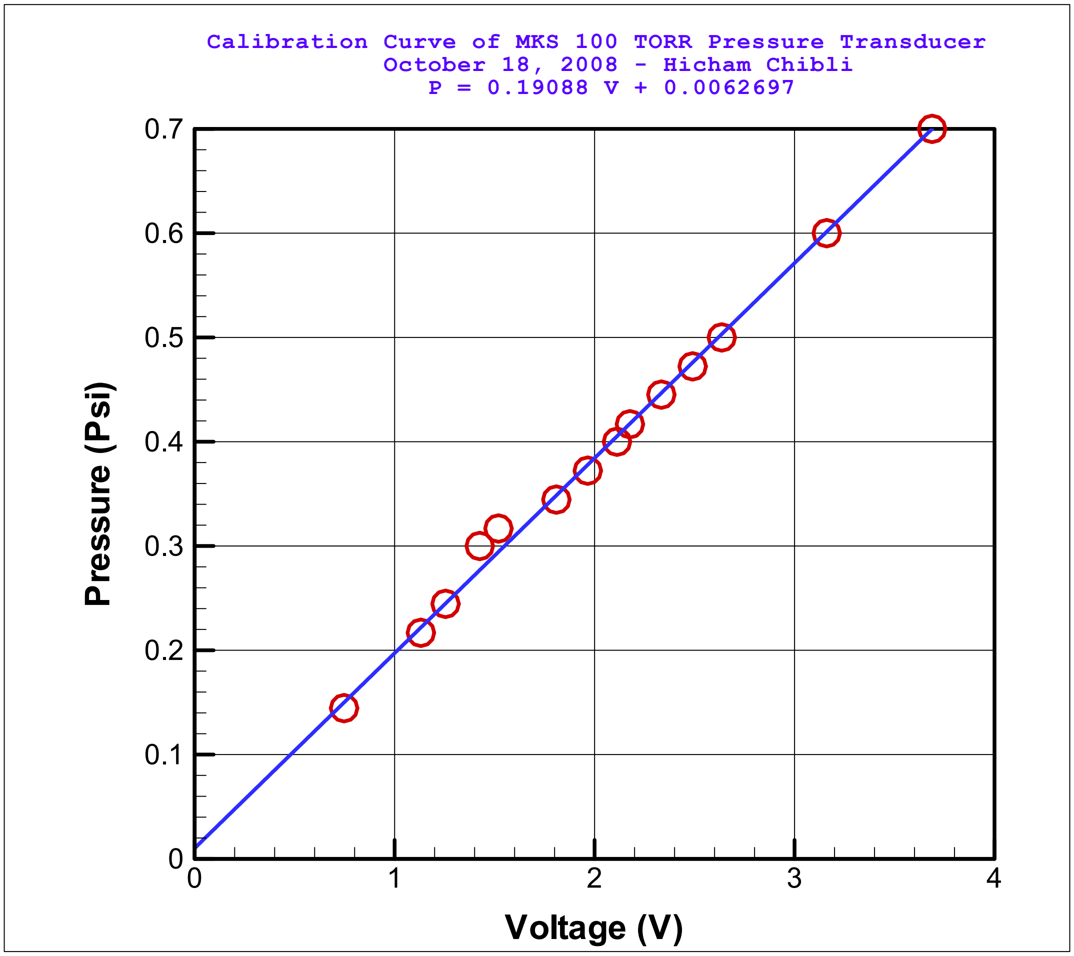


Figure E.1: MKS BARATRON pressure transducer calibration curve

APPENDIX F

BLADE ALUMINUM MOLDS

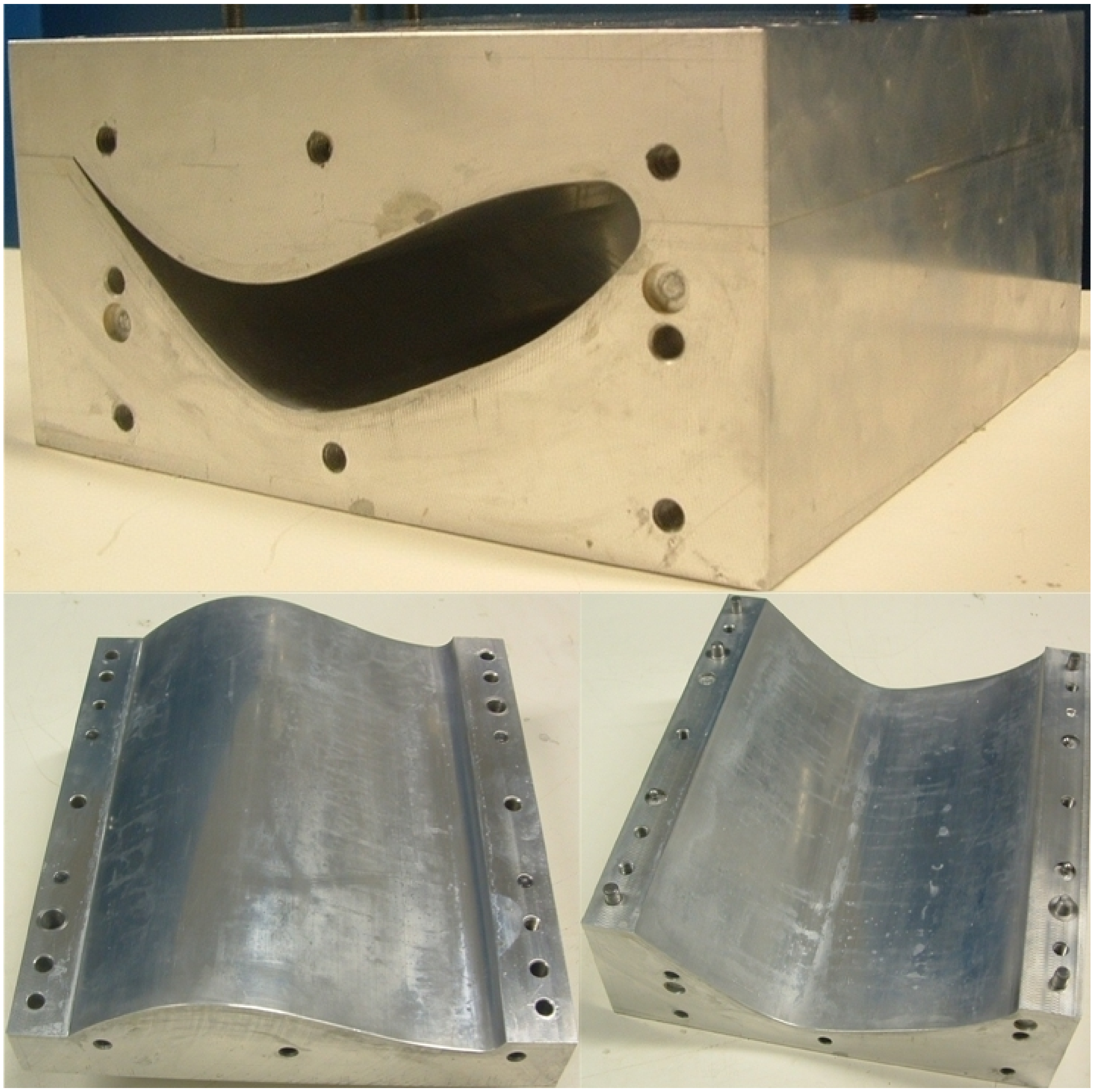


Figure F.1: Blade I aluminum mold (cascade flow incidence study)

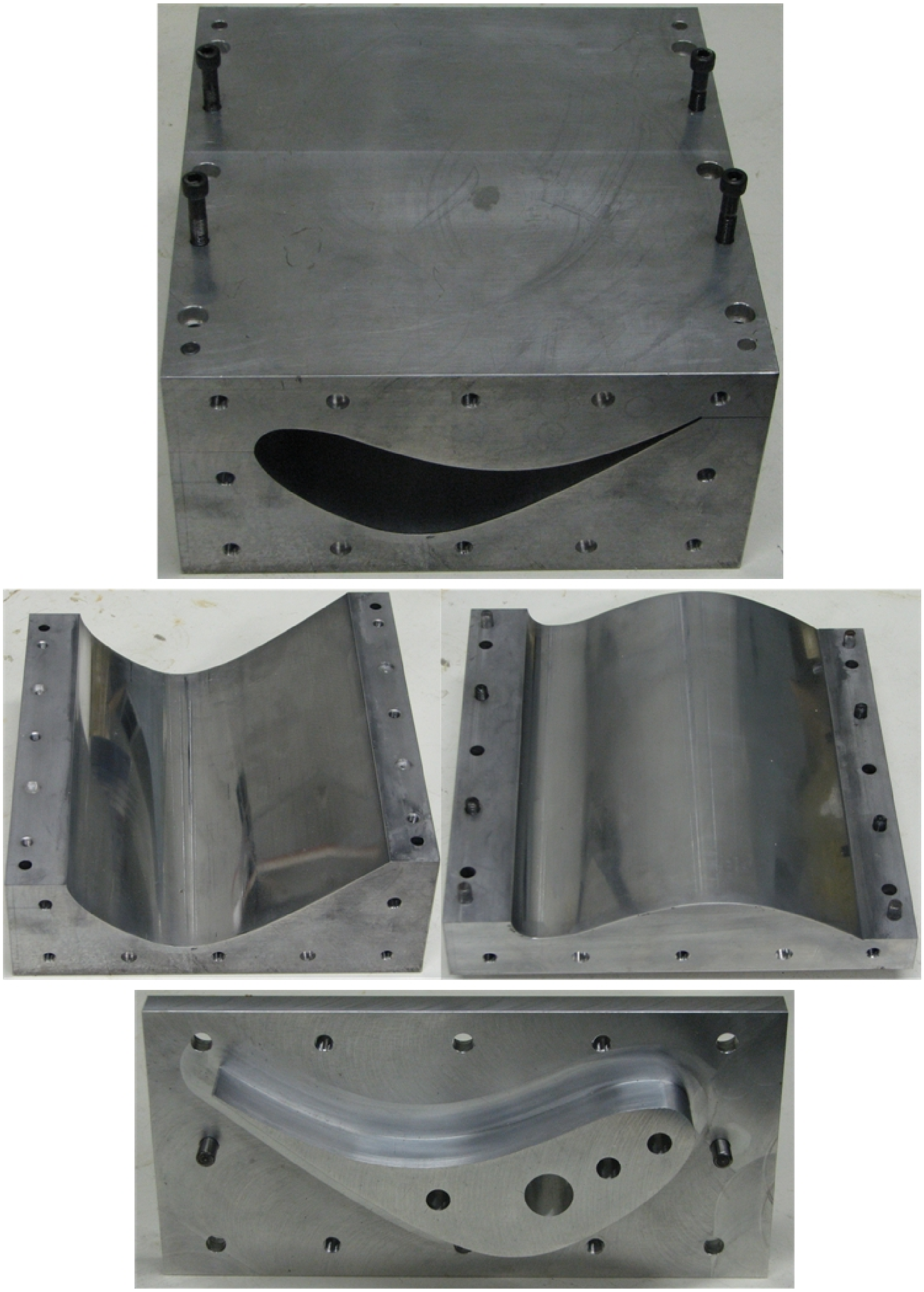
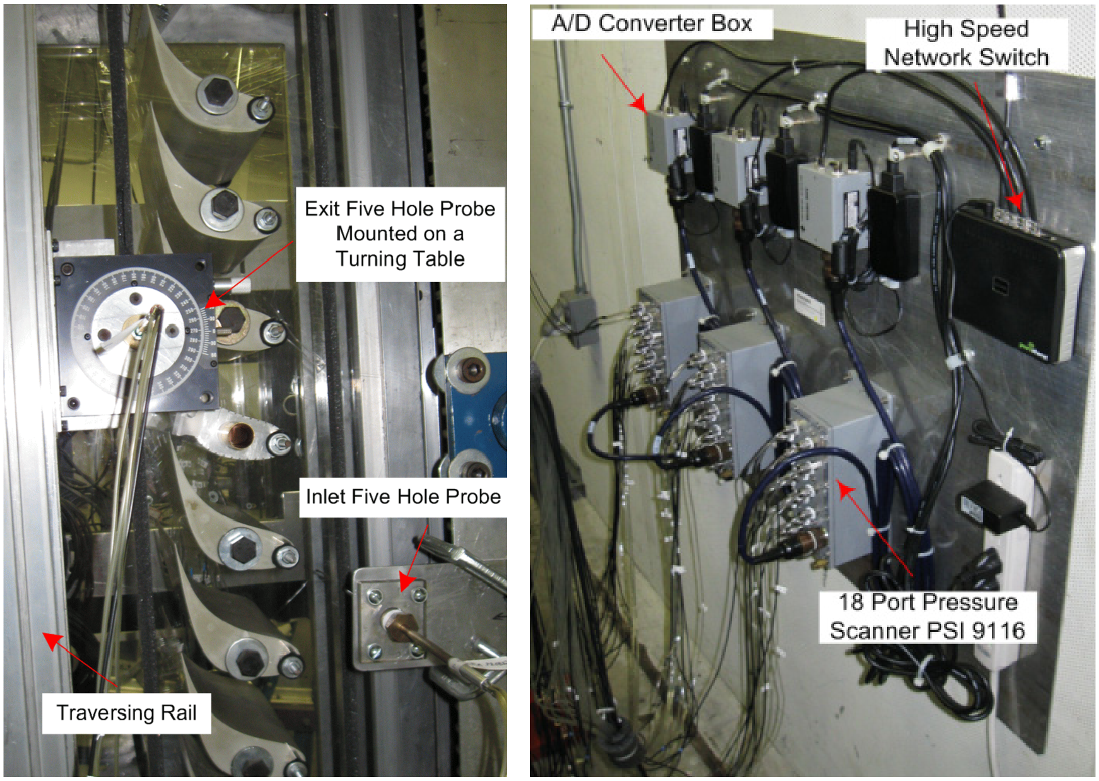


Figure F.2: Blade II aluminum mold (cascade stagger angle study)

APPENDIX G

LINEAR BLADE CASCADE TEST FACILITY


Exit Five Hole Probe Mounted on a Turning Table
Inlet Five Hole Probe
Traversing Rail
A/D Converter Box
High Speed Network Switch
18 Port Pressure Scanner PSI 9116

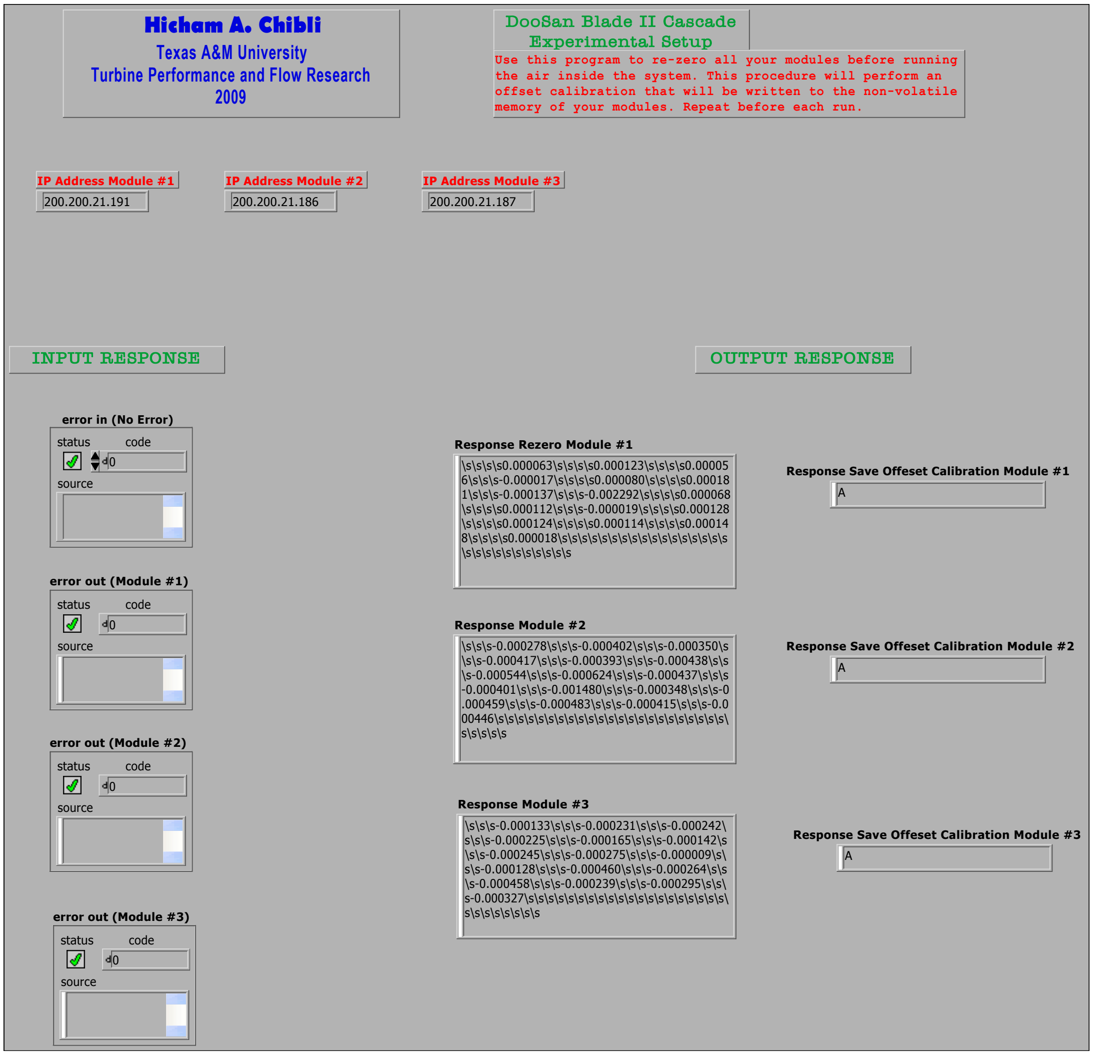
(a) Traversing system

(b) Data acquisition system

Figure G.1: Linear blade cascade test facility showing instrumentation, traversing and data acquisition systems

APPENDIX H

DATA ACQUISITION PROGRAMS



Hicham A. Chibli
Texas A&M University
Turbine Performance and Flow Research
2009

DooSan Blade II Cascade Experimental Setup

Use this program to re-zero all your modules before running the air inside the system. This procedure will perform an offset calibration that will be written to the non-volatile memory of your modules. Repeat before each run.

IP Address Module #1: 200.200.21.191

IP Address Module #2: 200.200.21.186

IP Address Module #3: 200.200.21.187

INPUT RESPONSE

error in (No Error)
status ☑ code 0
source

error out (Module #1)
status ☑ code 0
source

error out (Module #2)
status ☑ code 0
source

error out (Module #3)
status ☑ code 0
source

OUTPUT RESPONSE

Response Rezero Module #1

```
\s\s\s\s0.000063\s\s\s\s0.000123\s\s\s\s0.00005
6\s\s\s\s-0.000017\s\s\s\s0.000080\s\s\s\s0.00018
1\s\s\s\s-0.000137\s\s\s\s-0.002292\s\s\s\s0.000068
\s\s\s\s0.000112\s\s\s\s-0.000019\s\s\s\s0.000128
\s\s\s\s0.000124\s\s\s\s0.000114\s\s\s\s0.00014
8\s\s\s\s0.000018\s\s\s\s\s\s\s\s\s\s\s\s\s\s\s\s
\s\s\s\s\s\s\s\s\s\s\s
```

Response Save Offeset Calibration Module #1: A

Response Module #2

```
\s\s\s-0.000278\s\s\s-0.000402\s\s\s-0.000350\s
\s\s-0.000417\s\s\s-0.000393\s\s\s-0.000438\s\s
\s-0.000544\s\s\s-0.000624\s\s\s-0.000437\s\s\s
-0.000401\s\s\s-0.001480\s\s\s-0.000348\s\s\s-0
.000459\s\s\s-0.000483\s\s\s-0.000415\s\s\s-0.0
00446\s\s\s\s\s\s\s\s\s\s\s\s\s\s\s\s\s\s\s\s\s\s\s\
s\s\s\s\s
```

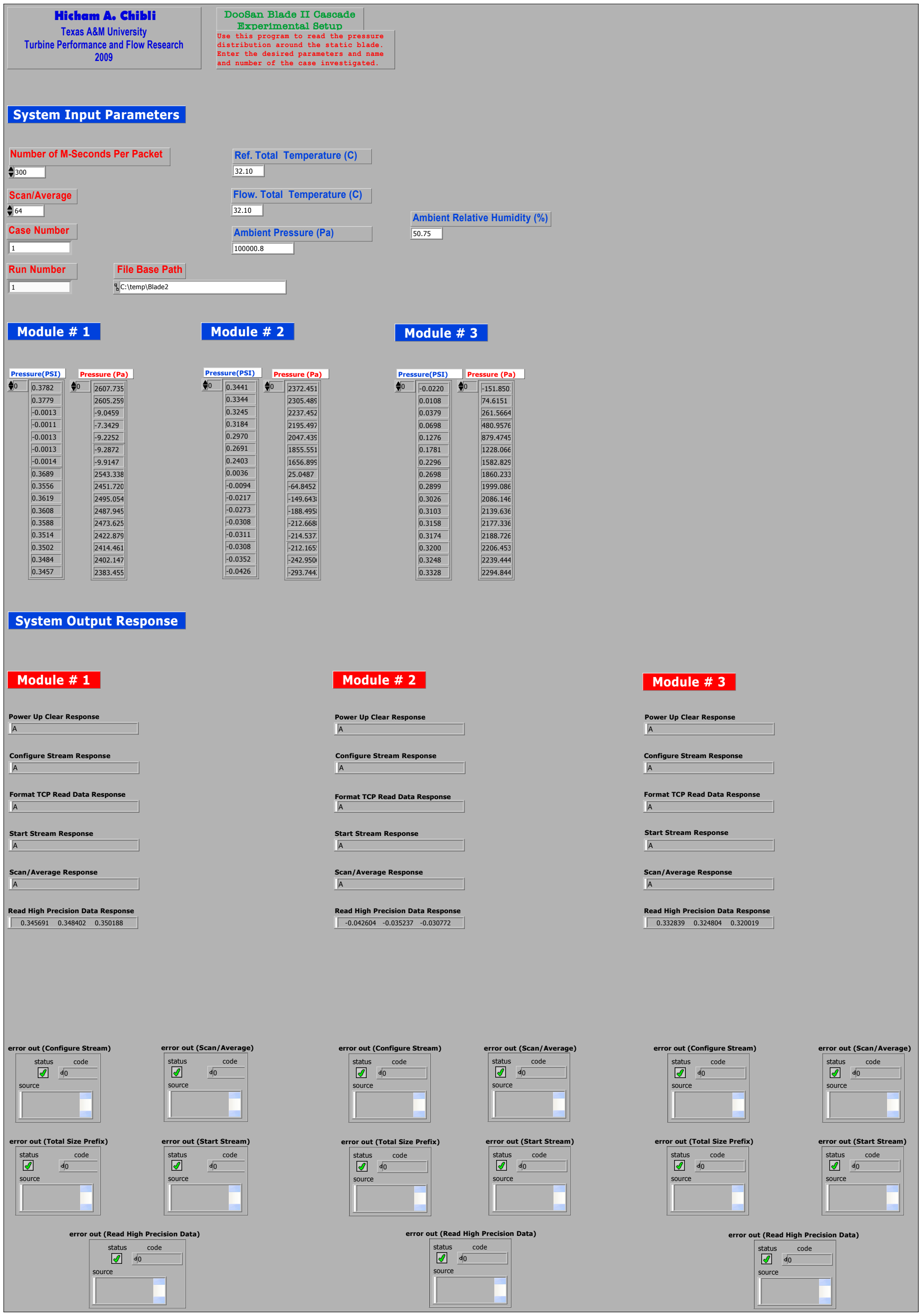
Response Save Offeset Calibration Module #2: A

Response Module #3

```
\s\s\s-0.000133\s\s\s-0.000231\s\s\s-0.000242\
s\s\s-0.000225\s\s\s-0.000165\s\s\s-0.000142\s
\s\s-0.000245\s\s\s-0.000275\s\s\s-0.000009\s\
s\s-0.000128\s\s\s-0.000460\s\s\s-0.000264\s\s
\s-0.000458\s\s\s-0.000239\s\s\s-0.000295\s\s\
s-0.000327\s\s\s\s\s\s\s\s\s\s\s\s\s\s\s\s\s\s\s\s\
s\s\s\s\s\s\s\s
```

Response Save Offeset Calibration Module #3: A

Figure H.1: LabVIEW data acquisition program for cascade stagger angle study



Hicham A. Chibli
Texas A&M University
Turbine Performance and Flow Research
2009

DooSan Blade II Cascade
Experimental Setup

```
Use this program to read the pressure
distribution around the static blade.
Enter the desired parameters and name
and number of the case investigated.
```

System Input Parameters

Number of M-Seconds Per Packet: 300

Scan/Average: 64

Case Number: 1

Run Number: 1

File Base Path: C:\temp\Blade2

Ref. Total Temperature (C): 32.10

Flow. Total Temperature (C): 32.10

Ambient Pressure (Pa): 100000.8

Ambient Relative Humidity (%): 50.75

Module # 1 Pressure(PSI)	Module # 1 Pressure (Pa)	Module # 2 Pressure(PSI)	Module # 2 Pressure (Pa)	Module # 3 Pressure(PSI)	Module # 3 Pressure (Pa)
0	0	0	0	0	0
0.3782	2607.735	0.3441	2372.451	-0.0220	-151.850
0.3779	2605.259	0.3344	2305.489	0.0108	74.6151
-0.0013	-9.0459	0.3245	2237.452	0.0379	261.5664
-0.0011	-7.3429	0.3184	2195.497	0.0698	480.9576
-0.0013	-9.2252	0.2970	2047.439	0.1276	879.4745
-0.0013	-9.2872	0.2691	1855.551	0.1781	1228.066
-0.0014	-9.9147	0.2403	1656.899	0.2296	1582.829
0.3689	2543.338	0.0036	25.0487	0.2698	1860.233
0.3556	2451.720	-0.0094	-64.8452	0.2899	1999.086
0.3619	2495.054	-0.0217	-149.643	0.3026	2086.146
0.3608	2487.945	-0.0273	-188.495	0.3103	2139.636
0.3588	2473.625	-0.0308	-212.668	0.3158	2177.336
0.3514	2422.879	-0.0311	-214.537	0.3174	2188.726
0.3502	2414.461	-0.0308	-212.165	0.3200	2206.453
0.3484	2402.147	-0.0352	-242.950	0.3248	2239.444
0.3457	2383.455	-0.0426	-293.744	0.3328	2294.844

System Output Response

Module # 1

Power Up Clear Response: A

Configure Stream Response: A

Format TCP Read Data Response: A

Start Stream Response: A

Scan/Average Response: A

Read High Precision Data Response: 0.345691 0.348402 0.350188

Module # 2

Power Up Clear Response: A

Configure Stream Response: A

Format TCP Read Data Response: A

Start Stream Response: A

Scan/Average Response: A

Read High Precision Data Response: -0.042604 -0.035237 -0.030772

Module # 3

Power Up Clear Response: A

Configure Stream Response: A

Format TCP Read Data Response: A

Start Stream Response: A

Scan/Average Response: A

Read High Precision Data Response: 0.332839 0.324804 0.320019

error out (Configure Stream): status ☑ code 0 source

error out (Scan/Average): status ☑ code 0 source

error out (Total Size Prefix): status ☑ code 0 source

error out (Start Stream): status ☑ code 0 source

error out (Read High Precision Data): status ☑ code 0 source

error out (Configure Stream): status ☑ code 0 source

error out (Scan/Average): status ☑ code 0 source

error out (Total Size Prefix): status ☑ code 0 source

error out (Start Stream): status ☑ code 0 source

error out (Read High Precision Data): status ☑ code 0 source

error out (Configure Stream): status ☑ code 0 source

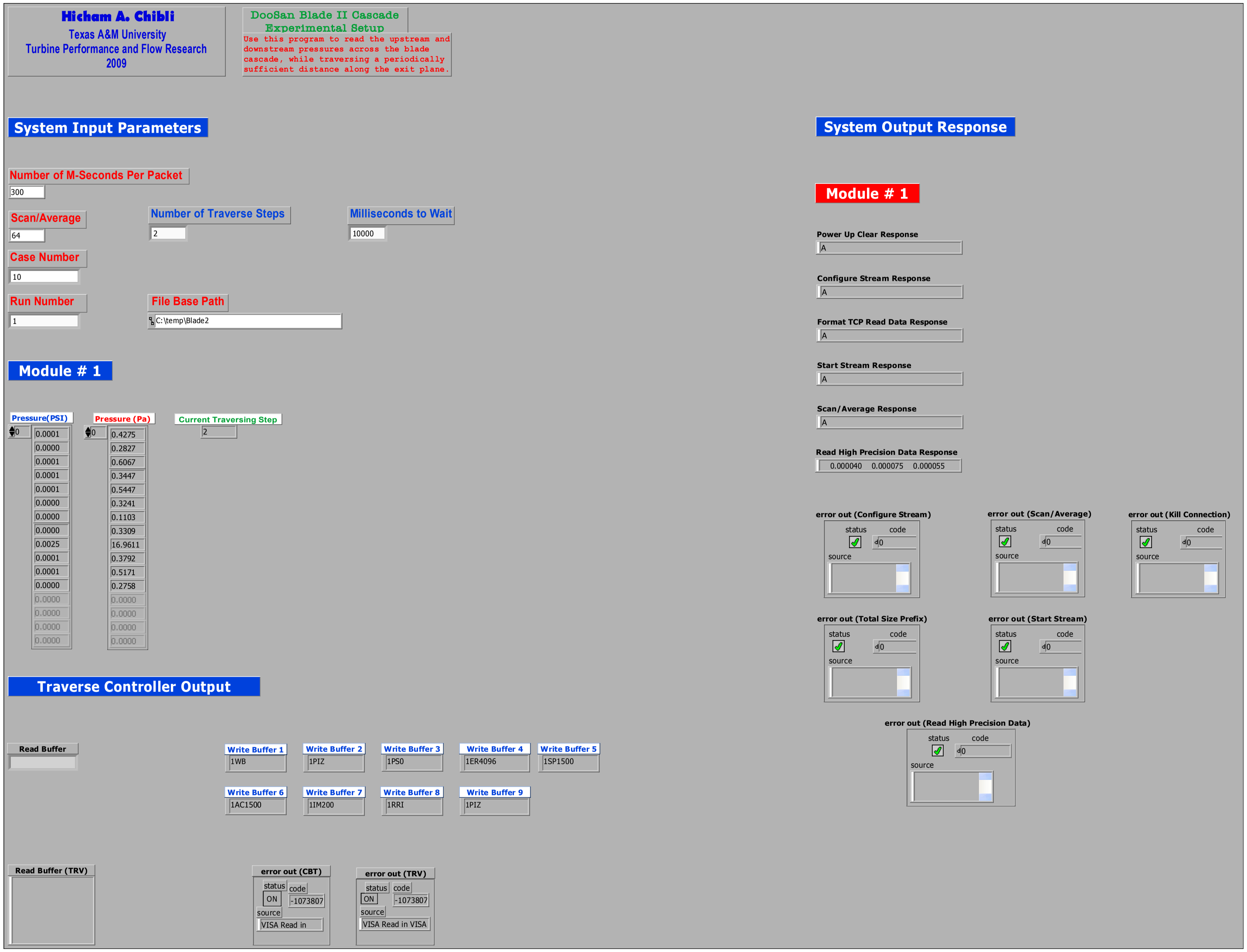
error out (Scan/Average): status ☑ code 0 source

error out (Total Size Prefix): status ☑ code 0 source

error out (Start Stream): status ☑ code 0 source

error out (Read High Precision Data): status ☑ code 0 source

Figure H.2: LabVIEW data acquisition program for cascade stagger angle study



Hicham A. Chibli
Texas A&M University
Turbine Performance and Flow Research
2009

DooSan Blade II Cascade
Experimental Setup

Use this program to read the upstream and downstream pressures across the blade cascade, while traversing a periodically sufficient distance along the exit plane.

System Input Parameters

Number of M-Seconds Per Packet: 300

Scan/Average: 64

Case Number: 10

Run Number: 1

Number of Traverse Steps: 2

Milliseconds to Wait: 10000

File Base Path: C:\temp\Blade2

Module # 1

Pressure(PSI)	Pressure (Pa)
0.0001	0.4275
0.0000	0.2827
0.0001	0.6067
0.0001	0.3447
0.0001	0.5447
0.0000	0.3241
0.0000	0.1103
0.0000	0.3309
0.0025	16.9611
0.0001	0.3792
0.0001	0.5171
0.0000	0.2758
0.0000	0.0000
0.0000	0.0000
0.0000	0.0000
0.0000	0.0000

Current Traversing Step: 2

Traverse Controller Output

Read Buffer

Write Buffer 1: 1WB
Write Buffer 2: 1PIZ
Write Buffer 3: 1PS0
Write Buffer 4: 1ER4096
Write Buffer 5: 1SP1500
Write Buffer 6: 1AC1500
Write Buffer 7: 1IM200
Write Buffer 8: 1RRI
Write Buffer 9: 1PIZ

Read Buffer (TRV)

error out (CBT)
status: ON
code: -1073807
source: VISA Read in

error out (TRV)
status: ON
code: -1073807
source: VISA Read in VISA

System Output Response

Module # 1

Power Up Clear Response: A

Configure Stream Response: A

Format TCP Read Data Response: A

Start Stream Response: A

Scan/Average Response: A

Read High Precision Data Response: 0.000040 0.000075 0.000055

error out (Configure Stream)
status: ✓ code: 0
source

error out (Scan/Average)
status: ✓ code: 0
source

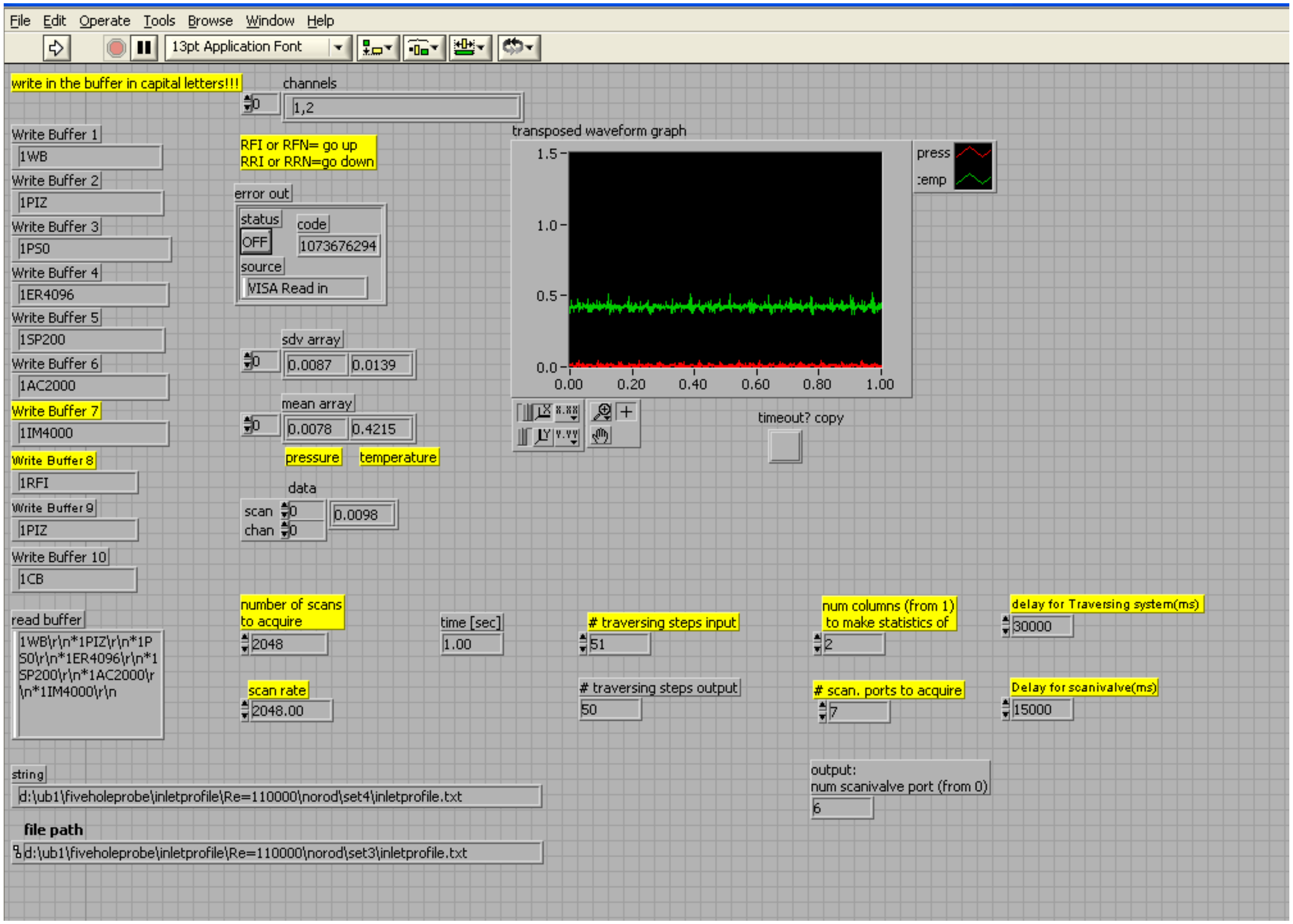
error out (Kill Connection)
status: ✓ code: 0
source

error out (Total Size Prefix)
status: ✓ code: 0
source

error out (Start Stream)
status: ✓ code: 0
source

error out (Read High Precision Data)
status: ✓ code: 0
source

Figure H.3: LabVIEW data acquisition program for cascade stagger angle study


File Edit Operate Tools Browse Window Help
13pt Application Font
transposed waveform graph
1.5
1.0
0.5
0.0
0.00
0.20
0.40
0.60
0.80
1.00
press
:emp

write in the buffer in capital letters!!!

channels: 0 | 1,2

Write Buffer 1: 1WB
Write Buffer 2: 1PIZ
Write Buffer 3: 1P50
Write Buffer 4: 1ER4096
Write Buffer 5: 1SP200
Write Buffer 6: 1AC2000
Write Buffer 7: 1IM4000
Write Buffer 8: 1RFI
Write Buffer 9: 1PIZ
Write Buffer 10: 1CB

read buffer:
1WB\r\n*1PIZ\r\n*1P
50\r\n*1ER4096\r\n*1
SP200\r\n*1AC2000\r
\n*1IM4000\r\n

RFI or RFN= go up
RRI or RRN=go down

error out
status: OFF
code: 1073676294
source: VISA Read in

sdv array: 0 | 0.0087 | 0.0139

mean array: 0 | 0.0078 | 0.4215

pressure temperature

data
scan: 0
chan: 0
0.0098

timeout? copy

number of scans to acquire: 2048

scan rate: 2048.00

time [sec]: 1.00

traversing steps input: 51

traversing steps output: 50

num columns (from 1) to make statistics of: 2

scan. ports to acquire: 7

output:
num scanivalve port (from 0): 6

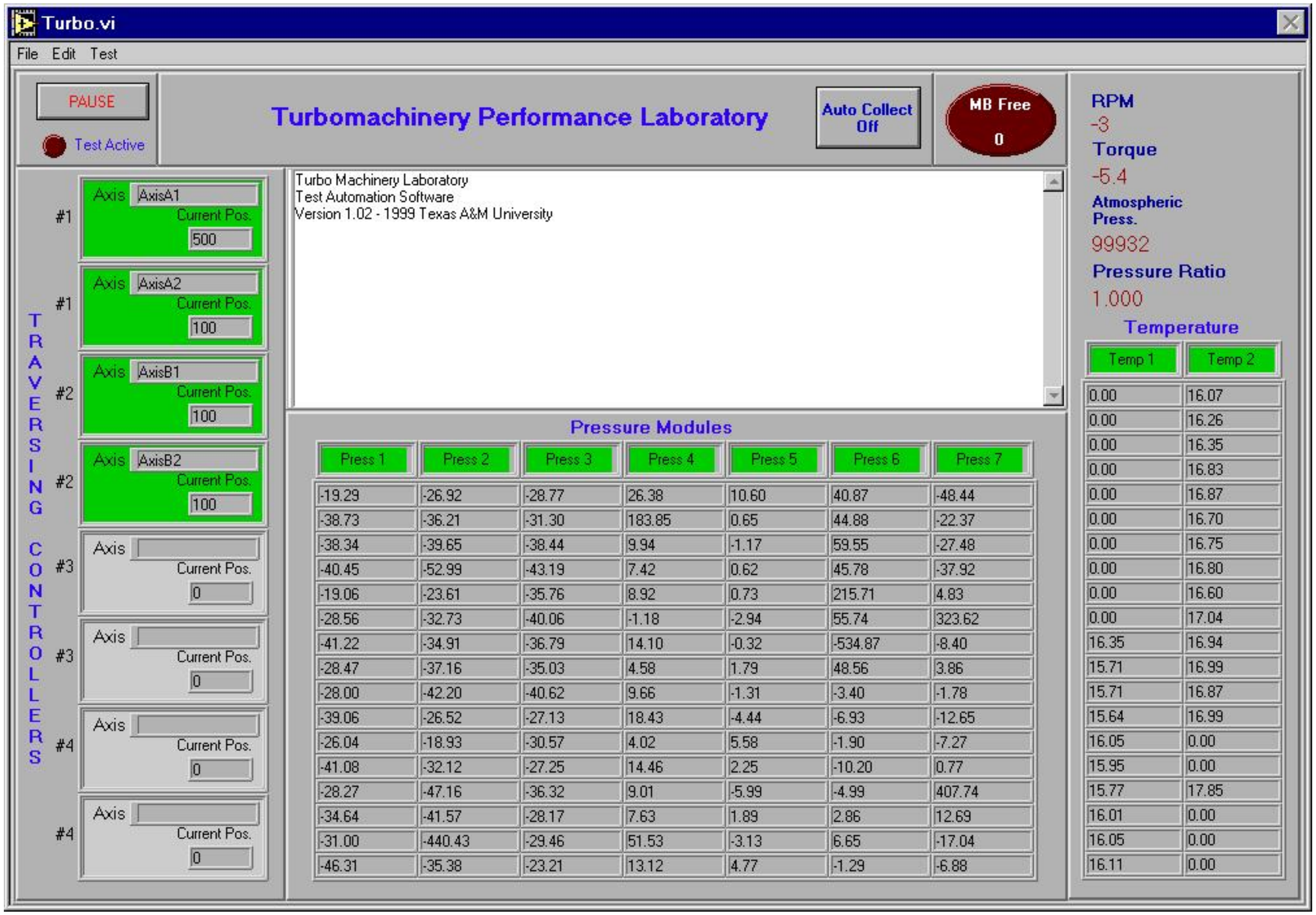
delay for Traversing system(ms): 30000

Delay for scanivalve(ms): 15000

string: d:\ub1\fiveholeprobe\inletprofile\Re=110000\norod\set4\inletprofile.txt

file path: d:\ub1\fiveholeprobe\inletprofile\Re=110000\norod\set3\inletprofile.txt

Figure H.4: LabVIEW data acquisition program for cascade flow incidence study



Turbomachinery Performance Laboratory

Turbo Machinery Laboratory
Test Automation Software
Version 1.02 - 1999 Texas A&M University

	Axis	Current Pos.
#1	AxisA1	500
#1	AxisA2	100
#2	AxisB1	100
#2	AxisB2	100
#3		0
#3		0
#4		0
#4		0

RPM
-3

Torque
-5.4

Atmospheric Press.
99932

Pressure Ratio
1.000

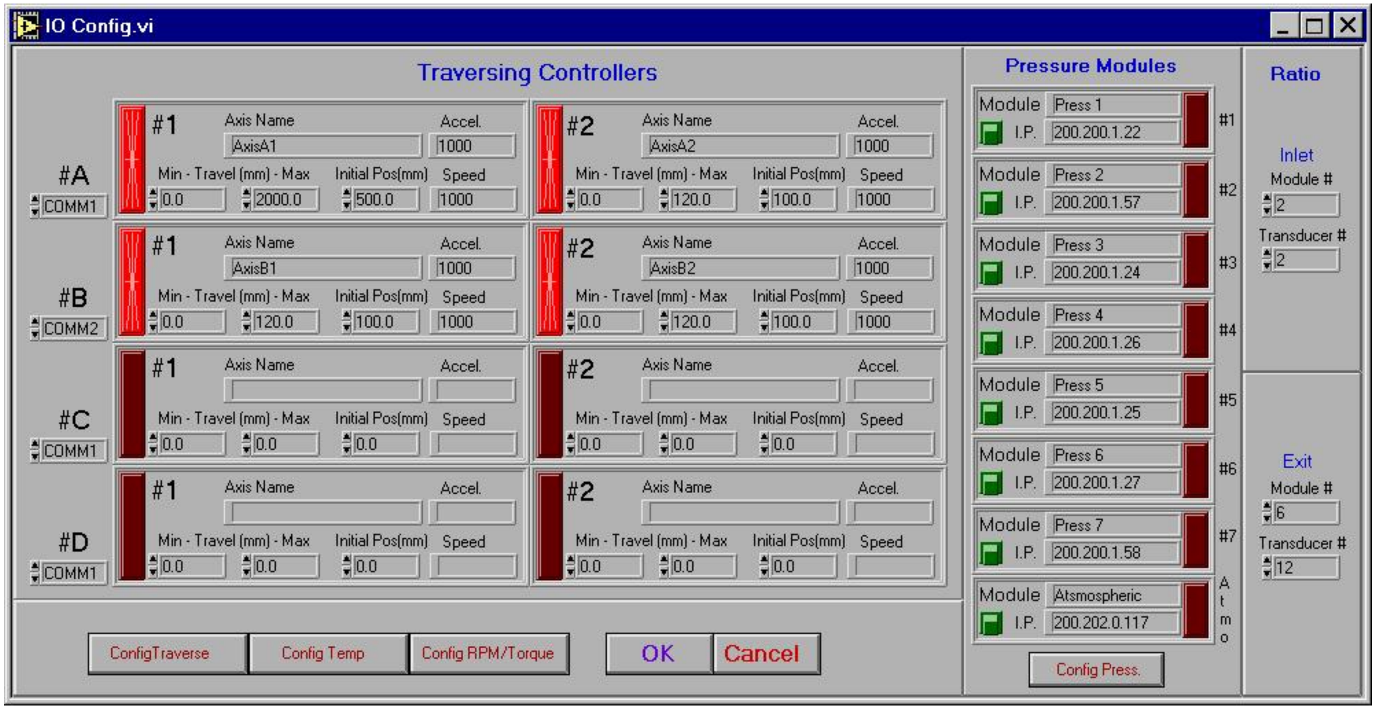
Pressure Modules

Press 1	Press 2	Press 3	Press 4	Press 5	Press 6	Press 7
-19.29	-26.92	-28.77	26.38	10.60	40.87	-48.44
-38.73	-36.21	-31.30	183.85	0.65	44.88	-22.37
-38.34	-39.65	-38.44	9.94	-1.17	59.55	-27.48
-40.45	-52.99	-43.19	7.42	0.62	45.78	-37.92
-19.06	-23.61	-35.76	8.92	0.73	215.71	4.83
-28.56	-32.73	-40.06	-1.18	-2.94	55.74	323.62
-41.22	-34.91	-36.79	14.10	-0.32	-534.87	-8.40
-28.47	-37.16	-35.03	4.58	1.79	48.56	3.86
-28.00	-42.20	-40.62	9.66	-1.31	-3.40	-1.78
-39.06	-26.52	-27.13	18.43	-4.44	-6.93	-12.65
-26.04	-18.93	-30.57	4.02	5.58	-1.90	-7.27
-41.08	-32.12	-27.25	14.46	2.25	-10.20	0.77
-28.27	-47.16	-36.32	9.01	-5.99	-4.99	407.74
-34.64	-41.57	-28.17	7.63	1.89	2.86	12.69
-31.00	-440.43	-29.46	51.53	-3.13	6.65	-17.04
-46.31	-35.38	-23.21	13.12	4.77	-1.29	-6.88

Temperature

Temp 1	Temp 2
0.00	16.07
0.00	16.26
0.00	16.35
0.00	16.83
0.00	16.87
0.00	16.70
0.00	16.75
0.00	16.80
0.00	16.60
0.00	17.04
16.35	16.94
15.71	16.99
15.71	16.87
15.64	16.99
16.05	0.00
15.95	0.00
15.77	17.85
16.01	0.00
16.05	0.00
16.11	0.00

Figure H.5: LabVIEW data acquisition program for two-stage turbine rig tests



IO Config.vi

Traversing Controllers

		Axis Name	Min - Travel (mm)	Max	Initial Pos(mm)	Accel.	Speed
#A COMM1	#1	AxisA1	0.0	2000.0	500.0	1000	1000
	#2	AxisA2	0.0	120.0	100.0	1000	1000
#B COMM2	#1	AxisB1	0.0	120.0	100.0	1000	1000
	#2	AxisB2	0.0	120.0	100.0	1000	1000
#C COMM1	#1		0.0	0.0	0.0		
	#2		0.0	0.0	0.0		
#D COMM1	#1		0.0	0.0	0.0		
	#2		0.0	0.0	0.0		

Pressure Modules

	Module	I.P.
#1	Press 1	200.200.1.22
#2	Press 2	200.200.1.57
#3	Press 3	200.200.1.24
#4	Press 4	200.200.1.26
#5	Press 5	200.200.1.25
#6	Press 6	200.200.1.27
#7	Press 7	200.200.1.58
Atmo	Atsmospheric	200.202.0.117

Ratio

Inlet

Module # 2

Transducer # 2

Exit

Module # 6

Transducer # 12

Figure H.6: LabVIEW data acquisition program for two-stage turbine rig tests

APPENDIX I

SCM SOLVER EXISTING LOSS MODEL SUMMARY

The main components of the loss model implemented in the existing SCM solver and used to numerically simulate the flow in two axial research turbines are presented in this section. This formulation is based on the classic methods of Ainley and Mathieson [8] and Baljé and Binsley [85, 86].

I.1 Profile Losses

The profile, or blade shape loss coefficient is defined as:

$$\zeta_p = \zeta_o \times f_{pc} \times f_{inc} \tag{I.1}$$

where ζ_o is the optimum profile loss coefficient due to the growth of the boundary layer on the blade pressure and suction surfaces, and is estimated as:

$$\zeta_o = 2\left(\frac{\theta}{l}\right)\left(\frac{l}{c}\right)\frac{1}{sin\alpha_2}\,\sigma_{opt} \tag{I.2}$$

where l and c are the blade camber and chord lengths, respectively. σ_{opt} is the cascade optimum solidity. The normalized boundary layer momentum thickness is empirically determined as a function of the cascade inlet and exit angles:

$$\left(\frac{\theta}{l}\right) = 0.0021\left[\frac{1-(sin\alpha_2/sin\alpha_1)^{4.5}}{1-sin\alpha_2/sin\alpha_1}\right]^{0.8} \tag{I.3}$$

The flow incidence correction factor is provided as a function of the blade stall and flow incidence angles:

$$f_{inc} = 1 + \left(\frac{i}{\alpha_{stall}}\right)^2 \tag{I.4}$$

The pitch/chord correction factor can be calculated as function of the Zweifel optimum solidity:

$$f_{pc} = \frac{2}{3}\frac{\sigma}{\sigma_{opt}} + \frac{1}{3}\left(\frac{\sigma_{opt}}{\sigma}\right)^2 \tag{I.5}$$

I.2 Secondary Losses

The secondary loss coefficient calculated using the endwall flow angles is defined as:

$$\zeta_s = 0.0668\left(\frac{c}{h}\right)\left(cot\alpha_2 - cot\alpha_1\right)^2 sin\alpha_2^3 \frac{1}{sin\alpha_\infty}\frac{1}{\sqrt{sin\alpha_1}} \times f_{inc} \tag{I.6}$$

where h is the blade height and α_∞ is the mean flow angle of the blade as defined by Zweifel [1].

APPENDIX J

FORTRAN 77 SOURCE CODE

J.1 Sample Five-Hole Probe Calibration Subroutine (Bohn's Method)

```
************************************************************************
* This subroutine will read the five probe pressures and use the       *
* calibration subroutines to calculate the total and static flow       *
* pressures as well as the flow angles. It utilizes the Bohn's method  *
* of calibration. Calibration data obtained on 11/09 for Turbine       *
* 5 hole probe 5HP#1. Calibration data 5HPT1R2.                        *
* Input Files : N/A                                                    *
* Output Files: N/A                                                    *
* Experimental Procedure: Turbine Test Facility                        *
************************************************************************
* Hicham A Chibli - Texas A&M University - 11/22/09                    *
************************************************************************
      SUBROUTINE TCAL12(P1,P2,P3,P4,P5,PS,PT,ALPHA,BETA,N)
C Define variables, parameters and constants to use
      DOUBLE PRECISION P1,P2,P3,P4,P5,PS,PT,ALPHA,BETA,PI,GUESS,ER,
     $                 LIMIT,INITIAL,CHECK,Q1,Q2,Q3,Q4,Q5,SOFT
      INTEGER N,M,KILL,RESET1,RESET2,RESET3,COUNTER,K
      PARAMETER(PI=3.141592654D0,INITIAL=0.7D0,LIMIT=0.0001D0,
     $          KILL=1000000,RESET1=1000,RESET2=10000,RESET3=100000)
      SOFT=LIMIT
      COUNTER=0
      PRINT*, '********************************************************'
      PRINT*, 'POINT NUMBER:',N
      PRINT*, '********************************************************'
    2 GUESS=INITIAL*P1
      COUNTER=COUNTER+1
      PRINT*, 'RUN COUNTER        :',COUNTER
      PRINT*, 'CONVERGENCE LIMIT  :',SOFT
      M=0
    1 Q1=(P4-P5)/(P1-GUESS)
      Q2=(P3-P2)/(P1-GUESS)
      M=M+1
C Call calibration subroutine GTAFA1 to calculate Alpha
      CALL GTAFA1(Q1,Q2,ALPHA)
C Call calibration subroutine GTBTA1 to calculate Beta
      CALL GTBTA1(Q1,Q2,BETA)
C Call calibration subroutine GTQ31 to calculate Q3
      CALL GTQ31(ALPHA,BETA,Q3)
C Call calibration subroutine GTQ41 to calculate Q4
      CALL GTQ41(ALPHA,BETA,Q4)
C Call calibration subroutine GTQ51 to calculate Q5
      CALL GTQ51(ALPHA,BETA,Q5)
      CHECK=P1-(Q3*0.5D0)*((P1-P4)/Q4+(P1-P5)/Q5)
C Kill diverging process
      IF(M.GT.KILL) THEN
        PRINT*, 'ITERATIONS EXCEEDED:',KILL,'     PROCESS TERMINATED'
        PRINT*, '-+-+-+-+-+-+-+-+-+- ERROR: BAD POINT -+-+-+-+-+-+-+',
     $          '-+-+-'
        PRINT*, '                                                   '
        PS=0.0D0
        PT=0.0D0
        ALPHA=0.0D0
        BETA=0.0D0
        RETURN
      ENDIF
C Reset convergence limits
      IF(M.GT.RESET3) THEN
        IF(COUNTER.EQ.3) THEN
          SOFT=100000.0D0*LIMIT
          K=M-1
          PRINT*, 'ITERATIONS EXCEEDED:',K,'     RESET NUMBER:    3'
          GO TO 2
        ENDIF
      ELSEIF(M.GT.RESET2) THEN
            IF(COUNTER.EQ.2) THEN
              SOFT=10000.0D0*LIMIT
              K=M-1
              PRINT*, 'ITERATIONS EXCEEDED:',K,
     $                '     RESET NUMBER:    2'
              GO TO 2
            ENDIF
      ELSEIF(M.GT.RESET1) THEN
            IF(COUNTER.EQ.1) THEN
              SOFT=1000.0D0*LIMIT
              K=M-1
              PRINT*, 'ITERATIONS EXCEEDED:',K,
     $                '     RESET NUMBER:    1'
              GO TO 2
            ENDIF
      ENDIF
C Check convergence
      ER=CHECK-GUESS
      IF(DABS(ER).LE.SOFT) THEN
        PS=GUESS
        PT=(P1-PS)/Q3+PS
C Interate
      ELSE
          GUESS=CHECK
          GO TO 1
      ENDIF
      PRINT*, '-+-+-+-+-+-+-+-+-+-+-  CONVERGENCE -+-+-+-+-+-+-+-+-+-+-'
      PRINT*, 'ITERATIONS REQUIRED:',M
      PRINT*, '                                                        '
      RETURN
      END
C Define the GTAFA1 calibration subroutine
      SUBROUTINE GTAFA1(X,Y,Z)
      DOUBLE PRECISION X,Y,Z
      Z=-5.658345428944627D0+
     $  X*(-17.55147149332653D0+X*(0.2392342891724455D0+
     $  X*(0.9788139491045860D0)))+Y*(-0.3711875303503090D0+
     $  Y*(0.3192536306929972D0+Y*(0.1108910989705018D0)))+
     $  X*Y*(0.2423053786467453D0+Y*(0.3651267457253491D0)+
     $  X*(0.1014394669577916D0))
      RETURN
      END
C Define the GTBTA1 calibration subroutine
      SUBROUTINE GTBTA1(X,Y,Z)
      DOUBLE PRECISION X,Y,Z
      DOUBLE PRECISION Z1,Z2,Z3,Z4
      Z1=0.1280270537807389D0+X*(0.4240519204189020D0+
     $   X*(-0.3048698473212721D0))
      Z2=Y*(14.02568402853777D0+
     $   Y*(0.6775750905238089D0+Y*(-0.4461086443127179D0)))
      Z3=1.000000000000000D0+X*(-0.07799013942182595D0+
     $   X*(-0.01754733439860260D0+X*(0.02300906103707801D0)))
      Z4=Y*(0.06648751119557493D0)
      Z=(Z1+Z2)/(Z3+Z4)
      RETURN
      END
C Define the GTQ31 calibration subroutine
      SUBROUTINE GTQ31(X,Y,Z)
      DOUBLE PRECISION X,Y,Z
      DOUBLE PRECISION Z1,Z2,Z3,Z4
      Z1=0.9877849174736605D0+X*(-0.009691424299726718D0+
     $   X*(-0.0001593605064950634D0+X*(1.814510800653724D-05)))
      Z2=Y*(-0.003004591361573864D0+
     $   Y*(-0.0003557096763005582D0))
      Z3=1.000000000000000D0+X*(-0.006295299947319825D0+
     $   X*(0.0002230841994909811D0+X*(2.056733069926757D-05)))
      Z4=Y*(-0.001914278687849987D0)
      Z=(Z1+Z2)/(Z3+Z4)
      RETURN
      END
C Define the GTQ41 calibration subroutine
      SUBROUTINE GTQ41(X,Y,Z)
      DOUBLE PRECISION X,Y,Z
      Z=0.9191933172260819D0+
     $  X*(0.02341418080692688D0+X*(-0.0003558007416109921D0+
     $  X*(-3.563667269356528D-06)))+Y*(0.0006611559793602247D0+
     $  Y*(-0.0003487793993196386D0+Y*(-5.533792298557268D-06)))+
     $  X*Y*(-2.396925071525935D-05+Y*(-8.081440606564024D-06)+
     $  X*(-4.101401821447164D-07))
      RETURN
      END
C Define the GTQ51 calibration subroutine
      SUBROUTINE GTQ51(X,Y,Z)
      DOUBLE PRECISION X,Y,Z
      Z=0.5927854025578972D0+
     $  X*(-0.03344241233652082D0+X*(-0.0001455751138735128D0+
     $  X*(8.751504801181858D-06)))+Y*(-0.0005824896145476819D0+
     $  Y*(-0.0001490433276596423D0+Y*(-2.583702185636681D-06)))+
     $  X*Y*(2.383412616836691D-06+Y*(6.688203796008452D-06)+
     $  X*(4.390406938297336D-07))
      RETURN
      END
```

J.2 Two-Stage Turbine Data Analysis Program

```
************************************************************************
* This program is the main component that calls upon all the utility   *
* subroutines to read and analyze the turbine experimental data.       *
* Input Files : N/A                                                    *
* Output Files: N/A                                                    *
* Experimental Procedure: Research Turbine Facility - TPFL             *
************************************************************************
* Hicham A Chibli - Texas A&M University - 10/31/09                    *
************************************************************************
      PROGRAM TURBO
C Define constants and parameters
      INTEGER M,N
C Integer to check for input error
      M=0
C Prompt user for input
      PRINT*, 'ENTER YOUR CHOICE BELOW, THEN HIT RETURN:'
      PRINT*, 'TO READ RAW DATA PRESS ------------------------------- 1'
      PRINT*, 'TO READ HUMIDITY RAW DATA PRESS ---------------------- 2'
      PRINT*, 'TO ANALYZE INTERSTAGE DATA PRESS --------------------- 3'
      PRINT*, 'TO PERFORM CIRCUMFERENTIAL AVERAGING PRESS ----------- 4'
      PRINT*, 'TO GENERATE LINE AND CONTOUR PLOTS PRESS ------------- 5'
      PRINT*, 'TO CONDUCT ROW PERFORMANCE ANALYSIS PRESS ------------ 6'
      PRINT*, 'TO EXIT THIS PROGRAM PRESS --------------------------- 9'
      READ*, N
C Call EXDATA subroutine to read and allocate raw data from turbine
      IF(N.EQ.1) THEN
        CALL EXDATA
        M=1
      ENDIF
C Call HUMID subroutine to read and analyze humidity data
      IF(N.EQ.2) THEN
        CALL HUMID
        M=1
      ENDIF
C Call FHP subroutine to analyze five hole probe data
      IF(N.EQ.3) THEN
        CALL FHP
        M=1
      ENDIF
C Call AVER subroutine to perform circum. averaging
      IF(N.EQ.4) THEN
        CALL AVER
        M=1
      ENDIF
C Call PLOT subroutine to plot data
      IF(N.EQ.5) THEN
        CALL PLOT
        M=1
      ENDIF
C Call LOSS subroutine to study row performance
      IF(N.EQ.6) THEN
        CALL PRFM
        M=1
      ENDIF
      IF(N.EQ.9) THEN
        PRINT*, 'NO ANALYSIS PERFORMED.'
        GO TO 1
      ENDIF
      IF(M.EQ.0) THEN
        PRINT*, 'INVALID CHOICE. PROGRAM WILL CLOSE.'
        GO TO 1
      ENDIF
    1 END
```

```
************************************************************************
* This subroutine reads and extracts the average pressures for the     *
* five hole probes on the three stations as well as pressure,          *
* temperature and rpm reference readings for correcting the flow.      *
* Input Files : parameter.txt, sample.txt, number.txt, rxx.txt         *
* Output Files: raws3.int, raws4.int, raws5.int,                       *
*               tring.int, htotal.int                                  *
* Experimental Procedure: Research Turbine Facility - TPFL             *
************************************************************************
* Hicham A Chibli - Texas A&M University - 12/02/09                    *
************************************************************************
      SUBROUTINE EXDATA
C Define variables and constants
      INTEGER M,N,K,J,I,L,Q,LUNIT,IMAX,SAMPLE,COLUMN,SLAYER
C Maximum number of samples allowed, data sublayes and array width
      PARAMETER(IMAX=10000,SLAYER=5,COLUMN=160)
C Define variables and constants
      DOUBLE PRECISION DATUM(COLUMN),MEAN(COLUMN),WINDOW(IMAX,COLUMN)
      DOUBLE PRECISION CIRCUM,RAD3,RAD4,RAD5,TT1,TT7,PT1,PS1
      CHARACTER TITLE*12,SDATE*8,SNUM*1
C Open files for input
      OPEN(UNIT=60,FILE='./inp/number.txt',STATUS='OLD')
      OPEN(UNIT=70,FILE='./inp/sample.txt',STATUS='OLD')
      OPEN(UNIT=80,FILE='./inp/parameter.txt',STATUS='OLD')
C Open files for output
      OPEN(UNIT=90,FILE='./tmp/raws3.int',STATUS='UNKNOWN')
      OPEN(UNIT=100,FILE='./tmp/raws4.int',STATUS='UNKNOWN')
      OPEN(UNIT=110,FILE='./tmp/raws5.int',STATUS='UNKNOWN')
      OPEN(UNIT=120,FILE='./tmp/tring.int',STATUS='UNKNOWN')
      OPEN(UNIT=130,FILE='./tmp/htotal.int',STATUS='UNKNOWN')
C Read number of data sublayers to analyze
      READ(80,*)
      READ(80,*) N
C Check if maximum sublayer number is bounded by (5)
      IF(N.GT.SLAYER) THEN
        PRINT*, 'MAXIMUM NUMBER OF SUBLAYERS ALLOWED EXCEEDED.'
        STOP
      ENDIF
      DO 8 L=1,N
C Read number of samples
         READ(70,*) SAMPLE
C Read input file number
         READ(60,*) SNUM
C Call NME subroutine to generate name string
         CALL NME('r',SNUM,TITLE,1)
C Open file for input
         LUNIT=10*L
         OPEN(UNIT=LUNIT,FILE=TITLE,STATUS='OLD')
C Reset counter
         K=0
         I=0
         M=0
C Skip headers
         READ(LUNIT,*)
         READ(LUNIT,*)
         READ(LUNIT,*)
C Read data
    1    READ(LUNIT,*,END=7) SDATE,DATUM
         K=K+1
C Populate temporary arrays
         DO 2 J=1,COLUMN
            WINDOW(K,J)=DATUM(J)
    2    CONTINUE
C Initialize all values to zero
         IF(K.EQ.SAMPLE) THEN
           CIRCUM=DATUM(1)
           RAD3=DATUM(4)
           RAD4=DATUM(3)
           RAD5=DATUM(2)
C Average data points over number of samples
           DO 3 J=1,COLUMN
              MEAN(J)=0.0D0
    3      CONTINUE
           DO 5 Q=1,COLUMN
              DO 4 J=1,SAMPLE
                 MEAN(Q)=MEAN(Q)+WINDOW(J,Q)
    4         CONTINUE
    5      CONTINUE
           DO 6 J=1,COLUMN
              MEAN(J)=MEAN(J)/DBLE(SAMPLE)
    6      CONTINUE
           TT1=(MEAN(148)+MEAN(149)+MEAN(150)+MEAN(151)+MEAN(152))/5.0D0
           PT1=(MEAN(6)+MEAN(7)+MEAN(8)+MEAN(9)+MEAN(10)+MEAN(11)+
     $          MEAN(12)+MEAN(13))/8.0D0
           PS1=(MEAN(22)+MEAN(23)+MEAN(25))/3.0D0
           TT7=(MEAN(136)+MEAN(138)+MEAN(139)+MEAN(140)+MEAN(141))/5.0D0
           K=0
           I=I+1
C Write data to output files
           WRITE(90,*) CIRCUM,RAD3,MEAN(38),MEAN(39),MEAN(40),MEAN(41),
     $                 MEAN(42),PT1,PS1,MEAN(118),TT1,TT7,MEAN(159)
           WRITE(100,*) CIRCUM,RAD4,MEAN(54),MEAN(55),MEAN(56),MEAN(57),
     $                  MEAN(58),PT1,PS1,MEAN(118),TT1,TT7,MEAN(159)
           WRITE(110,*) CIRCUM,RAD5,MEAN(59),MEAN(60),MEAN(62),MEAN(63),
     $                  MEAN(64),PT1,PS1,MEAN(118),TT1,TT7,MEAN(159)
         ENDIF
         M=M+1
         GO TO 1
    7    PRINT*, 'DATA READING FOR FILE NUMBER ',SNUM,' COMPLETE.'
         WRITE(130,*) I
    8 CONTINUE
      WRITE(120,*) MEAN(50),MEAN(51),MEAN(98)
      RETURN
      END
************************************************************************
* This subroutine reads the five hole probe data and makes calls to    *
* respective calibration subroutines to extract the basic flow         *
* parameters. Then it performs time correction based on                *
* Chibli's approach and transforms the flow angles to machine          *
* coordinate system together with the flow velocity components and     *
* the resulting thermal properties.                                    *
* Input Files : parameter.txt, humidity.int, tring.int, stemp.int,     *
*               layers3.txt, layers4.txt, layers5.txt,                 *
*               raws3.int, raws4.int, raws5.int                        *
* Output Files: fulls3.int, fulls4.int, fulls5.int,                    *
*               hubs3.int, hubs4.int, hubs5.int,                       *
*               basics3.int, basics4.int, basics5.int,                 *
*               xys3.int, xys4.int, xys5.int,                          *
*               xyst3.int, xyst4.int, xyst5.int,                       *
*               pitchs3.int, pitchs4.int, pitchs5.int,                 *
*               yaws3.int, yaws4.int, yaws5.int,                       *
*               totals3.int, totals4.int, totals5.int,                 *
*               chord.int                                              *
* Experimental Procedure: Research Turbine Facility - TPFL             *
************************************************************************
* Hicham A Chibli - Texas A&M University - 12/02/09                    *
************************************************************************
      SUBROUTINE STN(L,SNUM)
      DOUBLE PRECISION P1,P2,P3,P4,P5,PT,PS,V,RHO,MACH,MACHR,TT,TSCR,
     $                 ATT,ATS,APS,APT,ENS,ENT,ENTR,S,
     $                 PSCR,PTCR,RHOCR,MU,
     $                 VCR,VZCR,VRCR,VCCR,VMCR,WCR,WZCR,WRCR,
     $                 WCCR,UCR,
     $                 ATSCR,APSCR,APTCR,APRCR,
     $                 TT1,TT7,PT1,PS1,RHO1,
     $                 ATT1,ATS1,APS1,APT1,
     $                 STT1,SPT1,SPS1,SRHO1,
     $                 ASPT1,ASPS1,ASTT1,ASTS1,
     $                 ALPHA,BETA,APSI,APHI,RPHI,GMA,ANGLE(5,2),
     $                 CIRCUM,RADIAL,THETA,RADIUS,Y,X,
     $                 PATM,RPM,RHT,SRPM,SPATM,SRHT,
     $                 GAMMA,R,CP,KELVIN,PI,ZERO,AMAX,BMAX,DEGREE,
     $                 BH,HR,CHORD3,CHORD4,CHORD5,AXIAL3,AXIAL4,AXIAL5,
     $                 DUMMY,TIPRAD,CHECK,A1,A2,G1,G2,G3,G4,
     $                 TEMP,LIMIT,TERR,STEMP3,STEMP4,STEMP5,
     $                 APTH,APSH,APTHCR,APSHCR,APRHCR,MUH,ATTH,ATSH,
     $                 MACHRH,MACHH,RHOH,RHOHCR,UH,UHCR,VH,VHCR,
     $                 VCHCR,VRHCR,VZHCR,VMHCR,WHCR,WCHCR,WRHCR,WZHCR,
     $                 PST,APST,APTTCR,APSTCR,APRTCR,ATSHCR,TSHCR,
     $                 ATTT,ATST,ATSTCR,TSTCR,MUT,RHOT,RHOTCR,
     $                 MACHT,MACHRT,VT,VTCR,VCTCR,VRTCR,VZTCR,VMTCR,
     $                 WTCR,WCTCR,WRTCR,WZTCR,UTCR,
     $                 APSIT,APHIT,RPHIT,APSIH,APHIH,RPHIH,
     $                 CPH,GAMMAH,RH,SH,ENSH,ENTH,ENTHR,
     $                 CPT,GAMMAT,RT,ST,ENST,ENTT,ENTTR
      INTEGER N,I,K,J,L,Q,R3,S4,R5
      LOGICAL VALID,TEST
      CHARACTER SNUM*1,TITLE*100
C Convergence parameters
      PARAMETER(LIMIT=0.001D0,AMAX=25.0D0,BMAX=35.0D0)
C Thermo and general parameters
      PARAMETER(KELVIN=273.15D0,ZERO=0.0D0)
C General parameters
      PARAMETER(PI=3.141592654D0,DEGREE=180.0D0)
C Stage identifiers
      PARAMETER(R3=1,S4=2,R5=3)
C Call subroutine NME to generate input file name
      CALL NME('raws',SNUM,TITLE,3)
C Call subroutine CLEAN to clean file name
      CALL CLEAN(TITLE,Q)
C Open file for input
      OPEN(UNIT=10,FILE=TITLE(1:Q),STATUS='OLD')
C Call subroutine NME to generate input file name
      CALL NME('layers',SNUM,TITLE,1)
C Call subroutine CLEAN to clean file name
      CALL CLEAN(TITLE,Q)
C Open file for input
      OPEN(UNIT=40,FILE=TITLE(1:Q),STATUS='OLD')
C Open files for input
      OPEN(UNIT=20,FILE='./tmp/humidity.int',STATUS='OLD')
      OPEN(UNIT=30,FILE='./inp/parameter.txt',STATUS='OLD')
      OPEN(UNIT=60,FILE='./tmp/tring.int',STATUS='OLD')
C Check if stage temp data file was generated and open it for input
      INQUIRE(FILE='./tmp/stemp.int',EXIST=TEST)
      IF(TEST) OPEN(UNIT=70,FILE='./tmp/stemp.int',STATUS='OLD')
C Open file for output
      OPEN(UNIT=80,FILE='./tmp/chord.int',STATUS='UNKNOWN')
C Call subroutine NME to generate output file name
      CALL NME('xys',SNUM,TITLE,3)
C Call subroutine CLEAN to clean file name
      CALL CLEAN(TITLE,Q)
C Open file for output
      OPEN(UNIT=50,FILE=TITLE(1:Q),STATUS='UNKNOWN')
C Call subroutine NME to generate output file name
      CALL NME('xyst',SNUM,TITLE,3)
C Call subroutine CLEAN to clean file name
      CALL CLEAN(TITLE,Q)
C Open file for output
      OPEN(UNIT=150,FILE=TITLE(1:Q),STATUS='UNKNOWN')
C Call subroutine NME to generate output file name
      CALL NME('pitchs',SNUM,TITLE,3)
```

```
C Call subroutine CLEAN to clean file name
      CALL CLEAN(TITLE,Q)
C Open file for output
      OPEN(UNIT=110,FILE=TITLE(1:Q),STATUS='UNKNOWN')
C Call subroutine NME to generate output file name
      CALL NME('yaws',SNUM,TITLE,3)
C Call subroutine CLEAN to clean file name
      CALL CLEAN(TITLE,Q)
C Open file for output
      OPEN(UNIT=120,FILE=TITLE(1:Q),STATUS='UNKNOWN')
C Call subroutine NME to generate output file name
      CALL NME('fulls',SNUM,TITLE,3)
C Call subroutine CLEAN to clean file name
      CALL CLEAN(TITLE,Q)
C Open file for output
      OPEN(UNIT=130,FILE=TITLE(1:Q),STATUS='UNKNOWN')
C Call subroutine NME to generate output file name
      CALL NME('hubs',SNUM,TITLE,3)
C Call subroutine CLEAN to clean file name
      CALL CLEAN(TITLE,Q)
C Open file for output
      OPEN(UNIT=140,FILE=TITLE(1:Q),STATUS='UNKNOWN')
C Call subroutine NME to generate output file name
      CALL NME('totals',SNUM,TITLE,3)
C Call subroutine CLEAN to clean file name
      CALL CLEAN(TITLE,Q)
C Open file for output
      OPEN(UNIT=100,FILE=TITLE(1:Q),STATUS='UNKNOWN')
C Call subroutine NME to generate output file name
      CALL NME('basics',SNUM,TITLE,3)
C Call subroutine CLEAN to clean file name
      CALL CLEAN(TITLE,Q)
C Open file for output
      OPEN(UNIT=90,FILE=TITLE(1:Q),STATUS='UNKNOWN')
C Reset data counters and logical variables
      N=1
      J=0
      VALID=.TRUE.
C Read correction parameters
      READ(30,*)
      READ(30,*) DUMMY
      READ(30,*)
      READ(30,*) DUMMY
      READ(30,*)
      READ(30,*) SRPM
      READ(30,*)
      READ(30,*) SRHT
      READ(30,*)
      READ(30,*) SPATM
      READ(30,*)
      READ(30,*) STT1
      READ(30,*)
      READ(30,*) SPT1
      READ(30,*)
      READ(30,*) SPS1
C Read the geometric parameters
      READ(30,*)
      READ(30,*) G1
      READ(30,*)
      READ(30,*) G2
      READ(30,*)
      READ(30,*) G3
      READ(30,*)
      READ(30,*) BH
      READ(30,*)
      READ(30,*) HR
      TIPRAD=HR+BH
      READ(30,*)
      READ(30,*) CHORD3
      READ(30,*)
      READ(30,*) CHORD4
      READ(30,*)
      READ(30,*) CHORD5
      READ(30,*)
      READ(30,*) AXIAL3
      READ(30,*)
      READ(30,*) AXIAL4
      READ(30,*)
      READ(30,*) AXIAL5
      READ(30,*)
      IF(L.EQ.R3) READ(30,*) G4
      IF(L.EQ.S4) THEN
        READ(30,*) DUMMY
        READ(30,*)
        READ(30,*) G4
      ENDIF
      IF(L.EQ.R5) THEN
        READ(30,*) DUMMY
        READ(30,*)
        READ(30,*) DUMMY
        READ(30,*)
        READ(30,*) G4
      ENDIF
C Write data to output file
      WRITE(80,*) CHORD3,CHORD4,CHORD5
      WRITE(80,*) AXIAL3,AXIAL4,AXIAL5
C Get the radial preset probe gamma angles with the turbine fixed z-axis
      I=0
    1 READ(40,*,END=2) A1,A2
      I=I+1
      ANGLE(I,1)=A1
      ANGLE(I,2)=A2
      GO TO 1
C Calculate standard flow parameters
    2 ASTT1=STT1+KELVIN
      ASPT1=SPT1+SPATM
      ASPS1=SPS1+SPATM
C Call TSTAT subroutine to calculate static temperature
      CALL TSTAT(ASTT1,ASPT1,ASPS1,SRHT,ASTS1)
C Call DEN subroutine to calculate density
      CALL DEN(ASTS1,ASPS1,SRHT,SRHO1)
C Read input files
    3 READ(10,*,END=4) CIRCUM,RADIAL,P1,P2,P3,P4,P5,PT1,PS1,PATM,
     $                 TT1,TT7,RPM
      READ(20,*) RHT
C Read corrected stage temperature
      IF(TEST) THEN
        READ(70,*) STEMP3,STEMP4,STEMP5
        IF(L.EQ.R3) TT=STEMP3
        IF(L.EQ.S4) TT=STEMP4
        IF(L.EQ.R5) TT=STEMP5
      ENDIF
C Calculate location coordinates in fixed turbine r-theta-z axis system
      RADIUS=HR+G4+(G2-RADIAL)
      THETA=DATAN((CIRCUM-G1)/G3)
C Calculate location coordinates in fixed room x-y-z axis system
      Y=RADIUS*DCOS(THETA)
      X=-RADIUS*DSIN(THETA)
C Find the local probe gamma angle with the room fixed yz-plane
      DO 5 K=1,I-1
      IF(RADIAL.LE.ANGLE(K,1).AND.RADIAL.GT.ANGLE(K+1,1)) THEN
        GMA=ANGLE(K,2)*(PI/DEGREE)
      ENDIF
    5 CONTINUE
```

```
      IF(RADIAL.LE.ANGLE(I,1)) GMA=ANGLE(I,2)*(PI/DEGREE)
C Call calibration subroutine
      IF(L.EQ.R3) CALL TCAL13(P1,P2,P3,P4,P5,PS,PT,ALPHA,BETA,N)
      IF(L.EQ.S4) CALL TCAL22(P1,P2,P3,P4,P5,PS,PT,ALPHA,BETA,N)
      IF(L.EQ.R5) CALL TCAL32(P1,P2,P3,P4,P5,PS,PT,ALPHA,BETA,N)
C Remove bad data points
      IF(PS.GT.PT) THEN
        WRITE(150,*) X,Y
        GO TO 7
      ENDIF
      IF(DABS(ALPHA).GT.AMAX.OR.DABS(BETA).GT.BMAX) THEN
        WRITE(150,*) X,Y
        GO TO 7
      ENDIF
C Remove non-converging data points
      IF(PT.EQ.ZERO) THEN
        IF(PS.EQ.ZERO) THEN
          IF(ALPHA.EQ.ZERO) THEN
            IF(BETA.EQ.ZERO) THEN
              WRITE(150,*) X,Y
              GO TO 7
            ENDIF
          ENDIF
        ENDIF
      ENDIF
C Calculate reference flow parameters
      ATT1=TT1+KELVIN
      APT1=PT1+PATM
      APS1=PS1+PATM
C Call TSTAT subroutine to calculate static temperature
      CALL TSTAT(ATT1,APT1,APS1,RHT,ATS1)
C Call DEN subroutine to calculate density
      CALL DEN(ATS1,APS1,RHT,RHO1)
C Assume total temperature at current station for first iteration
      IF(.NOT.TEST) THEN
        IF(L.EQ.R3) TT=0.50D0*(TT1-TT7)+TT7
        IF(L.EQ.S4) TT=0.49D0*(TT1-TT7)+TT7
        IF(L.EQ.R5) TT=ZERO*(TT1-TT7)+TT7
      ENDIF
C Calculate current station flow parameters
      ATT=TT+KELVIN
      APT=PT+PATM
      APS=PS+PATM
C Call TSTAT subroutine to calculate static temperature
      CALL TSTAT(ATT,APT,APS,RHT,ATS)
C Call DEN subroutine to calculate density
      CALL DEN(ATS,APS,RHT,RHO)
C Calculate velocity
      V=DSQRT((PT-PS)*2.0D0/RHO)
C Perform flow correction
      VCR=(SRPM/RPM)*V
      RHOCR=(SRHO1/RHO1)*RHO
      PSCR=(SRHO1/RHO1)*((SRPM/RPM)**2.0D0)*PS
      PTCR=PSCR+0.5D0*RHOCR*VCR**2.0D0
      APSCR=PSCR+SPATM
      APTCR=PTCR+SPATM
C Calculate flow components in fixed turbine r-theta-z axis system
      VCCR=VCR*(-DSIN(BETA*(PI/DEGREE))*DCOS(GMA)+
     $     DCOS(BETA*(PI/DEGREE))*DCOS(ALPHA*(PI/DEGREE))*DSIN(GMA))
      VRCR=-VCR*DCOS(BETA*(PI/DEGREE))*DSIN(ALPHA*(PI/DEGREE))
VZCR=VCR*(DSIN(BETA*(PI/DEGREE))*DSIN(GMA)+
     $     DCOS(BETA*(PI/DEGREE))*DCOS(ALPHA*(PI/DEGREE))*DCOS(GMA))
      VMCR=DSQRT(VZCR**2.0D0+VRCR**2.0D0)
C Calculate absolute flow angles in fixed turbine r-theta-z axis system
      APHI=DACOS(VCCR/DSQRT(VZCR**2.0D0+VCCR**2.0D0))*(DEGREE/PI)
      APSI=DACOS(VRCR/VMCR)*(DEGREE/PI)
C Calculate corrected rotational velocity
```

```
      UCR=(0.001D0/30.0D0)*RADIUS*SRPM*PI
C Compute relative flow comp. in fixed turbine r-theta-z axis system
      WZCR=VZCR
      WRCR=VRCR
      WCCR=VCCR-UCR
      WCR=DSQRT(WZCR**2.0D0+WRCR**2.0D0+WCCR**2.0D0)
C Calculate relative flow angles in fixed turbine r-theta-z axis system
      RPHI=DACOS(WCCR/DSQRT(WZCR**2.0D0+WCCR**2.0D0))*(DEGREE/PI)
C Calculate the relative corrected total pressure
      APRCR=APSCR+0.5D0*RHOCR*(WCR**2.0D0-UCR**2.0D0)
C Call subroutine RDEN to calculate corrected absolute static temp.
      CALL RDEN(APSCR,SRHT,RHOCR,ATSCR)
      TSCR=ATSCR-KELVIN
C Calculate corrected flow viscosity using Sutherland's equation
      MU=1.46D-06*((ATSCR**1.5D0)/(ATSCR+111.0D0))
C Call PROP subroutine to calculate corrected thermodynamic properties
      CALL PROP(ATSCR,APSCR,SRHT,CP,R,GAMMA,ENS,S)
C Calculate absolute and relative Mach numbers
      MACH=(VCR)/((GAMMA*R*ATSCR)**0.5D0)
      MACHR=(WCR)/((GAMMA*R*ATSCR)**0.5D0)
C Calculate the total corrected enthalpy
      ENT=ENS+0.5D0*VCR**2.0D0
C Calculate the total corrected rothalpy
      ENTR=ENS+0.5D0*(WCR**2.0D0-UCR**2.0D0)
C Approximate data for hub
C The hub is rotating with the turbine
      IF(VALID) THEN
        CHECK=RADIUS
        VALID=.FALSE.
      ENDIF
      IF(RADIUS.EQ.CHECK) THEN
C Compute the actual and corrected hub velocity
        UH=(0.001D0/30.0D0)*HR*RPM*PI
        UHCR=(0.001D0/30.0D0)*HR*SRPM*PI
C Assume static pressure at hub equals static pressure near the hub
        APSH=APS
        ATTH=ATT
C Guess the static hub temperature
        ATSH=ATS
C Iterate to get exact static hub temperature
    6   CALL DEN(ATSH,APSH,RHT,RHOH)
        TEMP=ATSH
        VH=UH
        APTH=APSH+0.5D0*RHOH*VH**2.0D0
C Call TSTAT subroutine to calculate static temperature
        CALL TSTAT(ATTH,APTH,APSH,RHT,ATSH)
        TERR=DABS((ATSH-TEMP)/ATSH)
        IF(TERR.LE.LIMIT) GO TO 8
        GO TO 6
C Perform flow correction
    8   RHOHCR=(SRHO1/RHO1)*RHOH
        VHCR=VH*(SRPM/RPM)
        VCHCR=VHCR
        VRHCR=ZERO
        VZHCR=ZERO
        VMHCR=ZERO
        WHCR=ZERO
        WCHCR=ZERO
        WRHCR=ZERO
        WZHCR=ZERO
        APSIH=ZERO
        APHIH=ZERO
        RPHIH=ZERO
        APSHCR=APSH*(SRHO1/RHO1)*((SRPM/RPM)**2.0D0)
        APTHCR=APSHCR+0.5D0*RHOHCR*VHCR**2.0D0
C Call subroutine RDEN to calculate corrected absolute static temp.
        CALL RDEN(APSHCR,SRHT,RHOHCR,ATSHCR)
        TSHCR=ATSHCR-KELVIN
        APRHCR=APSHCR+0.5D0*RHOHCR*(WHCR**2.0D0-UHCR**2.0D0)
C Calculate corrected flow viscosity using Sutherland's equation
        MUH=1.46D-06*((ATSHCR**1.5D0)/(ATSHCR+111.0D0))
C Call PROP subroutine to calculate corrected thermodynamic properties
        CALL PROP(ATSHCR,APSHCR,SRHT,CPH,RH,GAMMAH,ENSH,SH)
C Calculate absolute and relative Mach numbers
        MACHH=(VHCR)/((GAMMAH*RH*ATSHCR)**0.5D0)
        MACHRH=(WHCR)/((GAMMAH*RH*ATSHCR)**0.5D0)
C Calculate the total corrected enthalpy
        ENTH=ENSH+0.5D0*VHCR**2.0D0
C Calculate the total corrected rothalpy
        ENTHR=ENSH+0.5D0*(WHCR**2.0D0-UHCR**2.0D0)
C Write hub data to output file
        WRITE(140,*) THETA,HR,APTHCR,APRHCR,APSHCR,VHCR,VCHCR,
     $               VRHCR,VZHCR,VMHCR,WHCR,WCHCR,WRHCR,WZHCR,UHCR,
     $               APSIH,APHIH,RPHIH,MACHH,MACHRH,TSHCR,RHOHCR,
     $               RH,CPH,SH,ENSH,ENTH,ENTHR,MUH
      ENDIF
C Write data to output files
      WRITE(50,*) X,Y
      WRITE(110,*) X,Y,ALPHA
      WRITE(120,*) X,Y,BETA
      WRITE(130,*) THETA,RADIUS,APTCR,APRCR,APSCR,VCR,VCCR,VRCR,
     $             VZCR,VMCR,WCR,WCCR,WRCR,WZCR,UCR,
     $             APSI,APHI,RPHI,MACH,MACHR,TSCR,RHOCR,
     $             R,CP,S,ENS,ENT,ENTR,MU
      WRITE(90,*) APS,APT
      J=J+1
    7 N=N+1
      GO TO 3
C Approximate data for tip
C The tip is stationary (casing)
    4 IF(L.EQ.1) THEN
        READ(60,*) PST,DUMMY,DUMMY
      ENDIF
      IF(L.EQ.2) THEN
        READ(60,*) DUMMY,PST,DUMMY
      ENDIF
      IF(L.EQ.3) THEN
        READ(60,*) DUMMY,DUMMY,PST
      ENDIF
C Absolute velocity at tip is nill
      VT=ZERO
      ATTT=ATT
      ATST=ATTT
      APST=PST+PATM
C Call subroutine DEN to calculate tip density
      CALL DEN(ATST,APST,RHT,RHOT)
C Correct flow
      RHOTCR=(SRHO1/RHO1)*RHOT
      APSTCR=APST*(SRHO1/RHO1)*((SRPM/RPM)**2.0D0)
      APTTCR=APSTCR
      VTCR=ZERO
      VCTCR=ZERO
      VRTCR=ZERO
      VZTCR=ZERO
      VMTCR=ZERO
      UTCR=(0.001D0/30.0D0)*TIPRAD*SRPM*PI
      WTCR=UTCR
      WCTCR=WTCR
      WRTCR=ZERO
      WZTCR=ZERO
      APSIT=ZERO
      APHIT=ZERO
      RPHIT=DEGREE
C Call subroutine RDEN to calculate corrected absolute static temp.
```

```
      CALL RDEN(APSTCR,SRHT,RHOTCR,ATSTCR)
      TSTCR=ATSTCR-KELVIN
C Calculate corrected flow viscosity using Sutherland's equation
      MUT=1.46D-06*((ATSTCR**1.5D0)/(ATSTCR+111.0D0))
      APRTCR=APSTCR
      MACHT=ZERO
C Call PROP subroutine to calculate corrected thermodynamic properties
      CALL PROP(ATSTCR,APSTCR,SRHT,CPT,RT,GAMMAT,ENST,ST)
C Calculate the relative Mach number
      MACHRT=(WTCR)/((GAMMAT*RT*ATSTCR)**0.5D0)
C Calculate the total corrected enthalpy
      ENTT=ENST+0.5D0*VTCR**2.0D0
C Calculate the total corrected rothalpy
      ENTTR=ENST+0.5D0*(WTCR**2.0D0-UTCR**2.0D0)
C Write tip data to output file
      WRITE(130,*) THETA,TIPRAD,APTTCR,APRTCR,APSTCR,VTCR,VCTCR,VRTCR,
     $             VZTCR,VMTCR,WTCR,WCTCR,WRTCR,WZTCR,UTCR,APSIT,
     $             APHIT,RPHIT,MACHT,MACHRT,TSTCR,RHOTCR,
     $             RT,CPT,ST,ENST,ENTT,ENTTR,MUT
C Print summary to screen
      PRINT*, 'ANALYSIS FOR STATION NUMBER ',SNUM,' COMPLETE.'
C Correct total number of points read
      N=N-1
      PRINT*, N,' POINTS READ. '
      PRINT*, J,' POINTS WRITTEN.'
C Correct total number of points read including tip
      J=J+1
C Write total number of points written to output file
      WRITE(100,*) J
      RETURN
      END
************************************************************************
* This subroutine reads the averaged circumferential data at each      *
* station and calculates the mass flow rate and the radially averaged  *
* flow quantities at each station in addition to the dimensionless     *
* loss coefficients and the thermodynamic efficiencies at each radial  *
* location using streamline analysis.                                  *
* Input Files : averages3.int, averages4.int, averages5.int, chord.int *
* Output Files: slines.dat, ptloss.dat, pdloss.dat,                    *
*               efficiency.dat, report.dat                             *
* Experimental Procedure: Research Turbine Facility - TPFL             *
************************************************************************
* Hicham A Chibli - Texas A&M University - 03/15/10                    *
************************************************************************
      SUBROUTINE SLOSS(RADIAL,GRID,ISKIP,PTRIM,MSPAN)
      INTEGER N,M,I,K,IMAX,RADIAL,GRID,ISKIP,NUNIT,N3,N4,N5,
     $        PSS3,PES3,PSS4,PES4,PSS5,PES5,IS3,IS4,IS5
      DOUBLE PRECISION MSPAN,PTRIM,LS3,LS4,LS5
      PARAMETER(IMAX=1000)
      DOUBLE PRECISION S3(IMAX,28),S4(IMAX,28),S5(IMAX,28),TEMP(28),
     $                 DA31(RADIAL*GRID+1),DA32(RADIAL*GRID+1),
     $                 DA41(RADIAL*GRID+1),DA42(RADIAL*GRID+1),
     $                 DA51(RADIAL*GRID+1),DA52(RADIAL*GRID+1),
     $                 DB32(RADIAL*GRID+1),DB42(RADIAL*GRID+1),
     $                 DB52(RADIAL*GRID+1),
     $                 ES3(RADIAL*GRID+1,28),ES4(RADIAL*GRID+1,28),
     $                 ES5(RADIAL*GRID+1,28),
     $                 AS3(28),AS4(28),AS5(28),
     $                 PAS3(28),PAS4(28),PAS5(28),
     $                 PTXIS(RADIAL+1,2),PDXIS(RADIAL+1,2),
     $                 PTXIR(RADIAL+1,2),PDXIR(RADIAL+1,2),
     $                 EFFS(RADIAL+1,2),EFFR(RADIAL+1,2),
     $                 EFFSI(RADIAL+1,2),EFFST(RADIAL+1,2)
      INTEGER STREAM3(RADIAL+1),STREAM4(RADIAL+1),STREAM5(RADIAL+1)
      DOUBLE PRECISION MFLOW3,MFLOW4,MFLOW5,PMFLOW3,PMFLOW4,PMFLOW5,
     $                 RE3,RE4,RE5,
     $                 PTLS,PDLS,PTLR,PDLR,PPDLS,SPDLS,PPDLR,SPDLR,
     $                 EFS,EFR,EFSI,EFST,
     $                 LAST4,NOW4,LINEAR4,LAST5,NOW5,LINEAR5,
     $                 AXIAL3,AXIAL4,AXIAL5,TIP,HUB,BH,
     $                 CHORD3,CHORD4,CHORD5,D3,D4,D5,
     $                 DHS,DH,TS,R,CP,GAMMA,
     $                 KELVIN,PI,UCNVRT,ZERO,ONE,NINETY
      CHARACTER STATOR*6,ROTOR*5,STAGE*5,ISENTROPIC*16
      PARAMETER(PI=3.141592D0,KELVIN=273.15D0)
      PARAMETER(UCNVRT=0.001D0,ZERO=0.0D0,ONE=1.0D0,NINETY=90.0D0)
C Open files for input
      OPEN(UNIT=10,FILE='./tmp/averages3.int',STATUS='OLD')
      OPEN(UNIT=20,FILE='./tmp/averages4.int',STATUS='OLD')
      OPEN(UNIT=30,FILE='./tmp/averages5.int',STATUS='OLD')
      OPEN(UNIT=90,FILE='./tmp/chord.int',STATUS='OLD')
C Open files for output
      OPEN(UNIT=40,FILE='./out/slines.dat',STATUS='UNKNOWN')
      OPEN(UNIT=50,FILE='./out/ptloss.dat',STATUS='UNKNOWN')
      OPEN(UNIT=80,FILE='./out/pdloss.dat',STATUS='UNKNOWN')
      OPEN(UNIT=60,FILE='./out/efficiency.dat',STATUS='UNKNOWN')
      OPEN(UNIT=70,FILE='./out/report.dat',STATUS='UNKNOWN')
C Read input geometric data
      READ(90,*) CHORD3,CHORD4,CHORD5
      READ(90,*) AXIAL3,AXIAL4,AXIAL5
C Read input data
      DO 1 I=1,3
         NUNIT=10*I
         K=0
    2    READ(NUNIT,*,END=3) TEMP
         K=K+1
         DO 4 M=1,28
            IF(I.EQ.1) S3(K,M)=TEMP(M)
            IF(I.EQ.2) S4(K,M)=TEMP(M)
            IF(I.EQ.3) S5(K,M)=TEMP(M)
    4    CONTINUE
         GO TO 2
    3    IF(I.EQ.1) N3=K
         IF(I.EQ.2) N4=K
         IF(I.EQ.3) N5=K
    1 CONTINUE
C Calculate the tip and hub radii and the blade height
      HUB=S4(1,1)
      TIP=S4(N4,1)
      BH=TIP-HUB
C Estimate the mass flow rate based on the un-expanded arrays
C used for checking purposes for comparing with the expanded math
      MFLOW3=0.0D0
      MFLOW4=0.0D0
      MFLOW5=0.0D0
C Station 3
      DO 68 N=1,N3-1
         MFLOW3=MFLOW3+0.5D0*(S3(N+1,1)-S3(N,1))*
     $          (S3(N,1)*S3(N,21)*S3(N,8)+
     $           S3(N+1,1)*S3(N+1,21)*S3(N+1,8))*
     $          (2.0D0*PI*UCNVRT**2.0D0)
   68 CONTINUE
C Station 4
      DO 69 N=1,N4-1
         MFLOW4=MFLOW4+0.5D0*(S4(N+1,1)-S4(N,1))*
     $          (S4(N,1)*S4(N,21)*S4(N,8)+
     $           S4(N+1,1)*S4(N+1,21)*S4(N+1,8))*
     $          (2.0D0*PI*UCNVRT**2.0D0)
   69 CONTINUE
C Station 5
      DO 70 N=1,N5-1
         MFLOW5=MFLOW5+0.5D0*(S5(N+1,1)-S5(N,1))*
     $          (S5(N,1)*S5(N,21)*S5(N,8)+
     $           S5(N+1,1)*S5(N+1,21)*S5(N+1,8))*
     $          (2.0D0*PI*UCNVRT**2.0D0)
   70 CONTINUE
      PRINT*
      PRINT*, '------------------- BASIC ESTIMATION -------------------'
      WRITE(*,65) MFLOW3,MFLOW4,MFLOW5
C Check for mass flow convergence at the 3 stations
      IF(MFLOW4/MFLOW3.GT.1.05D0.OR.MFLOW4/MFLOW3.LT.0.95D0) THEN
         PRINT*, 'ERROR: CHECK PROBE ANGLES.'
         STOP
      ENDIF
      IF(MFLOW5/MFLOW3.GT.1.05D0.OR.MFLOW5/MFLOW3.LT.0.95D0) THEN
         PRINT*, 'ERROR: CHECK PROBE ANGLES.'
         STOP
      ENDIF
C Call subroutine EXPAND to populate arrays
      CALL EXPAND(S3,ES3,IMAX,N3,RADIAL*GRID+1,28)
      CALL EXPAND(S4,ES4,IMAX,N4,RADIAL*GRID+1,28)
      CALL EXPAND(S5,ES5,IMAX,N5,RADIAL*GRID+1,28)
C Calculate the mass flow rate through the turbine at station 3
      DO 5 I=1,RADIAL*GRID+1
         DA31(I)=ES3(I,1)*UCNVRT
         DA32(I)=ES3(I,1)*ES3(I,21)*ES3(I,8)*(UCNVRT*2.0D0*PI)
    5 CONTINUE
      CALL INTG(DA31,DA32,RADIAL*GRID+1,1,RADIAL*GRID+1,MFLOW3)
C Calculate the mass flow rate through the turbine at station 4
      DO 6 I=1,RADIAL*GRID+1
         DA41(I)=ES4(I,1)*UCNVRT
         DA42(I)=ES4(I,1)*ES4(I,21)*ES4(I,8)*(UCNVRT*2.0D0*PI)
    6 CONTINUE
      CALL INTG(DA41,DA42,RADIAL*GRID+1,1,RADIAL*GRID+1,MFLOW4)
C Calculate the mass flow rate through the turbine at station 5
      DO 7 I=1,RADIAL*GRID+1
         DA51(I)=ES5(I,1)*UCNVRT
         DA52(I)=ES5(I,1)*ES5(I,21)*ES5(I,8)*(UCNVRT*2.0D0*PI)
    7 CONTINUE
      CALL INTG(DA51,DA52,RADIAL*GRID+1,1,RADIAL*GRID+1,MFLOW5)
C Write the mass flow to screen for angle calibration purposes
      PRINT*, '------------------ REFINED ESTIMATION ------------------'
      WRITE(*,65) MFLOW3,MFLOW4,MFLOW5
   65 FORMAT(/,25X,'STATION3',2X,'STATION4',2X,'STATION5',/,
     $       1X,'MASS FLOW [kg/sec]',6X,F8.4,2X,F8.4,2X,F8.4,/)
C Check for mass flow convergence at the 3 stations
      IF(MFLOW4/MFLOW3.GT.1.01D0.OR.MFLOW4/MFLOW3.LT.0.99D0) THEN
         PRINT*, 'ERROR: CHECK PROBE ANGLES.'
         STOP
      ENDIF
      IF(MFLOW5/MFLOW3.GT.1.01D0.OR.MFLOW5/MFLOW3.LT.0.99D0) THEN
         PRINT*, 'ERROR: CHECK PROBE ANGLES.'
         STOP
      ENDIF
C Perform radial mass averaging of data
      DO 66 M=2,28
         DO 67 I=1,RADIAL*GRID+1
            DB32(I)=ES3(I,M)*ES3(I,1)*ES3(I,21)*ES3(I,8)*
     $              (UCNVRT*2.0D0*PI)
            DB42(I)=ES4(I,M)*ES4(I,1)*ES4(I,21)*ES4(I,8)*
     $              (UCNVRT*2.0D0*PI)
            DB52(I)=ES5(I,M)*ES5(I,1)*ES5(I,21)*ES5(I,8)*
     $              (UCNVRT*2.0D0*PI)
   67    CONTINUE
         CALL INTG(DA31,DB32,RADIAL*GRID+1,1,RADIAL*GRID+1,AS3(M))
         CALL INTG(DA41,DB42,RADIAL*GRID+1,1,RADIAL*GRID+1,AS4(M))
         CALL INTG(DA51,DB52,RADIAL*GRID+1,1,RADIAL*GRID+1,AS5(M))
         AS3(M)=AS3(M)/MFLOW3
         AS4(M)=AS4(M)/MFLOW4
         AS5(M)=AS5(M)/MFLOW5
   66 CONTINUE
```

```
C Calculate the average Reynold's numbers at each station
      RE3=UCNVRT*CHORD3*AS3(21)*AS3(5)/AS3(28)
      RE4=UCNVRT*CHORD4*AS4(21)*AS4(5)/AS4(28)
      RE5=UCNVRT*CHORD5*AS5(21)*AS5(5)/AS5(28)
C Sort the 3 stations to find the start point of the profile array
      IS3=0
      IS4=0
      IS5=0
      DO 71 I=1,RADIAL*GRID+1
         LS3=(ES3(I,1)-ES3(1,1))/(ES3(RADIAL*GRID+1,1)-ES3(1,1))
         LS4=(ES4(I,1)-ES4(1,1))/(ES4(RADIAL*GRID+1,1)-ES4(1,1))
         LS5=(ES5(I,1)-ES5(1,1))/(ES5(RADIAL*GRID+1,1)-ES5(1,1))
         IF(IS3.EQ.0) THEN
           IF(LS3.GE.MSPAN) THEN
             PSS3=I
             IS3=1
           ENDIF
         ENDIF
         IF(IS4.EQ.0) THEN
           IF(LS4.GE.MSPAN) THEN
             PSS4=I
             IS4=1
           ENDIF
         ENDIF
         IF(IS5.EQ.0) THEN
           IF(LS5.GE.MSPAN) THEN
             PSS5=I
             IS5=1
           ENDIF
         ENDIF
   71 CONTINUE
C Sort the 3 stations to find the end point of the profile array
      IS3=0
      IS4=0
      IS5=0
      DO 72 I=RADIAL*GRID+1,1,-1
         LS3=(ES3(I,1)-ES3(1,1))/(ES3(RADIAL*GRID+1,1)-ES3(1,1))
         LS4=(ES4(I,1)-ES4(1,1))/(ES4(RADIAL*GRID+1,1)-ES4(1,1))
         LS5=(ES5(I,1)-ES5(1,1))/(ES5(RADIAL*GRID+1,1)-ES5(1,1))
         IF(IS3.EQ.0) THEN
           IF(LS3.LE.1.0D0-MSPAN) THEN
             PES3=I
             IS3=1
           ENDIF
         ENDIF
         IF(IS4.EQ.0) THEN
           IF(LS4.LE.1.0D0-MSPAN) THEN
             PES4=I
             IS4=1
           ENDIF
         ENDIF
         IF(IS5.EQ.0) THEN
           IF(LS5.LE.1.0D0-MSPAN) THEN
             PES5=I
             IS5=1
           ENDIF
         ENDIF
   72 CONTINUE
C Mass average the profile array data for the 3 stations
      CALL INTG(DA31,DA32,RADIAL*GRID+1,PSS3,PES3,PMFLOW3)
      CALL INTG(DA41,DA42,RADIAL*GRID+1,PSS4,PES4,PMFLOW4)
      CALL INTG(DA51,DA52,RADIAL*GRID+1,PSS5,PES5,PMFLOW5)
      DO 73 M=2,28
         DO 76 I=1,RADIAL*GRID+1
            DB32(I)=ES3(I,M)*ES3(I,1)*ES3(I,21)*ES3(I,8)*
     $              (UCNVRT*2.0D0*PI)
            DB42(I)=ES4(I,M)*ES4(I,1)*ES4(I,21)*ES4(I,8)*
     $              (UCNVRT*2.0D0*PI)
            DB52(I)=ES5(I,M)*ES5(I,1)*ES5(I,21)*ES5(I,8)*
     $              (UCNVRT*2.0D0*PI)
   76    CONTINUE
         CALL INTG(DA31,DB32,RADIAL*GRID+1,PSS3,PES3,PAS3(M))
         CALL INTG(DA41,DB42,RADIAL*GRID+1,PSS4,PES4,PAS4(M))
         CALL INTG(DA51,DB52,RADIAL*GRID+1,PSS5,PES5,PAS5(M))
         PAS3(M)=PAS3(M)/PMFLOW3
         PAS4(M)=PAS4(M)/PMFLOW4
         PAS5(M)=PAS5(M)/PMFLOW5
   73 CONTINUE
C Populate the stream location array at station 3
      DO 8 I=0,RADIAL
         STREAM3(I+1)=I*GRID+1
    8 CONTINUE
C Populate the bounds of the stream location arrays at station 4 and 5
      STREAM4(1)=1
      STREAM4(RADIAL+1)=RADIAL*GRID+1
      STREAM5(1)=1
      STREAM5(RADIAL+1)=RADIAL*GRID+1
C Populate the stream location arrays at station 4 and 5
      DO 9 I=1,RADIAL-1
         LAST4=ZERO
         LAST5=ZERO
         CALL INTG(DA31,DA32,RADIAL*GRID+1,STREAM3(I),STREAM3(I+1),D3)
         DO 10 N=2,STREAM4(RADIAL+1)-STREAM4(I),2
            CALL INTG(DA41,DA42,RADIAL*GRID+1,STREAM4(I),
     $                STREAM4(I)+N,D4)
            IF(D4.GE.D3) GO TO 12
            LAST4=D4
   10    CONTINUE
   11    PRINT*, 'ERROR: INCREASE GRID RESOLUTION.'
         STOP
   12    NOW4=LAST4+0.5D0*(DA42(STREAM4(I)+N-1)+DA42(STREAM4(I)+N-2))*
     $        (DA41(STREAM4(I)+N-1)-DA41(STREAM4(I)+N-2))
         IF(NOW4.GE.D3) THEN
           LINEAR4=(D3-LAST4)/(NOW4-LAST4)
           IF(LINEAR4.NE.1.0D0) THEN
             DO 13 M=1,28
                TEMP(M)=LINEAR4*(ES4(STREAM4(I)+N-1,M)-
     $                  ES4(STREAM4(I)+N-2,M))+ES4(STREAM4(I)+N-2,M)
                ES4(STREAM4(I)+N-2,M)=TEMP(M)
   13        CONTINUE
             STREAM4(I+1)=STREAM4(I)+N-2
             DA41(STREAM4(I)+N-2)=ES4(STREAM4(I)+N-2,1)*UCNVRT
             DA42(STREAM4(I)+N-2)=ES4(STREAM4(I)+N-2,1)*
     $                            ES4(STREAM4(I)+N-2,21)*
     $                            ES4(STREAM4(I)+N-2,8)*
     $                            (UCNVRT*2.0D0*PI)
           ELSE
               STREAM4(I+1)=STREAM4(I)+N-1
           ENDIF
         ELSE
             LINEAR4=(D3-NOW4)/(D4-NOW4)
             IF(LINEAR4.NE.1.0D0) THEN
               DO 14 M=1,28
                  TEMP(M)=LINEAR4*(ES4(STREAM4(I)+N,M)-
     $                    ES4(STREAM4(I)+N-1,M))+ES4(STREAM4(I)+N-1,M)
                  ES4(STREAM4(I)+N-1,M)=TEMP(M)
   14          CONTINUE
               STREAM4(I+1)=STREAM4(I)+N-1
               DA41(STREAM4(I)+N-2)=ES4(STREAM4(I)+N-2,1)*UCNVRT
               DA42(STREAM4(I)+N-2)=ES4(STREAM4(I)+N-2,1)*
     $                              ES4(STREAM4(I)+N-2,21)*
     $                              ES4(STREAM4(I)+N-2,8)*
     $                              (UCNVRT*2.0D0*PI)
             ELSE
```

```
                 STREAM4(I+1)=STREAM4(I)+N
             ENDIF
         ENDIF
         DO 15 N=2,STREAM5(RADIAL+1)-STREAM5(I),2
            CALL INTG(DA51,DA52,RADIAL*GRID+1,STREAM5(I),
     $                STREAM5(I)+N,D5)
            IF(D5.GE.D3) GO TO 17
            LAST5=D5
   15    CONTINUE
   16    PRINT*, 'ERROR: INCREASE GRID RESOLUTION.'
         STOP
   17    NOW5=LAST5+0.5D0*(DA52(STREAM5(I)+N-1)+DA52(STREAM5(I)+N-2))*
     $        (DA51(STREAM5(I)+N-1)-DA51(STREAM5(I)+N-2))
         IF(NOW5.GE.D3) THEN
           LINEAR5=(D3-LAST5)/(NOW5-LAST5)
           IF(LINEAR5.NE.1.0D0) THEN
             DO 18 M=1,28
                TEMP(M)=LINEAR5*(ES5(STREAM5(I)+N-1,M)-
     $                  ES5(STREAM5(I)+N-2,M))+ES5(STREAM5(I)+N-2,M)
                ES5(STREAM5(I)+N-2,M)=TEMP(M)
   18        CONTINUE
             STREAM5(I+1)=STREAM5(I)+N-2
             DA51(STREAM5(I)+N-2)=ES5(STREAM5(I)+N-2,1)*UCNVRT
             DA52(STREAM5(I)+N-2)=ES5(STREAM5(I)+N-2,1)*
     $                            ES5(STREAM5(I)+N-2,21)*
     $                            ES5(STREAM5(I)+N-2,8)*
     $                            (UCNVRT*2.0D0*PI)
           ELSE
               STREAM5(I+1)=STREAM5(I)+N-1
           ENDIF
         ELSE
             LINEAR5=(D3-NOW5)/(D5-NOW5)
             IF(LINEAR5.NE.1.0D0) THEN
               DO 19 M=1,28
                  TEMP(M)=LINEAR5*(ES5(STREAM5(I)+N,M)-
     $                    ES5(STREAM5(I)+N-1,M))+ES5(STREAM5(I)+N-1,M)
                  ES5(STREAM5(I)+N-1,M)=TEMP(M)
   19          CONTINUE
               STREAM5(I+1)=STREAM5(I)+N-1
               DA51(STREAM5(I)+N-2)=ES5(STREAM5(I)+N-2,1)*UCNVRT
               DA52(STREAM5(I)+N-2)=ES5(STREAM5(I)+N-2,1)*
     $                              ES5(STREAM5(I)+N-2,21)*
     $                              ES5(STREAM5(I)+N-2,8)*
     $                              (UCNVRT*2.0D0*PI)
             ELSE
                 STREAM5(I+1)=STREAM5(I)+N
             ENDIF
         ENDIF
    9 CONTINUE
C Write stream line data to output file
      WRITE(40,*) 'TITLE="STREAMLINES"'
      WRITE(40,*) 'Variables="Axial Position  [mm]", ',
     $            '"Radial Position  [mm]"'
      DO 20 I=1,RADIAL+1
         WRITE(40,22) I
         WRITE(40,*) AXIAL3,ES3(STREAM3(I),1)
         WRITE(40,*) AXIAL4,ES4(STREAM4(I),1)
         WRITE(40,*) AXIAL5,ES5(STREAM5(I),1)
   20 CONTINUE
   22 FORMAT(1X,'ZONE T="streamline ',I4,'"')
C Calculate the pressure loss coefficients for the stator and rotor rows
C using both inlet total and exit dynamic heads
      DO 44 I=1,RADIAL+1
C Stator row calculations
         PTXIS(I,1)=(0.5D0*(ES3(STREAM3(I),1)+
     $               ES4(STREAM4(I),1))-HUB)/BH
         PTXIS(I,2)=(ES3(STREAM3(I),2)-ES4(STREAM4(I),2))/
     $              ES3(STREAM3(I),2)
         PDXIS(I,1)=PTXIS(I,1)
         PDXIS(I,2)=(ES3(STREAM3(I),2)-ES4(STREAM4(I),2))/
     $             (AS3(2)-AS4(4))
C Rotor row calculations
         PTXIR(I,1)=(0.5D0*(ES4(STREAM4(I),1)+
     $               ES5(STREAM5(I),1))-HUB)/BH
         PTXIR(I,2)=(ES4(STREAM4(I),3)-ES5(STREAM5(I),3))/
     $              ES4(STREAM4(I),3)
         PDXIR(I,1)=PTXIR(I,1)
         PDXIR(I,2)=(ES4(STREAM4(I),3)-ES5(STREAM5(I),3))/
     $             (AS4(3)-AS5(4))
   44 CONTINUE
C Initialize Tecplot headers
      STATOR=’STATOR’
      ROTOR=’ROTOR’
      STAGE=’STAGE’
      ISENTROPIC=’STAGE ISENTROPIC’
C Write total pressure loss data to output file
      WRITE(50,*) ’TITLE="LOSS COEFFICIENT"’
      WRITE(50,*) ’Variables="Immersion Ratio r^* ‘= ’,
     $           ’(r ‘- r_h_u_b) ~$ (r_t_i_p ‘- r_h_u_b)", ’,’"‘z"’
      WRITE(50,47) STATOR
      WRITE(80,*) ’TITLE="LOSS COEFFICIENT"’
      WRITE(80,*) ’Variables="Immersion Ratio r^* ‘= ’,
     $           ’(r ‘- r_h_u_b) ~$ (r_t_i_p ‘- r_h_u_b)", ’,’"‘z"’
      WRITE(80,47) STATOR
      DO 45 I=1,RADIAL+1
         IF(PTXIS(I,2).LT.ZERO) GO TO 45
         IF(PTXIS(I,1).GE.PTRIM.AND.PTXIS(I,1).LE.1.0D0-PTRIM) THEN
           IF(MOD(I,ISKIP).EQ.0) GO TO 77
           GO TO 45
         ENDIF
   77    WRITE(50,*) PTXIS(I,1),PTXIS(I,2)
   45 CONTINUE
      DO 53 I=1,RADIAL+1
         IF(PDXIS(I,2).LT.ZERO) GO TO 53
         IF(PDXIS(I,1).GE.PTRIM.AND.PDXIS(I,1).LE.1.0D0-PTRIM) THEN
           IF(MOD(I,ISKIP).EQ.0) GO TO 79
           GO TO 53
         ENDIF
   79    WRITE(80,*) PDXIS(I,1),PDXIS(I,2)
   53 CONTINUE
      WRITE(50,47) ROTOR
      WRITE(80,47) ROTOR
      DO 46 I=1,RADIAL+1
         IF(PTXIR(I,2).LT.ZERO) GO TO 46
         IF(PTXIR(I,1).GE.PTRIM.AND.PTXIR(I,1).LE.1.0D0-PTRIM) THEN
           IF(MOD(I,ISKIP).EQ.0) GO TO 78
           GO TO 46
         ENDIF
   78    WRITE(50,*) PTXIR(I,1),PTXIR(I,2)
   46 CONTINUE
      DO 54 I=1,RADIAL+1
         IF(PDXIR(I,2).LT.ZERO) GO TO 54
         IF(PDXIR(I,1).GE.PTRIM.AND.PDXIR(I,1).LE.1.0D0-PTRIM) THEN
           IF(MOD(I,ISKIP).EQ.0) GO TO 81
           GO TO 54
         ENDIF
   81    WRITE(80,*) PDXIR(I,1),PDXIR(I,2)
   54 CONTINUE
C Calculate the efficiency for the individual rows and the stage
      DO 49 I=1,RADIAL+1
C Stator row calculations
            CP=(ES3(STREAM3(I),23)+ES4(STREAM4(I),23))*0.5D0
            R=(ES3(STREAM3(I),22)+ES4(STREAM4(I),22))*0.5D0
            GAMMA=CP/(CP-R)
            TS=((ES3(STREAM3(I),20)+KELVIN)*(ES4(STREAM4(I),4)/
     $           ES3(STREAM3(I),4))**((GAMMA-1.0D0)/GAMMA))-KELVIN
            DHS=CP*(ES3(STREAM3(I),20)-TS)
            EFFS(I,1)=PTXIS(I,1)
            EFFS(I,2)=(0.5D0*ES4(STREAM4(I),5)**2.0D0)/
     $                (DHS+0.5D0*ES3(STREAM3(I),5)**2.0D0)
C Rotor row calculations
            CP=(ES4(STREAM4(I),23)+ES5(STREAM5(I),23))*0.5D0
            R=(ES4(STREAM4(I),22)+ES5(STREAM5(I),22))*0.5D0
            GAMMA=CP/(CP-R)
            TS=((ES4(STREAM4(I),20)+KELVIN)*(ES5(STREAM5(I),4)/
     $           ES4(STREAM4(I),4))**((GAMMA-1.0D0)/GAMMA))-KELVIN
            DHS=CP*(ES4(STREAM4(I),20)-TS)
            EFFR(I,1)=PTXIR(I,1)
            EFFR(I,2)=0.5D0*(ES5(STREAM5(I),10)**2.0D0-
     $                ES5(STREAM5(I),14)**2.0D0)/(DHS+0.5D0*
     $                (ES4(STREAM4(I),10)**2.0D0-
     $                 ES4(STREAM4(I),14)**2.0D0))
C Stage calculations
            CP=(ES3(STREAM3(I),23)+ES5(STREAM5(I),23))*0.5D0
            R=(ES3(STREAM3(I),22)+ES5(STREAM5(I),22))*0.5D0
            GAMMA=CP/(CP-R)
            TS=((ES3(STREAM3(I),20)+KELVIN)*(ES5(STREAM5(I),4)/
     $           ES3(STREAM3(I),4))**((GAMMA-1.0D0)/GAMMA))-KELVIN
            DHS=CP*(ES3(STREAM3(I),20)-TS)
            DH=CP*(ES3(STREAM3(I),20)-ES5(STREAM5(I),20))
            EFFSI(I,1)=(0.5D0*(ES3(STREAM3(I),1)+ES5(STREAM5(I),1))-
     $                  HUB)/BH
            EFFSI(I,2)=(DH+0.5D0*(ES3(STREAM3(I),5)**2.0D0-
     $                  ES5(STREAM5(I),5)**2.0D0))/
     $                 (DHS+0.5D0*ES3(STREAM3(I),5)**2.0D0)
            EFFST(I,1)=EFFSI(I,1)
            EFFST(I,2)=0.5D0*(ES5(STREAM5(I),10)**2.0D0-
     $                 ES5(STREAM5(I),14)**2.0D0+
     $                 ES4(STREAM4(I),5)**2.0D0)/(DHS+0.5D0*
     $                 (ES3(STREAM3(I),5)**2.0D0+
     $                  ES4(STREAM4(I),10)**2.0D0-
     $                  ES4(STREAM4(I),14)**2.0D0))
   49 CONTINUE
C Write efficiency data to output file
      WRITE(60,*) 'TITLE="EFFICIENCY"'
      WRITE(60,*) 'Variables="Immersion Ratio r^* ‘= ',
     $           '(r ‘- r_h_u_b) ~$ (r_t_i_p ‘- r_h_u_b)", ',
     $           '"‘h"'
      WRITE(60,47) STATOR
      DO 50 I=1,RADIAL+1
C Cleaning of efficiency write-out data
C Algorithm only applies with actual temperature measurements
C        IF(EFFS(I,2).GT.ONE) GO TO 50
         IF(EFFS(I,1).GE.PTRIM.AND.EFFS(I,1).LE.1.0D0-PTRIM) THEN
           IF(MOD(I,ISKIP).EQ.0) GO TO 82
           GO TO 50
         ENDIF
   82    WRITE(60,*) EFFS(I,1),EFFS(I,2)
   50 CONTINUE
      WRITE(60,47) ROTOR
      DO 51 I=1,RADIAL+1
C Cleaning of efficiency write-out data
C Algorithm only applies with actual temperature measurements
C        IF(EFFR(I,2).GT.ONE) GO TO 51
         IF(EFFR(I,1).GE.PTRIM.AND.EFFR(I,1).LE.1.0D0-PTRIM) THEN
           IF(MOD(I,ISKIP).EQ.0) GO TO 83
           GO TO 51
         ENDIF
   83    WRITE(60,*) EFFR(I,1),EFFR(I,2)
   51 CONTINUE
      WRITE(60,47) STAGE
      DO 63 I=1,RADIAL+1
C Cleaning of efficiency write-out data
C Algorithm only applies with actual temperature measurements
C        IF(EFFST(I,2).GT.ONE) GO TO 63
         IF(EFFST(I,1).GE.PTRIM.AND.EFFST(I,1).LE.1.0D0-PTRIM) THEN
           IF(MOD(I,ISKIP).EQ.0) GO TO 84
           GO TO 63
         ENDIF
   84    WRITE(60,*) EFFST(I,1),EFFST(I,2)
   63 CONTINUE
      WRITE(60,47) ISENTROPIC
      DO 52 I=1,RADIAL+1
C Cleaning of efficiency write-out data
C Algorithm only applies with actual temperature measurements
C        IF(EFFSI(I,2).GT.ONE) GO TO 52
         IF(EFFSI(I,1).GE.PTRIM.AND.EFFSI(I,1).LE.1.0D0-PTRIM) THEN
           IF(MOD(I,ISKIP).EQ.0) GO TO 85
           GO TO 52
         ENDIF
   85    WRITE(60,*) EFFSI(I,1),EFFSI(I,2)
   52 CONTINUE
C Calculate the averaged stator/rotor loss coefficients
      PTLS=(AS3(2)-AS4(2))/(AS3(2))
      PDLS=(AS3(2)-AS4(2))/(AS3(2)-AS4(4))
      PTLR=(AS4(3)-AS5(3))/(AS4(3))
      PDLR=(AS4(3)-AS5(3))/(AS4(3)-AS5(4))
C Calculate the averaged stator/rotor profile loss coefficients
      PPDLS=(PAS3(2)-PAS4(2))/(PAS3(2)-PAS4(4))
      PPDLR=(PAS4(3)-PAS5(3))/(PAS4(3)-PAS5(4))
C Calculate the averaged stator/rotor secondary loss coefficients
      SPDLS=PDLS-PPDLS
      SPDLR=PDLR-PPDLR
C Calculate the averaged stator efficiency
      CP=(AS3(23)+AS4(23))*0.5D0
      R=(AS3(22)+AS4(22))*0.5D0
      GAMMA=CP/(CP-R)
      TS=((AS3(20)+KELVIN)*(AS4(4)/AS3(4))**((GAMMA-1.0D0)/GAMMA))-
     $     KELVIN
      DHS=CP*(AS3(20)-TS)
      EFS=(0.5D0*AS4(5)**2.0D0)/(DHS+0.5D0*AS3(5)**2.0D0)
C Calculate the averaged rotor efficiency
      CP=(AS4(23)+AS5(23))*0.5D0
      R=(AS4(22)+AS5(22))*0.5D0
      GAMMA=CP/(CP-R)
      TS=((AS4(20)+KELVIN)*(AS5(4)/AS4(4))**((GAMMA-1.0D0)/GAMMA))-
     $   KELVIN
      DHS=CP*(AS4(20)-TS)
      EFR=0.5D0*(AS5(10)**2.0D0-AS5(14)**2.0D0)/
     $    (DHS+0.5D0*(AS4(10)**2.0D0-AS4(14)**2.0D0))
C Calculate the averaged stage efficiency
      CP=(AS3(23)+AS5(23))*0.5D0
      R=(AS3(22)+AS5(22))*0.5D0
      GAMMA=CP/(CP-R)
      TS=((AS3(20)+KELVIN)*(AS5(4)/
     $    AS3(4))**((GAMMA-1.0D0)/GAMMA))-KELVIN
      DHS=CP*(AS3(20)-TS)
      DH=CP*(AS3(20)-AS5(20))
      EFSI=(DH+0.5D0*(AS3(5)**2.0D0-AS5(5)**2.0D0))/
     $     (DHS+0.5D0*AS3(5)**2.0D0)
      EFST=0.5D0*(AS5(10)**2.0D0-AS5(14)**2.0D0+AS4(5)**2.0D0)/
     $     (DHS+0.5D0*(AS3(5)**2.0D0+AS4(10)**2.0D0-AS4(14)**2.0D0))
C Format specifiers for Tecplot headers
   47 FORMAT(1X,'ZONE T="',A,'"')
C Change the flow angles to conform to Schobeiri's notations
      AS3(15)=NINETY-AS3(15)
      AS4(15)=NINETY-AS4(15)
      AS5(15)=NINETY-AS5(15)
```

```
C Write output summary of respective station data
      WRITE(70,43)
   43 FORMAT(32X,'SUMMARY OF STATION ANALYSIS',/,32X,
     $       '===========================',//)
      WRITE(70,27)
   27 FORMAT(40X,'STATION 3',11X,'STATION 4',11X,'STATION 5',/,40X,
     $       '=========',11X,'=========',11X,'=========',/)
      WRITE(70,28) AS3(4),AS4(4),AS5(4)
   28 FORMAT(1X,'STATIC PRESSURE [PA]',18X,F10.1,10X,F10.1,10X,F10.1,/)
      WRITE(70,29) AS3(2),AS4(2),AS5(2)
   29 FORMAT(1X,'TOTAL PRESSURE [PA]',19X,F10.1,10X,F10.1,10X,F10.1,/)
      WRITE(70,30) AS3(3),AS4(3),AS5(3)
   30 FORMAT(1X,'RELATIVE TOTAL PRESSURE [PA]',10X,F10.1,10X,F10.1,10X,
     $       F10.1,/)
      WRITE(70,31) AS3(5),AS4(5),AS5(5)
   31 FORMAT(1X,'ABSOLUTE VELOCITY [M/SEC]',17X,F6.1,14X,F6.1,
     $       14X,F6.1,/)
      WRITE(70,32) AS3(10),AS4(10),AS5(10)
   32 FORMAT(1X,'RELATIVE VELOCITY [M/SEC]',17X,F6.1,14X,F6.1,
     $       14X,F6.1,/)
      WRITE(70,33) AS3(18),AS4(18),AS5(18)
   33 FORMAT(1X,'MACH NUMBER',32X,F5.3,15X,F5.3,15X,F5.3,/)
      WRITE(70,99) AS3(19),AS4(19),AS5(19)
   99 FORMAT(1X,'RELATIVE MACH NUMBER',23X,F5.3,15X,F5.3,15X,F5.3,/)
      WRITE(70,34) AS3(15),AS4(15),AS5(15)
   34 FORMAT(1X,'MERIDIONAL ANGLE [DEG]',19X,F7.2,13X,F7.2,13X,F7.2,/)
      WRITE(70,35) AS3(16),AS4(16),AS5(16)
   35 FORMAT(1X,'ALPHA [DEG]',30X,F7.2,13X,F7.2,13X,F7.2,/)
      WRITE(70,36) AS3(17),AS4(17),AS5(17)
   36 FORMAT(1X,'BETA [DEG]',31X,F7.2,13X,F7.2,13X,F7.2,/)
      WRITE(70,37) AS3(20),AS4(20),AS5(20)
   37 FORMAT(1X,'STATIC TEMPERATURE [C]',20X,F6.2,14X,F6.2,14X,F6.2,/)
      WRITE(70,38) AS3(21),AS4(21),AS5(21)
   38 FORMAT(1X,'DENSITY [KG/M**2]',26X,F5.3,15X,F5.3,15X,F5.3,/)
      WRITE(70,39) AS3(25),AS4(25),AS5(25)
   39 FORMAT(1X,'STATIC ENTHALPY [J/KG]',15X,F11.1,9X,F11.1,9X,F11.1,/)
      WRITE(70,40) AS3(26),AS4(26),AS5(26)
   40 FORMAT(1X,'TOTAL ENTHALPY [J/KG]',16X,F11.1,9X,F11.1,9X,F11.1,/)
      WRITE(70,41) AS3(27),AS4(27),AS5(27)
   41 FORMAT(1X,'RELATIVE TOTAL ENTHALPY [J/KG]',7X,F11.1,9X,F11.1,9X,
     $       F11.1,/)
      WRITE(70,42) MFLOW3,MFLOW4,MFLOW5
   42 FORMAT(1X,'MASS FLOW RATE [KG/SEC]',19X,F6.4,14X,F6.4,14X,F6.4,/)
      WRITE(70,56) RE3,RE4,RE5
   56 FORMAT(1X,'REYNOLDS NUMBER',23X,E10.3,10X,E10.3,10X,E10.3,/)
      WRITE(70,*) '=============================================',
     $            '============================================='
      WRITE(70,*)
      WRITE(70,57) PTLS,PDLS
   57 FORMAT(1X,'STATOR LOSS COEFFICIENT (TOTAL, DYNAMIC)',
     $       19X,F9.5,11X,F9.5,/)
      WRITE(70,74) PPDLS,SPDLS
   74 FORMAT(1X,'STATOR LOSS COEFFICIENT (PROFILE, SECONDARY)',
     $       15X,F9.5,11X,F9.5,/)
      WRITE(70,58) PTLR,PDLR
   58 FORMAT(1X,'ROTOR LOSS COEFFICIENT (TOTAL, DYNAMIC)',
     $       20X,F9.5,11X,F9.5,/)
      WRITE(70,75) PPDLR,SPDLR
   75 FORMAT(1X,'ROTOR LOSS COEFFICIENT (PROFILE, SECONDARY)',
     $       16X,F9.5,11X,F9.5,/)
      WRITE(70,*) '=============================================',
     $            '============================================='
      WRITE(70,*)
      WRITE(70,59) EFS
   59 FORMAT(1X,'STATOR EFFICIENCY',44X,F7.4,/)
      WRITE(70,60) EFR
   60 FORMAT(1X,'ROTOR EFFICIENCY',45X,F7.4,/)
```

```
      WRITE(70,62) EFST
   62 FORMAT(1X,'STAGE EFFICIENCY',45X,F7.4,/)
      WRITE(70,61) EFSI
   61 FORMAT(1X,'STAGE ISENTROPIC EFFICIENCY',34X,F7.4,/)
      PRINT*, 'PERFORMANCE ANALYSIS COMPLETE.'
      RETURN
      END
************************************************************************
* This subroutine expands arrays by interpolating their data with a    *
* combination of linear and spline derived points.                     *
* Input Files : N/A                                                    *
* Output Files: N/A                                                    *
* Experimental Procedure: Research Turbine Facility - Room 131         *
************************************************************************
* Hicham A Chibli - Texas A&M University - 12/13/09                    *
************************************************************************
      SUBROUTINE EXPAND(AIN,AOUT,NMAX,N,M,J)
      INTEGER NMAX,N,M,J,I,K,IST,IEN
      DOUBLE PRECISION AIN(NMAX,J),AOUT(M,J),NX(N),NY(N),SPACE,ST,EN
C Populate x-value base array
      DO 1 I=1,N
         NX(I)=AIN(I,1)
    1 CONTINUE
      SPACE=(NX(N)-NX(1))/DBLE(M-1)
      ST=(NX(2)-NX(1))/SPACE
      EN=(NX(N)-NX(N-1))/SPACE
      IST=IDINT(ST)
      IEN=IDINT(EN)
      IF(DBLE(IST).LT.ST) IST=IST+1
      IF(DBLE(IEN).LT.EN) IEN=IEN+1
C Populate x-value expanded array
      DO 2 I=1,M
         AOUT(I,1)=NX(1)+DBLE(I-1)*SPACE
    2 CONTINUE
C Populate y-arrays
      DO 7 K=2,J
C Populate y-value base arrays
         DO 3 I=1,N
            NY(I)=AIN(I,K)
    3    CONTINUE
C Populate y-value expanded arrays
         DO 4 I=1+IST,M-IEN
            CALL SPLINE(NX,NY,AOUT(I,1),AOUT(I,K),N,0,0.0D0,0.0D0)
    4    CONTINUE
         DO 5 I=1,IST
            AOUT(I,K)=NY(1)+(AOUT(I,1)-NX(1))/(NX(2)-NX(1))*
     $                (NY(2)-NY(1))
    5    CONTINUE
         DO 6 I=M-IEN+1,M
            AOUT(I,K)=NY(N-1)+(AOUT(I,1)-NX(N-1))/(NX(N)-NX(N-1))*
     $                (NY(N)-NY(N-1))
    6    CONTINUE
    7 CONTINUE
      RETURN
      END
************************************************************************
* This subroutine performs Lagrangian interpolation using the Aitken   *
* Method.                                                              *
* Input Files : N/A                                                    *
* Output Files: N/A                                                    *
* Experimental Procedure: Research Turbine Facility - Room 131         *
************************************************************************
* Hicham A Chibli - Texas A&M University - 10/30/09                    *
************************************************************************
      SUBROUTINE AITKEN(N,XI,FI,X,F)
      INTEGER N,NMAX,I,J
      PARAMETER(NMAX=1000)
      DOUBLE PRECISION XI(NMAX),FI(NMAX),X,F,FT(NMAX),F1,F2,X1,X2,CHECK
C Check if given dimension is too large
      IF(N.GT.NMAX) THEN
        PRINT*, 'INTERPOLATION ERROR: TOO MANY POINTS USED.'
        STOP
      ENDIF
C Check for errors in input data
      DO 2 I=1,N-1
         CHECK=XI(I)
         DO 1 J=I+1,N
            IF(CHECK.EQ.XI(J)) THEN
              PRINT*, 'INTERPOLATION ERROR: INPUT POINTS NOT UNIQUE.'
              STOP
            ENDIF
    1    CONTINUE
    2 CONTINUE
C Populate arrays
      DO 3 I=1,N
         FT(I)=FI(I)
    3 CONTINUE
      DO 5 I=1,N-1
         DO 4 J=1,N-I
            X1=XI(J)
            X2=XI(J+I)
            F1=FT(J)
            F2=FT(J+1)
            FT(J)=(X-X1)/(X2-X1)*F2+(X-X2)/(X1-X2)*F1
    4    CONTINUE
    5 CONTINUE
C Interpolant computed
      F=FT(1)
      RETURN
      END
************************************************************************
* This subroutine reads formatted input data files and performs        *
* simple arithmetic averaging of variables in the circumferential      *
* direction.                                                           *
* Input Files : hubs3.int, hubs4.int, hubs5.int,                       *
*               fulls3.int, fulls4.int, fulls5.int                     *
* Output Files: averages3.int, averages4.int, averages5.int,           *
*               irs3.int, irs4.int, irs5.int                           *
* Experimental Procedure: Research Turbine Facility - TPFL             *
************************************************************************
* Hicham A Chibli - Texas A&M University - 11/30/09                    *
************************************************************************
      SUBROUTINE SAVER(N,SNUM)
      INTEGER N,M,K,J,I,L,P,Q,IMAX
C The maximum number of points measured radially or circumferentially
      PARAMETER(IMAX=1000)
      DOUBLE PRECISION DATUM(N+IMAX,29),RADIAL(IMAX),SUMS(27),TEMP(29)
      DOUBLE PRECISION RADIUS,APTCR,APRCR,APSCR,VCR,VCCR,VRCR,
     $                 VZCR,VMCR,WCR,WCCR,WRCR,WZCR,UCR,
     $                 APSI,APHI,RPSI,RPHI,MACH,MACHR,TSCR,MU,
     $                 RHOCR,HUB,TIP,IRATIO,
     $                 R,CP,S,ENS,ENT,ENTR
      CHARACTER SNUM*1,TITLE*100
C Call subroutine NME to generate input file name
      CALL NME('hubs',SNUM,TITLE,3)
C Call subroutine CLEAN to clean file name
      CALL CLEAN(TITLE,M)
C Open file for input
      OPEN(UNIT=10,FILE=TITLE(1:M),STATUS='OLD')
C Call subroutine NME to generate input file name
      CALL NME('fulls',SNUM,TITLE,3)
C Call subroutine CLEAN to clean file name
      CALL CLEAN(TITLE,M)
C Open file for input
```

```
      OPEN(UNIT=20,FILE=TITLE(1:M),STATUS='OLD')
C Call subroutine NME to create output file name
      CALL NME('irs',SNUM,TITLE,3)
C Call subroutine CLEAN to clean file name
      CALL CLEAN(TITLE,M)
C Open file for output
      OPEN(UNIT=30,FILE=TITLE(1:M),STATUS='UNKNOWN')
C Call subroutine NME to create output file name
      CALL NME('averages',SNUM,TITLE,3)
C Call subroutine CLEAN to clean file name
      CALL CLEAN(TITLE,M)
C Open file for output
      OPEN(UNIT=40,FILE=TITLE(1:M),STATUS='UNKNOWN')
C Read hub data array
      Q=0
    8 READ(10,*,END=10) TEMP
      Q=Q+1
      DO 9 J=1,29
            DATUM(Q,J)=TEMP(J)
    9 CONTINUE
      GO TO 8
C Read data array
   10 DO 12 I=1+Q,N+Q
         READ(20,*) TEMP
         DO 11 J=1,29
            DATUM(I,J)=TEMP(J)
   11    CONTINUE
   12 CONTINUE
      HUB=DATUM(1,2)
      TIP=DATUM(N+Q,2)
C Initialize radial array
      DO 1 I=1,IMAX
         RADIAL(I)=-1
    1 CONTINUE
C Sort the data array
      DO 2 I=1,N+Q
C Initialize variables
         K=0
         DO 3 J=1,27
            SUMS(J)=0.0D0
    3    CONTINUE
         RADIUS=DATUM(I,2)
         IRATIO=(RADIUS-HUB)/(TIP-HUB)
C Check for redundancy
         DO 4 L=1,IMAX
            IF(RADIUS.EQ.RADIAL(L)) GO TO 2
            IF(RADIAL(L).EQ.-1) THEN
              RADIAL(L)=RADIUS
              GO TO 5
            ENDIF
    4    CONTINUE
C Perform circumferential simple averaging of data
    5    DO 6 J=1,N+Q
            IF(DATUM(J,2).EQ.RADIUS) THEN
              K=K+1
              DO 7 P=1,27
                 SUMS(P)=SUMS(P)+DATUM(J,P+2)
    7         CONTINUE
            ENDIF
    6    CONTINUE
         APTCR=SUMS(1)/DBLE(K)
         APRCR=SUMS(2)/DBLE(K)
         APSCR=SUMS(3)/DBLE(K)
         VCR=SUMS(4)/DBLE(K)
         VCCR=SUMS(5)/DBLE(K)
         VRCR=SUMS(6)/DBLE(K)
         VZCR=SUMS(7)/DBLE(K)
         VMCR=SUMS(8)/DBLE(K)
         WCR=SUMS(9)/DBLE(K)
         WCCR=SUMS(10)/DBLE(K)
         WRCR=SUMS(11)/DBLE(K)
         WZCR=SUMS(12)/DBLE(K)
         UCR=SUMS(13)/DBLE(K)
         APSI=SUMS(14)/DBLE(K)
         APHI=SUMS(15)/DBLE(K)
         RPHI=SUMS(16)/DBLE(K)
         MACH=SUMS(17)/DBLE(K)
         MACHR=SUMS(18)/DBLE(K)
         TSCR=SUMS(19)/DBLE(K)
         RHOCR=SUMS(20)/DBLE(K)
         R=SUMS(21)/DBLE(K)
         CP=SUMS(22)/DBLE(K)
         S=SUMS(23)/DBLE(K)
         ENS=SUMS(24)/DBLE(K)
         ENT=SUMS(25)/DBLE(K)
         ENTR=SUMS(26)/DBLE(K)
         MU=SUMS(27)/DBLE(K)
C Write data to output file
         WRITE(30,*) IRATIO
C Write data to output file
         WRITE(40,*) RADIUS,APTCR,APRCR,APSCR,VCR,VCCR,VRCR,
     $               VZCR,VMCR,WCR,WCCR,WRCR,WZCR,UCR,
     $               APSI,APHI,RPHI,MACH,MACHR,TSCR,RHOCR,
     $               R,CP,S,ENS,ENT,ENTR,MU
    2 CONTINUE
      PRINT*, 'CIRCUMFERENTIAL AVERAGING FOR STATION NUMBER ',SNUM,
     $        ' COMPLETE.'
      RETURN
      END
************************************************************************
* This subroutine calls the averaging subroutines to perform required  *
* circumferential averaging of data.                                   *
* Input Files : totals3.int, totals4.int, totals5.int                  *
* Output Files: N/A                                                    *
* Experimental Procedure: Research Turbine Facility - TPFL             *
************************************************************************
* Hicham A Chibli - Texas A&M University - 11/30/09                    *
************************************************************************
      SUBROUTINE AVER
      INTEGER J3,J4,J5
C Open files for input
      OPEN(UNIT=10,FILE='./tmp/totals3.int',STATUS='OLD')
      OPEN(UNIT=20,FILE='./tmp/totals4.int',STATUS='OLD')
      OPEN(UNIT=30,FILE='./tmp/totals5.int',STATUS='OLD')
C Read respective total number of points to average
      READ(10,*) J3
      READ(20,*) J4
      READ(30,*) J5
C Place calls to circumferential averaging subroutine
      CALL SAVER(J3,'3')
      CALL SAVER(J4,'4')
      CALL SAVER(J5,'5')
      RETURN
      END
************************************************************************
* This subroutine returns the non-blank component of a string.         *
* Input Files : N/A                                                    *
* Output Files: N/A                                                    *
* Experimental Procedure: Research Turbine Facility - TPFL             *
************************************************************************
* Hicham A Chibli - Texas A&M University - 12/01/09                    *
************************************************************************
      SUBROUTINE CLEAN(TITLE,N)
C Define variables and constants
```

```
      INTEGER N
      CHARACTER*(*) TITLE
C Find the blank part of file name
      N=INDEX(TITLE,' ')
C Get the size of the non-blank part of file name
      N=N-1
      RETURN
      END
***********************************************************************
* This subroutine calculates the density of humid air.               *
* Input Files : N/A                                                   *
* Output Files: N/A                                                   *
* Experimental Procedure: Research Turbine Facility - Room 131        *
***********************************************************************
* Hicham A Chibli - Texas A&M University - 10/30/09                   *
***********************************************************************
      SUBROUTINE DEN(ATS,APS,RH,RHO)
C Define variables and constants to use
      DOUBLE PRECISION RA,RV,KELVIN,SVP,VPP,DPP,RH,APS,ATS,RHO
      PARAMETER(RA=286.9D0,RV=461.495D0,KELVIN=273.15D0)
C Calculate saturation vapor pressure
      SVP=6.1078D0*100.0D0*10.0D0**((7.5D0*(ATS-KELVIN))/
     $    (237.3D0+ATS-KELVIN))
C Calculate vapor partial pressure
      VPP=SVP*RH/100.0D0
C Calculate dry air partial pressure
      DPP=APS-VPP
C Calculate flow density
      RHO=DPP/(RA*ATS)+VPP/(RV*ATS)
      RETURN
      END
***********************************************************************
* This subroutine calls the data analysis subroutines for each of the *
* three stations to be analyzed.                                      *
* Input Files : N/A                                                   *
* Output Files: N/A                                                   *
* Experimental Procedure: Research Turbine Facility - TPFL            *
***********************************************************************
* Hicham A Chibli - Texas A&M University - 11/30/09                   *
***********************************************************************
      SUBROUTINE FHP
      INTEGER N
C Prompt user for input
      PRINT*, 'PRESS 3 TO ANALYZE STATION 3'
      PRINT*, 'PRESS 4 TO ANALYZE STATION 4'
      PRINT*, 'PRESS 5 TO ANALYZE STATION 5'
      PRINT*, 'PRESS 9 TO CORRECT STAGE TEMPERATURES'
      READ*, N
C Place calls to simple averaging subroutine
      IF(N.EQ.3) THEN
        CALL STN(1,'3')
      ELSEIF(N.EQ.4) THEN
            CALL STN(2,'4')
      ELSEIF(N.EQ.5) THEN
            CALL STN(3,'5')
      ELSEIF(N.EQ.9) THEN
            CALL TEMP
      ELSE
          PRINT*, 'INVALID CHOICE. PROGRAM WILL CLOSE.'
      ENDIF
      RETURN
      END
***********************************************************************
* This subroutine calculates the humidity ratio of humid air as well  *
* as the weight ratio of water vapor to humid air mixture.            *
* Input Files : N/A                                                   *
* Output Files: N/A                                                   *
* Experimental Procedure: Research Turbine Facility - Room 131         *
************************************************************************
* Hicham A Chibli - Texas A&M University - 01/13/10                    *
************************************************************************
      SUBROUTINE HRATIO(ATS,APS,RH,HR,HRM)
C Define variables  and constants to use
      DOUBLE PRECISION RA,RV,KELVIN,SVP,VPP,DPP,RH,APS,ATS,RHO,HR,HRM
      PARAMETER(RA=286.9D0,RV=461.495D0,KELVIN=273.15D0)
C Calculate saturation vapor pressure
      SVP=6.1078D0*100.0D0*10.0D0**((7.5D0*(ATS-KELVIN))/
     $    (237.3D0+ATS-KELVIN))
C Calculate vapor partial pressure
      VPP=SVP*RH/100.0D0
C Calculate dry air partial pressure
      DPP=APS-VPP
C Calculate humidity ratio
      HR=0.62198D0*VPP/DPP
      HRM=HR/(1.0D0+HR)
      RETURN
      END
************************************************************************
* This subroutine reads the raw humidity data and generates a smooth   *
* profile to be used to estimate the humidity corresponding to each    *
* data point in the experimental grid.                                 *
* Input Files : parameter.txt, number.txt, hxx.txt, htotal.int         *
* Output Files: humidity.int                                           *
* Experimental Procedure: Research Turbine Facility - Room 131         *
************************************************************************
* Hicham A Chibli - Texas A&M University - 10/29/09                    *
************************************************************************
      SUBROUTINE HUMID
C Define variables and constants to use
      DOUBLE PRECISION EXACT,BASE
      INTEGER L,M,N,HOUR,MINUTE,RHCNT,I,IMAX,RHMAX,LUNIT,SLAYER
C Maximum number of measured points and sublayers
      PARAMETER(IMAX=1000000,RHMAX=1000,SLAYER=5)
      DOUBLE PRECISION RH(RHMAX),ERH(IMAX),HT(RHMAX),TME(IMAX)
      CHARACTER TITLE*12,SNUM*1
C Open files for i/o
      OPEN(UNIT=60,FILE='./tmp/htotal.int',STATUS='OLD')
      OPEN(UNIT=70,FILE='./inp/number.txt',STATUS='OLD')
      OPEN(UNIT=80,FILE='./inp/parameter.txt',STATUS='OLD')
      OPEN(UNIT=90,FILE='./tmp/humidity.int',STATUS='UNKNOWN')
C Read number of humidity sublayers to analyze
      READ(80,*)
      READ(80,*) M
C Check if maximum sublayer number is bounded by (5)
      IF(M.GT.SLAYER) THEN
        PRINT*, 'MAXIMUM NUMBER OF SUBLAYERS ALLOWED EXCEEDED.'
        STOP
      ENDIF
      DO 5 L=1,M
      RHCNT=1
C Read input file number
         READ(70,*) SNUM
C Read number of points to interpolate
         READ(60,*) N
C Call NME subroutine to generate name string
         CALL NME('h',SNUM,TITLE,1)
C Open file for input
         LUNIT=10*L
         OPEN(UNIT=LUNIT,FILE=TITLE,STATUS='OLD')
C Read raw humidity data
    1    READ(LUNIT,2,END=3) HOUR,MINUTE,RH(RHCNT)
    2    FORMAT(I2,I2,1X,F4.1)
         EXACT=DBLE(HOUR)*60.0D0+DBLE(MINUTE)
         IF(RHCNT.EQ.1) BASE=EXACT
         HT(RHCNT)=EXACT-BASE
         RHCNT=RHCNT+1
         GO TO 1
    3    PRINT*, 'DATA READING FOR FILE NUMBER ',SNUM,' COMPLETE.'
C Correct array size
         RHCNT=RHCNT-1
C Populate location arrays
         DO 4 I=1,N
            TME(I)=DBLE(I-1)*HT(RHCNT)/DBLE(N-1)
            CALL AITKEN(RHCNT,HT,RH,TME(I),ERH(I))
C Write data to output file
            WRITE(90,*) ERH(I)
    4    CONTINUE
    5 CONTINUE
      PRINT*, 'HUMIDITY DATA ANALYSIS COMPLETE.'
      RETURN
      END
************************************************************************
* This subroutine performs Simpson's integration.                      *
* Input Files : N/A                                                    *
* Output Files: N/A                                                    *
* Experimental Procedure: Research Turbine Facility - Room 131         *
************************************************************************
* Hicham A Chibli - Texas A&M University - 12/11/09                    *
************************************************************************
      SUBROUTINE INTG(NX,NY,K,M,N,A)
      INTEGER K,M,N,I,J
      DOUBLE PRECISION NX(K),NY(K),A
      J=N-M+1
C Check if number of points is odd
      IF(MOD(J,2).EQ.0) THEN
        PRINT*, 'ODD NUMBER OF POINTS NEEDED FOR INTEGRATION.'
        STOP
      ENDIF
C Initialize integral
      A=0.0D0
C Populate arrays
      DO 1 I=M,N-2
         IF(MOD(I,2).EQ.0) GO TO 1
         A=A+(1.0D0/3.0D0)*(NX(I+1)-NX(I))*(NY(I)+4.0D0*NY(I+1)+NY(I+2))
    1 CONTINUE
      RETURN
      END
************************************************************************
* This subroutine performs cubic spline interpolation.                 *
* INPUT FILES : N/A                                                    *
* OUTPUT FILES: N/A                                                    *
* EXPERIMENTAL PROCEDURE: RESEARCH TURBINE FACILITY - ROOM 131         *
************************************************************************
* HICHAM A CHIBLI - TEXAS A&M UNIVERSITY - 12/18/09                    *
************************************************************************
      SUBROUTINE SPLINE(X,Y,XINT,YINT,N,M,SLOPE1,SLOPEN)
      INTEGER M,N,NMAX,I,K,KHI,KLO
      DOUBLE PRECISION ULIMIT,LLIMIT,Y1P1,Y1PN,P,QN,SIG,UN,
     $                 A,B,H,XINT,YINT,SLOPE1,SLOPEN
      PARAMETER(NMAX=10000,ULIMIT=0.99D+30,LLIMIT=0.01D-20)
      DOUBLE PRECISION X(N),Y(N),Y2(N),U(NMAX)
C Check whether the natural or fixed slope option is given
      IF(M.EQ.0) THEN
C Calculate the first derivative at the first point of the sequence
        Y1P1=(Y(2)-Y(1))/(X(2)-X(1))
C Calculate the first derivative at the last point of the sequence
        Y1PN=(Y(N-1)-Y(N))/(X(N-1)-X(N))
      ELSEIF(M.EQ.1) THEN
C Use specified values
            Y1P1=SLOPE1
            Y1PN=SLOPEN
      ELSE
          PRINT*, 'ERROR: INVALID INPUT IN SPLINE.'
          STOP
      ENDIF
C Check if the first derivative is equal to infinity
      IF(Y1P1.GT.ULIMIT) THEN
        Y2(1)=0.0D0
        U(1)=0.0D0
      ELSE
          Y2(1)=-0.5D0
          U(1)=(3.0D0/(X(2)-X(1)))*((Y(2)-Y(1))/(X(2)-X(1))-Y1P1)
      ENDIF
      DO 1 I=2,N-1
         SIG=(X(I)-X(I-1))/(X(I+1)-X(I-1))
         P=SIG*Y2(I-1)+2.0D0
         Y2(I)=(SIG-1.0D0)/P
         U(I)=(6.0D0*((Y(I+1)-Y(I))/(X(I+1)-X(I))-(Y(I)-Y(I-1))/
     $        (X(I)-X(I-1)))/(X(I+1)-X(I-1))-SIG*U(I-1))/P
    1 CONTINUE
C Check if the first derivative is equal to infinity
      IF(Y1PN.GT.ULIMIT) THEN
        QN=0.0D0
        UN=0.0D0
      ELSE
          QN=0.5D0
          UN=(3.0D0/(X(N)-X(N-1)))*(Y1PN-(Y(N)-Y(N-1))/(X(N)-X(N-1)))
      ENDIF
      Y2(N)=(UN-QN*U(N-1))/(QN*Y2(N-1)+1.0D0)
      DO 2 K=N-1,1,-1
         Y2(K)=Y2(K)*Y2(K+1)+U(K)
    2 CONTINUE
C Bracket the x-interpolation point
      KLO=1
      KHI=N
    3 IF((KHI-KLO).GT.1) THEN
         K=(KHI+KLO)/2
         IF(X(K).GT.XINT)THEN
           KHI=K
         ELSE
             KLO=K
         ENDIF
         GO TO 3
      ENDIF
      H=X(KHI)-X(KLO)
C Check for non unique input x-array points
      IF(DABS(H).LT.LLIMIT) THEN
        PRINT*, 'NON-UNIQUE X-ARRAY POINT: INTERPOLATION TERMINATED.'
        STOP
      ENDIF
      A=(X(KHI)-XINT)/H
      B=(XINT-X(KLO))/H
      YINT=A*Y(KLO)+B*Y(KHI)+(((A**3.0D0)-A)*
     $     Y2(KLO)+((B**3.0D0)-B)*Y2(KHI))*
     $     (H**2.0D0)/6.0D0
      RETURN
      END
************************************************************************
* This subroutine generates the file name to be subsequently read and  *
* analyzed.                                                            *
* Input Files : N/A                                                    *
* Output Files: N/A                                                    *
* Experimental Procedure: Research Turbine Facility - TPFL             *
************************************************************************
* Hicham A Chibli - Texas A&M University - 11/19/09                    *
************************************************************************
      SUBROUTINE NME(ADD,SNUM,TITLE,N)
C Define variables and constants
```

```
      CHARACTER SNUM*1,EXT*4,DIR*6
      CHARACTER*(*) TITLE,ADD
      INTEGER N
C Generate input file name
      IF(N.EQ.1) THEN
        EXT='.txt'
        DIR='./inp/'
      ENDIF
      IF(N.EQ.2) THEN
        EXT='.dat'
        DIR='./out/'
      ENDIF
      IF(N.EQ.3) THEN
        EXT='.int'
        DIR='./tmp/'
      ENDIF
      TITLE=DIR//ADD//SNUM//EXT
      RETURN
      END
************************************************************************
* This subroutine generates the formatted line plot file to Tecplot.   *
* Input Files : averages3.int, averages4.int, averages5.int,           *
*               fulls3.int, fulls4.int, fulls5.int, irs3.int,          *
*               irs4.int, irs5.int, xys3.int, xys4.int, xys5.int       *
* Output Files: lplot.dat, pt.dat, ptr.dat, ps.dat,                    *
*               mach.dat, machr.dat,                                   *
*               v.dat, vz.dat, vr.dat, vtheta.dat, vm.dat,             *
*               w.dat, wtheta.dat,                                     *
*               alpha.dat, beta.dat, gamma.dat,                        *
*               vector.dat, wector.dat                                 *
* Experimental Procedure: Research Turbine Facility - TPFL             *
************************************************************************
* Hicham A Chibli - Texas A&M University - 03/06/10                    *
************************************************************************
      SUBROUTINE PLOT
      INTEGER N,M,K,J,NUM1,NUM2,NUM3,NUM4
      CHARACTER TEST*10
      DOUBLE PRECISION TEMP1(28),TEMP2(29)
      DOUBLE PRECISION IR,X,Y,NINETY,CONVERT
      PARAMETER(NINETY=90.0D0,CONVERT=1000.0D0)
C Open files for input/output
      OPEN(UNIT=10,FILE='./tmp/averages3.int',STATUS='OLD')
      OPEN(UNIT=20,FILE='./tmp/averages4.int',STATUS='OLD')
      OPEN(UNIT=30,FILE='./tmp/averages5.int',STATUS='OLD')
      OPEN(UNIT=40,FILE='./tmp/irs3.int',STATUS='OLD')
      OPEN(UNIT=60,FILE='./tmp/irs4.int',STATUS='OLD')
      OPEN(UNIT=80,FILE='./tmp/irs5.int',STATUS='OLD')
      OPEN(UNIT=50,FILE='./tmp/xys3.int',STATUS='OLD')
      OPEN(UNIT=70,FILE='./tmp/xys4.int',STATUS='OLD')
      OPEN(UNIT=90,FILE='./tmp/xys5.int',STATUS='OLD')
      OPEN(UNIT=100,FILE='./tmp/fulls3.int',STATUS='OLD')
      OPEN(UNIT=140,FILE='./tmp/fulls4.int',STATUS='OLD')
      OPEN(UNIT=180,FILE='./tmp/fulls5.int',STATUS='OLD')
      OPEN(UNIT=200,FILE='./out/lplot.dat',STATUS='UNKNOWN')
      OPEN(UNIT=210,FILE='./out/pt.dat',STATUS='UNKNOWN')
      OPEN(UNIT=220,FILE='./out/ptr.dat',STATUS='UNKNOWN')
      OPEN(UNIT=230,FILE='./out/ps.dat',STATUS='UNKNOWN')
      OPEN(UNIT=240,FILE='./out/mach.dat',STATUS='UNKNOWN')
      OPEN(UNIT=250,FILE='./out/machr.dat',STATUS='UNKNOWN')
      OPEN(UNIT=260,FILE='./out/v.dat',STATUS='UNKNOWN')
      OPEN(UNIT=270,FILE='./out/vz.dat',STATUS='UNKNOWN')
      OPEN(UNIT=350,FILE='./out/vr.dat',STATUS='UNKNOWN')
      OPEN(UNIT=360,FILE='./out/vtheta.dat',STATUS='UNKNOWN')
      OPEN(UNIT=280,FILE='./out/vm.dat',STATUS='UNKNOWN')
      OPEN(UNIT=290,FILE='./out/w.dat',STATUS='UNKNOWN')
      OPEN(UNIT=370,FILE='./out/wtheta.dat',STATUS='UNKNOWN')
      OPEN(UNIT=300,FILE='./out/alpha.dat',STATUS='UNKNOWN')
      OPEN(UNIT=310,FILE='./out/beta.dat',STATUS='UNKNOWN')
      OPEN(UNIT=320,FILE='./out/gamma.dat',STATUS='UNKNOWN')
      OPEN(UNIT=330,FILE='./out/vector.dat',STATUS='UNKNOWN')
      OPEN(UNIT=340,FILE='./out/wector.dat',STATUS='UNKNOWN')
C Promplt user to enter test date and test rpm information
      PRINT*, 'ENTER TEST RPM (XXXX)'
      READ(*,6) N
      PRINT*, 'ENTER TEST DATE (MM.DD.YYYY)'
      READ(*,7) TEST
    6 FORMAT(I4)
    7 FORMAT(A10)
C Write Tecplot headers
      WRITE(200,*) 'TITLE="INTERSTAGE FLOW DATA"'
      WRITE(200,*) 'VARIABLES="Immersion Ratio r^* `= ',
     $             '(r `- r_h_u_b) ~$ (r_t_i_p `- r_h_u_b)",',
     $             '"Total Pressure  [kPa]",',
     $             '"Relative Total Pressure  [kPa]",',
     $             '"Static Pressure  [kPa]",',
     $             '"Absolute Velocity  [m/sec]",',
     $             '"Absolute Circumferential Velocity  [m/sec]",',
     $             '"Absolute Radial Velocity  [m/sec]",',
     $             '"Absolute Axial Velocity  [m/sec]",',
     $             '"Absolute Meridional Velocity  [m/sec]",',
     $             '"Relative Velocity  [m/sec]",',
     $             '"Relative Circumferential Velocity  [m/sec]",',
     $             '"Relative Radial Velocity  [m/sec]",',
     $             '"Relative Axial Velocity  [m/sec]",',
     $             '"Rotational Velocity  [m/sec]",',
     $             '"`g  [deg]","`a  [deg]","`b  [deg]",',
     $             '"Mach Number","Relative Mach Number",',
     $             '"Static Temperature  [^oC]","Density  [kg/m^3]",',
     $             '"Specific Gas Constant - R  [kj/kg/K]",',
     $             '"Specific Heat Capacity - C_p  [kj/kg/K]",',
     $             '"Entropy  [kj/kg/K]","Enthalpy  [kj/kg]",',
     $             '"Total Enthalpy  [kj/kg]",',
     $             '"Relative Total Enthalpy  [kj/kg]",',
     $             '"Dynamic Viscosity - `m  [kg/sec/m]"'
      WRITE(210,*) 'TITLE="TOTAL PRESSURE CONTOUR"'
      WRITE(210,*) 'VARIABLES="X [mm]","Y [mm]","P_t  [kPa]"'
      WRITE(220,*) 'TITLE="RELATIVE TOTAL PRESSURE CONTOUR"'
      WRITE(220,*) 'VARIABLES="X [mm]","Y [mm]","P_t_r  [kPa]"'
      WRITE(230,*) 'TITLE="STATIC PRESSURE CONTOUR"'
      WRITE(230,*) 'VARIABLES="X [mm]","Y [mm]","P_s  [kPa]"'
      WRITE(240,*) 'TITLE="MACH NUMBER CONTOUR"'
      WRITE(240,*) 'VARIABLES="X [mm]","Y [mm]","M"'
      WRITE(250,*) 'TITLE="RELATIVE MACH NUMBER CONTOUR"'
      WRITE(250,*) 'VARIABLES="X [mm]","Y [mm]","M_r"'
      WRITE(260,*) 'TITLE="ABSOLUTE VELOCITY CONTOUR"'
      WRITE(260,*) 'VARIABLES="X [mm]","Y [mm]","V  [m/sec]"'
      WRITE(270,*) 'TITLE="ABSOLUTE AXIAL VELOCITY CONTOUR"'
      WRITE(270,*) 'VARIABLES="X [mm]","Y [mm]","V_z  [m/sec]"'
      WRITE(280,*) 'TITLE="ABSOLUTE MERIDIONAL VELOCITY CONTOUR"'
      WRITE(280,*) 'VARIABLES="X [mm]","Y [mm]","V_m  [m/sec]"'
      WRITE(350,*) 'TITLE="ABSOLUTE RADIAL VELOCITY CONTOUR"'
      WRITE(350,*) 'VARIABLES="X [mm]","Y [mm]","V_r  [m/sec]"'
      WRITE(360,*) 'TITLE="ABSOLUTE TANGENTIAL VELOCITY CONTOUR"'
      WRITE(360,*) 'VARIABLES="X [mm]","Y [mm]","V_`q  [m/sec]"'
      WRITE(290,*) 'TITLE="RELATIVE VELOCITY CONTOUR"'
      WRITE(290,*) 'VARIABLES="X [mm]","Y [mm]","W  [m/sec]"'
      WRITE(370,*) 'TITLE="RELATIVE TANGENTIAL VELOCITY CONTOUR"'
      WRITE(370,*) 'VARIABLES="X [mm]","Y [mm]","W_`q  [m/sec]"'
      WRITE(300,*) 'TITLE="ALPHA CONTOUR"'
      WRITE(300,*) 'VARIABLES="X [mm]","Y [mm]","`a  [deg]"'
      WRITE(310,*) 'TITLE="BETA CONTOUR"'
      WRITE(310,*) 'VARIABLES="X [mm]","Y [mm]","`b  [deg]"'
      WRITE(320,*) 'TITLE="GAMMA CONTOUR"'
      WRITE(320,*) 'VARIABLES="X [mm]","Y [mm]","`g  [deg]"'
      WRITE(330,*) 'TITLE="VELOCITY VECTORS"'
      WRITE(330,*) 'VARIABLES="X [mm]","Y [mm]","u  [m/sec]"',
     $             ',"v  [m/sec]"'
      WRITE(340,*) 'TITLE="VELOCITY VECTORS"'
      WRITE(340,*) 'VARIABLES="X [mm]","Y [mm]","u  [m/sec]"',
     $             ',"v  [m/sec]"'
C Populate line plot output file
      DO 1 M=1,3
         NUM1=M*10
         NUM2=(M+1)*20
         WRITE(200,9) M+2
C Find the number of records in averaged data input files
         K=0
         J=0
    2    READ(NUM2,*,END=3)
         K=K+1
         GO TO 2
    3    REWIND(UNIT=NUM2)
C Read data from input files
    4    READ(NUM1,*,END=1) TEMP1
         READ(NUM2,*) IR
         J=J+1
C Replace the radial location by dimensionless immersion ratio
         TEMP1(1)=IR
C Change the flow angles to conform to Schobeiri's notations
         TEMP1(15)=NINETY-TEMP1(15)
C Change units to kPa,kJ/kg,kj/kg/K
         TEMP1(2)=TEMP1(2)/CONVERT
         TEMP1(3)=TEMP1(3)/CONVERT
         TEMP1(4)=TEMP1(4)/CONVERT
         TEMP1(22)=TEMP1(22)/CONVERT
         TEMP1(23)=TEMP1(23)/CONVERT
         TEMP1(24)=TEMP1(24)/CONVERT
         TEMP1(25)=TEMP1(25)/CONVERT
         TEMP1(26)=TEMP1(26)/CONVERT
         TEMP1(27)=TEMP1(27)/CONVERT
C Write data to output file
         IF(J.EQ.1.OR.J.EQ.K) GO TO 4
         WRITE(200,*) TEMP1
         GO TO 4
    1 CONTINUE
C Populate contour plot output files
      DO 5 M=1,3
         NUM3=(2*M+3)*10
         NUM4=(2*M+3)*20
         WRITE(210,9) M+2
         WRITE(220,9) M+2
         WRITE(230,9) M+2
         WRITE(240,9) M+2
         WRITE(250,9) M+2
         WRITE(260,9) M+2
         WRITE(270,9) M+2
         WRITE(280,9) M+2
         WRITE(290,9) M+2
         WRITE(300,9) M+2
         WRITE(310,9) M+2
         WRITE(320,9) M+2
         WRITE(330,9) M+2
         WRITE(340,9) M+2
         WRITE(350,9) M+2
         WRITE(360,9) M+2
         WRITE(370,9) M+2
C Read data from input files
    8    READ(NUM3,*,END=5) X,Y
         READ(NUM4,*) TEMP2
         J=J+1
C Change the flow angles to conform to Schobeiri's notations
```

```
         TEMP2(16)=NINETY-TEMP2(16)
C Write data to output files
         WRITE(210,*) X,Y,TEMP2(3)/CONVERT
         WRITE(220,*) X,Y,TEMP2(4)/CONVERT
         WRITE(230,*) X,Y,TEMP2(5)/CONVERT
         WRITE(240,*) X,Y,TEMP2(19)
         WRITE(250,*) X,Y,TEMP2(20)
         WRITE(260,*) X,Y,TEMP2(6)
         WRITE(270,*) X,Y,TEMP2(9)
         WRITE(280,*) X,Y,TEMP2(10)
         WRITE(290,*) X,Y,TEMP2(11)
         WRITE(300,*) X,Y,TEMP2(17)
         WRITE(310,*) X,Y,TEMP2(18)
         WRITE(320,*) X,Y,TEMP2(16)
         WRITE(350,*) X,Y,TEMP2(8)
         WRITE(360,*) X,Y,TEMP2(7)
         WRITE(370,*) X,Y,TEMP2(12)
         WRITE(330,*) X,Y,
     $                -1.0D0*TEMP2(8)*DSIN(TEMP2(1))-
     $                TEMP2(7)*DCOS(TEMP2(1)),
     $                TEMP2(8)*DCOS(TEMP2(1))-
     $                TEMP2(7)*DSIN(TEMP2(1))
         WRITE(340,*) X,Y,
     $                -1.0D0*TEMP2(13)*DSIN(TEMP2(1))-
     $                TEMP2(12)*DCOS(TEMP2(1)),
     $                TEMP2(13)*DCOS(TEMP2(1))-
     $                TEMP2(12)*DSIN(TEMP2(1))
         GO TO 8
    5 CONTINUE
C Write rpm/date text box for data sets
      WRITE(200,10) N,TEST
      WRITE(210,10) N,TEST
      WRITE(220,10) N,TEST
      WRITE(230,10) N,TEST
      WRITE(240,10) N,TEST
      WRITE(250,10) N,TEST
      WRITE(260,10) N,TEST
      WRITE(270,10) N,TEST
      WRITE(280,10) N,TEST
      WRITE(290,10) N,TEST
      WRITE(300,10) N,TEST
      WRITE(310,10) N,TEST
      WRITE(320,10) N,TEST
      WRITE(330,10) N,TEST
      WRITE(340,10) N,TEST
      WRITE(350,10) N,TEST
      WRITE(360,10) N,TEST
      WRITE(370,10) N,TEST
C Format statement for plot file headers/footers
    9 FORMAT(1X,'ZONE T="Station ',I1,'"')
   10 FORMAT(1X,'TEXT X=45,Y=85,H=4.5,F=TIMES-ITALIC,',
     $       'T="  ',I4,' rpm \\n',A10,'"')
      PRINT*, 'FILES GENERATED FOR TECPLOT CONTOUR AND LINE PLOTS.'
      RETURN
      END
************************************************************************
* This subroutine calls loss subroutine to run required radial         *
* averaging and performnace analysis.                                  *
* Input Files : grid.txt                                               *
* Output Files: N/A                                                    *
* Experimental Procedure: Research Turbine Facility - TPFL             *
************************************************************************
* Hicham A Chibli - Texas A&M University - 01/25/10                    *
************************************************************************
      SUBROUTINE PRFM
      INTEGER GRID,RADIAL,ISKIP
      DOUBLE PRECISION MSPAN,PTRIM
```

```
C Open input file
      OPEN(UNIT=10,FILE='./inp/grid.txt',STATUS='OLD')
C Read grid analysis criteria
      READ(10,*)
      READ(10,*) RADIAL
      READ(10,*)
      READ(10,*) GRID
      READ(10,*)
      READ(10,*) ISKIP
      READ(10,*)
      READ(10,*) PTRIM
      READ(10,*)
      READ(10,*) MSPAN
C Place calls to averaging/performance analysis subroutine
      CALL SLOSS(RADIAL,GRID,ISKIP,PTRIM,MSPAN)
      RETURN
      END
*********************************************************************
* This subroutine calculates the thermodynamic properties of air.      *
* Input Files : N/A                                                   *
* Output Files: N/A                                                   *
* Experimental Procedure: Research Turbine Facility - TPFL            *
*********************************************************************
* Hicham A Chibli - Texas A&M University - 01/15/10                   *
*********************************************************************
      SUBROUTINE PROP(ATS,APS,RH,CP,R,GAMMA,ENS,S)
C Define variables and constants
      DOUBLE PRECISION ATS,APS,RH,CP,R,GAMMA,ENS,S,HRM,HR,ZERO,BAR
      REAL Z(13),HIGAS
      PARAMETER(ZERO=0.0D0,BAR=1.0D5)
C Initialize the parameters for HIGAS1 subroutine
      Z(1)=SNGL(ZERO)
C Call HRATIO subroutine to obtain the humidity ratio
      CALL HRATIO(ATS,APS,RH,HR,HRM)
      Z(2)=SNGL(HRM)
C Call HIGAS1 subroutine to calculate the humid air gas constant
      CALL HIGAS1(Z)
      R=DBLE(Z(4))*1000.0D0
C Use HIGAS function to calculate Cp of humid air
      CP=DBLE(HIGAS(1,SNGL(ATS),SNGL(APS/BAR),Z))*1000.0D0
C Calculate the gamma of humid air
      GAMMA=CP/(CP-R)
C Use HIGAS function to calculate enthalpy of humid air
      ENS=DBLE(HIGAS(2,SNGL(ATS),SNGL(APS/BAR),Z))*1000.0D0
C Use HIGAS function to calculate entropy of humid air
      S=DBLE(HIGAS(3,SNGL(ATS),SNGL(APS/BAR),Z))*1000.0D0
      RETURN
      END
*********************************************************************
* This subroutine iterates for the corrected temperature of humid air. *
* Input Files : N/A                                                   *
* Output Files: N/A                                                   *
* Experimental Procedure: Research Turbine Facility - Room 131        *
*********************************************************************
* Hicham A Chibli - Texas A&M University - 10/30/09                   *
*********************************************************************
      SUBROUTINE RDEN(APS,RH,RHO,ATS)
C Define variables and constants to use
      DOUBLE PRECISION RA,RV,KELVIN,SVP,VPP,DPP,RH,APS,ATS,RHO,ERROR,
     $                 LIMIT
      PARAMETER(RA=286.9D0,RV=461.495D0,KELVIN=273.15D0,LIMIT=0.001D0)
C Calculate first estimate for temperature
      ATS=(APS)/(RA*RHO)
C Calculate saturation vapor pressure
    1 SVP=6.1078D0*100.0D0*10.0D0**((7.5D0*(ATS-KELVIN))/
     $    (237.3D0+ATS-KELVIN))
C Calculate vapor partial pressure
```

```
      VPP=SVP*RH/100.0D0
C Calculate dry air partial pressure
      DPP=APS-VPP
C Calculate the error and check for convergence
      ERROR=RHO-DPP/(RA*ATS)-VPP/(RV*ATS)
      IF(DABS(ERROR/RHO).LE.LIMIT) RETURN
      ATS=(1.0D0/RHO)*(DPP/RA+VPP/RV)
      GO TO 1
      RETURN
      END
************************************************************************
* This subroutine calculates the average stage total temperatures      *
* using the total-static isentropic efficiency for the entire turbine. *
* Input Files : parameter.txt, raws4.int, basics3.int, basics5.int,    *
*              humidity.int                                            *
* Output Files: stemp.int                                              *
* Experimental Procedure: Research Turbine Facility - TPFL             *
************************************************************************
* Hicham A Chibli - Texas A&M University - 01/15/10                    *
************************************************************************
      SUBROUTINE TEMP
C Define variables and constants
      INTEGER N
C Define variables and constants
      DOUBLE PRECISION AD1(13),D1,D2,D3,D4,D5
      DOUBLE PRECISION TT1,ATT1,PT1,APT1,PS1,APS1,TT7,ATT7,TT3,ATT3,
     $                 APT3,APS3,TT4,TT5,ATT5,APT5,APS5,PATM,TTAVER,
     $                 PTAVER,ATS3S,ATS5S,CP,R,GAMMA,EFF,HUMID,HRM,HR,
     $                 KELVIN,ZERO,BAR
      REAL Z(13),HIGAS
      CHARACTER HUMIDITY*21,ATMOSPHERE*34,TEMPERATURE*41,
     $          TOTAL*39,STATIC*40
      PARAMETER(ZERO=0.0D0,KELVIN=273.15D0,BAR=1.0D5)
C Populate strings for screen output
      HUMIDITY='AVERAGE HUMIDITY (%)'
      ATMOSPHERE='AVERAGE ATMOSPHERIC PRESSURE (PA)'
      TEMPERATURE='AVERAGE TOTAL TEMPERATURE (C) - STATION 1'
      TOTAL='AVERAGE TOTAL PRESSURE (PA) - STATION 1'
      STATIC='AVERAGE STATIC PRESSURE (PA) - STATION 1'
C Initialize the parameters for HIGAS1 subroutine
      Z(1)=SNGL(ZERO)
C Open files for input
      OPEN(UNIT=10,FILE='./inp/parameter.txt',STATUS='OLD')
      OPEN(UNIT=20,FILE='./tmp/humidity.int',STATUS='OLD')
      OPEN(UNIT=30,FILE='./tmp/raws4.int',STATUS='OLD')
      OPEN(UNIT=40,FILE='./tmp/basics3.int',STATUS='OLD')
      OPEN(UNIT=50,FILE='./tmp/basics5.int',STATUS='OLD')
C Open files for output
      OPEN(UNIT=60,FILE='./tmp/stemp.int',STATUS='UNKNOWN')
C Read the total-static turbine efficiency
      READ(10,*)
      READ(10,*) D1
      READ(10,*)
      READ(10,*) EFF
C Read and average station 3 total and static pressures
      APS3=ZERO
      APT3=ZERO
      N=0
    3 READ(40,*,END=4) D1,D2
      APS3=APS3+D1
      APT3=APT3+D2
      N=N+1
      GO TO 3
    4 APS3=APS3/DBLE(N)
      APT3=APT3/DBLE(N)
C Read and average station 5 total and static pressures
      APS5=ZERO
      APT5=ZERO
      N=0
    5 READ(50,*,END=6) D1,D2
      APS5=APS5+D1
      APT5=APT5+D2
      N=N+1
      GO TO 5
    6 APS5=APS5/DBLE(N)
      APT5=APT5/DBLE(N)
C Read humidity data
      D1=ZERO
      D2=ZERO
      D3=ZERO
      D4=ZERO
      D5=ZERO
      N=0
    1 READ(20,*,END=2) HUMID
C Read turbine pressures and temperatures
      READ(30,*) AD1
      TT1=AD1(11)
      PT1=AD1(8)
      PS1=AD1(9)
      TT7=AD1(12)
      PATM=AD1(10)
C Calculate the average total temperature and pressure of the turbine
      ATT1=TT1+KELVIN
      ATT7=TT7+KELVIN
      APT1=PT1+PATM
      APS1=PS1+PATM
      TTAVER=(ATT1+ATT7)/2.0D0
      PTAVER=(APT1+APT5)/2.0D0
C Sum up turbine performance reference data
      D1=HUMID+D1
      D2=PATM+D2
      D3=TT1+D3
      D4=PT1+D4
      D5=PS1+D5
      N=N+1
C Call HRATIO subroutine to obtain the humidity ratio
      CALL HRATIO(TTAVER,PTAVER,HUMID,HR,HRM)
      Z(2)=SNGL(HRM)
C Call HIGAS1 subroutine to calculate the humid air gas constant
      CALL HIGAS1(Z)
      R=DBLE(Z(4))*1000.0D0
C Use HIGAS function to calculate Cp of humid air
      CP=DBLE(HIGAS(1,SNGL(TTAVER),SNGL(PTAVER/BAR),Z))*1000.0D0
C Calculate the gamma of humid air
      GAMMA=CP/(CP-R)
C Calculate isentrpoic static temperature for station 3
      ATS3S=ATT1*(APS3/APT1)**((GAMMA-1.0D0)/GAMMA)
C Calculate the polytrpoic total temperature for station 3
      ATT3=ATT1-(ATT1-ATS3S)*EFF
      TT3=ATT3-KELVIN
C Calculate the total temperature for station 4 including heat loss
      TT4=0.98D0*TT3
C Calculate isentrpoic static temperature for station 5
      ATS5S=ATT3*(APS5/APT3)**((GAMMA-1.0D0)/GAMMA)
C Calculate the polytrpoic total temperature for station 5
      ATT5=ATT3-(ATT3-ATS5S)*EFF
      TT5=ATT5-KELVIN
C Write temperatures to out file
      WRITE(60,*) TT3,TT4,TT5
      GO TO 1
C Average the turbine performance reference data
    2 HUMID=D1/DBLE(N)
      PATM=D2/DBLE(N)
      TT1=D3/DBLE(N)
      PT1=D4/DBLE(N)
      PS1=D5/DBLE(N)
C Write correction parameters to screen
      WRITE(*,*) 'UPDATE THE PARAMETER FILE AS FOLLOWS:'
      WRITE(*,*) '===================================='
      WRITE(*,9) HUMIDITY,HUMID
      WRITE(*,11) ATMOSPHERE,PATM
      WRITE(*,10) TEMPERATURE,TT1
      WRITE(*,11) TOTAL,PT1
      WRITE(*,11) STATIC,PS1
    9 FORMAT(1X,'REFERENCE',1X,A,1X,'=',1X,F6.2)
   10 FORMAT(1X,'REFERENCE',1X,A,1X,'=',1X,F6.2)
   11 FORMAT(1X,'REFERENCE',1X,A,1X,'=',1X,F11.2)
C Print to screen
      PRINT*, 'STAGE TOTAL TEMPERATURES CORRECTED.'
      RETURN
      END
*************************************************************************
* This subroutine calculates the static temperature using the           *
* isentropic relationships and the corrected thermodynamic properties. *
* Input Files : N/A                                                     *
* Output Files: N/A                                                     *
* Experimental Procedure: Research Turbine Facility - TPFL              *
*************************************************************************
* Hicham A Chibli - Texas A&M University - 01/15/10                     *
*************************************************************************
      SUBROUTINE TSTAT(ATT,APT,APS,RH,ATS)
C Define variables and constants
      DOUBLE PRECISION ATT,ATS,APT,APS,RH,CP,R,GAMMA,HRM,HR,
     $                 LIMIT,OLD,ZERO,BAR
      REAL Z(13),HIGAS
      PARAMETER(LIMIT=0.001D0,ZERO=0.0D0,BAR=1.0D5)
C Initialize the parameters for HIGAS1 subroutine
      Z(1)=SNGL(ZERO)
C Make a first guess for gamma
      GAMMA=1.4D0
C Use ideal gas relationship for a first guess of static temperature
      ATS=ATT*(APS/APT)**((GAMMA-1.0D0)/GAMMA)
C Call HRATIO subroutine to obtain the humidity ratio
    1 CALL HRATIO(ATS,APS,RH,HR,HRM)
      Z(2)=SNGL(HRM)
C Call HIGAS1 subroutine to calculate the humid air gas constant
      CALL HIGAS1(Z)
      R=DBLE(Z(4))*1000.0D0
C Use HIGAS function to calculate Cp of humid air
      CP=DBLE(HIGAS(1,SNGL(ATS),SNGL(APS/BAR),Z))*1000.0D0
C Calculate the gamma of humid air
      GAMMA=CP/(CP-R)
C Store old static temperature
      OLD=ATS
C Use ideal gas relationship to compute corrected static temperature
      ATS=ATT*(APS/APT)**((GAMMA-1.0D0)/GAMMA)
      IF(DABS((ATS-OLD)/ATS).LE.LIMIT) RETURN
      GO TO 1
      RETURN
      END
```

J.3 Blade Cascade Flow Incidence Study Data Analysis Programs

```
**********************************************************************
* This program reads the inlet and exit flow profile data and analyzes *
* it to calculate various flow properties, forces and losses.         *
* Input Files : exit.dat, info.dat, aver.dat, blade.dat               *
* Output Files: casexx.dat, cpxx.dat, xplotxx.data, nxplotxx.dat      *
* Experimental Procedure: Cascade Facility of the DOOSAN I Blade      *
**********************************************************************
* Hicham A Chibli - Texas A&M University - 09/10/10                   *
**********************************************************************
      PROGRAM ANLS
C Define variables, parameters and constants to use
      INTEGER K,J,W1,W2,P1,P2,F1,F2,M1,M2,BPL,EPL,ISIZE,IDEN
      DOUBLE PRECISION WKS,WKE,PRS,PRE,MID,PITCH,CHORD,ACHORD,LL,
     $                 UL,BPLOT,EPLOT,STEP,RHOREF,VXREF,REXREF,
     $                 IRHO,IMU,IMACH,IRE,IVXE,IREXE,IVXW,IREXW,
     $                 EPTM,EPDM,EPSM,EVM,EVXM,EVYM,EREM,ERHOM,
     $                 EMUM,EMACHM,EMFL,EFL,WPTM,WPDM,WPSM,WVM,
     $                 WVXM,WVYM,WREM,WRHOM,WMUM,WMACHM,WMFL,WFL,
     $                 FPTM,FPDM,FPSM,FVM,FVXM,FVYM,FMFL,
     $                 ELSS1,ELSS2,WLSS1,WLSS2,A1,AI,IVX,IV,PI,
     $                 LASTR,LASTX,INCD,MULT,PFT,CPFT,
     $                 DEL2,SDEL2,PDEL2,DLTA2,CPDEL2,CSDEL2,CDLTA2,
     $                 DEL1,SDEL1,PDEL1,DLTA1,CPDEL1,CSDEL1,CDLTA1
      CHARACTER CASNME*10,EXT*4,CASNUM*2,CP*8,XPLOT*11,NXPLOT*12
      PARAMETER(K=195,ISIZE=42,STEP=1.5D0,CHORD=0.15D0,ACHORD=0.12453D0,
     $          AI=5.5D0,PI=3.141592654D0)
      DOUBLE PRECISION EPT(K),EPS(K),EPD(K),EV(K),EVY(K),EVX(K),
     $                 ERHO(K),EMU(K),ERE(K),EMACH(K),AXIAL(K),
     $                 CEPT(K,2),CEPS(K,2),CEPD(K,2),CEV(K,2),
     $                 CEVY(K,2),CEVX(K,2),
     $                 NEPT(K,2),NEPS(K,2),NEPD(K,2),NEV(K,2),
     $                 NEVY(K,2),NEVX(K,2),NAXIAL(K,2),
     $                 NAVER(6),
     $                 P(ISIZE),PP(ISIZE),DIST(ISIZE),NDIST(ISIZE)
C Open data files for input
      OPEN(UNIT=20,FILE='./out/exit.dat',STATUS='OLD')
      OPEN(UNIT=40,FILE='./in/info.dat',STATUS='OLD')
      OPEN(UNIT=90,FILE='./out/aver.dat',STATUS='OLD')
C Read data from input file aver.dat
      READ(90,*) LASTR,LASTX
      READ(90,*) IRHO,IMU,IRE,IMACH
C Define angle unit transformation
      MULT=PI/180.0D0
C Read data from input file info.dat
      READ(40,*)
      READ(40,*) CASNUM
      READ(40,*)
      READ(40,*) INCD
      READ(40,*)
      READ(40,*) RHOREF
      READ(40,*)
      READ(40,*) VXREF
      READ(40,*)
      READ(40,*) REXREF
      READ(40,*)
      READ(40,*) PRS
      READ(40,*)
      READ(40,*) PRE
      READ(40,*)
      READ(40,*) WKS
      READ(40,*)
      READ(40,*) WKE
      READ(40,*)
      READ(40,*) LL
      READ(40,*)
      READ(40,*) UL
      READ(40,*)
      READ(40,*) IDEN
C Generate input/output file names
      EXT='.dat'
      CASNME='case'//CASNUM//EXT
      CP='cp'//CASNUM//EXT
      XPLOT='xplot'//CASNUM//EXT
      NXPLOT='nxplot'//CASNUM//EXT
C Open data file for input
      OPEN(UNIT=60,FILE='./in/blade.dat',STATUS='OLD')
C Open data files for output
      OPEN(UNIT=30,FILE='./out/'//CASNME,STATUS='UNKNOWN')
      OPEN(UNIT=50,FILE='./out/'//XPLOT,STATUS='UNKNOWN')
      OPEN(UNIT=51,FILE='./out/'//NXPLOT,STATUS='UNKNOWN')
      OPEN(UNIT=52,FILE='./out/'//CP,STATUS='UNKNOWN')
C Read data from iput file exit.dat
      READ(20,*)
      READ(20,*)
      DO 5 J=1,K
         READ(20,6) EPT(J),EPD(J),EPS(J),EV(J),EVY(J),EVX(J),
     $              ERHO(J),EMU(J),EMACH(J),ERE(J),AXIAL(J)
    6    FORMAT(22X,F21.5,25X,F23.5,24X,F22.5,44X,F18.5,23X,F19.5,2X,
     $          F19.5,2X,F19.10,2X,F22.10,2X,F13.4,2X,F17.1,2X,F19.1)
    5 CONTINUE
C Call PTRIM and WAKE subroutines
      CALL PTRIM(K,P1,P2,PRS,PRE,AXIAL)
      CALL PTRIM(K,W1,W2,WKS,WKE,AXIAL)
      CALL WAKE(K,W1,W2,F1,F2,STEP,LL,UL,EPT)
      MID=(AXIAL(W1)+AXIAL(W2))/2.0D0
      CALL PTRIM(K,W1,M1,WKS,MID,AXIAL)
      CALL PTRIM(K,M2,W2,MID,WKE,AXIAL)
C Calculate the plotting period size
      PITCH=AXIAL(W2)-AXIAL(W1)
      BPLOT=AXIAL(W1)-PITCH/2.0D0
      EPLOT=AXIAL(W2)+PITCH/2.0D0
      CALL PTRIM(K,BPL,EPL,BPLOT,EPLOT,AXIAL)
C Write to screen the trimming criteria
      WRITE(*,26) AXIAL(P1),AXIAL(P2)
   26 FORMAT(1X,'EXIT PROFILE DATA HAS BEEN TRIMMED BETWEEN',/,1X,
     $       'Y1 =',1X,F6.2,2X,'AND',2X,'Y2 =',1X,F6.2,//)
      WRITE(*,27) AXIAL(W1),AXIAL(W2)
   27 FORMAT(1X,'COMPLETE WAKE PROFILE HAS BEEN TRIMMED BETWEEN',/,1X,
     $       'Y1 =',1X,F6.2,2X,'AND',2X,'Y2 =',1X,F6.2,//)
      WRITE(*,28) AXIAL(F1),AXIAL(F2)
   28 FORMAT(1X,'FREE STREAM WAKE PROFILE HAS BEEN TRIMMED BETWEEN',/,
     $       1X,'Y1 =',1X,F6.2,2X,'AND',2X,'Y2 =',1X,F6.2,//)
C Integrate using Trapezoidal Rule the full trimmed exit profile
      EPTM=0.0D0
      EPDM=0.0D0
      EPSM=0.0D0
      EVM=0.0D0
      EVXM=0.0D0
      EVYM=0.0D0
      EREM=0.0D0
      ERHOM=0.0D0
      EMUM=0.0D0
      EMACHM=0.0D0
      EMFL=0.0D0
      EFL=0.0D0
      DO 7 J=P1,P2-1
         EPTM=EPTM+0.5D0*(EPT(J)*EVX(J)*ERHO(J)+
     $        EPT(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         EPDM=EPDM+0.5D0*(EPD(J)*EVX(J)*ERHO(J)+
     $        EPD(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
```

```
         EPSM=EPSM+0.5D0*(EPS(J)*EVX(J)*ERHO(J)+
     $        EPS(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         EVM=EVM+0.5D0*(EV(J)*EVX(J)*ERHO(J)+
     $       EV(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         EVXM=EVXM+0.5D0*(EVX(J)*EVX(J)*ERHO(J)+
     $        EVX(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         EVYM=EVYM+0.5D0*(EVY(J)*EVX(J)*ERHO(J)+
     $        EVY(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         ERHOM=ERHOM+0.5D0*(ERHO(J)*EVX(J)*ERHO(J)+
     $         ERHO(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         EMUM=EMUM+0.5D0*(EMU(J)*EVX(J)*ERHO(J)+
     $        EMU(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         EREM=EREM+0.5D0*(ERE(J)*EVX(J)*ERHO(J)+
     $        ERE(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         EMACHM=EMACHM+0.5D0*(EMACH(J)*EVX(J)*ERHO(J)+
     $          EMACH(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         EMFL=EMFL+0.5D0*(EVX(J)*ERHO(J)+EVX(J+1)*ERHO(J+1))*
     $        (AXIAL(J+1)-AXIAL(J))
    7 CONTINUE
      EPTM=EPTM/EMFL
      EPDM=EPDM/EMFL
      EPSM=EPSM/EMFL
      EVM=EVM/EMFL
      EVXM=EVXM/EMFL
      EVYM=EVYM/EMFL
      ERHOM=ERHOM/EMFL
      EMUM=EMUM/EMFL
      EREM=EREM/EMFL
      EMACHM=EMACHM/EMFL
      EFL=EMFL/(AXIAL(P2)-AXIAL(P1))
      EMFL=EMFL/1000.0D0
C Integrate using Trapezoidal Rule the wake trimmed exit profile
      WPTM=0.0D0
      WPDM=0.0D0
      WPSM=0.0D0
      WVM=0.0D0
      WVXM=0.0D0
      WVYM=0.0D0
      WREM=0.0D0
      WRHOM=0.0D0
      WMUM=0.0D0
      WMACHM=0.0D0
      WMFL=0.0D0
      WFL=0.0D0
      DO 25 J=W1,W2-1
         WPTM=WPTM+0.5D0*(EPT(J)*EVX(J)*ERHO(J)+
     $        EPT(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         WPDM=WPDM+0.5D0*(EPD(J)*EVX(J)*ERHO(J)+
     $        EPD(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         WPSM=WPSM+0.5D0*(EPS(J)*EVX(J)*ERHO(J)+
     $        EPS(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         WVM=WVM+0.5D0*(EV(J)*EVX(J)*ERHO(J)+
     $       EV(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         WVXM=WVXM+0.5D0*(EVX(J)*EVX(J)*ERHO(J)+
     $        EVX(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         WVYM=WVYM+0.5D0*(EVY(J)*EVX(J)*ERHO(J)+
     $        EVY(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         WRHOM=WRHOM+0.5D0*(ERHO(J)*EVX(J)*ERHO(J)+
     $         ERHO(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         WMUM=WMUM+0.5D0*(EMU(J)*EVX(J)*ERHO(J)+
     $        EMU(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         WREM=WREM+0.5D0*(ERE(J)*EVX(J)*ERHO(J)+
     $        ERE(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         WMACHM=WMACHM+0.5D0*(EMACH(J)*EVX(J)*ERHO(J)+
     $          EMACH(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         WMFL=WMFL+0.5D0*(EVX(J)*ERHO(J)+EVX(J+1)*ERHO(J+1))*
     $        (AXIAL(J+1)-AXIAL(J))
```

```
   25 CONTINUE
      WPTM=WPTM/WMFL
      WPDM=WPDM/WMFL
      WPSM=WPSM/WMFL
      WVM=WVM/WMFL
      WVXM=WVXM/WMFL
      WVYM=WVYM/WMFL
      WRHOM=WRHOM/WMFL
      WMUM=WMUM/WMFL
      WREM=WREM/WMFL
      WMACHM=WMACHM/WMFL
      WFL=WMFL/(AXIAL(W2)-AXIAL(W1))
      WMFL=WMFL/1000.0D0
C Integrate using Trapezoidal Rule the free stream exit profile
      FPTM=0.0D0
      FPDM=0.0D0
      FPSM=0.0D0
      FVM=0.0D0
      FVXM=0.0D0
      FVYM=0.0D0
      FMFL=0.0D0
      DO 24 J=F1,F2-1
         FPTM=FPTM+0.5D0*(EPT(J)*EVX(J)*ERHO(J)+
     $        EPT(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         FPDM=FPDM+0.5D0*(EPD(J)*EVX(J)*ERHO(J)+
     $        EPD(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         FPSM=FPSM+0.5D0*(EPS(J)*EVX(J)*ERHO(J)+
     $        EPS(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         FVM=FVM+0.5D0*(EV(J)*EVX(J)*ERHO(J)+
     $       EV(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         FVXM=FVXM+0.5D0*(EVX(J)*EVX(J)*ERHO(J)+
     $        EVX(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         FVYM=FVYM+0.5D0*(EVY(J)*EVX(J)*ERHO(J)+
     $        EVY(J+1)*EVX(J+1)*ERHO(J+1))*(AXIAL(J+1)-AXIAL(J))
         FMFL=FMFL+0.5D0*(EVX(J)*ERHO(J)+EVX(J+1)*ERHO(J+1))*
     $        (AXIAL(J+1)-AXIAL(J))
   24 CONTINUE
      FPTM=FPTM/FMFL
      FPDM=FPDM/FMFL
      FPSM=FPSM/FMFL
      FVM=FVM/FMFL
      FVXM=FVXM/FMFL
      FVYM=FVYM/FMFL
C find locally corrected axial inlet V and RE based on full profile
      IVXE=EFL/IRHO
      IREXE=IVXE*IRHO*CHORD/IMU
C find locally corrected axial inlet V and RE based on wake profile
      IVXW=WFL/IRHO
      IREXW=IVXW*IRHO*CHORD/IMU
C Compute loss coefficient for trimmed full profile
      ELSS1=(FPTM-EPTM)/EPDM
C Compute loss coefficient for trimmed wake profile
      WLSS1=(FPTM-WPTM)/WPDM
C Compute the PFEIL globally corrected loss coefficient for tfp
      ELSS2=ELSS1*(IREXE/REXREF)**0.2D0
C Compute the PFEIL globally corrected loss coefficient for twp
      WLSS2=WLSS1*(IREXW/REXREF)**0.2D0
C Compute the momentum deficiency thickness for pressure side wake
      PDEL2=0.0D0
      DO 29 J=W1,F1-1,1
         PDEL2=PDEL2+(STEP/2.0D0)*(DEL2(EV(J),FVM)+DEL2(EV(J+1),FVM))
   29 CONTINUE
C Compute the momentum deficiency thickness for suction side wake
      SDEL2=0.0D0
      DO 30 J=W2,F2+1,-1
         SDEL2=SDEL2+(STEP/2.0D0)*(DEL2(EV(J),FVM)+DEL2(EV(J-1),FVM))
   30 CONTINUE
```

```
      DLTA2=PDEL2+SDEL2
C Compute the displacement deficiency thickness for pressure side wake
      PDEL1=0.0D0
      DO 62 J=W1,F1-1,1
         PDEL1=PDEL1+(STEP/2.0D0)*(DEL1(EV(J),FVM)+DEL1(EV(J+1),FVM))
   62 CONTINUE
C Compute the displacement deficiency thickness for suction side wake
      SDEL1=0.0D0
      DO 63 J=W2,F2+1,-1
         SDEL1=SDEL1+(STEP/2.0D0)*(DEL1(EV(J),FVM)+DEL1(EV(J-1),FVM))
   63 CONTINUE
      DLTA1=PDEL1+SDEL1
C Compute the PFEIL globally correctd boundary layer values
      CPDEL2=0.5D0*PDEL2*((IREXW/REXREF)**0.2D0+(IREXE/REXREF)**0.2D0)
      CSDEL2=0.5D0*SDEL2*((IREXW/REXREF)**0.2D0+(IREXE/REXREF)**0.2D0)
      CDLTA2=CPDEL2+CSDEL2
      CPDEL1=0.5D0*PDEL1*((IREXW/REXREF)**0.2D0+(IREXE/REXREF)**0.2D0)
      CSDEL1=0.5D0*SDEL1*((IREXW/REXREF)**0.2D0+(IREXE/REXREF)**0.2D0)
      CDLTA1=CPDEL1+CSDEL1
C Globally correct the exit profiles
      DO 39 J=1,K
         CEPT(J,1)=EPT(J)*(RHOREF/IRHO)*(VXREF/IVXE)**2.0D0
         CEPT(J,2)=EPT(J)*(RHOREF/IRHO)*(VXREF/IVXW)**2.0D0
         CEPD(J,1)=EPD(J)*(RHOREF/IRHO)*(VXREF/IVXE)**2.0D0
         CEPD(J,2)=EPD(J)*(RHOREF/IRHO)*(VXREF/IVXW)**2.0D0
         CEPS(J,1)=EPS(J)*(RHOREF/IRHO)*(VXREF/IVXE)**2.0D0
         CEPS(J,2)=EPS(J)*(RHOREF/IRHO)*(VXREF/IVXW)**2.0D0
         CEV(J,1)=EV(J)*(VXREF/IVXE)
         CEV(J,2)=EV(J)*(VXREF/IVXW)
         CEVX(J,1)=EVX(J)*(VXREF/IVXE)
         CEVX(J,2)=EVX(J)*(VXREF/IVXW)
         CEVY(J,1)=EVY(J)*(VXREF/IVXE)
         CEVY(J,2)=EVY(J)*(VXREF/IVXW)
   39 CONTINUE
C Establish the base values for normilizing the exit profile
      NAVER(1)=(EPT(M1)+EPT(M2))/2.0D0
      NAVER(2)=(EPD(M1)+EPD(M2))/2.0D0
      NAVER(3)=(EPS(M1)+EPS(M2))/2.0D0
      NAVER(4)=(EV(M1)+EV(M2))/2.0D0
      NAVER(5)=(EVX(M1)+EVX(M2))/2.0D0
      NAVER(6)=(EVY(M1)+EVY(M2))/2.0D0
C Normalize the exit profile variabels using the trimmed full profile
      DO 40 J=P1,P2
         NEPT(J,1)=EPT(J)/NAVER(1)
         NEPD(J,1)=EPD(J)/NAVER(2)
         NEPS(J,1)=EPS(J)/NAVER(3)
         NEV(J,1)=EV(J)/NAVER(4)
         NEVX(J,1)=EVX(J)/NAVER(5)
         NEVY(J,1)=EVY(J)/NAVER(6)
         NAXIAL(J,1)=(AXIAL(J)-AXIAL(P1))/(AXIAL(P2)-AXIAL(P1))
   40 CONTINUE
C Normalize the exit profile variabels using the trimmed wake profile
      DO 41 J=BPL,EPL
         NEPT(J,2)=EPT(J)/NAVER(1)
         NEPD(J,2)=EPD(J)/NAVER(2)
         NEPS(J,2)=EPS(J)/NAVER(3)
         NEV(J,2)=EV(J)/NAVER(4)
         NEVX(J,2)=EVX(J)/NAVER(5)
         NEVY(J,2)=EVY(J)/NAVER(6)
         NAXIAL(J,2)=(AXIAL(J)-AXIAL(BPL))/(AXIAL(EPL)-AXIAL(BPL))
   41 CONTINUE
C Find the inlet locally corrected velocity
      A1=AI-INCD
      IVX=(IVXW+IVXE)/2.0D0
      IV=IVX/DCOS(A1*MULT)
C Read data from inupt file blade.dat
      DO 22 J=1,ISIZE
```

```
         IF(J.EQ.19) THEN
           NDIST(J)=1.0D0
           DIST(J)=NDIST(J)*ACHORD
         ELSE
             READ(60,*) NDIST(J),P(J)
             DIST(J)=NDIST(J)*ACHORD
C Locally correct pressure if needed
             IF(IDEN.EQ.1) P(J)=P(J)*(IRHO/LASTR)*LASTX
             PP(J)=(P(J)-FPTM)/FPTM
         ENDIF
   22 CONTINUE
      P(19)=0.5D0*(P(18)+P(20))
      PP(19)=(P(19)-FPTM)/FPTM
C Write Cp distribution data to cpxx.dat output file
      WRITE(52,*) 'VARIABLES="X",','"C_p"'
      WRITE(52,33) INCD
   33 FORMAT('ZONE T="exp.  i = ',F5.1,' ^o"')
      DO 23 J=1,ISIZE
         IF(J.EQ.19) GO TO 23
         WRITE(52,*) NDIST(J),PP(J)
   23 CONTINUE
C Integrate the pressure distribution to calculate tang. force on blade
      PFT=0.0D0
      DO 35 J=1,ISIZE-1
         PFT=PFT+0.5D0*(P(J)+P(J+1))*(DIST(J+1)-DIST(J))
   35 CONTINUE
      PFT=DABS(PFT)
C Calculate the globally corrected force
      CPFT=PFT*(RHOREF/IRHO)*(VXREF/IVXE)**2.0D0
C Write output to casex.dat
      WRITE(30,8) CASNUM,INCD
    8 FORMAT(1X,'CASE NUMBER:',1X,A,/,1X,'INCIDENCE ANGLE (DEG):',
     $       1X,F5.1,//)
C Write profile data summary
      WRITE(30,9) FPTM,IRHO,IMU,IRE,IMACH
    9 FORMAT(1X,'MEASURED INLET PROFILE SUMMARY',/,1X,
     $       '******************************',/,
     $       1X,'AVERAGE TOTAL PRESSURE [PA]',2X,'= ',F11.1,/,
     $       1X,'AVERAGE DENSITY [KG/M**3]',4X,'= ',F11.3,/,
     $       1X,'AVERAGE VISCOSITY [KG/M/SEC]',1X,'= ',F11.9,/,
     $       1X,'AVERAGE REYNOLDS NUMBER',6X,'= ',F11.1,/,
     $       1X,'AVERAGE MACH NUMBER',10X,'= ',F11.4,//)
      WRITE(30,10) EPTM,EPSM,EPDM,EVM,EVXM,EVYM,ERHOM,
     $             EMUM,EREM,EMACHM,EFL
   10 FORMAT(1X,'MEASURED EXIT PROFILE SUMMARY - FULL',/,1X,
     $       '************************************',/,
     $       1X,'AVERAGE TOTAL PRESSURE [PA]',9X,'= ',F11.1,/,
     $       1X,'AVERAGE STATIC PRESSURE [PA]',8X,'= ',F11.1,/,
     $       1X,'AVERAGE DYNAMIC PRESSURE [PA]',7X,'= ',F11.1,/,
     $       1X,'AVERAGE VELOCITY [M/SE]',13X,'= ',F11.3,/,
     $       1X,'AVERAGE AXIAL VELOCITY [M/SEC]',6X,'= ',F11.3,/,
     $       1X,'AVERAGE TANGENTIAL VELOCITY [M/SEC]',1X,'= ',F11.3,/,
     $       1X,'AVERAGE DENSITY [KG/M**3]',11X,'= ',F11.3,/,
     $       1X,'AVERAGE VISCOSITY [KG/M/SEC]',8X,'= ',F11.9,/,
     $       1X,'AVERAGE REYNOLDS NUMBER',13X,'= ',F11.1,/,
     $       1X,'AVERAGE MACH NUMBER',17X,'= ',F11.4,/,
     $       1X,'MASS FLOW RATE [KG/SEC/M**2]',8X,'= ',F11.4,//)
      WRITE(30,12) WPTM,WPSM,WPDM,WVM,WVXM,WVYM,WRHOM,
     $             WMUM,WREM,WMACHM,WFL
   12 FORMAT(1X,'MEASURED EXIT PROFILE SUMMARY - LIMITED',/,1X,
     $       '***************************************',/,
     $       1X,'AVERAGE TOTAL PRESSURE [PA]',9X,'= ',F11.1,/,
     $       1X,'AVERAGE STATIC PRESSURE [PA]',8X,'= ',F11.1,/,
     $       1X,'AVERAGE DYNAMIC PRESSURE [PA]',7X,'= ',F11.1,/,
     $       1X,'AVERAGE VELOCITY [M/SE]',13X,'= ',F11.3,/,
     $       1X,'AVERAGE AXIAL VELOCITY [M/SEC]',6X,'= ',F11.3,/,
     $       1X,'AVERAGE TANGENTIAL VELOCITY [M/SEC]',1X,'= ',F11.3,/,
     $       1X,'AVERAGE DENSITY [KG/M**3]',11X,'= ',F11.3,/,
     $       1X,'AVERAGE VISCOSITY [KG/M/SEC]',8X,'= ',F11.9,/,
     $       1X,'AVERAGE REYNOLDS NUMBER',13X,'= ',F11.1,/,
     $       1X,'AVERAGE MACH NUMBER',17X,'= ',F11.3,/,
     $       1X,'MASS FLOW RATE [KG/SEC/M**2]',8X,'= ',F11.4,//)
C Write loss coefficient summary
      WRITE(30,13) ELSS1,WLSS1,ELSS2,WLSS2
   13 FORMAT(1X,'PROFILE LOSS SUMMARY',/,1X,'********************',/,
     $       1X,'MEASURED PROFILE LOSS COEFFICICNET - FULL     =',
     $       1X,F11.9,/,
     $       1X,'MEASURED PROFILE LOSS COEFFICICNET - LIMITED  =',
     $       1X,F11.9,/,
     $       1X,'CORRECTED PROFILE LOSS COEFFICICNET - FULL    =',
     $       1X,F11.9,/,
     $       1X,'CORRECTED PROFILE LOSS COEFFICICNET - LIMITED =',
     $       1X,F11.9,//)
C Write boundary layer summary
      WRITE(30,31) SDEL2,CSDEL2,PDEL2,CPDEL2,DLTA2,CDLTA2
   31 FORMAT(1X,'MOMENTUM DEFICIENCY THICKNESS CALCULATIONS',/,
     $       1X,'******************************************',/,
     $       1X,'M. SUCTION SIDE MOMENTUM DEFICIENCY THICKNESS  [MM] =',
     $       1X,F12.9,/,
     $       1X,'C. SUCTION SIDE MOMENTUM DEFICIENCY THICKNESS  [MM] =',
     $       1X,F12.9,/,
     $       1X,'M. PRESSURE SIDE MOMENTUM DEFICIENCY THICKNESS [MM] =',
     $       1X,F12.9,/,
     $       1X,'C. PRESSURE SIDE MOMENTUM DEFICIENCY THICKNESS [MM] =',
     $       1X,F12.9,/,
     $       1X,'M. TOTAL MOMENTUM DEFICIENCY THICKNESS         [MM] =',
     $       1X,F12.9,/,
     $       1X,'C. TOTAL MOMENTUM DEFICIENCY THICKNESS         [MM] =',
     $       1X,F12.9,//)
      WRITE(30,64) SDEL1,CSDEL1,PDEL1,CPDEL1,DLTA1,CDLTA1
   64 FORMAT(1X,'DISPLACEMENT DEFICIENCY THICKNESS CALCULATIONS',/,
     $       1X,'**********************************************',/,
     $       1X,'M. SUCTION SIDE DISP. DEFICIENCY THICKNESS  [MM] =',
     $       1X,F12.9,/,
     $       1X,'C. SUCTION SIDE DISP. DEFICIENCY THICKNESS  [MM] =',
     $       1X,F12.9,/,
     $       1X,'M. PRESSURE SIDE DISP. DEFICIENCY THICKNESS [MM] =',
     $       1X,F12.9,/,
     $       1X,'C. PRESSURE SIDE DISP. DEFICIENCY THICKNESS [MM] =',
     $       1X,F12.9,/,
     $       1X,'M. TOTAL DISPLACEMENT DEFICIENCY THICKNESS  [MM] =',
     $       1X,F12.9,/,
     $       1X,'C. TOTAL DISPLACEMENT DEFICIENCY THICKNESS  [MM] =',
     $       1X,F12.9,//)
C Write the blade tangential pressure force calculated
      WRITE(30,36) PFT,CPFT
   36 FORMAT(1X,'PRESSURE TANGENTIAL BLADE FORCE',/,
     $       1X,'*******************************',/,
     $       1X,'MEASURED [N/M]',2X,' = ',F10.4,/,
     $       1X,'CORRECTED [N/M]',2X,'= ',F10.4,//)
C Write correction parameters used
      WRITE(30,42) IVXE,IVXW,VXREF,IREXE,IREXW,REXREF,IRHO,RHOREF
   42 FORMAT(1X,'CORRECTION PARAMETERS',/,
     $       1X,'*********************',/,
     $       1X,'INLET AXIAL VELOCITY - FULL [M/SEC]',6X,'= ',F8.3,/,
     $       1X,'INLET AXIAL VELOCITY - LIMITED [M/SEC]',3X,'= ',F8.3,/,
     $       1X,'REFERENCE AXIAL VELOCITY [M/SEC]',9X,'= ',F8.3,/,
     $       1X,'INLET AXIAL REYNOLDS NUMBER - FULL',7X,'= ',F8.1,/,
     $       1X,'INLET AXIAL REYNOLDS NUMBER - LIMITED',4X,'= ',F8.1,/,
     $       1X,'REFERENCE AXIAL REYNOLDS NUMBER',10X,'= ',F8.1,/,
     $       1X,'INLET DENSITY [KG/M**3]',18X,'= ',F8.3,/,
     $       1X,'REFERENCE DENSITY [KG/M**3]',14X,'= ',F8.3,//)
C Write output to xplot.dat
      WRITE(50,*) 'VARIABLES="Y",',
     $            '"P_t [Pa]",',
     $            '"P_s [Pa]",',
     $            '"P_d [Pa]",',
     $            '"V [m/sec]"',
     $            '"V_x [m/sec]"',
     $            '"V_y [m/sec]"'
      WRITE(50,33) INCD
      DO 37 J=P1,P2
         WRITE(50,*) NAXIAL(J,1),CEPT(J,1),CEPS(J,1),CEPD(J,1),
     $               CEV(J,1),CEVX(J,1),CEVY(J,1)
   37 CONTINUE
      WRITE(50,33) INCD
      DO 38 J=BPL,EPL
         WRITE(50,*) NAXIAL(J,2),CEPT(J,2),CEPS(J,2),CEPD(J,2),
     $               CEV(J,2),CEVX(J,2),CEVY(J,2)
   38 CONTINUE
C Write output to nxplot.dat
      WRITE(51,*) 'VARIABLES="Y",',
     $            '"P_t_n",',
     $            '"P_s_n",',
     $            '"P_d_n",',
     $            '"V_n"',
     $            '"V_x_n"',
     $            '"V_y_n"'
      WRITE(51,33) INCD
      DO 43 J=P1,P2
         WRITE(51,*) NAXIAL(J,1),NEPT(J,1),NEPS(J,1),NEPD(J,1),
     $               NEV(J,1),NEVX(J,1),NEVY(J,1)
   43 CONTINUE
      WRITE(51,33) INCD
      DO 44 J=BPL,EPL
         WRITE(51,*) NAXIAL(J,2),NEPT(J,2),NEPS(J,2),NEPD(J,2),
     $               NEV(J,2),NEVX(J,2),NEVY(J,2)
   44 CONTINUE
      PRINT*, 'END OF ANALYSIS'
      END
************************************************************************
C Define subroutine PTRIM
************************************************************************
      SUBROUTINE PTRIM(K,M,N,X,Y,Z)
      INTEGER M,N,K,J,I
      DOUBLE PRECISION X,Y
      DOUBLE PRECISION Z(K)
      I=0
      DO 14 J=1,K
         IF(I.EQ.0) THEN
           IF(Z(J).GE.X) THEN
             I=1
             M=J
           ENDIF
         ENDIF
   14 CONTINUE
      I=0
      DO 15 J=1,K
         IF(I.EQ.0) THEN
           IF(Z(J).GT.Y) THEN
             I=1
             N=J-1
           ENDIF
         ENDIF
   15 CONTINUE
      RETURN
      END
************************************************************************
C Define subroutine WAKE
************************************************************************
      SUBROUTINE WAKE(K,M,N,I,J,S,LL,UL,Z)
      INTEGER M,N,K,J,I,L
      DOUBLE PRECISION ERROR,LL,UL,S
      DOUBLE PRECISION Z(K),LAREA(M:N-1),UAREA(M+1:N)
      DO 16 L=M,N-1,1
C Integrate using Trapezoidal Rule from lower end
         LAREA(L)=(S*0.5D0)*(Z(L+1)+Z(L))
   16 CONTINUE
C Check convergence criteria
      DO 17 L=M,N-2,1
         IF(LAREA(L).GE.LAREA(L+1)) THEN
           ERROR=1.0D0-LAREA(L+1)/LAREA(L)
         ELSE
             ERROR=1.0D0-LAREA(L)/LAREA(L+1)
         ENDIF
         IF(ERROR.LE.LL) THEN
           I=L+1
           GO TO 18
         ENDIF
   17 CONTINUE
   18 DO 19 L=N,M+1,-1
C Integrate using Simpson’s Rule from upper end after rotation
         UAREA(L)=(S*0.5D0)*(Z(L-1)+Z(L))
   19 CONTINUE
C Check convergence criteria
      DO 20 L=N,M+2,-1
         IF(UAREA(L).GE.UAREA(L-1)) THEN
           ERROR=1.0D0-UAREA(L-1)/UAREA(L)
         ELSE
             ERROR=1.0D0-UAREA(L)/UAREA(L-1)
         ENDIF
         IF(ERROR.LE.UL) THEN
          J=L-1
           GO TO 21
         ENDIF
   20 CONTINUE
   21 RETURN
      END
************************************************************************
C Define function DEL2
************************************************************************
      DOUBLE PRECISION FUNCTION DEL2(U,UMEAN)
      DOUBLE PRECISION U,UMEAN
      DEL2=(U/UMEAN)*(1.0D0-U/UMEAN)
      RETURN
      END
************************************************************************
C Define function DEL1
************************************************************************
      DOUBLE PRECISION FUNCTION DEL1(U,UMEAN)
      DOUBLE PRECISION U,UMEAN
      DEL1=(1.0D0-U/UMEAN)
      RETURN
      END
************************************************************************
* This program will calculate the flow angles as well as the velocity, *
* total and static pressures using the Bohn’s calibration method of    *
* the five hole probe. Local  and ambient pressure and temperature     *
* measurements as well as the inlet flow density will be utilized to   *
* provide the needed corrections to account for mass flow variations   *
* through the experimental cascade. All calibration coefficients are   *
* intended for the 5HP#1 (old cobra probe) at 0.197 Mach (06/08) and   *
* the MKS 100 TORR pressure transducer (02/08).                        *
* Input Files : press.dat                                              *
* Output Files: exit.dat, aver.dat                                     *
* Experimental Procedure: Cascade Facility of the DOOSAN I Blade       *
************************************************************************
* Hicham A Chibli - Texas A&M University - 09/10/10                    *
```

```
************************************************************************
      PROGRAM CLB3
C Define variables, parameters and constants to use
      DOUBLE PRECISION V,CV,ALPHA,BETA,PS,PT,PD,P1,P2,P3,P4,P5,TSA,RHO,
     $                 RH,MU,Q1,Q2,Q3,Q4,Q5,RE,ER,GUESS,CHECK,GAMMA,
     $                 MACH,DIST,IRHO,RHOREF,RHREF,STEP,PI,R,INITIAL,
     $                 LIMIT,ZROT,BAR,TTOT,TTOTA,PSA,PTA,MULT,
     $                 CPT,CPS,CPD,CTSA,CRHO,VX,VY,VZ,CVX,CVY,CVZ,
     $                 Z,CAL1,CAL2,FACTOR,CHORD,BARREF,SVP,VPP,DPP,
     $                 CSVP,CVPP,CDPP,ERROR,MARGIN,IMU,IRE,IMACH,XRATIO
      DOUBLE PRECISION AVER(3)
      INTEGER N,M
      PARAMETER(PI=3.141592654D0,R=286.9D0,GAMMA=1.4D0,CHORD=0.15D0,
     $          FACTOR=6894.75D0,CAL1=0.03181519D0,CAL2=5.249319434D0)
      PARAMETER(INITIAL=0.75D0,LIMIT=0.01D0,MARGIN=0.0001D0,STEP=1.5D0,
     $          ZROT=77.14D0)
C Open data files for I/O
      OPEN(UNIT=10,FILE='./out/press.dat',STATUS='OLD')
      OPEN(UNIT=20,FILE='./out/exit.dat',STATUS='UNKNOWN')
      OPEN(UNIT=30,FILE='./out/aver.dat',STATUS='UNKNOWN')
C Write headers to output file exit.dat
      WRITE(20,100)
  100 FORMAT(1X,'TOTAL PRESSURE (PA)',2X,'C-TOTAL PRESSURE (PA)',2X,
     $       'DYNAMIC PRESSURE (PA)',2X,'C-DYNAMIC PRESSURE (PA)',2X,
     $       'STATIC PRESSURE (PA)',2X,'C-STATIC PRESSURE (PA)',2X,
     $       'PITCH (DEG)',2X,'YAW (DEG)',2X,'VELOCITY (M/SEC)',2X,
     $       'C-VELOCITY (M/SEC)',2X,'C-ZVELOCITY (M/SEC)',2X,
     $       'C-YVELOCITY (M/SEC)',2X,'C-XVELOCITY (M/SEC)',2X,
     $       'C-DENSITY (KG/M**3)',2X,'C-VISCOSITY (KG/M/SEC)',2X,
     $       'C-MACH NUMBER',2X,'C-REYNOLDS NUMBER',2X,
     $       'AXIAL LOCATION (MM)',/)
      N=0
      MULT=PI/180.0D0
C Zero average array
      AVER(1)=0.0D0
      AVER(2)=0.0D0
      AVER(3)=0.0D0
C Read data from iput file press.dat
    4 READ(10,*,END=5)P1,P2,P3,P4,P5,BAR,TTOT,RH,
   $                IRHO,IMU,IRE,IMACH,XRATIO
C Find reference local correction density
      IF(N.EQ.0) THEN
        RHOREF=IRHO
        BARREF=BAR
        RHREF=RH
      ENDIF
C Convert voltage signals into pressure readings
      P1=((P1+CAL1)/CAL2)*FACTOR
      P2=((P2+CAL1)/CAL2)*FACTOR
      P3=((P3+CAL1)/CAL2)*FACTOR
      P4=((P4+CAL1)/CAL2)*FACTOR
      P5=((P5+CAL1)/CAL2)*FACTOR
      M=0
      N=N+1
C Sum inlet flow properties
      AVER(1)=AVER(1)+IMU
      AVER(2)=AVER(2)+IRE
      AVER(3)=AVER(3)+IMACH
      Z=DBLE(N)
      WRITE(*,800) N
  800 FORMAT(/,1X,'STATION NO.',1X,I5,/)
      GUESS=INITIAL*P1
    3 M=M+1
      WRITE(*,850) M,GUESS
  850 FORMAT(1X,'ITERATION NO. ',I5,3X,'PS =',F15.5)
      Q1=(P4-P5)/(P1-GUESS)
      Q2=(P3-P2)/(P1-GUESS)
```

```
C Call calibration subroutines
      CALL ALPHA2(Q2,Q1,ALPHA)
      CALL BETA2(Q2,Q1,BETA)
      CALL Q32(BETA,ALPHA,Q3)
      CALL Q42(BETA,ALPHA,Q4)
      CALL Q52(BETA,ALPHA,Q5)
      CHECK=P1-Q3*((P1-P4)/(2.0D0*Q4)+(P1-P5)/(2.0D0*Q5))
C Check convergence
      ER=CHECK-GUESS
      IF(DABS(ER).LE.LIMIT) THEN
        PS=GUESS
        PT=(P1-PS)/Q3+PS
        PD=(PT-PS)
C Compute exit static temperature
        PSA=BAR+PS
        PTA=BAR+PT
        TTOTA=TTOT+273.15D0
        TSA=(TTOTA)*((PSA/PTA)**((GAMMA-1.0D0)/(GAMMA)))
C Calculate saturation vapor pressure
        SVP=6.1078D0*100.0D0*10.0D0**((7.5D0*(TSA-273.15D0))/
     $      (237.3D0+TSA-273.15D0))
C Calculate vapor partial pressure
        VPP=SVP*RH/100.0D0
C Calculate dry air partial pressure
        DPP=PSA-VPP
C Calculate flow density
        RHO=DPP/(286.9D0*TSA)+VPP/(461.495D0*TSA)
C Calculate air velocity
        V=(2.0D0*(PT-PS)/(RHO))**0.5D0
C Calculate components using turbine notation
        VX=V*DCOS(BETA*MULT)*DCOS(ALPHA*MULT)*DCOS(ZROT*MULT)+
     $     V*DSIN(BETA*MULT)*DSIN(ZROT*MULT)
        VY=V*DCOS(BETA*MULT)*DCOS(ALPHA*MULT)*DSIN(ZROT*MULT)-
     $     V*DSIN(BETA*MULT)*DCOS(ZROT*MULT)
        VZ=V*DCOS(BETA*MULT)*DSIN(ALPHA*MULT)
        DIST=(Z-1.0D0)*STEP
C Locally correct pressures, velocities and flow density
        CPS=PS*(RHOREF/IRHO)*XRATIO
        CPT=PT*(RHOREF/IRHO)*XRATIO
        CPD=CPT-CPS
        CRHO=RHO*(RHOREF/IRHO)
        CV=V*DSQRT(XRATIO)
        CVX=VX*DSQRT(XRATIO)
        CVY=VY*DSQRT(XRATIO)
        CVZ=VZ*DSQRT(XRATIO)
C Iterate to locally correct static temperature
        CTSA=(CPS+BARREF)/(286.9D0*RHOREF)
    9   CSVP=6.1078D0*100.0D0*10.0D0**((7.5D0*(CTSA-273.15D0))/
     $      (237.3D0+CTSA-273.15D0))
        CVPP=CSVP*RHREF/100.0D0
        CDPP=(CPS+BARREF)-CVPP
        ERROR=RHOREF-CDPP/(286.9D0*CTSA)-CVPP/(461.495D0*CTSA)
        IF(DABS(ERROR).LE.MARGIN) GO TO 8
        CTSA=CDPP/((RHOREF-CVPP/(461.495D0*CTSA))*286.9D0)
        GO TO 9
C Calculate locally corrected flow viscosity using Sutherland's equation
    8   MU=1.46D-06*((CTSA**1.5D0)/(CTSA+111.0D0))
C Calculate  locally corrected flow Reynold's and Mach numbers
        RE=(V*CHORD*CRHO)/(MU)
        MACH=(V)/((GAMMA*R*CTSA)**0.5D0)
        WRITE(*,700)
  700   FORMAT(/,1X,'CONVERGENCE SUCCESS',/)
C Interate
      ELSE
          GUESS=CHECK
          GO TO 3
      ENDIF
```

```
C Write data to output file exit.dat
      WRITE(20,200) PT,CPT,PD,CPD,PS,CPS,ALPHA,BETA,V,CV,CVZ,
     $              CVY,CVX,CRHO,MU,MACH,RE,DIST
  200 FORMAT(1X,F19.5,2X,F21.5,2X,F21.5,2X,F23.5,2X,F20.5,2X,F22.5,2X,
     $       F11.3,2X,F9.3,2X,F16.5,2X,F18.5,2X,F19.5,2X,F19.5,2X,
     $       F19.5,2X,F19.10,2X,F22.10,2X,F13.4,2X,F17.1,2X,F19.1)
      GO TO 4
C Write data end correction factors
    5 WRITE(30,*) IRHO,XRATIO
C Average locally corrected inlet flow properties
      IMU=AVER(1)/Z
      IRE=AVER(2)/Z
      IMACH=AVER(3)/Z
C Write averaged values to aver.dat
      WRITE(30,*) RHOREF,IMU,IRE,IMACH
      PRINT*, 'END OF ANALYSIS'
      END
************************************************************************
* This program reads the barometric pressure and humidity and the flow *
* total exit temperature as well as raw data from the inlet prandtl    *
* and static tubes, and analyzes them to calculate the corrected inlet *
* static, total and dynamic pressures, velocity, density and Reynold's *
* number. All calibration coefficients are intended for the MKS 100    *
* TORR pressure transducer (02/08).                                    *
* Input Files : ambnt.dat, new.dat                                     *
* Output Files: inlet.dat                                              *
* Experimental Procedure: Cascade Facility of the DOOSAN I Blade       *
************************************************************************
* Hicham A Chibli - Texas A&M University - 09/05/10                    *
************************************************************************
      PROGRAM PITOT
C Define variables, parameters and constants to use
      DOUBLE PRECISION STEP,I,J,K,RH,RE,BAR,TTC,TTK,TS,RHO,PT,PRT,PS,PD,
     $                 VEL,MU,SVP,VPP,DPP,CPT,CPD,CPS,CTS,AXIAL,FACTOR,
     $                 RHOREF,GAMMA,R,MACH,CSVP,CVPP,CDPP,ERROR,LIMIT,
     $                 RHREF,BARREF,PREFT,PREFS,XRATIO,CAL1,CAL2,
     $                 CHORD,CVEL
      INTEGER COUNTER
      PARAMETER(STEP=1.5D0,FACTOR=6894.75D0,CAL1=0.03181519D0,R=286.9D0,
     $          CAL2=5.249319434D0,CHORD=0.15D0,GAMMA=1.4D0,
     $          LIMIT=1.0D-10)
C Open data files for I/O
      OPEN(UNIT=10,FILE='./out/ambnt.dat',STATUS='OLD')
      OPEN(UNIT=20,FILE='./out/new.dat',STATUS='OLD')
      OPEN(UNIT=30,FILE='./out/inlet.dat',STATUS='UNKNOWN')
      WRITE(30,6)
    6 FORMAT(1X,'TOTAL PRESSURE (PA)',2X,'C-TOTAL PRESSURE (PA)',
     $       2X,'DYNAMIC PRESSURE (PA)',2X,'C-DYNAMIC PRESSURE (PA)',2X,
     $      'STATIC PRESSURE (PA)',2X,'C-STATIC PRESSURE (PA)',2X,
     $      'VELOCITY (M/SEC)',2X,'C-VELOCITY (M/SEC)',2X,
     $      'DENSITY (KG/M**3)',2X,'C-VISCOSITY (KG/M/SEC)',2X,
     $      'C-REYNOLDS NUMBER',2X,'C-MACH NUMBER',2X,
     $      'AXIAL CORRECTION FACTOR',2X,
     $      'AXIAL LOCATION (MM)',/)
      COUNTER=0
C Read data from iput file ambnt.dat and new.dat
    1 READ(10,2,END=3) BAR,TTC,RH
    2 FORMAT(1X,F9.1,1X,F5.2,1X,F4.1)
      READ(20,5) I,J,K
    5 FORMAT(1X,F13.10,1X,F13.10,1X,F13.10)
C Convert voltage signals into pressure readings
      AXIAL=DBLE(COUNTER)*STEP
      PT=((K+CAL1)/CAL2)*FACTOR
      PRT=((J+CAL1)/CAL2)*FACTOR
      PS=((I+CAL1)/CAL2)*FACTOR
      PD=PRT-PS
C Compute exit static temperature
```

```
      TTK=TTC+273.15D0
      TS=TTK*((PS+BAR)/(PRT+BAR))**((GAMMA-1.0D0)/GAMMA)
C Calculate saturation vapor pressure
      SVP=6.1078D0*100.0D0*10.0D0**((7.5D0*(TS-273.15D0))/
     $    (237.3D0+TS-273.15D0))
C Calculate vapor partial pressure
      VPP=SVP*RH/100.0D0
C Calculate dry air partial pressure
      DPP=(PS+BAR)-VPP
C Calculate flow density
      RHO=DPP/(286.9D0*TS)+VPP/(461.495D0*TS)
C Calculate flow velocity
      VEL=(2.0D0*PD/RHO)**0.5D0
C Find the local correction reference density
      IF(COUNTER.EQ.0) THEN
        RHOREF=RHO
        RHREF=RH
        BARREF=BAR
        PREFT=PRT
        PREFS=PS
      ENDIF
C Calculate correction factor for axial velocity
      XRATIO=(RHO/RHOREF)*((((PREFS/PS)**2.0D0+
     $       (PREFT/PRT)**2.0D0)/2.0D0)**0.5D0)
C Locally correct the inlet profile data
      CPS=PS*(RHOREF/RHO)*XRATIO
      CPT=PRT*(RHOREF/RHO)*XRATIO
      CPD=CPT-CPS
      CVEL=VEL*DSQRT(XRATIO)
C Iterate to locally correct static temperature
      CTS=(CPS+BARREF)/(286.9D0*RHOREF)
    9 CSVP=6.1078D0*100.0D0*10.0D0**((7.5D0*(CTS-273.15D0))/
     $     (237.3D0+CTS-273.15D0))
      CVPP=CSVP*RHREF/100.0D0
      CDPP=(CPS+BARREF)-CVPP
      ERROR=RHOREF-CDPP/(286.9D0*CTS)-CVPP/(461.495D0*CTS)
      IF(ABS(ERROR).LE.LIMIT) GO TO 8
      CTS=CDPP/((RHOREF-CVPP/(461.495D0*CTS))*286.9D0)
      GO TO 9
C Calculate locally corrected flow viscosity using Sutherland's equation
    8 MU=1.46D-06*((CTS**1.5D0)/(CTS+111.0D0))
C Calculate locally corrected flow Reynold's and Mach numbers
      RE=(CVEL*CHORD*RHOREF)/(MU)
      MACH=(CVEL)/((GAMMA*R*CTS)**0.5D0)
C Write data to output file inlet.dat
      WRITE(30,7) PRT,CPT,PD,CPD,PS,CPS,VEL,CVEL,RHO,MU,RE,MACH,XRATIO,
     $            AXIAL
    7 FORMAT(1X,F19.2,2X,F21.2,2X,F21.2,2X,F23.2,2X,F20.2,2X,F22.2,2X,
     $       F16.2,2X,F18.2,2X,F17.10,2X,F22.15,2X,F17.2,2X,F13.5,
     $       2X,F23.15,2X,F19.1)
      COUNTER=COUNTER+1
      PRINT*, 'DATA POINT',COUNTER
      GO TO 1
    3 PRINT*, ' END OF ANALYSIS'
      END
************************************************************************
* This program cleans up the voltage data file containing the inlet    *
* and exit pressures and prepares an 8 column data set for analysis.   *
* Input Files : old.dat                                                *
* Output Files: new.dat                                                *
* Experimental Procedure: Cascade Facility of the DOOSAN I Blade       *
************************************************************************
* Hicham A Chibli - Texas A&M University - 09/10/10                    *
************************************************************************
      PROGRAM UTL1
      DOUBLE PRECISION X1,X2,X3,X4,X5,X6,X7,X8
      CHARACTER*10 DSCRD1,DSCRD2,DSCRD3
```

```
C Open data files for I/O
      OPEN(UNIT=200,FILE='./out/new.dat',STATUS='UNKNOWN')
      OPEN(UNIT=100,FILE='./in/old.dat',STATUS='OLD')
C Reading characters to discard
    3 READ(100,*,END=6) DSCRD1,DSCRD2,DSCRD3
      READ(100,*) X1,X2,X3,X4,X5,X6,X7,X8
      WRITE(200,5) X1,X2,X3,X4,X5,X6,X7,X8
    5 FORMAT(1X,F13.10,1X,F13.10,1X,F13.10,1X,F13.10,1X,F13.10,1X,
     $       F13.10,1X,F13.10,1X,F13.10)
      PRINT*, DSCRD1,DSCRD2,DSCRD3
      PRINT*, X1,X2,X3,X4,X5,X6,X7,X8
      GO TO 3
    6 PRINT*, 'END OF ANALYSIS'
      END
************************************************************************
* This program reads the total exit temperature (C), the barometric    *
* pressure (inch HG) and the relative humidity (%) with the            *
* corresponding time of readings, then interpolates the data using a   *
* shape preserving cubic spline to generate smooth temperature,        *
* pressure and humidity profiles to be used in density calculations.   *
* Input Files : fpt.dat, fht.dat                                       *
* Output Files: opt.dat, orh.dat, ambnt.dat                            *
* Experimental Procedure: Cascade Facility of the DOOSAN I Blade       *
************************************************************************
* Hicham A Chibli - Texas A&M University - 09/10/10                    *
************************************************************************
      PROGRAM UTL2
C Define variables, parameters and constants to use
      DOUBLE PRECISION STEP,EXACT,BASE,J,K
      INTEGER TOTAL,HOUR,MINUTE,PTCNT,RHCNT,I,ITYPE,IDER
      PARAMETER(STEP=1.5D0,TOTAL=195,ITYPE=3,IDER=0)
      DOUBLE PRECISION PB(TOTAL),EPB(TOTAL),TT(TOTAL),ETT(TOTAL),
     $                 RH(TOTAL),ERH(TOTAL),PT(TOTAL),HT(TOTAL),
     $                 TME(TOTAL),LOCATE(TOTAL),PTLCT(TOTAL),
     $                 RHLCT(TOTAL)
C Read data from iput files fpt.dat and fht.dat
      OPEN(UNIT=10,FILE='./in/fpt.dat',STATUS='OLD')
      OPEN(UNIT=20,FILE='./in/fht.dat',STATUS='OLD')
      PTCNT=1
    1 READ(10,2,END=3) HOUR,MINUTE,J,TT(PTCNT)
    2 FORMAT(I2,I2,1X,F5.2,1X,F4.1)
      PB(PTCNT)=3386.388D0*J
      EXACT=DBLE(HOUR)*60.D0+DBLE(MINUTE)
      IF(PTCNT.EQ.1) BASE=EXACT
      PT(PTCNT)=EXACT-BASE
      PTCNT=PTCNT+1
      GO TO 1
    3 RHCNT=1
    6 READ(20,4,END=5) HOUR,MINUTE,RH(RHCNT)
    4 FORMAT(I2,I2,1X,F4.1)
      EXACT=DBLE(HOUR)*60.D0+DBLE(MINUTE)
      IF(PTCNT.EQ.1) BASE=EXACT
      HT(RHCNT)=EXACT-BASE
      RHCNT=RHCNT+1
      GO TO 6
    5 PRINT*, 'END READING DATA'
C Correct array sizes
      PTCNT=PTCNT-1
      RHCNT=RHCNT-1
C Populate location arrays
      EXACT=(PT(PTCNT)+HT(RHCNT))/2.D0
      DO 9 I=1,TOTAL
           TME(I)=DBLE(I-1)*EXACT/DBLE(TOTAL-1)
           LOCATE(I)=DBLE(I-1)*STEP
    9 CONTINUE
      DO 14 I=1,PTCNT
            PTLCT(I)=(PT(I)/PT(PTCNT))*STEP*DBLE(TOTAL-1)
```

```
   14 CONTINUE
      DO 15 I=1,RHCNT
            RHLCT(I)=(PT(I)/PT(RHCNT))*STEP*DBLE(TOTAL-1)
   15 CONTINUE
C Compute cubic spline interpolant using IMSL subroutine DSPLEZ
      CALL DSPLEZ(PTCNT,PT,PB,ITYPE,IDER,TOTAL,TME,EPB)
      CALL DSPLEZ(PTCNT,PT,TT,ITYPE,IDER,TOTAL,TME,ETT)
      CALL DSPLEZ(RHCNT,HT,RH,ITYPE,IDER,TOTAL,TME,ERH)
C Write data to output files ambnt.dat,opt.dat and orh.dat
      OPEN(UNIT=30,FILE='./out/ambnt.dat',STATUS='UNKNOWN')
      OPEN(UNIT=40,FILE='./out/opt.dat',STATUS='UNKNOWN')
      OPEN(UNIT=50,FILE='./out/orh.dat',STATUS='UNKNOWN')
      DO 7 I=1,TOTAL
         WRITE(30,8) EPB(I),ETT(I),ERH(I),LOCATE(I)
    8    FORMAT(1X,F9.1,1X,F5.2,1X,F4.1,1X,F5.1)
    7 CONTINUE
      DO 10 I=1,PTCNT
         WRITE(40,11) PB(I),TT(I),PTLCT(I)
   11    FORMAT(1X,F9.1,1X,F5.2,1X,F5.1)
   10 CONTINUE
      DO 12 I=1,RHCNT
         WRITE(50,13) RH(I),RHLCT(I)
   13    FORMAT(1X,F4.1,1X,F5.1)
   12 CONTINUE
      PRINT*, ' END OF ANALYSIS'
      END
************************************************************************
* This program reads the barometric pressure and the total exit        *
* temperature as well as raw data for the five hole probe and the flow *
* inlet density to prepare an intermediate file for exit flow full     *
* analysis.                                                            *
* Input Files : ambnt.dat, inlet.dat, new.dat                          *
* Output Files: press.dat                                              *
* Experimental Procedure: Cascade Facility of the DOOSAN I Blade       *
************************************************************************
* Hicham A Chibli - Texas A&M University - 09/10/10                    *
************************************************************************
      PROGRAM UTL3
C Define variables, parameters and constants to use
      INTEGER TOTAL,J
      PARAMETER(TOTAL=195)
      DOUBLE PRECISION P1(TOTAL),P2(TOTAL),P3(TOTAL),P4(TOTAL),
     $                 P5(TOTAL),BAR(TOTAL),TOT(TOTAL),RH(TOTAL),
     $                 RHO(TOTAL),MU(TOTAL),RE(TOTAL),MACH(TOTAL),
     $                 XRATIO(TOTAL)
C Read data from iput files ambnt.dat, new.dat and inlet.dat
      OPEN(UNIT=10,FILE='./out/inlet.dat',STATUS='OLD')
      OPEN(UNIT=20,FILE='./out/new.dat',STATUS='OLD')
      OPEN(UNIT=30,FILE='./out/ambnt.dat',STATUS='OLD')
      READ(10,*)
      READ(10,*)
      DO 3 J=1,TOTAL
         READ(10,2) RHO(J),MU(J),RE(J),MACH(J),XRATIO(J)
    2    FORMAT(178X,F17.10,2X,F22.15,2X,F17.2,2X,F13.5,2X,F23.15)
    3 CONTINUE
      DO 4 J=1,TOTAL
         READ(20,5) P1(J),P2(J),P3(J),P4(J),P5(J)
    5    FORMAT(43X,F13.10,1X,F13.10,1X,F13.10,1X,F13.10,1X,F13.10)
    4 CONTINUE
      DO 6 J=1,TOTAL
         READ(30,7) BAR(J),TOT(J),RH(J)
    7    FORMAT(1X,F9.1,1X,F5.2,1X,F4.1)
    6    CONTINUE
C Write data to output file press.dat
      OPEN(UNIT=40,FILE='./out/press.dat',STATUS='UNKNOWN')
      DO 8 J=1,TOTAL
         WRITE(40,9) P1(J),P2(J),P3(J),P4(J),P5(J),BAR(J),TOT(J),RH(J),
     $               RHO(J),MU(J),RE(J),MACH(J),XRATIO(J)
    9    FORMAT(1X,F13.10,1X,F13.10,1X,F13.10,1X,F13.10,1X,F13.10,1X,
     $          F9.1,1X,F5.2,1X,F4.1,1X,F17.10,1X,F22.15,1X,F17.2,1X,
     $          F13.5,1X,F23.15)
    8 CONTINUE
      PRINT*, ’END OF ANALYSIS’
      END
************************************************************************
*  This code automates the analysis of Cascade-I numerical results     *
*  and computes the loss coefficients as well as the forces and        *
*  prepares the output plot data files.                                *
************************************************************************
*  Hicham A Chibli - Texas A&M University - 08/2010                    *
************************************************************************
      PROGRAM NCAS1
C define variables and parameters
      REAL PI,CHORD,PITCH,EANGLE,INCID,
     $     REREF,REREFX,VREF,VREFX,RHOREF
      INTEGER W1,W2,L1,L2,C
      PARAMETER(W1=10,W2=2,L1=1000,L2=1000,C=2,PI=3.14159265,
     $          EANGLE=77.14,CHORD=0.15,PITCH=0.069,
     $          REREF=150000.0,VREF=15.0,
     $          REREFX=111852.0,VREFX=11.442,
     $          RHOREF=1.203)
      INTEGER N,M,K,BDE,IPE,OPE,FPE,NUM,OW,OS,FW,FS,OE,FE,OFS,FFS
      REAL REIN,REINX,RHOIN,VIN,VINX,MUIN,PIN,
     $     POLSS,PFLSS,POLSSX,PFLSSX,MLOSS,PMLOSS,
     $     MIMF,MOMF,MFMF,CIMF,COMF,CFMF,
     $     MOFT,MFFT,COFT,CFFT,BFT,CBFT,
     $     MOFN,MFFN,COFN,CFFN,
     $     IMXM,IMYM,OMXM,OMYM,FMXM,FMYM,
     $     IMXC,IMYC,OMXC,OMYC,FMXC,FMYC,
     $     TEMP,BMIN,BMAX,PMAX,PMIN,
     $     OGAP,FGAP
      REAL INP(C+1,L1,W1),OUTP(C+1,L1,W1),FARP(C+1,L1,W1),
     $     EINP(C+1,2*L1,W1),EOUTP(C+1,2*L1,W1),EFARP(C+1,2*L1,W1),
     $     BD(L2,W2),CP(L2,W2),
     $     AINP(C+1,W1),AOUTP(C+1,W1),AFARP(C+1,W1),
     $     OLOSS(C+1),FLOSS(C+1),
     $     TB(W2),TP(W1)
      CHARACTER LOCATE*25,CASNUM*25,NEW*50
C open file list and directory
      OPEN(UNIT=19,FILE=’./list/list.txt’,STATUS=’OLD’)
      OPEN(UNIT=20,FILE=’./list/incidence.txt’,STATUS=’OLD’)
      OPEN(UNIT=21,FILE=’./summary/all.dat’,STATUS=’UNKNOWN’)
      OPEN(UNIT=22,FILE=’./summary/cnplot.dat’,STATUS=’UNKNOWN’)
      OPEN(UNIT=23,FILE=’./summary/ccp.dat’,STATUS=’UNKNOWN’)
      OPEN(UNIT=24,FILE=’./summary/closs.dat’,STATUS=’UNKNOWN’)
C write header to all-summary files
      WRITE(21,*) ’VARIABLES="i [deg]",’,
     $            ’"‘z_o_u_t___m_e_a_s_u_r_e_d",’,
     $            ’"‘z_o_u_t___P_f_e_i_l",’,
     $            ’"‘z_o_u_t___P_f_e_i_l___x",’,
     $            ’"‘z_f_a_r___m_e_a_s_u_r_e_d",’,
     $            ’"‘z_f_a_r___P_f_e_i_l",’,
     $            ’"‘z_f_a_r___P_f_e_i_l___x",’,
     $            ’"‘z_m_i_x___m_e_a_s_u_r_e_d",’,
     $            ’"‘z_m_i_x___c_o_r_r_e_c_t_e_d",’,
     $            ’"Drag/Lift_o_u_t___m_e_a_s_u_r_e_d",’,
     $            ’"Drag/Lift_o_u_t___c_o_r_r_e_c_t_e_d",’,
     $            ’"Drag/Lift_f_a_r___m_e_a_s_u_r_e_d",’,
     $            ’"Drag/Lift_f_a_r___c_o_r_r_e_c_t_e_d",’,
     $            ’"Re_x___m_e_a_s_u_r_e_d","Re_x___r_e_f",’,
     $            ’"‘r_m_e_a_s_u_r_e_d [kg/m^3]",’,
     $            ’"‘r_c_o_r_r_e_c_t_e_d [kg/m^3]",’,
     $            ’"mflow_m_e_a_s_u_r_e_d [kg/sec]",’,
     $            ’"mflow_c_o_r_r_e_c_t_e_d [kg/sec]",’,
     $            '"F_T___m_e_a_s_u_r_e_d [N]",',
     $            '"F_T___c_o_r_r_e_c_t_e_d [N]"'
      WRITE(24,*) 'VARIABLES="i [deg]",','"`z"'
      WRITE(24,*) 'ZONE T=" numerical"'
      WRITE(22,*) 'VARIABLES="Y",',
     $            '"P_t_n",',
     $            '"P_s_n",',
     $            '"P_d_n",',
     $            '"V_n"',
     $            '"V_x_n"',
     $            '"V_y_n"'
      WRITE(23,*) 'VARIABLES="X",','"C_p"'
C read list of files and working directory
      NUM=0
      READ(20,*)
      READ(19,*)
      READ(19,*) LOCATE
      READ(19,*)
   30 READ(19,*,END=31) CASNUM
      READ(20,*) INCID
      PRINT*
      NUM=NUM+1
C trim blanks from strings
      CALL CLEAN(LOCATE,N)
      CALL CLEAN(CASNUM,M)
      NEW='./'//LOCATE(1:N)//'/'//CASNUM(1:M)
      CALL CLEAN(NEW,K)
      PRINT*, '***************************************************'
      PRINT*
      PRINT*, 'CASE NUMBER: ',CASNUM(1:M)
C write case header to all-summary files
      WRITE(23,93) INCID
      WRITE(22,93) INCID
   93 FORMAT('ZONE T="num.  i = ',F5.1,' ^o"')
C open files for input
      OPEN(UNIT=15,FILE=NEW(1:K)//'/in/blade.dat',STATUS='OLD')
      OPEN(UNIT=16,FILE=NEW(1:K)//'/in/inlet.dat',STATUS='OLD')
      OPEN(UNIT=17,FILE=NEW(1:K)//'/in/outlet.dat',STATUS='OLD')
      OPEN(UNIT=18,FILE=NEW(1:K)//'/in/far.dat',STATUS='OLD')
C open files for output
      OPEN(UNIT=25,FILE=NEW(1:K)//'/out/cp.dat',STATUS='UNKNOWN')
      OPEN(UNIT=26,FILE=NEW(1:K)//'/out/cplt.dat',STATUS='UNKNOWN')
      OPEN(UNIT=27,FILE=NEW(1:K)//'/out/ncplt.dat',STATUS='UNKNOWN')
      OPEN(UNIT=29,FILE=NEW(1:K)//'/out/case.dat',STATUS='UNKNOWN')
C zero profile data arrays used
      DO 1 N=1,C+1
         DO 2 M=1,L1
            DO 3 K=1,W1
               INP(N,M,K)=0.0
               OUTP(N,M,K)=0.0
               FARP(N,M,K)=0.0
    3       CONTINUE
    2    CONTINUE
    1 CONTINUE
      DO 95 N=1,C+1
         DO 94 M=1,2*L1
            DO 96 K=1,W1
               EINP(N,M,K)=0.0
               EOUTP(N,M,K)=0.0
               EFARP(N,M,K)=0.0
   96       CONTINUE
   94    CONTINUE
   95 CONTINUE
C zero average data arrays used
      DO 4 N=1,C+1
         DO 5 M=1,W1
            AINP(N,M)=0.0
            AOUTP(N,M)=0.0
            AFARP(N,M)=0.0
    5    CONTINUE
    4 CONTINUE
C zero blade pressure ditribution arrays used
      DO 6 N=1,L2
         DO 7 M=1,W2
            BD(N,M)=0.0
    7    CONTINUE
    6 CONTINUE
C zero blade cp ditribution array used
      DO 8 N=1,L2
         DO 9 M=1,W2
            CP(N,M)=0.0
    9    CONTINUE
    8 CONTINUE
C zero loss and array used
      DO 10 N=1,C+1
         OLOSS(N)=0.0
         FLOSS(N)=0.0
   10 CONTINUE
C zero counters
      BDE=0
      IPE=0
      OPE=0
      FPE=0
C read the blade pressure ditribution input data and find the bounds
      BMIN=10000.0
      BMAX=-10000.0
   11 READ(15,*,END=12) TB
      BDE=BDE+1
      DO 13 N=1,W2
         BD(BDE,N)=TB(N)
         IF(N.EQ.1) THEN
           IF(BD(BDE,N).GT.BMAX) BMAX=BD(BDE,N)
           IF(BD(BDE,N).LT.BMIN) BMIN=BD(BDE,N)
         ENDIF
   13 CONTINUE
C repeat
      GO TO 11
   12 BDE=BDE+1
      BD(BDE,1)=BD(1,1)
      BD(BDE,2)=BD(1,2)
      PRINT*
      PRINT*, '**************************************************'
      PRINT*, 'READING BLADE PRESSURE DISTRIBUTION DATA COMPLETE'
      PRINT*, 'NUMBER OF POINTS PER SET:         ',BDE
      PRINT*, 'MAXIMUM NUMBER OF POINTS PER SET: ',L2
      PRINT*
C read the inlet profile input data
C M
   14 READ(16,*,END=15) TP
      IPE=IPE+1
      DO 16 N=1,W1
         INP(1,IPE,N)=TP(N)
   16 CONTINUE
C reset the 8th column to handle dynamic pressure head
      INP(1,IPE,8)=INP(1,IPE,2)-INP(1,IPE,3)
C repeat
      GO TO 14
   15 PRINT*, 'READING INLET PROFILE DATA COMPLETE'
      PRINT*, 'NUMBER OF POINTS PER SET:         ',IPE
      PRINT*, 'MAXIMUM NUMBER OF POINTS PER SET: ',L1
      PRINT*
C read the outlet profile input data
C M
   17 READ(17,*,END=18) TP
```

```
      OPE=OPE+1
      DO 19 N=1,W1
         OUTP(1,OPE,N)=TP(N)
   19 CONTINUE
C reset the 8th column to handle dynamic pressure head
      OUTP(1,OPE,8)=OUTP(1,OPE,2)-OUTP(1,OPE,3)
C repeat
      GO TO 17
   18 PRINT*, ’READING OUTLET PROFILE DATA COMPLETE’
      PRINT*, ’NUMBER OF POINTS PER SET:         ’,OPE
      PRINT*, ’MAXIMUM NUMBER OF POINTS PER SET: ’,L1
      PRINT*
C read the far profile input data
C M
   20 READ(18,*,END=21) TP
      FPE=FPE+1
      DO 22 N=1,W1
         FARP(1,FPE,N)=TP(N)
   22 CONTINUE
C reset the 8th column to handle dynamic pressure head
      FARP(1,FPE,8)=FARP(1,FPE,2)-FARP(1,FPE,3)
C repeat
      GO TO 20
   21 PRINT*, ’READING FAR PROFILE DATA COMPLETE’
      PRINT*, ’NUMBER OF POINTS PER SET:         ’,FPE
      PRINT*, ’MAXIMUM NUMBER OF POINTS PER SET: ’,L1
      PRINT*
C manipulate the profile data to establish periodic conditions
      DO 58 K=2,W1
            INP(1,1,K)=(INP(1,1,K)+INP(1,IPE,K))/2.0
            INP(1,IPE,K)=INP(1,1,K)
            OUTP(1,1,K)=(OUTP(1,1,K)+OUTP(1,OPE,K))/2.0
            OUTP(1,OPE,K)=OUTP(1,1,K)
            FARP(1,1,K)=(FARP(1,1,K)+FARP(1,FPE,K))/2.0
            FARP(1,FPE,K)=FARP(1,1,K)
   58 CONTINUE
C sort the far arrays in ascending order of axial location
      DO 68 K=1,IPE-1
         DO 69 N=K+1,IPE
            IF(INP(1,K,1).GT.INP(1,N,1)) THEN
              DO 70 M=1,W1
                 TEMP=INP(1,K,M)
                 INP(1,K,M)=INP(1,N,M)
                 INP(1,N,M)=TEMP
   70         CONTINUE
            ENDIF
   69    CONTINUE
   68 CONTINUE
C sort the out arrays in ascending order of axial location
      DO 62 K=1,OPE-1
         DO 63 N=K+1,OPE
            IF(OUTP(1,K,1).GT.OUTP(1,N,1)) THEN
              DO 64 M=1,W1
                 TEMP=OUTP(1,K,M)
                 OUTP(1,K,M)=OUTP(1,N,M)
                 OUTP(1,N,M)=TEMP
   64         CONTINUE
            ENDIF
   63    CONTINUE
   62 CONTINUE
C sort the far arrays in ascending order of axial location
      DO 65 K=1,FPE-1
         DO 66 N=K+1,FPE
            IF(FARP(1,K,1).GT.FARP(1,N,1)) THEN
              DO 67 M=1,W1
                 TEMP=FARP(1,K,M)
                 FARP(1,K,M)=FARP(1,N,M)
                 FARP(1,N,M)=TEMP
   67         CONTINUE
            ENDIF
   66    CONTINUE
   65 CONTINUE
C average inlet profile data
C M
      CALL INTG(INP,C+1,L1,W1,1,IPE,2,AINP(1,2),TEMP,MIMF)
      CALL INTG(INP,C+1,L1,W1,1,IPE,3,AINP(1,3),TEMP,MIMF)
      CALL INTG(INP,C+1,L1,W1,1,IPE,4,AINP(1,4),TEMP,MIMF)
      CALL INTG(INP,C+1,L1,W1,1,IPE,5,AINP(1,5),TEMP,MIMF)
      CALL INTG(INP,C+1,L1,W1,1,IPE,6,AINP(1,6),TEMP,MIMF)
      CALL INTG(INP,C+1,L1,W1,1,IPE,8,AINP(1,8),TEMP,MIMF)
      CALL INTG(INP,C+1,L1,W1,1,IPE,9,AINP(1,9),IMXM,MIMF)
      CALL INTG(INP,C+1,L1,W1,1,IPE,10,AINP(1,10),IMYM,MIMF)
      PIN=AINP(1,2)
      VIN=AINP(1,4)
      VINX=AINP(1,9)
      RHOIN=AINP(1,5)
      MUIN=AINP(1,6)
      REIN=CHORD*RHOIN*VIN/MUIN
      REINX=CHORD*RHOIN*VINX/MUIN
C average outlet profile data
C M
      CALL INTG(OUTP,C+1,L1,W1,1,OPE,2,AOUTP(1,2),TEMP,MOMF)
      CALL INTG(OUTP,C+1,L1,W1,1,OPE,3,AOUTP(1,3),TEMP,MOMF)
      CALL INTG(OUTP,C+1,L1,W1,1,OPE,4,AOUTP(1,4),TEMP,MOMF)
      CALL INTG(OUTP,C+1,L1,W1,1,OPE,5,AOUTP(1,5),TEMP,MOMF)
      CALL INTG(OUTP,C+1,L1,W1,1,OPE,6,AOUTP(1,6),TEMP,MOMF)
      CALL INTG(OUTP,C+1,L1,W1,1,OPE,8,AOUTP(1,8),TEMP,MOMF)
      CALL INTG(OUTP,C+1,L1,W1,1,OPE,9,AOUTP(1,9),OMXM,MOMF)
      CALL INTG(OUTP,C+1,L1,W1,1,OPE,10,AOUTP(1,10),OMYM,MOMF)
C average far profile data
C M
      CALL INTG(FARP,C+1,L1,W1,1,FPE,2,AFARP(1,2),TEMP,MFMF)
      CALL INTG(FARP,C+1,L1,W1,1,FPE,3,AFARP(1,3),TEMP,MFMF)
      CALL INTG(FARP,C+1,L1,W1,1,FPE,4,AFARP(1,4),TEMP,MFMF)
      CALL INTG(FARP,C+1,L1,W1,1,FPE,5,AFARP(1,5),TEMP,MFMF)
      CALL INTG(FARP,C+1,L1,W1,1,FPE,6,AFARP(1,6),TEMP,MFMF)
      CALL INTG(FARP,C+1,L1,W1,1,FPE,8,AFARP(1,8),TEMP,MFMF)
      CALL INTG(FARP,C+1,L1,W1,1,FPE,9,AFARP(1,9),FMXM,MFMF)
      CALL INTG(FARP,C+1,L1,W1,1,FPE,10,AFARP(1,10),FMYM,MFMF)
C compute cp distribution around blade
C M
      DO 23 N=1,BDE
         CP(N,1)=(BD(N,1)-BMIN)/(BMAX-BMIN)
         CP(N,2)=(BD(N,2)-PIN)/PIN
   23 CONTINUE
C perform correction of inlet profiles
C this is done be specifying  axial-inlet reference
C parameters and correcting all the values accordingly
      DO 24 N=1,IPE
C C1
         INP(2,N,1)=INP(1,N,1)
         INP(2,N,2)=INP(1,N,2)*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
         INP(2,N,3)=INP(1,N,3)*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
         INP(2,N,8)=INP(1,N,8)*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
         INP(2,N,5)=INP(1,N,5)*RHOREF/RHOIN
         INP(2,N,4)=INP(1,N,4)*VREFX/VINX
         INP(2,N,9)=INP(1,N,9)*VREFX/VINX
         INP(2,N,10)=INP(1,N,10)*VREFX/VINX
   24 CONTINUE
C perform correction of outlet profiles
      DO 25 N=1,OPE
C C1
         OUTP(2,N,1)=OUTP(1,N,1)
         OUTP(2,N,2)=OUTP(1,N,2)*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
         OUTP(2,N,3)=OUTP(1,N,3)*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
         OUTP(2,N,8)=OUTP(1,N,8)*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
         OUTP(2,N,5)=OUTP(1,N,5)*RHOREF/RHOIN
         OUTP(2,N,4)=OUTP(1,N,4)*VREFX/VINX
         OUTP(2,N,9)=OUTP(1,N,9)*VREFX/VINX
         OUTP(2,N,10)=OUTP(1,N,10)*VREFX/VINX
   25 CONTINUE
C perform correction of far profiles
      DO 26 N=1,FPE
C C1
         FARP(2,N,1)=FARP(1,N,1)
         FARP(2,N,2)=FARP(1,N,2)*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
         FARP(2,N,3)=FARP(1,N,3)*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
         FARP(2,N,8)=FARP(1,N,8)*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
         FARP(2,N,5)=FARP(1,N,5)*RHOREF/RHOIN
         FARP(2,N,4)=FARP(1,N,4)*VREFX/VINX
         FARP(2,N,9)=FARP(1,N,9)*VREFX/VINX
         FARP(2,N,10)=FARP(1,N,10)*VREFX/VINX
   26 CONTINUE
C average corrected inlet profile data
C C1
      CALL INTG(INP,C+1,L1,W1,2,IPE,2,AINP(2,2),TEMP,CIMF)
      CALL INTG(INP,C+1,L1,W1,2,IPE,3,AINP(2,3),TEMP,CIMF)
      CALL INTG(INP,C+1,L1,W1,2,IPE,4,AINP(2,4),TEMP,CIMF)
      CALL INTG(INP,C+1,L1,W1,2,IPE,5,AINP(2,5),TEMP,CIMF)
      CALL INTG(INP,C+1,L1,W1,2,IPE,8,AINP(2,8),TEMP,CIMF)
      CALL INTG(INP,C+1,L1,W1,2,IPE,9,AINP(2,9),IMXC,CIMF)
      CALL INTG(INP,C+1,L1,W1,2,IPE,10,AINP(2,10),IMYC,CIMF)
C average corrected outlet profile data
C C1
      CALL INTG(OUTP,C+1,L1,W1,2,OPE,2,AOUTP(2,2),TEMP,COMF)
      CALL INTG(OUTP,C+1,L1,W1,2,OPE,3,AOUTP(2,3),TEMP,COMF)
      CALL INTG(OUTP,C+1,L1,W1,2,OPE,5,AOUTP(2,5),TEMP,COMF)
      CALL INTG(OUTP,C+1,L1,W1,2,OPE,8,AOUTP(2,8),TEMP,COMF)
      CALL INTG(OUTP,C+1,L1,W1,2,OPE,9,AOUTP(2,9),OMXC,COMF)
      CALL INTG(OUTP,C+1,L1,W1,2,OPE,10,AOUTP(2,10),OMYC,COMF)
C average corrected far profile data
C C1
      CALL INTG(FARP,C+1,L1,W1,2,FPE,2,AFARP(2,2),TEMP,CFMF)
      CALL INTG(FARP,C+1,L1,W1,2,FPE,3,AFARP(2,3),TEMP,CFMF)
      CALL INTG(FARP,C+1,L1,W1,2,FPE,5,AFARP(2,5),TEMP,CFMF)
      CALL INTG(FARP,C+1,L1,W1,2,FPE,8,AFARP(2,8),TEMP,CFMF)
      CALL INTG(FARP,C+1,L1,W1,2,FPE,9,AFARP(2,9),FMXC,CFMF)
      CALL INTG(FARP,C+1,L1,W1,2,FPE,10,AFARP(2,10),FMYC,CFMF)
C compute out loss coefficients
      OLOSS(1)=(AINP(1,2)-AOUTP(1,2))/AOUTP(1,8)
      OLOSS(2)=(AINP(2,2)-AOUTP(2,2))/AOUTP(2,8)
C compute far loss coefficients
      FLOSS(1)=(AINP(1,2)-AFARP(1,2))/AFARP(1,8)
      FLOSS(2)=(AINP(2,2)-AFARP(2,2))/AFARP(2,8)
C compute the PFEIL corrected loss coefficient
      POLSS=OLOSS(1)*(REIN/REREF)**0.2
      PFLSS=FLOSS(1)*(REIN/REREF)**0.2
      POLSSX=OLOSS(1)*(REINX/REREFX)**0.2
      PFLSSX=FLOSS(1)*(REINX/REREFX)**0.2
C compute the mixing loss
      MLOSS=FLOSS(1)-OLOSS(1)
      PMLOSS=0.5*(PFLSSX-POLSSX+PFLSS-POLSS)
C compute the measured blade reaction forces (normal and tengential)
      MOFN=ABS((IMXM-OMXM)-(AINP(1,3)-AOUTP(1,3))*PITCH)
      MFFN=ABS((IMXM-FMXM)-(AINP(1,3)-AFARP(1,3))*PITCH)
      MOFT=ABS(IMYM-OMYM)
      MFFT=ABS(IMYM-FMYM)
C compute the corrected blade reaction forces (normal and tengential)
      COFN=ABS((IMXC-OMXC)-(AINP(2,3)-AOUTP(2,3))*PITCH)
      CFFN=ABS((IMXC-FMXC)-(AINP(2,3)-AFARP(2,3))*PITCH)
      COFT=ABS(IMYC-OMYC)
      CFFT=ABS(IMYC-FMYC)
C find the center of the wake in the out profile
      TEMP=100000.0
      DO 47 N=1,OPE
         IF(OUTP(1,N,2).LT.TEMP) THEN
           TEMP=OUTP(1,N,2)
           OW=N
         ENDIF
   47 CONTINUE
C find the center of the wake in the far profile
      TEMP=100000.0
      DO 48 N=1,FPE
         IF(FARP(1,N,2).LT.TEMP) THEN
           TEMP=FARP(1,N,2)
           FW=N
         ENDIF
   48 CONTINUE
C non-dimensionalize the profile at inlet plane
      PMAX=-10000.0
      PMIN=10000.0
      DO 32 N=1,IPE
         IF(INP(1,N,1).GT.PMAX) PMAX=INP(1,N,1)
         IF(INP(1,N,1).LT.PMIN) PMIN=INP(1,N,1)
   32 CONTINUE
      DO 33 N=1,IPE
         INP(1,N,1)=(INP(1,N,1)-PMIN)/(PMAX-PMIN)
         INP(2,N,1)=INP(1,N,1)
         INP(3,N,1)=INP(1,N,1)
         INP(3,N,2)=INP(2,N,2)/AINP(2,2)
         INP(3,N,3)=INP(2,N,3)/AINP(2,3)
         INP(3,N,4)=INP(2,N,4)/AINP(2,4)
         INP(3,N,8)=INP(2,N,8)/AINP(2,8)
         INP(3,N,9)=INP(2,N,9)/AINP(2,9)
         INP(3,N,10)=INP(2,N,10)/AINP(2,10)
   33 CONTINUE
C non-dimensionalize the location array of the outlet plane
      PMAX=-10000.0
      PMIN=10000.0
      DO 36 N=1,OPE
         IF(OUTP(1,N,1).GT.PMAX) PMAX=OUTP(1,N,1)
         IF(OUTP(1,N,1).LT.PMIN) PMIN=OUTP(1,N,1)
   36 CONTINUE
      DO 37 N=1,OPE
         OUTP(1,N,1)=(OUTP(1,N,1)-PMIN)/(PMAX-PMIN)
         OUTP(2,N,1)=OUTP(1,N,1)
         OUTP(3,N,1)=OUTP(1,N,1)
   37 CONTINUE
C find the free-stream location for non-dimensioning the out profiles
      OGAP=(OUTP(1,OW,1)-0.5)
      IF(OGAP.GE.0.0) THEN
        DO 84 K=1,OPE
           IF(OUTP(1,K,1)-OGAP.GE.0.0) THEN
             OFS=K
             GO TO 85
           ENDIF
   84   CONTINUE
      ELSE
          DO 86 K=1,OPE
             IF(OUTP(1,K,1)-OGAP.GE.1.0) THEN
             OFS=K
             GO TO 85
             ENDIF
   86     CONTINUE
   85 ENDIF
C non-dimensionalize the profile at the out plane
      DO 87 N=1,OPE
         OUTP(3,N,2)=OUTP(2,N,2)/OUTP(2,OFS,2)
         OUTP(3,N,3)=OUTP(2,N,3)/OUTP(2,OFS,3)
         OUTP(3,N,4)=OUTP(2,N,4)/OUTP(2,OFS,4)
         OUTP(3,N,8)=OUTP(2,N,8)/OUTP(2,OFS,8)
         OUTP(3,N,9)=OUTP(2,N,9)/OUTP(2,OFS,9)
         OUTP(3,N,10)=OUTP(2,N,10)/OUTP(2,OFS,10)
   87 CONTINUE
C non-dimensionalize the location array of the profile at far plane
      PMAX=-10000.0
      PMIN=10000.0
      DO 38 N=1,FPE
         IF(FARP(1,N,1).GT.PMAX) PMAX=FARP(1,N,1)
         IF(FARP(1,N,1).LT.PMIN) PMIN=FARP(1,N,1)
   38 CONTINUE
      DO 39 N=1,FPE
         FARP(1,N,1)=(FARP(1,N,1)-PMIN)/(PMAX-PMIN)
         FARP(2,N,1)=FARP(1,N,1)
         FARP(3,N,1)=FARP(1,N,1)
   39 CONTINUE
C find the free-stream location for non-dimensioning the far profiles
      FGAP=(FARP(1,FW,1)-0.5)
      IF(FGAP.GE.0.0) THEN
        DO 89 K=1,FPE
           IF(FARP(1,K,1)-FGAP.GE.0.0) THEN
             FFS=K
             GO TO 91
           ENDIF
   89   CONTINUE
      ELSE
          DO 90 K=1,FPE
             IF(FARP(1,K,1)-FGAP.GE.1.0) THEN
             FFS=K
             GO TO 91
             ENDIF
   90     CONTINUE
   91 ENDIF
C non-dimensionalize the profile at the far plane
      DO 92 N=1,FPE
         FARP(3,N,2)=FARP(2,N,2)/FARP(2,FFS,2)
         FARP(3,N,3)=FARP(2,N,3)/FARP(2,FFS,3)
         FARP(3,N,4)=FARP(2,N,4)/FARP(2,FFS,4)
         FARP(3,N,8)=FARP(2,N,8)/FARP(2,FFS,8)
         FARP(3,N,9)=FARP(2,N,9)/FARP(2,FFS,9)
         FARP(3,N,10)=FARP(2,N,10)/FARP(2,FFS,10)
   92 CONTINUE
C compute the tangential pressure force on the blade
      BFT=0.0
      CBFT=0.0
      DO 27 N=1,BDE-1
            BFT=BFT+(BD(N+1,1)-BD(N,1))*(BD(N+1,2)+BD(N,2))*0.5
   27 CONTINUE
      BFT=ABS(BFT)
      CBFT=BFT*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
C print out averaged corrected data to summary file
      WRITE(29,*) '***************************************************'
      WRITE(29,*) 'CORRECTION PARAMETERS'
      WRITE(29,*)
      WRITE(29,*) 'VREF     [M/SEC]  =',VREF
      WRITE(29,*) 'VREFX    [M/SEC]  =',VREFX
      WRITE(29,*) 'REREF             =',REREF
      WRITE(29,*) 'REREFX            =',REREFX
      WRITE(29,*) 'MUREF [KG/M/SEC]  = ',VREF*RHOREF*CHORD/REREF
      WRITE(29,*) 'RHOREF [KG/M**3]  =',RHOREF
      WRITE(29,*)
      WRITE(29,*) 'VIN     [M/SEC]   =',VIN
      WRITE(29,*) 'VINX    [M/SEC]   =',VINX
      WRITE(29,*) 'REIN              =',REIN
      WRITE(29,*) 'REINX             =',REINX
      WRITE(29,*) 'MUIN [KG/M/SEC]   = ',MUIN
      WRITE(29,*) 'RHOIN [KG/M**3]   =',RHOIN
      WRITE(29,*) '****************************************************'
      WRITE(29,*) 'INLET PROFILE CORRECTION (PA)'
      WRITE(29,*)
      WRITE(29,*) 'PT: ','(M)',AINP(1,2),'(C)',AINP(2,2)
      WRITE(29,*) 'PS: ','(M)',AINP(1,3),'(C)',AINP(2,3)
      WRITE(29,*) 'PD: ','(M)',AINP(1,8),'(C)',AINP(2,8)
      WRITE(29,*) '****************************************************'
      WRITE(29,*) 'OUTLET PROFILE CORRECTION (PA)'
      WRITE(29,*)
      WRITE(29,*) 'PT: ','(M)',AOUTP(1,2),'(C)',AOUTP(2,2)
      WRITE(29,*) 'PS: ','(M)',AOUTP(1,3),'(C)',AOUTP(2,3)
      WRITE(29,*) 'PD: ','(M)',AOUTP(1,8),'(C)',AOUTP(2,8)
      WRITE(29,*) '****************************************************'
      WRITE(29,*) 'FAR PROFILE CORRECTION (PA)'
      WRITE(29,*)
      WRITE(29,*) 'PT: ','(M)',AFARP(1,2),'(C)',AFARP(2,2)
      WRITE(29,*) 'PS: ','(M)',AFARP(1,3),'(C)',AFARP(2,3)
      WRITE(29,*) 'PD: ','(M)',AFARP(1,8),'(C)',AFARP(2,8)
      WRITE(29,*) '****************************************************'
      WRITE(29,*) 'MASS FLOW (KG/SEC/M)'
      WRITE(29,*)
      WRITE(29,*) 'INLET PROFILE  :','(M)',MIMF,'(C)',CIMF
      WRITE(29,*) 'OUTLET PROFILE :','(M)',MOMF,'(C)',COMF
      WRITE(29,*) 'FAR PROFILE    :','(M)',MFMF,'(C)',CFMF
      WRITE(29,*) '****************************************************'
      WRITE(29,*) 'OUTLET PLANE LOSS COEFFICIENTS'
      WRITE(29,*)
      WRITE(29,*) 'MEASURED PROFILE      :',OLOSS(1)
      WRITE(29,*) 'CORRECTED PROFILE     :',OLOSS(2)
      WRITE(29,*) 'PFEIL CORRECTED       :',POLSS
      WRITE(29,*) 'PFEIL AXIAL-CORRECTED :',POLSSX
      WRITE(29,*) '****************************************************'
      WRITE(29,*) 'FAR PLANE LOSS COEFFICIENTS'
      WRITE(29,*)
      WRITE(29,*) 'MEASURED PROFILE      :',FLOSS(1)
      WRITE(29,*) 'CORRECTED PROFILE     :',FLOSS(2)
      WRITE(29,*) 'PFEIL CORRECTED       :',PFLSS
      WRITE(29,*) 'PFEIL AXIAL-CORRECTED :',PFLSSX
      WRITE(29,*) '****************************************************'
      WRITE(29,*) 'MIXING LOSS COEFFICIENTS'
      WRITE(29,*)
      WRITE(29,*) 'MEASURED PROFILE        :',MLOSS
      WRITE(29,*) 'PFEIL CORRECTED PROFILE :',PMLOSS
      WRITE(29,*) '****************************************************'
      WRITE(29,*) 'DRAG/LIFT RATIO'
      WRITE(29,*)
      WRITE(29,*) 'OUTLET MEASURED PROFILE  :',
     $            MOFN/SQRT(MOFT**2.0+MOFN**2.0)
      WRITE(29,*) 'FAR MEASURED PROFILE     :',
     $            MFFN/SQRT(MFFT**2.0+MFFN**2.0)
      WRITE(29,*) 'OUTLET CORRECTED PROFILE :',
     $            COFN/SQRT(COFT**2.0+COFN**2.0)
      WRITE(29,*) 'FAR CORRECTED PROFILE    :',
     $            CFFN/SQRT(CFFT**2.0+CFFN**2.0)
      WRITE(29,*) '****************************************************'
      WRITE(29,*) 'PRESSURE LIFT FORCE (N/M)'
      WRITE(29,*)
      WRITE(29,*) '(M)',BFT,'(C)',CBFT
      WRITE(29,*) '****************************************************'
C write all-summary out to output file
      WRITE(21,*) INCID,
     $            OLOSS(1),POLSS,POLSSX,
     $            FLOSS(1),PFLSS,PFLSSX,
     $            MLOSS,PMLOSS,
     $            MOFN/SQRT(MOFT**2.0+MOFN**2.0),
     $            MFFN/SQRT(MFFT**2.0+MFFN**2.0),
     $            COFN/SQRT(COFT**2.0+COFN**2.0),
     $            CFFN/SQRT(CFFT**2.0+CFFN**2.0),
     $            REINX,REREFX,
     $            RHOIN,RHOREF,
     $            MIMF,CIMF,
     $            BFT,CBFT
      WRITE(24,*) INCID,POLSSX
C linearly translate the axial location of the points on out profile
      DO 49 N=1,OPE
         OUTP(1,N,1)=OUTP(1,N,1)-OGAP
         OUTP(2,N,1)=OUTP(1,N,1)
         OUTP(3,N,1)=OUTP(1,N,1)
   49 CONTINUE
C linearly translate the axial location of the points on far profile
      DO 50 N=1,FPE
         FARP(1,N,1)=FARP(1,N,1)-FGAP
         FARP(2,N,1)=FARP(1,N,1)
         FARP(3,N,1)=FARP(1,N,1)
   50 CONTINUE
C find the new start-point of the out profile
      DO 51 N=1,OPE
         IF(OUTP(1,N,1).GE.0.0) THEN
           OS=N
           GO TO 52
         ENDIF
   51 CONTINUE
C find the new end-point of the out profile
   52 DO 74 N=1,OPE
         IF(OUTP(1,N,1).GE.1.0) THEN
           OE=N
           GO TO 75
         ENDIF
   74 CONTINUE
C find the new start-point of the far profile
   75 DO 53 N=1,FPE
         IF(FARP(1,N,1).GE.0.0) THEN
           FS=N
           GO TO 54
         ENDIF
   53 CONTINUE
C find the new end-point of the far profile
   54 DO 76 N=1,FPE
         IF(FARP(1,N,1).GE.1.0) THEN
           FE=N
           GO TO 77
         ENDIF
   76 CONTINUE
C populate the extended in arrays
   77 DO 71 N=1,C+1
         DO 72 M=1,IPE
            DO 73 K=1,W1
               IF(K.EQ.1) THEN
                 EINP(N,M,K)=INP(N,M,K)
                 EINP(N,M+IPE-1,K)=INP(N,M,K)+1.0
               ENDIF
               IF(K.GT.1) THEN
                 EINP(N,M,K)=INP(N,M,K)
                 EINP(N,M+IPE-1,K)=INP(N,M,K)
               ENDIF
   73       CONTINUE
   72    CONTINUE
   71 CONTINUE
C populate the extended out arrays - first half
      DO 55 N=1,C+1
         DO 56 M=1,OPE
            DO 57 K=1,W1
```

```
               IF(OGAP.GE.0.0) THEN
                 IF(M.LE.OS) THEN
                   IF(K.EQ.1) THEN
                     EOUTP(N,M+OPE-OS,K)=OUTP(N,M,K)+1.0
                   ELSE
                       EOUTP(N,M+OPE-OS,K)=OUTP(N,M,K)
                   ENDIF
                 ENDIF
                 IF(M.GE.OS) THEN
                   EOUTP(N,M-OS+1,K)=OUTP(N,M,K)
                   EOUTP(N,M-OS+1,K)=OUTP(N,M,K)
                 ENDIF
               ELSE
                   IF(M.GE.OE) THEN
                     IF(K.EQ.1) THEN
                       EOUTP(N,M-OE+1,K)=OUTP(N,M,K)-1.0
                     ELSE
                         EOUTP(N,M-OE+1,K)=OUTP(N,M,K)
                     ENDIF
                   ENDIF
                   IF(M.LE.OE) THEN
                     EOUTP(N,M+OPE-OE,K)=OUTP(N,M,K)
                     EOUTP(N,M+OPE-OE,K)=OUTP(N,M,K)
                   ENDIF
               ENDIF
   57       CONTINUE
   56    CONTINUE
   55 CONTINUE
C populate the extended out arrays - second half
      DO 59 N=1,C+1
         DO 60 M=1,OPE
            DO 61 K=1,W1
               IF(K.EQ.1) EOUTP(N,M+OPE-1,K)=EOUTP(N,M,K)+1.0
               IF(K.GT.1) EOUTP(N,M+OPE-1,K)=EOUTP(N,M,K)
   61       CONTINUE
   60    CONTINUE
   59 CONTINUE
C populate the extended far arrays - first half
      DO 78 N=1,C+1
         DO 79 M=1,FPE
            DO 80 K=1,W1
               IF(OGAP.GE.0.0) THEN
                 IF(M.LE.FS) THEN
                   IF(K.EQ.1) THEN
                     EFARP(N,M+FPE-FS,K)=FARP(N,M,K)+1.0
                   ELSE
                       EFARP(N,M+FPE-FS,K)=FARP(N,M,K)
                   ENDIF
                 ENDIF
                 IF(M.GE.FS) THEN
                   EFARP(N,M-FS+1,K)=FARP(N,M,K)
                   EFARP(N,M-FS+1,K)=FARP(N,M,K)
                 ENDIF
               ELSE
                   IF(M.GE.FE) THEN
                     IF(K.EQ.1) THEN
                       EFARP(N,M-FE+1,K)=FARP(N,M,K)-1.0
                     ELSE
                         EFARP(N,M-FE+1,K)=FARP(N,M,K)
                     ENDIF
                   ENDIF
                   IF(M.LE.FE) THEN
                     EFARP(N,M+FPE-FE,K)=FARP(N,M,K)
                     EFARP(N,M+FPE-FE,K)=FARP(N,M,K)
                   ENDIF
               ENDIF
   80       CONTINUE
   79    CONTINUE
   78 CONTINUE
C populate the extended far arrays - second half
      DO 81 N=1,C+1
         DO 82 M=1,FPE
            DO 83 K=1,W1
               IF(K.EQ.1) EFARP(N,M+FPE-1,K)=EFARP(N,M,K)+1.0
               IF(K.GT.1) EFARP(N,M+FPE-1,K)=EFARP(N,M,K)
   83       CONTINUE
   82    CONTINUE
   81 CONTINUE
C write data to output file cplt.dat
      WRITE(26,*) 'VARIABLES="Y",',
     $            '"P_t [Pa]",',
     $            '"P_s [Pa]",',
     $            '"P_d [Pa]",',
     $            '"V [m/sec]"',
     $            '"V_x [m/sec]"',
     $            '"V_y [m/sec]"'
C write inlet data
      WRITE(26,*) 'ZONE T="cfd-in-m"'
      DO 28 N=1,2*IPE-1
         WRITE(26,*) 1.0-EINP(1,N,1)/2.0,EINP(1,N,2),
     $               EINP(1,N,3),EINP(1,N,8),
     $               EINP(1,N,4),EINP(1,N,9),EINP(1,N,10)
   28 CONTINUE
      WRITE(26,*) 'ZONE T="cfd-in-c"'
      DO 29 N=1,2*IPE-1
         WRITE(26,*) 1.0-EINP(2,N,1)/2.0,EINP(2,N,2),
     $               EINP(2,N,3),EINP(2,N,8),
     $               EINP(2,N,4),EINP(2,N,9),EINP(2,N,10)
   29 CONTINUE
C write outlet data
      WRITE(26,*) 'ZONE T="cfd-out-m"'
      DO 34 N=1,2*OPE-1
         WRITE(26,*) 1.0-EOUTP(1,N,1)/2.0,EOUTP(1,N,2),
     $               EOUTP(1,N,3),EOUTP(1,N,8),
     $               EOUTP(1,N,4),EOUTP(1,N,9),EOUTP(1,N,10)
   34 CONTINUE
      WRITE(26,*) 'ZONE T="cfd-out-c"'
      DO 35 N=1,2*OPE-1
         WRITE(26,*) 1.0-EOUTP(2,N,1)/2.0,EOUTP(2,N,2),
     $               EOUTP(2,N,3),EOUTP(2,N,8),
     $               EOUTP(2,N,4),EOUTP(2,N,9),EOUTP(2,N,10)
   35 CONTINUE
C write far data
      WRITE(26,*) 'ZONE T="cfd-far-m"'
      DO 40 N=1,2*FPE-1
         WRITE(26,*) 1.0-EFARP(1,N,1)/2.0,EFARP(1,N,2),
     $               EFARP(1,N,3),EFARP(1,N,8),
     $               EFARP(1,N,4),EFARP(1,N,9),EFARP(1,N,10)
   40 CONTINUE
      WRITE(26,*) 'ZONE T="cfd-far-c"'
      DO 41 N=1,2*FPE-1
         WRITE(26,*) 1.0-EFARP(2,N,1)/2.0,EFARP(2,N,2),
     $               EFARP(2,N,3),EFARP(2,N,8),
     $               EFARP(2,N,4),EFARP(2,N,9),EFARP(2,N,10)
   41 CONTINUE
C write data to output file cnplt.dat and all-summary output file
      WRITE(27,*) 'VARIABLES="Y",',
     $            '"P_t_n",',
     $            '"P_s_n",',
     $            '"P_d_n",',
     $            '"V_n"',
     $            '"V_x_n"',
     $            '"V_y_n"'
C write inlet data
```

```
      WRITE(27,*) 'ZONE T="cfd-in-n"'
      DO 42 N=1,2*IPE-1
         WRITE(27,*) 1.0-EINP(3,N,1)/2.0,EINP(3,N,2),
     $               EINP(3,N,3),EINP(3,N,8),
     $               EINP(3,N,4),EINP(3,N,9),EINP(3,N,10)
   42 CONTINUE
C write outlet data
      WRITE(27,*) 'ZONE T="cfd-out-n"'
      DO 43 N=1,2*OPE-1
         WRITE(27,*) 1.0-EOUTP(3,N,1)/2.0,EOUTP(3,N,2),
     $               EOUTP(3,N,3),EOUTP(3,N,8),
     $               EOUTP(3,N,4),EOUTP(3,N,9),EOUTP(3,N,10)
C         WRITE(22,*) 1.0-EOUTP(3,N,1)/2.0,EOUTP(3,N,2),
C     $               EOUTP(3,N,3),EOUTP(3,N,8),
C     $               EOUTP(3,N,4),EOUTP(3,N,9),EOUTP(3,N,10)
   43 CONTINUE
C write far data
      WRITE(27,*) 'ZONE T="cfd-far-n"'
      DO 44 N=1,2*FPE-1
         WRITE(27,*) 1.0-EFARP(3,N,1)/2.0,EFARP(3,N,2),
     $               EFARP(3,N,3),EFARP(3,N,8),
     $               EFARP(3,N,4),EFARP(3,N,9),EFARP(3,N,10)
         WRITE(22,*) 1.0-EFARP(3,N,1)/2.0,EFARP(3,N,2),
     $               EFARP(3,N,3),EFARP(3,N,8),
     $               EFARP(3,N,4),EFARP(3,N,9),EFARP(3,N,10)
   44 CONTINUE
C write cp data to output file cp.dat and all-summary output file
      WRITE(25,*) 'VARIABLES="X",',"C_p"'
      DO 46 N=1,BDE
         WRITE(25,*) CP(N,1),CP(N,2)
         WRITE(23,*) CP(N,1),CP(N,2)
   46 CONTINUE
C terminate files for current case
      CLOSE(15)
      CLOSE(16)
      CLOSE(17)
      CLOSE(18)
      CLOSE(25)
      CLOSE(26)
      CLOSE(27)
      CLOSE(28)
      CLOSE(29)
C loop
      GO TO 30
   31 PRINT*, '================================='
      PRINT*, 'TOTAL CASES STUDIED: ',NUM
      PRINT*, '================================='
      END
************************************************************************
* This subroutine performs mass averaging using Trap. integration.     *
************************************************************************
* Hicham A Chibli - Texas A&M University - 12/11/09                    *
************************************************************************
      SUBROUTINE INTG(FULL,C,L,W,SET,K,CMN,MA,MINT,MFLOW)
      INTEGER C,L,W,SET,K,CMN,I
      REAL FULL(C,L,W),NX(K),NY(K),NZ(K),MA,MFLOW,MINT
C populate integration arrays
      DO 2 I=1,K
         NX(I)=FULL(SET,I,1)
         NZ(I)=FULL(SET,I,5)*FULL(SET,I,9)
         NY(I)=FULL(SET,I,CMN)*NZ(I)
    2 CONTINUE
C initialize integral
      MFLOW=0.0
      MINT=0.0
C calculate mass flow rate
      DO 3 I=1,K-1
```

```
         MFLOW=MFLOW+0.5*(NX(I+1)-NX(I))*(NZ(I)+NZ(I+1))
    3 CONTINUE
C integrate
      DO 1 I=1,K-1
         MINT=MINT+0.5*(NX(I+1)-NX(I))*(NY(I)+NY(I+1))
    1 CONTINUE
      MA=MINT/MFLOW
      MFLOW=ABS(MFLOW)
      RETURN
      END
************************************************************************
* This subroutine returns the non-blank component of a string.         *
************************************************************************
* Hicham A Chibli - Texas A&M University - 12/01/09                    *
************************************************************************
      SUBROUTINE CLEAN(TITLE,N)
C Define variables and constants
      INTEGER N
      CHARACTER*(*) TITLE
C Find the blank part of file name
      N=INDEX(TITLE,’ ’)
C Get the size of the non-blank part of file name
      N=N-1
      RETURN
      END
```

J.4 Blade Cascade Stagger Angle Study Data Analysis Programs

```
************************************************************************
* This program reads the inlet and exit flow profile data and analyzes *
* it to calculate various flow properties, forces and losses.          *
* Input Files : xx.dat, info.dat, blade.dat, ambnt.dat                 *
* Output Files: casexx.dat, xplot.data, nxplot.dat, cpxx.dat, exit.dat *
* Experimental Procedure: Cascade Facility of the DOOSAN II Blade      *
************************************************************************
* Hicham A Chibli - Texas A&M University - 10/10/10                    *
************************************************************************
      PROGRAM DB2ALS
      DOUBLE PRECISION GAMMA,PI,KELVIN
      INTEGER MTOTAL,ISIZE
      PARAMETER (MTOTAL=2000,PI=3.14159265D0,GAMMA=1.4D0,
     $           KELVIN=273.15D0,ISIZE=48)
      INTEGER N,M,K,J,TOTAL,IDEN,WCTR,
C------------array indices
     $        P1,P2,W1,W2,F1,F2,M1,M2,BPL,EPL
C----------------------reference values
      DOUBLE PRECISION BPTREF,SWAP,CPT,CRHO,CRH,CBAR,
C----------------------profile trimming points and convergence limits
     $                PRS,PRE,WKS,WKE,LL,UL,
C----------------------cascade parameters
     $                CHORD,XCHORD,ZROT,ALIGN,STEP,STGR,DEGREE,
     $                PITCH,BPLOT,EPLOT,MID,
C----------------------global correction parameters
     $                IRHORF,IVXRF,IREXRF,
C----------------------locally corrected averaged values
     $                IRHO,IVX,IV,IRE,IREX,IMU,IPT,IPD,IPS,ITS,FEPT,
     $                EMU,ERE,EPT,EPS,EPD,EV,EVX,EVY,EVZ,ETS,ERHO,FVM,
C----------------------calculated performance parameters
     $                ELSS1,ELSS2,CPFT,PFT,MFL,FL,FMFL,PDEL2,SDEL2,
     $                DEL2,CDEL2,CPDEL2,CSDEL2,DEL
C---------------strings holding file names and working directories
      CHARACTER CASNUM*2,EXT*4,CP*8,NXPLOT*12,CASNME*6,
     $          NCASE*10,XPLOT*11
C----------------------upstream reference presssures
      DOUBLE PRECISION PT(MTOTAL),PS(MTOTAL),
C----------------------locally corrected values
     $                UP1(MTOTAL),UP2(MTOTAL),UP3(MTOTAL),UP4(MTOTAL),
     $                UP5(MTOTAL),DP1(MTOTAL),DP2(MTOTAL),DP3(MTOTAL),
     $                DP4(MTOTAL),DP5(MTOTAL),
     $                UPT(MTOTAL),UPS(MTOTAL),UPD(MTOTAL),
     $                DPT(MTOTAL),DPS(MTOTAL),DPD(MTOTAL),
     $                UALPHA(MTOTAL),UBETA(MTOTAL),
     $                DALPHA(MTOTAL),DBETA(MTOTAL),
     $                UV(MTOTAL),
     $                DV(MTOTAL),DVX(MTOTAL),DVY(MTOTAL),DVZ(MTOTAL),
     $                Y(MTOTAL),
C----------------------blade pressure distribution
     $                P(ISIZE),PP(ISIZE),DIST(ISIZE),NDIST(ISIZE),
C----------------------mid free stream normalization variables
     $                NAVER(6),
C----------------------globally corrected values
     $                CDPT(MTOTAL),CDPS(MTOTAL),CDPD(MTOTAL),
     $                CDV(MTOTAL),CDVX(MTOTAL),
     $                CDVY(MTOTAL),CDVZ(MTOTAL),
C----------------------normalized plotting values
     $                NDPT(MTOTAL),NDPS(MTOTAL),NDPD(MTOTAL),
     $                NDV(MTOTAL),NDVX(MTOTAL),
     $                NDVY(MTOTAL),NDVZ(MTOTAL),NY(MTOTAL),
C----------------------measured values
     $                BAR(MTOTAL),RHO(MTOTAL),TT(MTOTAL),RH(MTOTAL),
C----------------------measured/corrected values
```

```
     $                 UTS(MTOTAL,2),URHO(MTOTAL,2),
     $                 DTS(MTOTAL,2),DRHO(MTOTAL,2)
C open files for input/output
      OPEN(UNIT=40,FILE=’./in/info.dat’,STATUS=’OLD’)
      OPEN(UNIT=90,FILE=’./in/blade.dat’,STATUS=’OLD’)
      OPEN(UNIT=70,FILE=’./out/ambnt.dat’,STATUS=’OLD’)
      OPEN(UNIT=20,FILE=’./out/exit.dat’,STATUS=’UNKNOWN’)
C reaf basic data from input file
      READ(40,*)
      READ(40,*) TOTAL
      READ(40,*)
      READ(40,*) STEP
      READ(40,*)
      READ(40,*) CASNUM
      READ(40,*)
      READ(40,*) STGR
      READ(40,*)
      READ(40,*) ALIGN
      READ(40,*)
      READ(40,*) ZROT
      READ(40,*)
      READ(40,*) IRHORF
      READ(40,*)
      READ(40,*) IVXRF
      READ(40,*)
      READ(40,*) IREXRF
      READ(40,*)
      READ(40,*) PRS
      READ(40,*)
      READ(40,*) PRE
      READ(40,*)
      READ(40,*) WKS
      READ(40,*)
      READ(40,*) WKE
      READ(40,*)
      READ(40,*) WCTR
      READ(40,*)
      READ(40,*) LL
      READ(40,*)
      READ(40,*) UL
      READ(40,*)
      READ(40,*) IDEN
      READ(40,*)
      READ(40,*) BPTREF
      READ(40,*)
      READ(40,*) CHORD
      READ(40,*)
      READ(40,*) XCHORD
C Generate input/output file names
      EXT=’.dat’
      CASNME=CASNUM//EXT
      CP=’cp’//CASNUM//EXT
      NXPLOT=’nxplot’//CASNUM//EXT
      XPLOT=’xplot’//CASNUM//EXT
      NCASE=’case’//CASNUM//EXT
C Open data files for input/output
      OPEN(UNIT=10,FILE=’./in/’//CASNME,STATUS=’OLD’)
      OPEN(UNIT=30,FILE=’./out/’//NCASE,STATUS=’UNKNOWN’)
      OPEN(UNIT=50,FILE=’./out/’//NXPLOT,STATUS=’UNKNOWN’)
      OPEN(UNIT=80,FILE=’./out/’//XPLOT,STATUS=’UNKNOWN’)
      OPEN(UNIT=60,FILE=’./out/’//CP,STATUS=’UNKNOWN’)
C zero array variables
      DO 3 N=1,MTOTAL
         PT(N)=0.0D0
         PS(N)=0.0D0
         UP1(N)=0.0D0
         UP2(N)=0.0D0
         UP3(N)=0.0D0
         UP4(N)=0.0D0
         UP5(N)=0.0D0
         DP1(N)=0.0D0
         DP2(N)=0.0D0
         DP3(N)=0.0D0
         DP4(N)=0.0D0
         DP5(N)=0.0D0
         UPT(N)=0.0D0
         UPS(N)=0.0D0
         UPD(N)=0.0D0
         UV(N)=0.0D0
         UALPHA(N)=0.0D0
         UBETA(N)=0.0D0
         DPT(N)=0.0D0
         DPS(N)=0.0D0
         DPD(N)=0.0D0
         DV(N)=0.0D0
         DVX(N)=0.0D0
         DVY(N)=0.0D0
         DVZ(N)=0.0D0
         DALPHA(N)=0.0D0
         DBETA(N)=0.0D0
         CDV(N)=0.0D0
         CDVX(N)=0.0D0
         CDVY(N)=0.0D0
         CDVZ(N)=0.0D0
         CDPT(N)=0.0D0
         CDPS(N)=0.0D0
         CDPD(N)=0.0D0
         NDV(N)=0.0D0
         NDVX(N)=0.0D0
         NDVY(N)=0.0D0
         NDVZ(N)=0.0D0
         NDPT(N)=0.0D0
         NDPS(N)=0.0D0
         NDPD(N)=0.0D0
         NY(N)=0.0D0
         Y(N)=0.0D0
         BAR(N)=0.0D0
         RHO(N)=0.0D0
         TT(N)=0.0D0
         RH(N)=0.0D0
    3 CONTINUE
      DO 4 N=1,MTOTAL
         DO 5 M=1,2
            UTS(N,M)=0.0D0
            URHO(N,M)=0.0D0
            DTS(N,M)=0.0D0
            DRHO(N,M)=0.0D0
    5    CONTINUE
    4 CONTINUE
C write header to full output data set
      WRITE(20,100)
  100 FORMAT(1X,’Y [MM]’,5X,’PTREF [PA]’,2X,’UPT [PA]’,
     $      2X,’UPS [PA]’,2X,’UPD [PA]’,2X,’UV [M/SEC]’,2X,
     $      2X,’DPT [PA]’,2X,’DPS [PA]’,2X,’DPD [PA]’,
     $      2X,’DV [M/SEC]’,2X,’DVX [M/SEC]’,2X,’DVY [M/SEC]’,
     $      2X,’DVZ [M/SEC]’,2X,’EXIT DENSITY [KG/M**3]’,/)
C initialize counters
      N=1
      DEGREE=PI/180.0D0
C read input pressure data
      READ(10,*)
    1 READ(10,*,END=2) PT(N),PS(N),UP1(N),UP2(N),UP3(N),UP4(N),
     $                 UP5(N),DP1(N),DP2(N),DP3(N),DP4(N),DP5(N)
C read input ambient conditions data
```

```
      READ(70,*) BAR(N),TT(N),RH(N),Y(N)
C change the total temperature unit to Kelvin
      TT(N)=TT(N)+KELVIN
C swap static and total pressure to minimize transducer errors
      IF(PT(N).LT.PS(N)) THEN
         SWAP=PT(N)
         PT(N)=PS(N)
         PS(N)=SWAP
      ENDIF
C set the local reference total pressure, ambient pressure and humidity
C for the flow correction procedure
      IF(N.EQ.1) THEN
        CPT=PT(N)
        CBAR=BAR(N)
        CRH=RH(N)
      ENDIF
C locally correct all the measured pressures
      UP1(N)=UP1(N)*CPT/PT(N)
      UP2(N)=UP2(N)*CPT/PT(N)
      UP3(N)=UP3(N)*CPT/PT(N)
      UP4(N)=UP4(N)*CPT/PT(N)
      UP5(N)=UP5(N)*CPT/PT(N)
      DP1(N)=DP1(N)*CPT/PT(N)
      DP2(N)=DP2(N)*CPT/PT(N)
      DP3(N)=DP3(N)*CPT/PT(N)
      DP4(N)=DP4(N)*CPT/PT(N)
      DP5(N)=DP5(N)*CPT/PT(N)
C call calibration subroutines to reduce 5-hole-probe data
      CALL CLB8(UP1(N),UP2(N),UP3(N),UP4(N),UP5(N),UPS(N),UPT(N),
     $          UALPHA(N),UBETA(N),N)
      CALL CLB6(DP1(N),DP2(N),DP3(N),DP4(N),DP5(N),DPS(N),DPT(N),
     $          DALPHA(N),DBETA(N),N)
C calculate the dynamic heads
      UPD(N)=UPT(N)-UPS(N)
      DPD(N)=DPT(N)-DPS(N)
C calculate the uncorrected static temperatures at each analysis plane
      CALL TSTAT(TT(N),UPT(N)*PT(N)/CPT+BAR(N),UPS(N)*PT(N)/CPT+BAR(N),
     $           UTS(N,1))
      CALL TSTAT(TT(N),DPT(N)*PT(N)/CPT+BAR(N),DPS(N)*PT(N)/CPT+BAR(N),
     $           DTS(N,1))
C calculate the uncorrected densities at each analysis plane
      CALL DEN(TT(N),PT(N)+BAR(N),RH(N),RHO(N))
      CALL DEN(UTS(N,1),UPS(N)*PT(N)/CPT+BAR(N),RH(N),URHO(N,1))
      CALL DEN(DTS(N,1),DPS(N)*PT(N)/CPT+BAR(N),RH(N),DRHO(N,1))
C set the local reference density for the flow correction procedure
      IF(N.EQ.1) CRHO=RHO(N)
C locally correct the measured densities
      URHO(N,2)=URHO(N,1)*CRHO/RHO(N)
      DRHO(N,2)=DRHO(N,1)*CRHO/RHO(N)
C calculate locally corrected velocities at the inlet and exit planes
      UV(N)=(2.0D0*(UPT(N)-UPS(N))/(URHO(N,2)))**0.5D0
      DV(N)=(2.0D0*(DPT(N)-DPS(N))/(DRHO(N,2)))**0.5D0
      DVX(N)=DV(N)*DCOS(DBETA(N)*DEGREE)*DCOS(DALPHA(N)*DEGREE)*
     $       DCOS(ZROT*DEGREE)+DV(N)*DSIN(DBETA(N)*
     $       DEGREE)*DSIN(ZROT*DEGREE)
      DVY(N)=DV(N)*DCOS(DBETA(N)*DEGREE)*DCOS(DALPHA(N)*DEGREE)*
     $       DSIN(ZROT*DEGREE)-DV(N)*DSIN(DBETA(N)*
     $       DEGREE)*DCOS(ZROT*DEGREE)
      DVZ(N)=DV(N)*DCOS(DBETA(N)*DEGREE)*DSIN(DALPHA(N)*DEGREE)
C write full data to output file
      WRITE(20,300) Y(N),CPT,UPT(N),UPS(N),UPD(N),UV(N),
     $              DPT(N),DPS(N),DPD(N),DV(N),DVX(N),DVY(N),DVZ(N),
     $              DRHO(N,2)
  300 FORMAT(1X,F6.2,8X,F7.1,3X,F7.1,3X,F7.1,3X,F7.1,6X,F6.2,
     $      5X,F7.1,3X,F7.1,3X,F7.1,6X,F6.2,7X,F6.2,7X,F6.2,7X,F6.2,
     $      19X,F5.3)
C correct the static temperatures
```

```
      CALL RSTAT(UPS(N),URHO(N,2),CBAR,CRH,UTS(N,2))
      CALL RSTAT(DPS(N),DRHO(N,2),CBAR,CRH,DTS(N,2))
      N=N+1
      GO TO 1
C arithmetically average the locally corrected inlet profile data
    2 CALL ARTH(UPT,MTOTAL,TOTAL,1,1,IPT)
      CALL ARTH(UPS,MTOTAL,TOTAL,1,1,IPS)
      CALL ARTH(UPD,MTOTAL,TOTAL,1,1,IPD)
      CALL ARTH(UTS,MTOTAL,TOTAL,2,2,ITS)
      CALL ARTH(URHO,MTOTAL,TOTAL,2,2,IRHO)
C trim the exit profile to 2 periods
      CALL PTRIM(MTOTAL,TOTAL,P1,P2,PRS,PRE,Y)
      CALL PTRIM(MTOTAL,TOTAL,W1,W2,WKS,WKE,Y)
C find the location of the free-stream component of the profile
      CALL WAKE(MTOTAL,W1,W2,F1,F2,STEP,LL,UL,DPT)
C find the location of the mid-free-stream used for normalizing data
      MID=(Y(W1)+Y(W2))/2.0D0
      CALL PTRIM(MTOTAL,TOTAL,W1,M1,WKS,MID,Y)
      CALL PTRIM(MTOTAL,TOTAL,M2,W2,MID,WKE,Y)
C calculate the experimental pitch
      PITCH=Y(W2)-Y(W1)
C calculate the location of the start/end of plot arrays
      IF(WCTR.EQ.1) THEN
        BPLOT=Y(W1)-PITCH/2.0D0
        EPLOT=Y(W1)+PITCH/2.0D0
      ENDIF
      IF(WCTR.EQ.2) THEN
        BPLOT=Y(W2)-PITCH/2.0D0
        EPLOT=Y(W2)+PITCH/2.0D0
      ENDIF
      CALL PTRIM(MTOTAL,TOTAL,BPL,EPL,BPLOT,EPLOT,Y)
C write to screen the trimming criteria
      WRITE(*,26) Y(P1),Y(P2),P2-P1
   26 FORMAT(//,1X,'EXIT PROFILE DATA HAS BEEN TRIMMED BETWEEN',/,1X,
     $       'Y1 =',1X,F6.2,2X,'AND',2X,'Y2 =',1X,F6.2,/,
     $       1X,'TOTAL OF ',I3,' POINTS'//)
      WRITE(*,27) Y(W1),Y(W2),W2-W1
   27 FORMAT(1X,'COMPLETE WAKE PROFILE HAS BEEN TRIMMED BETWEEN',/,1X,
     $       'Y1 =',1X,F6.2,2X,'AND',2X,'Y2 =',1X,F6.2,/,
     $       1X,'TOTAL OF ',I3,' POINTS'//)
      WRITE(*,28) Y(F1),Y(F2),F2-F1
   28 FORMAT(1X,'FREE STREAM WAKE PROFILE HAS BEEN TRIMMED BETWEEN',/,
     $       1X,'Y1 =',1X,F6.2,2X,'AND',2X,'Y2 =',1X,F6.2,/,
     $       1X,'TOTAL OF ',I3,' POINTS'//)
C Integrate using Trapezoidal Rule the full trimmed exit profile
      EPT=0.0D0
      EPD=0.0D0
      EPS=0.0D0
      EV=0.0D0
      EVX=0.0D0
      EVY=0.0D0
      ERHO=0.0D0
      ETS=0.0D0
      MFL=0.0D0
      FL=0.0D0
      DO 7 J=P1,P2-1
         EPT=EPT+0.5D0*(DPT(J)*DVX(J)*DRHO(J,2)+
     $       DPT(J+1)*DVX(J+1)*DRHO(J+1,2))*(Y(J+1)-Y(J))
         EPD=EPD+0.5D0*(DPD(J)*DVX(J)*DRHO(J,2)+
     $       DPD(J+1)*DVX(J+1)*DRHO(J+1,2))*(Y(J+1)-Y(J))
         EPS=EPS+0.5D0*(DPS(J)*DVX(J)*DRHO(J,2)+
     $       DPS(J+1)*DVX(J+1)*DRHO(J+1,2))*(Y(J+1)-Y(J))
         EV=EV+0.5D0*(DV(J)*DVX(J)*DRHO(J,2)+
     $      DV(J+1)*DVX(J+1)*DRHO(J+1,2))*(Y(J+1)-Y(J))
         EVX=EVX+0.5D0*(DVX(J)*DVX(J)*DRHO(J,2)+
     $       DVX(J+1)*DVX(J+1)*DRHO(J+1,2))*(Y(J+1)-Y(J))
         EVY=EVY+0.5D0*(DVY(J)*DVX(J)*DRHO(J,2)+
     $       DVY(J+1)*DVX(J+1)*DRHO(J+1,2))*(Y(J+1)-Y(J))
         ERHO=ERHO+0.5D0*(DRHO(J,2)*DVX(J)*DRHO(J,2)+
     $        DRHO(J+1,2)*DVX(J+1)*DRHO(J+1,2))*(Y(J+1)-Y(J))
         ETS=ETS+0.5D0*(DTS(J,2)*DVX(J)*DRHO(J,2)+
     $       DTS(J+1,2)*DVX(J+1)*DRHO(J+1,2))*(Y(J+1)-Y(J))
         MFL=MFL+0.5D0*(DVX(J)*DRHO(J,2)+DVX(J+1)*DRHO(J+1,2))*
     $      (Y(J+1)-Y(J))
    7 CONTINUE
      EPT=EPT/MFL
      EPD=EPD/MFL
      EPS=EPS/MFL
      EV=EV/MFL
      EVX=EVX/MFL
      EVY=EVY/MFL
      ERHO=ERHO/MFL
      ETS=ETS/MFL
      FL=MFL/(Y(P2)-Y(P1))
C Integrate using Trapezoidal Rule the free-stream exit profile
      FEPT=0.0D0
      FVM=0.0D0
      FMFL=0.0D0
      DO 8 J=F1,F2-1
         FEPT=FEPT+0.5D0*(DPT(J)*DVX(J)*DRHO(J,2)+
     $        DPT(J+1)*DVX(J+1)*DRHO(J+1,2))*(Y(J+1)-Y(J))
         FVM=FVM+0.5D0*(DV(J)*DVX(J)*DRHO(J,2)+
     $        DV(J+1)*DVX(J+1)*DRHO(J+1,2))*(Y(J+1)-Y(J))
         FMFL=FMFL+0.5D0*(DVX(J)*DRHO(J,2)+DVX(J+1)*DRHO(J+1,2))*
     $       (Y(J+1)-Y(J))
    8 CONTINUE
      FEPT=FEPT/FMFL
      FVM=FVM/FMFL
C Calculate locally corrected flow viscosity using Sutherland's equation
      IMU=1.46D-06*((ITS**1.5D0)/(ITS+111.0D0))
      EMU=1.46D-06*((ETS**1.5D0)/(ETS+111.0D0))
C find locally corrected inlet V and RE based on exit profile
      IVX=FL/IRHO
      IV=IVX/DCOS(ALIGN*DEGREE)
      IREX=0.001D0*IVX*IRHO*CHORD/IMU
      IRE=0.001D0*IV*IRHO*CHORD/IMU
C Calculate the locally corrected exit Reynold's
      ERE=0.001D0*EV*ERHO*CHORD/EMU
C Compute loss coefficient
      ELSS1=(FEPT-EPT)/EPD
C Compute the PFEIL globally corrected loss coefficient
      ELSS2=ELSS1*(IREX/IREXRF)**0.2D0
C Compute the momentum deficiency thickness for pressure side wake
      PDEL2=0.0D0
      DO 29 J=W2,F2+1,-1
         PDEL2=PDEL2+(STEP/2.0D0)*(DEL(DV(J),FVM)+DEL(DV(J+1),FVM))
   29 CONTINUE
C Compute the momentum deficiency thickness for suction side wake
      SDEL2=0.0D0
      DO 30 J=W1,F1-1,1
         SDEL2=SDEL2+(STEP/2.0D0)*(DEL(DV(J),FVM)+DEL(DV(J-1),FVM))
   30 CONTINUE
      DEL2=PDEL2+SDEL2
C Compute the PFEIL globally correctd boundary layer values
      CPDEL2=PDEL2*(IREX/IREXRF)**0.2D0
      CSDEL2=SDEL2*(IREX/IREXRF)**0.2D0
      CDEL2=CPDEL2+CSDEL2
C Globally correct the exit profile
      DO 39 J=1,TOTAL
         CDPT(J)=DPT(J)*(IRHORF/IRHO)*(IVXRF/IVX)**2.0D0
         CDPD(J)=DPD(J)*(IRHORF/IRHO)*(IVXRF/IVX)**2.0D0
         CDPS(J)=DPS(J)*(IRHORF/IRHO)*(IVXRF/IVX)**2.0D0
         CDV(J)=DV(J)*(IVXRF/IVX)
         CDVX(J)=DVX(J)*(IVXRF/IVX)
         CDVY(J)=DVY(J)*(IVXRF/IVX)
   39 CONTINUE
C Establish the base values for normilizing the exit profile
      NAVER(1)=(DPT(M1)+DPT(M2))/2.0D0
      NAVER(2)=(DPD(M1)+DPD(M2))/2.0D0
      NAVER(3)=(DPS(M1)+DPS(M2))/2.0D0
      NAVER(4)=(DV(M1)+DV(M2))/2.0D0
      NAVER(5)=(DVX(M1)+DVX(M2))/2.0D0
      NAVER(6)=(DVY(M1)+DVY(M2))/2.0D0
C Normalize the exit profile
      DO 40 J=BPL,EPL
         NDPT(J)=DPT(J)/NAVER(1)
         NDPD(J)=DPD(J)/NAVER(2)
         NDPS(J)=DPS(J)/NAVER(3)
         NDV(J)=DV(J)/NAVER(4)
         NDVX(J)=DVX(J)/NAVER(5)
         NDVY(J)=DVY(J)/NAVER(6)
         NY(J)=(Y(J)-Y(BPL))/(Y(EPL)-Y(BPL))
   40 CONTINUE
C Read data from inupt file blade.dat
      DO 22 J=1,ISIZE
         IF(J.EQ.22) THEN
           NDIST(J)=1.0D0
           DIST(J)=NDIST(J)*XCHORD
         ELSE
             READ(90,*) NDIST(J),P(J)
             DIST(J)=NDIST(J)*XCHORD
C Locally correct pressure
             P(J)=P(J)*(CPT/BPTREF)
             PP(J)=(P(J)-FEPT)/FEPT
         ENDIF
   22 CONTINUE
      P(22)=0.5D0*(P(21)+P(23))
      PP(22)=(P(22)-FEPT)/FEPT
C Write Cp distribution data to cpxx.dat output file
      WRITE(60,*) 'VARIABLES="X",','"C_p"'
      WRITE(60,33) STGR
   33 FORMAT('ZONE T="exp.  ‘g = ',F5.1,' ^o"')
      DO 23 J=1,ISIZE
         IF(J.EQ.22) GO TO 23
         WRITE(60,*) NDIST(J),PP(J)
   23 CONTINUE
C Integrate the pressure distribution to calculate tang. force on blade
      PFT=0.0D0
      DO 35 J=1,ISIZE-1
         PFT=PFT+0.5D0*(P(J)+P(J+1))*(DIST(J+1)-DIST(J))
   35 CONTINUE
      PFT=0.001D0*DABS(PFT)
C Calculate the globally corrected force
      CPFT=PFT*(IRHORF/IRHO)*(IVXRF/IVX)**2.0D0
C Write output to casex.dat
      WRITE(30,11) CASNUM,STGR
   11 FORMAT(1X,'CASE NUMBER:',1X,A,/,1X,'STAGGER ANGLE (DEG):',
     $       1X,F5.1,//)
C Write profile data summary
      WRITE(30,9) FEPT,IV,IRHO,IMU,IRE
    9 FORMAT(1X,'MEASURED INLET PROFILE SUMMARY',/,1X,
     $       '******************************',/,
     $       1X,'AVERAGE TOTAL PRESSURE [PA]',2X,'= ',F11.1,/,
     $       1X,'AVERAGE VELOCITY [M/SEC]',5X,'= ',F11.3,/
     $       1X,'AVERAGE DENSITY [KG/M**3]',4X,'= ',F11.3,/,
     $       1X,'AVERAGE VISCOSITY [KG/M/SEC]',1X,'= ',F11.9,/,
     $       1X,'AVERAGE REYNOLDS NUMBER',6X,'= ',F11.1,//)
      WRITE(30,14) EPT,EPS,EPD,EV,EVX,EVY,ERHO,
     $             EMU,ERE,FL
   14 FORMAT(1X,'MEASURED EXIT PROFILE SUMMARY',/,1X,
     $       '*****************************',/,
     $       1X,'AVERAGE TOTAL PRESSURE [PA]',9X,'= ',F11.1,/,
     $       1X,'AVERAGE STATIC PRESSURE [PA]',8X,'= ',F11.1,/,
     $       1X,'AVERAGE DYNAMIC PRESSURE [PA]',7X,'= ',F11.1,/,
     $       1X,'AVERAGE VELOCITY [M/SEC]',12X,'= ',F11.3,/,
     $       1X,'AVERAGE AXIAL VELOCITY [M/SEC]',6X,'= ',F11.3,/,
     $       1X,'AVERAGE TANGENTIAL VELOCITY [M/SEC]',1X,'= ',F11.3,/,
     $       1X,'AVERAGE DENSITY [KG/M**3]',11X,'= ',F11.3,/,
     $       1X,'AVERAGE VISCOSITY [KG/M/SEC]',8X,'= ',F11.9,/,
     $       1X,'AVERAGE REYNOLDS NUMBER',13X,'= ',F11.1,/,
     $       1X,'MASS FLOW RATE [KG/SEC/M**2]',8X,'= ',F11.4,//)
C Write loss coefficient summary
      WRITE(30,13) ELSS1,ELSS2
   13 FORMAT(1X,'PROFILE LOSS SUMMARY',/,1X,'********************',/,
     $       1X,'MEASURED PROFILE LOSS COEFFICICNET     =',
     $       1X,F11.9,/,
     $       1X,'CORRECTED PROFILE LOSS COEFFICICNET    =',
     $       1X,F11.9,//)
C Write boundary layer summary
      WRITE(30,31) SDEL2,CSDEL2,PDEL2,CPDEL2,DEL2,CDEL2
   31 FORMAT(1X,'MOMENTUM DEFICIENCY THICKNESS CALCULATIONS',/,
     $       1X,'******************************************',/,
     $       1X,'M. SUCTION SIDE MOMENTUM DEFICIENCY THICKNESS  [MM] =',
     $       1X,F12.9,/,
     $       1X,'C. SUCTION SIDE MOMENTUM DEFICIENCY THICKNESS  [MM] =',
     $       1X,F12.9,/,
     $       1X,'M. PRESSURE SIDE MOMENTUM DEFICIENCY THICKNESS [MM] =',
     $       1X,F12.9,/,
     $       1X,'C. PRESSURE SIDE MOMENTUM DEFICIENCY THICKNESS [MM] =',
     $       1X,F12.9,/,
     $       1X,'M. AVERAGE MOMENTUM DEFICIENCY THICKNESS       [MM] =',
     $       1X,F12.9,/,
     $       1X,'C. AVERAGE MOMENTUM DEFICIENCY THICKNESS       [MM] =',
     $       1X,F12.9,//)
C Write the blade tangential pressure force calculated
      WRITE(30,36) PFT,CPFT
   36 FORMAT(1X,'PRESSURE TANGENTIAL BLADE FORCE',/,
     $       1X,'*******************************',/,
     $       1X,'MEASURED [N/M]',2X,' = ',F10.4,/,
     $       1X,'CORRECTED [N/M]',2X,'= ',F10.4,//)
C Write correction parameters used
      WRITE(30,42) IVX,IVXRF,IREX,IREXRF,IRHO,IRHORF
   42 FORMAT(1X,'CORRECTION PARAMETERS',/,
     $       1X,'*********************',/,
     $       1X,'INLET AXIAL VELOCITY [M/SEC]',6X,'= ',F8.3,/,
     $       1X,'REFERENCE AXIAL VELOCITY [M/SEC]',2X,'= ',F8.3,/,
     $       1X,'INLET AXIAL REYNOLDS NUMBER',7X,'= ',F8.1,/,
     $       1X,'REFERENCE AXIAL REYNOLDS NUMBER',3X,'= ',F8.1,/,
     $       1X,'INLET DENSITY [KG/M**3]',11X,'= ',F8.3,/,
     $       1X,'REFERENCE DENSITY [KG/M**3]',7X,'= ',F8.3,//)
C Write output to xplot.dat
      WRITE(80,*) 'VARIABLES="Y",',
     $           '"P_t [Pa]",',
     $           '"P_s [Pa]",',
     $           '"P_d [Pa]",',
     $           '"V [m/sec]"',
     $           '"V_x [m/sec]"',
     $           '"V_y [m/sec]"'
      WRITE(80,33) STGR
      DO 38 J=BPL,EPL
         WRITE(80,*) NY(J),CDPT(J),CDPS(J),CDPD(J),
     $               CDV(J),CDVX(J),CDVY(J)
   38 CONTINUE
C Write output to nxplot.dat
      WRITE(50,*) 'VARIABLES="Y",',
     $           '"P_t_n",',
     $           '"P_s_n",',
     $           '"P_d_n",',
     $            '"V_n"',
     $            '"V_x_n"',
     $            '"V_y_n"'
      WRITE(50,33) STGR
      DO 44 J=BPL,EPL
         WRITE(50,*) NY(J),NDPT(J),NDPS(J),NDPD(J),
     $               NDV(J),NDVX(J),NDVY(J)
   44 CONTINUE
      PRINT*, 'END OF ANALYSIS'
      END
************************************************************************
* This subroutine calculates the density of humid air.                 *
* Input Files : N/A                                                    *
* Output Files: N/A                                                    *
************************************************************************
* Hicham A Chibli - Texas A&M University - 10/30/09                    *
************************************************************************
      SUBROUTINE DEN(ATS,APS,RH,RHO)
C define variables, parameters and constants to use
      DOUBLE PRECISION RA,RV,KELVIN,SVP,VPP,DPP,RH,APS,ATS,RHO
      PARAMETER(RA=286.9D0,RV=461.495D0,KELVIN=273.15D0)
C calculate saturation vapor pressure
      SVP=6.1078D0*100.0D0*10.0D0**((7.5D0*(ATS-KELVIN))/
     $    (237.3D0+ATS-KELVIN))
C calculate vapor partial pressure
      VPP=SVP*RH/100.0D0
C calculate dry air partial pressure
      DPP=APS-VPP
C calculate flow density
      RHO=DPP/(RA*ATS)+VPP/(RV*ATS)
      RETURN
      END
************************************************************************
* This subroutine calculates the static temperature.                   *
* Input Files : N/A                                                    *
* Output Files: N/A                                                    *
************************************************************************
* Hicham A Chibli - Texas A&M University - 10/05/10                    *
************************************************************************
      SUBROUTINE TSTAT(ATT,APT,APS,ATS)
C define variables and constants
      DOUBLE PRECISION ATT,ATS,APT,APS,GAMMA
      PARAMETER(GAMMA=1.4D0)
C use ideal gas relationship to calculate the static temperature
      ATS=ATT*(APS/APT)**((GAMMA-1.0D0)/GAMMA)
      RETURN
      END
************************************************************************
* This subroutine reverse calculates the corrected static temperature. *
* Input Files : N/A                                                    *
* Output Files: N/A                                                    *
************************************************************************
* Hicham A Chibli - Texas A&M University - 10/05/10                    *
************************************************************************
      SUBROUTINE RSTAT(CPS,CRHO,CBAR,CRH,CTSA)
C define variables and constants
      DOUBLE PRECISION MARGIN,ERROR,CTSA,CPS,CRHO,CBAR,CRH,
     $                 CSVP,CVPP,CDPP
      PARAMETER(MARGIN=0.0001D0)
C estimate the initial guess for the static temperature
      CTSA=(CPS+CBAR)/(286.9D0*CRHO)
C calculate the saturation vapor pressure
    9 CSVP=6.1078D0*100.0D0*10.0D0**((7.5D0*(CTSA-273.15D0))/
     $    (237.3D0+CTSA-273.15D0))
C calculate vapor partial pressure
      CVPP=CSVP*CRH/100.0D0
C calculate dry air partial pressure
```

```
      CDPP=(CPS+CBAR)-CVPP
C calculate the error
      ERROR=CRHO-CDPP/(286.9D0*CTSA)-CVPP/(461.495D0*CTSA)
      IF(DABS(ERROR).LE.MARGIN) GO TO 8
C loop
      CTSA=CDPP/((CRHO-CVPP/(461.495D0*CTSA))*286.9D0)
      GO TO 9
    8 RETURN
      END
************************************************************************
* This subroutine arithmetically averages data in an array.            *
* Input Files : N/A                                                    *
* Output Files: N/A                                                    *
************************************************************************
* Hicham A Chibli - Texas A&M University - 10/05/10                    *
************************************************************************
      SUBROUTINE ARTH(ARRAY,LENGTH,ACTUAL,WIDTH,COLUMN,AVER)
      INTEGER LENGTH,ACTUAL,WIDTH,COLUMN,N
      DOUBLE PRECISION AVER,TEMP
      DOUBLE PRECISION ARRAY(LENGTH,WIDTH)
      TEMP=0.0D0
      DO 1 N=1,ACTUAL
         TEMP=TEMP+ARRAY(N,COLUMN)
    1 CONTINUE
      AVER=TEMP/ACTUAL
      RETURN
      END
************************************************************************
C Define subroutine PTRIM
************************************************************************
      SUBROUTINE PTRIM(K,L,M,N,X,Y,Z)
      INTEGER M,N,K,J,I,L
      DOUBLE PRECISION X,Y
      DOUBLE PRECISION Z(K)
      I=0
      DO 14 J=1,L
         IF(I.EQ.0) THEN
           IF(Z(J).GE.X) THEN
             I=1
             M=J
           ENDIF
         ENDIF
   14 CONTINUE
      I=0
      DO 15 J=1,L
         IF(I.EQ.0) THEN
           IF(Z(J).GT.Y) THEN
             I=1
             N=J-1
           ENDIF
         ENDIF
   15 CONTINUE
      RETURN
      END
************************************************************************
C Define subroutine WAKE
C Use this subprogram with highly noisy data sets
************************************************************************
      SUBROUTINE WAKE(K,M,N,I,J,S,LL,UL,Z)
      INTEGER M,N,K,J,I,L
      DOUBLE PRECISION ERROR,LL,UL,S
      DOUBLE PRECISION Z(K),LAREA(M:N-1),UAREA(M+1:N)
      DO 16 L=M,N-1,1
C Integrate using Trapezoidal Rule from lower end
         LAREA(L)=(S*0.5D0)*(Z(L+1)+Z(L))
   16 CONTINUE
C Check convergence criteria
```

```
      DO 17 L=M,N-2,1
         IF(LAREA(L).GE.LAREA(L+1)) THEN
           ERROR=1.0D0-LAREA(L+1)/LAREA(L)
         ELSE
             ERROR=1.0D0-LAREA(L)/LAREA(L+1)
         ENDIF
         IF(ERROR.LE.LL) THEN
           I=L+1
           GO TO 18
         ENDIF
   17 CONTINUE
   18 DO 19 L=N,M+1,-1
C Integrate using Simpson's Rule from upper end after rotation
         UAREA(L)=(S*0.5D0)*(Z(L-1)+Z(L))
   19 CONTINUE
C Check convergence criteria
      DO 20 L=N,M+2,-1
         IF(UAREA(L).GE.UAREA(L-1)) THEN
           ERROR=1.0D0-UAREA(L-1)/UAREA(L)
         ELSE
             ERROR=1.0D0-UAREA(L)/UAREA(L-1)
         ENDIF
         IF(ERROR.LE.UL) THEN
          J=L-1
           GO TO 21
         ENDIF
   20 CONTINUE
   21 RETURN
      END
************************************************************************
C Define function DEL
************************************************************************
      DOUBLE PRECISION FUNCTION DEL(U,UMEAN)
      DOUBLE PRECISION U,UMEAN
      DEL=(U/UMEAN)*(1.0D0-U/UMEAN)
      RETURN
      END
************************************************************************
* This program reads the total exit temperature (C), the barometric    *
* pressure (inch HG) and the relative humidity (%) with the            *
* corresponding time of readings, then interpolates the data using a   *
* shape preserving cubic spline to generate smooth temperature,        *
* pressure and humidity profiles to be used in density calculations.   *
* Input Files : fpt.dat, fht.dat, info.dat                             *
* Output Files: opt.dat, orh.dat, ambnt.dat                            *
* Experimental Procedure: Cascade Facility of the DOOSAN II Blade      *
************************************************************************
* Hicham A Chibli - Texas A&M University - 09/10/10                    *
************************************************************************
      PROGRAM UTL2
C Define variables, parameters and constants to use
      DOUBLE PRECISION STEP,EXACT,BASE,J,K
      INTEGER MTOTAL,TOTAL,HOUR,MINUTE,PTCNT,RHCNT,I,ITYPE,IDER
      PARAMETER(MTOTAL=1000,ITYPE=3,IDER=0)
      DOUBLE PRECISION PB(MTOTAL),EPB(MTOTAL),TT(MTOTAL),ETT(MTOTAL),
     $                 RH(MTOTAL),ERH(MTOTAL),PT(MTOTAL),HT(MTOTAL),
     $                 TME(MTOTAL),LOCATE(MTOTAL),PTLCT(MTOTAL),
     $                 RHLCT(MTOTAL)
C Read data from iput files fpt.dat and fht.dat
      OPEN(UNIT=10,FILE='./in/fpt.dat',STATUS='OLD')
      OPEN(UNIT=20,FILE='./in/fht.dat',STATUS='OLD')
      OPEN(UNIT=60,FILE='./in/info.dat',STATUS='OLD')
      READ(60,*)
      READ(60,*) TOTAL
      READ(60,*)
      READ(60,*) STEP
      PTCNT=1
```

```
    1 READ(10,2,END=3) HOUR,MINUTE,J,TT(PTCNT)
    2 FORMAT(I2,I2,1X,F5.2,1X,F4.1)
      PB(PTCNT)=3386.388D0*J
      EXACT=DBLE(HOUR)*60.D0+DBLE(MINUTE)
      IF(PTCNT.EQ.1) BASE=EXACT
      PT(PTCNT)=EXACT-BASE
      PTCNT=PTCNT+1
      GO TO 1
    3 RHCNT=1
    6 READ(20,4,END=5) HOUR,MINUTE,RH(RHCNT)
    4 FORMAT(I2,I2,1X,F4.1)
      EXACT=DBLE(HOUR)*60.D0+DBLE(MINUTE)
      IF(PTCNT.EQ.1) BASE=EXACT
      HT(RHCNT)=EXACT-BASE
      RHCNT=RHCNT+1
      GO TO 6
    5 PRINT*, 'END READING DATA'
C Correct array sizes
      PTCNT=PTCNT-1
      RHCNT=RHCNT-1
C Populate location arrays
      EXACT=(PT(PTCNT)+HT(RHCNT))/2.D0
      DO 9 I=1,TOTAL
           TME(I)=DBLE(I-1)*EXACT/DBLE(TOTAL-1)
           LOCATE(I)=DBLE(I-1)*STEP
    9 CONTINUE
      DO 14 I=1,PTCNT
            PTLCT(I)=(PT(I)/PT(PTCNT))*STEP*DBLE(TOTAL-1)
   14 CONTINUE
      DO 15 I=1,RHCNT
            RHLCT(I)=(PT(I)/PT(RHCNT))*STEP*DBLE(TOTAL-1)
   15 CONTINUE
C Compute cubic spline interpolant using IMSL subroutine DSPLEZ
      CALL DSPLEZ(PTCNT,PT,PB,ITYPE,IDER,TOTAL,TME,EPB)
      CALL DSPLEZ(PTCNT,PT,TT,ITYPE,IDER,TOTAL,TME,ETT)
      CALL DSPLEZ(RHCNT,HT,RH,ITYPE,IDER,TOTAL,TME,ERH)
C Write data to output files ambnt.dat,opt.dat and orh.dat
      OPEN(UNIT=30,FILE='./out/ambnt.dat',STATUS='UNKNOWN')
      OPEN(UNIT=40,FILE='./out/opt.dat',STATUS='UNKNOWN')
      OPEN(UNIT=50,FILE='./out/orh.dat',STATUS='UNKNOWN')
      DO 7 I=1,TOTAL
         WRITE(30,8) EPB(I),ETT(I),ERH(I),LOCATE(I)
    8    FORMAT(1X,F9.1,1X,F5.2,1X,F4.1,1X,F5.1)
    7 CONTINUE
      DO 10 I=1,PTCNT
         WRITE(40,11) PB(I),TT(I),PTLCT(I)
   11    FORMAT(1X,F9.1,1X,F5.2,1X,F5.1)
   10 CONTINUE
      DO 12 I=1,RHCNT
         WRITE(50,13) RH(I),RHLCT(I)
   13    FORMAT(1X,F4.1,1X,F5.1)
   12 CONTINUE
      PRINT*, ' END OF ANALYSIS'
      END
************************************************************************
*  This code automates the analysis of Cascade-II numerical results    *
*  and computes the loss coefficients as well as the forces and        *
*  prepares the output plot data files.                                *
************************************************************************
*  Hicham A Chibli - Texas A&M University - 08/2010                    *
************************************************************************
      PROGRAM NCAS3
C define variables and parameters
      REAL PI,CHORD,PITCH,EANGLE,STGR,
     $     REREF,REREFX,VREF,VREFX,RHOREF,DUMMY
      INTEGER W1,W2,L1,L2,C
      PARAMETER(W1=10,W2=2,L1=1000,L2=1000,C=2,PI=3.14159265,
     $          EANGLE=77.14,CHORD=0.15,PITCH=0.09540,
     $          REREF=200000.0,VREF=20.0,
     $          REREFX=173655.7,VREFX=16.451,
     $          RHOREF=1.240)
      INTEGER N,M,K,BDE,IPE,OPE,FPE,NUM,OW,OS,FW,FS,OE,FE,REV
      REAL REIN,REINX,RHOIN,VIN,VINX,MUIN,PIN,
     $     POLSS,PFLSS,POLSSX,PFLSSX,MLOSS,PMLOSS,
     $     MIMF,MOMF,MFMF,CIMF,COMF,CFMF,
     $     MOFT,MFFT,COFT,CFFT,BFT,CBFT,
     $     MOFN,MFFN,COFN,CFFN,
     $     IMXM,IMYM,OMXM,OMYM,FMXM,FMYM,
     $     IMXC,IMYC,OMXC,OMYC,FMXC,FMYC,
     $     TEMP,BMIN,BMAX,PMAX,PMIN
      REAL INP(C+1,L1,W1),OUTP(C+1,L1,W1),FARP(C+1,L1,W1),
     $     BD(L2,W2),CP(L2,W2),
     $     AINP(C+1,W1),AOUTP(C+1,W1),AFARP(C+1,W1),
     $     OLOSS(C+1),FLOSS(C+1),
     $     TB(W2),TP(W1)
      CHARACTER LOCATE*25,CASNUM*25,NEW*50
C open file list and directory
      OPEN(UNIT=19,FILE=’./list/list.txt’,STATUS=’OLD’)
      OPEN(UNIT=20,FILE=’./list/stagger.txt’,STATUS=’OLD’)
      OPEN(UNIT=30,FILE=’./list/rev.txt’,STATUS=’OLD’)
      OPEN(UNIT=21,FILE=’./summary/all.dat’,STATUS=’UNKNOWN’)
      OPEN(UNIT=22,FILE=’./summary/cnplot.dat’,STATUS=’UNKNOWN’)
      OPEN(UNIT=23,FILE=’./summary/ccp.dat’,STATUS=’UNKNOWN’)
      OPEN(UNIT=24,FILE=’./summary/closs.dat’,STATUS=’UNKNOWN’)
C write header to all-summary files
      WRITE(21,*) ’VARIABLES="‘g [deg]",’,
     $            ’"‘z_o_u_t___m_e_a_s_u_r_e_d",’,
     $            ’"‘z_o_u_t___P_f_e_i_l",’,
     $            ’"‘z_o_u_t___P_f_e_i_l___x",’,
     $            ’"‘z_f_a_r___m_e_a_s_u_r_e_d",’,
     $            ’"‘z_f_a_r___P_f_e_i_l",’,
     $            ’"‘z_f_a_r___P_f_e_i_l___x",’,
     $            ’"‘z_m_i_x___m_e_a_s_u_r_e_d",’,
     $            ’"‘z_m_i_x___c_o_r_r_e_c_t_e_d",’,
     $            ’"Drag/Lift_o_u_t___m_e_a_s_u_r_e_d",’,
     $            ’"Drag/Lift_o_u_t___c_o_r_r_e_c_t_e_d",’,
     $            ’"Drag/Lift_f_a_r___m_e_a_s_u_r_e_d",’,
     $            ’"Drag/Lift_f_a_r___c_o_r_r_e_c_t_e_d",’,
     $            ’"Re_x___m_e_a_s_u_r_e_d","Re_x___r_e_f",’,
     $            ’"‘r_m_e_a_s_u_r_e_d [kg/m^3]",’,
     $            ’"‘r_c_o_r_r_e_c_t_e_d [kg/m^3]",’,
     $            ’"mflow_m_e_a_s_u_r_e_d [kg/sec]",’,
     $            ’"mflow_c_o_r_r_e_c_t_e_d [kg/sec]",’,
     $            ’"F_T___m_e_a_s_u_r_e_d [N]",’,
     $            ’"F_T___c_o_r_r_e_c_t_e_d [N]"’
      WRITE(24,*) ’VARIABLES="‘g [deg]",’,’"‘z",’,’"error"’
      WRITE(24,*) ’ZONE T=" numerical"’
      WRITE(22,*) ’VARIABLES="Y",’,
     $            ’"P_t_n",’,
     $            ’"P_s_n",’,
     $            ’"P_d_n",’,
     $            ’"V_n"’,
     $            ’"V_x_n"’,
     $            ’"V_y_n"’
      WRITE(23,*) ’VARIABLES="X",’,’"C_p"’
C read list of files and working directory
      NUM=0
      REV=0
      READ(20,*)
      READ(19,*)
      READ(19,*) LOCATE
      READ(19,*)
   30 READ(19,*,END=31) CASNUM
      READ(20,*) STGR
      READ(30,*) REV
      PRINT*
      NUM=NUM+1
C trim blanks from strings
      CALL CLEAN(LOCATE,N)
      CALL CLEAN(CASNUM,M)
      NEW='./'//LOCATE(1:N)//'/'//CASNUM(1:M)
      CALL CLEAN(NEW,K)
      PRINT*, '**************************************************'
      PRINT*
      PRINT*, 'CASE NUMBER: ',CASNUM(1:M)
C write case header to all-summary files
      WRITE(23,93) STGR
      WRITE(22,93) STGR
   93 FORMAT('ZONE T="num.  ‘g = ',F5.1,' ^o"')
C open files for input
      OPEN(UNIT=15,FILE=NEW(1:K)//'/in/blade.dat',STATUS='OLD')
      OPEN(UNIT=16,FILE=NEW(1:K)//'/in/inlet.dat',STATUS='OLD')
      OPEN(UNIT=17,FILE=NEW(1:K)//'/in/outlet.dat',STATUS='OLD')
      OPEN(UNIT=18,FILE=NEW(1:K)//'/in/far.dat',STATUS='OLD')
C open files for output
      OPEN(UNIT=25,FILE=NEW(1:K)//'/out/cp.dat',STATUS='UNKNOWN')
      OPEN(UNIT=26,FILE=NEW(1:K)//'/out/cplt.dat',STATUS='UNKNOWN')
      OPEN(UNIT=27,FILE=NEW(1:K)//'/out/ncplt.dat',STATUS='UNKNOWN')
      OPEN(UNIT=29,FILE=NEW(1:K)//'/out/case.dat',STATUS='UNKNOWN')
C zero profile data arrays used
      DO 1 N=1,C+1
         DO 2 M=1,L1
            DO 3 K=1,W1
               INP(N,M,K)=0.0
               OUTP(N,M,K)=0.0
               FARP(N,M,K)=0.0
    3       CONTINUE
    2    CONTINUE
    1 CONTINUE
C zero average data arrays used
      DO 4 N=1,C+1
         DO 5 M=1,W1
            AINP(N,M)=0.0
            AOUTP(N,M)=0.0
            AFARP(N,M)=0.0
    5    CONTINUE
    4 CONTINUE
C zero blade pressure ditribution arrays used
      DO 6 N=1,L2
         DO 7 M=1,W2
            BD(N,M)=0.0
    7    CONTINUE
    6 CONTINUE
C zero blade cp ditribution array used
      DO 8 N=1,L2
         DO 9 M=1,W2
            CP(N,M)=0.0
    9    CONTINUE
    8 CONTINUE
C zero loss and array used
      DO 10 N=1,C+1
         OLOSS(N)=0.0
         FLOSS(N)=0.0
   10 CONTINUE
C zero counters
      BDE=0
      IPE=0
      OPE=0
      FPE=0
C read the blade pressure ditribution input data and find the bounds
      BMIN=10000.0
      BMAX=-10000.0
   11 READ(15,*,END=12) TB
      BDE=BDE+1
      DO 13 N=1,W2
C CFX x-ref is opposite direction of experiments
         BD(BDE,N)=TB(N)
         IF(N.EQ.1) THEN
           BD(BDE,N)=-1.0*BD(BDE,N)
           IF(BD(BDE,N).GT.BMAX) BMAX=BD(BDE,N)
           IF(BD(BDE,N).LT.BMIN) BMIN=BD(BDE,N)
         ENDIF
   13 CONTINUE
C repeat
      GO TO 11
   12 BDE=BDE+1
      BD(BDE,1)=BD(1,1)
      BD(BDE,2)=BD(1,2)
      PRINT*
      PRINT*, '*************************************************'
      PRINT*, 'READING BLADE PRESSURE DISTRIBUTION DATA COMPLETE'
      PRINT*, 'NUMBER OF POINTS PER SET:         ',BDE
      PRINT*, 'MAXIMUM NUMBER OF POINTS PER SET: ',L2
      PRINT*
C read the inlet profile input data
C M
   14 READ(16,*,END=15) DUMMY,TP
C determine whether the profile needs be reversed
      IF(REV.EQ.1) TP(1)=1.0-TP(1)
      IPE=IPE+1
      DO 16 N=1,W1
         INP(1,IPE,N)=TP(N)
C CFX x-ref is opposite direction of experiments
         IF(N.EQ.9) INP(1,IPE,N)=-1.0*INP(1,IPE,N)
   16 CONTINUE
C reset the 8th column to handle dynamic pressure head
      INP(1,IPE,8)=INP(1,IPE,2)-INP(1,IPE,3)
C repeat
      GO TO 14
   15 PRINT*, 'READING INLET PROFILE DATA COMPLETE'
      PRINT*, 'NUMBER OF POINTS PER SET:         ',IPE
      PRINT*, 'MAXIMUM NUMBER OF POINTS PER SET: ',L1
      PRINT*
C read the outlet profile input data
C M
   17 READ(17,*,END=18) DUMMY,TP
C determine whether the profile needs be reversed
      IF(REV.EQ.1) TP(1)=1.0-TP(1)
      OPE=OPE+1
      DO 19 N=1,W1
         OUTP(1,OPE,N)=TP(N)
C CFX x-ref is opposite direction of experiments
         IF(N.EQ.9) OUTP(1,OPE,N)=-1.0*OUTP(1,OPE,N)
   19 CONTINUE
C reset the 8th column to handle dynamic pressure head
      OUTP(1,OPE,8)=OUTP(1,OPE,2)-OUTP(1,OPE,3)
C repeat
      GO TO 17
   18 PRINT*, 'READING OUTLET PROFILE DATA COMPLETE'
      PRINT*, 'NUMBER OF POINTS PER SET:         ',OPE
      PRINT*, 'MAXIMUM NUMBER OF POINTS PER SET: ',L1
      PRINT*
C read the far profile input data
C M
   20 READ(18,*,END=21) DUMMY,TP
C determine whether the profile needs be reversed
      IF(REV.EQ.1) TP(1)=1.0-TP(1)
      FPE=FPE+1
      DO 22 N=1,W1
         FARP(1,FPE,N)=TP(N)
C CFX x-ref is opposite direction of experiments
         IF(N.EQ.9) FARP(1,FPE,N)=-1.0*FARP(1,FPE,N)
   22 CONTINUE
C reset the 8th column to handle dynamic pressure head
      FARP(1,FPE,8)=FARP(1,FPE,2)-FARP(1,FPE,3)
C repeat
      GO TO 20
   21 PRINT*, 'READING FAR PROFILE DATA COMPLETE'
      PRINT*, 'NUMBER OF POINTS PER SET:         ',FPE
      PRINT*, 'MAXIMUM NUMBER OF POINTS PER SET: ',L1
      PRINT*
C manipulate the profile data to establish periodic conditions
      DO 58 K=2,W1
            INP(1,1,K)=(INP(1,1,K)+INP(1,IPE,K))/2.0
            INP(1,IPE,K)=INP(1,1,K)
            OUTP(1,1,K)=(OUTP(1,1,K)+OUTP(1,OPE,K))/2.0
            OUTP(1,OPE,K)=OUTP(1,1,K)
            FARP(1,1,K)=(FARP(1,1,K)+FARP(1,FPE,K))/2.0
            FARP(1,FPE,K)=FARP(1,1,K)
   58 CONTINUE
C sort the far arrays in ascending order of axial location
      DO 68 K=1,IPE-1
         DO 69 N=K+1,IPE
            IF(INP(1,K,1).GT.INP(1,N,1)) THEN
              DO 70 M=1,W1
                 TEMP=INP(1,K,M)
                 INP(1,K,M)=INP(1,N,M)
                 INP(1,N,M)=TEMP
   70         CONTINUE
            ENDIF
   69    CONTINUE
   68 CONTINUE
C sort the out arrays in ascending order of axial location
      DO 62 K=1,OPE-1
         DO 63 N=K+1,OPE
            IF(OUTP(1,K,1).GT.OUTP(1,N,1)) THEN
              DO 64 M=1,W1
                 TEMP=OUTP(1,K,M)
                 OUTP(1,K,M)=OUTP(1,N,M)
                 OUTP(1,N,M)=TEMP
   64         CONTINUE
            ENDIF
   63    CONTINUE
   62 CONTINUE
C sort the far arrays in ascending order of axial location
      DO 65 K=1,FPE-1
         DO 66 N=K+1,FPE
            IF(FARP(1,K,1).GT.FARP(1,N,1)) THEN
              DO 67 M=1,W1
                 TEMP=FARP(1,K,M)
                 FARP(1,K,M)=FARP(1,N,M)
                 FARP(1,N,M)=TEMP
   67         CONTINUE
            ENDIF
   66    CONTINUE
   65 CONTINUE
C average inlet profile data
C M
      CALL INTG(INP,C+1,L1,W1,1,IPE,2,AINP(1,2),TEMP,MIMF)
      CALL INTG(INP,C+1,L1,W1,1,IPE,3,AINP(1,3),TEMP,MIMF)
      CALL INTG(INP,C+1,L1,W1,1,IPE,4,AINP(1,4),TEMP,MIMF)
      CALL INTG(INP,C+1,L1,W1,1,IPE,5,AINP(1,5),TEMP,MIMF)
      CALL INTG(INP,C+1,L1,W1,1,IPE,6,AINP(1,6),TEMP,MIMF)
      CALL INTG(INP,C+1,L1,W1,1,IPE,8,AINP(1,8),TEMP,MIMF)
      CALL INTG(INP,C+1,L1,W1,1,IPE,9,AINP(1,9),IMXM,MIMF)
      CALL INTG(INP,C+1,L1,W1,1,IPE,10,AINP(1,10),IMYM,MIMF)
      PIN=AINP(1,2)
      VIN=AINP(1,4)
      VINX=AINP(1,9)
      RHOIN=AINP(1,5)
      MUIN=AINP(1,6)
      REIN=CHORD*RHOIN*VIN/MUIN
      REINX=CHORD*RHOIN*VINX/MUIN
C average outlet profile data
C M
      CALL INTG(OUTP,C+1,L1,W1,1,OPE,2,AOUTP(1,2),TEMP,MOMF)
      CALL INTG(OUTP,C+1,L1,W1,1,OPE,3,AOUTP(1,3),TEMP,MOMF)
      CALL INTG(OUTP,C+1,L1,W1,1,OPE,4,AOUTP(1,4),TEMP,MOMF)
      CALL INTG(OUTP,C+1,L1,W1,1,OPE,5,AOUTP(1,5),TEMP,MOMF)
      CALL INTG(OUTP,C+1,L1,W1,1,OPE,6,AOUTP(1,6),TEMP,MOMF)
      CALL INTG(OUTP,C+1,L1,W1,1,OPE,8,AOUTP(1,8),TEMP,MOMF)
      CALL INTG(OUTP,C+1,L1,W1,1,OPE,9,AOUTP(1,9),OMXM,MOMF)
      CALL INTG(OUTP,C+1,L1,W1,1,OPE,10,AOUTP(1,10),OMYM,MOMF)
C average far profile data
C M
      CALL INTG(FARP,C+1,L1,W1,1,FPE,2,AFARP(1,2),TEMP,MFMF)
      CALL INTG(FARP,C+1,L1,W1,1,FPE,3,AFARP(1,3),TEMP,MFMF)
      CALL INTG(FARP,C+1,L1,W1,1,FPE,4,AFARP(1,4),TEMP,MFMF)
      CALL INTG(FARP,C+1,L1,W1,1,FPE,5,AFARP(1,5),TEMP,MFMF)
      CALL INTG(FARP,C+1,L1,W1,1,FPE,6,AFARP(1,6),TEMP,MFMF)
      CALL INTG(FARP,C+1,L1,W1,1,FPE,8,AFARP(1,8),TEMP,MFMF)
      CALL INTG(FARP,C+1,L1,W1,1,FPE,9,AFARP(1,9),FMXM,MFMF)
      CALL INTG(FARP,C+1,L1,W1,1,FPE,10,AFARP(1,10),FMYM,MFMF)
C compute cp distribution around blade
C M
      DO 23 N=1,BDE
         CP(N,1)=(BD(N,1)-BMIN)/(BMAX-BMIN)
         CP(N,2)=(BD(N,2)-PIN)/PIN
   23 CONTINUE
C perform correction of inlet profiles
C this is done be specifying  axial-inlet reference
C parameters and correcting all the values accordingly
      DO 24 N=1,IPE
C C1
         INP(2,N,1)=INP(1,N,1)
         INP(2,N,2)=INP(1,N,2)*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
         INP(2,N,3)=INP(1,N,3)*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
         INP(2,N,8)=INP(1,N,8)*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
         INP(2,N,5)=INP(1,N,5)*RHOREF/RHOIN
         INP(2,N,4)=INP(1,N,4)*VREFX/VINX
         INP(2,N,9)=INP(1,N,9)*VREFX/VINX
         INP(2,N,10)=INP(1,N,10)*VREFX/VINX
   24 CONTINUE
C perform correction of outlet profiles
      DO 25 N=1,OPE
C C1
         OUTP(2,N,1)=OUTP(1,N,1)
         OUTP(2,N,2)=OUTP(1,N,2)*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
         OUTP(2,N,3)=OUTP(1,N,3)*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
         OUTP(2,N,8)=OUTP(1,N,8)*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
         OUTP(2,N,5)=OUTP(1,N,5)*RHOREF/RHOIN
         OUTP(2,N,4)=OUTP(1,N,4)*VREFX/VINX
         OUTP(2,N,9)=OUTP(1,N,9)*VREFX/VINX
         OUTP(2,N,10)=OUTP(1,N,10)*VREFX/VINX
   25 CONTINUE
C perform correction of far profiles
      DO 26 N=1,FPE
C C1
         FARP(2,N,1)=FARP(1,N,1)
         FARP(2,N,2)=FARP(1,N,2)*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
         FARP(2,N,3)=FARP(1,N,3)*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
         FARP(2,N,8)=FARP(1,N,8)*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
         FARP(2,N,5)=FARP(1,N,5)*RHOREF/RHOIN
         FARP(2,N,4)=FARP(1,N,4)*VREFX/VINX
         FARP(2,N,9)=FARP(1,N,9)*VREFX/VINX
         FARP(2,N,10)=FARP(1,N,10)*VREFX/VINX
   26 CONTINUE
C average corrected inlet profile data
C C1
      CALL INTG(INP,C+1,L1,W1,2,IPE,2,AINP(2,2),TEMP,CIMF)
      CALL INTG(INP,C+1,L1,W1,2,IPE,3,AINP(2,3),TEMP,CIMF)
      CALL INTG(INP,C+1,L1,W1,2,IPE,4,AINP(2,4),TEMP,CIMF)
      CALL INTG(INP,C+1,L1,W1,2,IPE,5,AINP(2,5),TEMP,CIMF)
      CALL INTG(INP,C+1,L1,W1,2,IPE,8,AINP(2,8),TEMP,CIMF)
      CALL INTG(INP,C+1,L1,W1,2,IPE,9,AINP(2,9),IMXC,CIMF)
      CALL INTG(INP,C+1,L1,W1,2,IPE,10,AINP(2,10),IMYC,CIMF)
C average corrected outlet profile data
C C1
      CALL INTG(OUTP,C+1,L1,W1,2,OPE,2,AOUTP(2,2),TEMP,COMF)
      CALL INTG(OUTP,C+1,L1,W1,2,OPE,3,AOUTP(2,3),TEMP,COMF)
      CALL INTG(OUTP,C+1,L1,W1,2,OPE,5,AOUTP(2,5),TEMP,COMF)
      CALL INTG(OUTP,C+1,L1,W1,2,OPE,8,AOUTP(2,8),TEMP,COMF)
      CALL INTG(OUTP,C+1,L1,W1,2,OPE,9,AOUTP(2,9),OMXC,COMF)
      CALL INTG(OUTP,C+1,L1,W1,2,OPE,10,AOUTP(2,10),OMYC,COMF)
C average corrected far profile data
C C1
      CALL INTG(FARP,C+1,L1,W1,2,FPE,2,AFARP(2,2),TEMP,CFMF)
      CALL INTG(FARP,C+1,L1,W1,2,FPE,3,AFARP(2,3),TEMP,CFMF)
      CALL INTG(FARP,C+1,L1,W1,2,FPE,5,AFARP(2,5),TEMP,CFMF)
      CALL INTG(FARP,C+1,L1,W1,2,FPE,8,AFARP(2,8),TEMP,CFMF)
      CALL INTG(FARP,C+1,L1,W1,2,FPE,9,AFARP(2,9),FMXC,CFMF)
      CALL INTG(FARP,C+1,L1,W1,2,FPE,10,AFARP(2,10),FMYC,CFMF)
C compute out loss coefficients
      OLOSS(1)=(AINP(1,2)-AOUTP(1,2))/AOUTP(1,8)
      OLOSS(2)=(AINP(2,2)-AOUTP(2,2))/AOUTP(2,8)
C compute far loss coefficients
      FLOSS(1)=(AINP(1,2)-AFARP(1,2))/AFARP(1,8)
      FLOSS(2)=(AINP(2,2)-AFARP(2,2))/AFARP(2,8)
C compute the PFEIL corrected loss coefficient
      POLSS=OLOSS(1)*(REIN/REREF)**0.2
      PFLSS=FLOSS(1)*(REIN/REREF)**0.2
      POLSSX=OLOSS(1)*(REINX/REREFX)**0.2
      PFLSSX=FLOSS(1)*(REINX/REREFX)**0.2
C compute the mixing loss
      MLOSS=FLOSS(1)-OLOSS(1)
      PMLOSS=0.5*(PFLSSX-POLSSX+PFLSS-POLSS)
C compute the measured blade reaction forces (normal and tengential)
      MOFN=ABS((IMXM-OMXM)-(AINP(1,3)-AOUTP(1,3))*PITCH)
      MFFN=ABS((IMXM-FMXM)-(AINP(1,3)-AFARP(1,3))*PITCH)
      MOFT=ABS(IMYM-OMYM)
      MFFT=ABS(IMYM-FMYM)
C compute the corrected blade reaction forces (normal and tengential)
      COFN=ABS((IMXC-OMXC)-(AINP(2,3)-AOUTP(2,3))*PITCH)
      CFFN=ABS((IMXC-FMXC)-(AINP(2,3)-AFARP(2,3))*PITCH)
      COFT=ABS(IMYC-OMYC)
      CFFT=ABS(IMYC-FMYC)
C find the center of the middle wake in the out profile
      TEMP=100000.0
      DO 47 N=OPE/3,2*OPE/3
         IF(OUTP(1,N,2).LT.TEMP) THEN
           TEMP=OUTP(1,N,2)
           OW=N
         ENDIF
   47 CONTINUE
C find the center of the middle wake in the far profile
      TEMP=100000.0
      DO 48 N=FPE/3,2*FPE/3
         IF(FARP(1,N,2).LT.TEMP) THEN
           TEMP=FARP(1,N,2)
           FW=N
         ENDIF
   48 CONTINUE
C non-dimensionalize the profile at inlet plane
      PMAX=-10000.0
      PMIN=10000.0
      DO 32 N=IPE/3,2*IPE/3
         IF(INP(1,N,1).GT.PMAX) PMAX=INP(1,N,1)
         IF(INP(1,N,1).LT.PMIN) PMIN=INP(1,N,1)
   32 CONTINUE
      DO 33 N=1,IPE
         INP(1,N,1)=(INP(1,N,1)-PMIN)/(PMAX-PMIN)
         INP(2,N,1)=INP(1,N,1)
         INP(3,N,1)=INP(1,N,1)
         INP(3,N,2)=INP(2,N,2)/AINP(2,2)
         INP(3,N,3)=INP(2,N,3)/AINP(2,3)
         INP(3,N,4)=INP(2,N,4)/AINP(2,4)
         INP(3,N,8)=INP(2,N,8)/AINP(2,8)
         INP(3,N,9)=INP(2,N,9)/AINP(2,9)
         INP(3,N,10)=INP(2,N,10)/AINP(2,10)
   33 CONTINUE
C locate the edges of the out-plot area
      DO 36 N=1,OPE,1
         IF(OUTP(1,N,1).GE.OUTP(1,OW,1)-PITCH/2.0) THEN
           PMIN=OUTP(1,N,1)
           OS=N
           GO TO 87
         ENDIF
   36 CONTINUE
   87 DO 88 N=OPE,1,-1
         IF(OUTP(1,N,1).LE.OUTP(1,OW,1)+PITCH/2.0) THEN
           PMAX=OUTP(1,N,1)
           OE=N
           GO TO 95
         ENDIF
   88 CONTINUE
C non-dimensionalize the profile at the out plane
   95 DO 37 N=1,OPE
         OUTP(1,N,1)=(OUTP(1,N,1)-PMIN)/(PMAX-PMIN)
         OUTP(2,N,1)=OUTP(1,N,1)
         OUTP(3,N,1)=OUTP(1,N,1)
         OUTP(3,N,2)=2.0*OUTP(2,N,2)/(OUTP(2,OS,2)+OUTP(2,OE,2))
         OUTP(3,N,3)=2.0*OUTP(2,N,3)/(OUTP(2,OS,3)+OUTP(2,OE,3))
         OUTP(3,N,4)=2.0*OUTP(2,N,4)/(OUTP(2,OS,4)+OUTP(2,OE,4))
         OUTP(3,N,8)=2.0*OUTP(2,N,8)/(OUTP(2,OS,8)+OUTP(2,OE,8))
         OUTP(3,N,9)=2.0*OUTP(2,N,9)/(OUTP(2,OS,9)+OUTP(2,OE,9))
         OUTP(3,N,10)=2.0*OUTP(2,N,10)/(OUTP(2,OS,10)+OUTP(2,OE,10))
   37 CONTINUE
C locate the edges of the far-plot area
C non-dimensionalize the profile at the far plane
      DO 38 N=1,FPE,1
         IF(FARP(1,N,1).GE.FARP(1,FW,1)-PITCH/2.0) THEN
           PMIN=FARP(1,N,1)
           FS=N
           GO TO 92
         ENDIF
   38 CONTINUE
   92 DO 89 N=FPE,1,-1
         IF(FARP(1,N,1).LE.FARP(1,FW,1)+PITCH/2.0) THEN
           PMAX=FARP(1,N,1)
           FE=N
           GO TO 91
         ENDIF
   89 CONTINUE
   91 DO 39 N=1,FPE
         FARP(1,N,1)=(FARP(1,N,1)-PMIN)/(PMAX-PMIN)
         FARP(2,N,1)=FARP(1,N,1)
         FARP(3,N,1)=FARP(1,N,1)
         FARP(3,N,2)=2.0*FARP(2,N,2)/(FARP(2,FS,2)+FARP(2,FE,2))
         FARP(3,N,3)=2.0*FARP(2,N,3)/(FARP(2,FS,3)+FARP(2,FE,3))
         FARP(3,N,4)=2.0*FARP(2,N,4)/(FARP(2,FS,4)+FARP(2,FE,4))
         FARP(3,N,8)=2.0*FARP(2,N,8)/(FARP(2,FS,8)+FARP(2,FE,8))
         FARP(3,N,9)=2.0*FARP(2,N,9)/(FARP(2,FS,9)+FARP(2,FE,9))
         FARP(3,N,10)=2.0*FARP(2,N,10)/(FARP(2,FS,10)+FARP(2,FE,10))
   39 CONTINUE
C compute the tangential pressure force on the blade
      BFT=0.0
      CBFT=0.0
      DO 27 N=1,BDE-1
            BFT=BFT+(BD(N+1,1)-BD(N,1))*(BD(N+1,2)+BD(N,2))*0.5
   27 CONTINUE
      BFT=ABS(BFT)
      CBFT=BFT*(RHOREF/RHOIN)*(VREFX/VINX)**2.0
C print out averaged corrected data to summary file
      WRITE(29,*) '****************************************************'
      WRITE(29,*) 'CORRECTION PARAMETERS'
      WRITE(29,*)
      WRITE(29,*) 'VREF     [M/SEC]  =',VREF
      WRITE(29,*) 'VREFX    [M/SEC]  =',VREFX
      WRITE(29,*) 'REREF             =',REREF
      WRITE(29,*) 'REREFX            =',REREFX
      WRITE(29,*) 'MUREF [KG/M/SEC]  = ',VREF*RHOREF*CHORD/REREF
      WRITE(29,*) 'RHOREF [KG/M**3]  =',RHOREF
      WRITE(29,*)
      WRITE(29,*) 'VIN     [M/SEC]   =',VIN
      WRITE(29,*) 'VINX    [M/SEC]   =',VINX
      WRITE(29,*) 'REIN              =',REIN
      WRITE(29,*) 'REINX             =',REINX
      WRITE(29,*) 'MUIN [KG/M/SEC]   = ',MUIN
      WRITE(29,*) 'RHOIN [KG/M**3]   =',RHOIN
      WRITE(29,*) '****************************************************'
      WRITE(29,*) 'INLET PROFILE CORRECTION (PA)'
      WRITE(29,*)
      WRITE(29,*) 'PT: ','(M)',AINP(1,2),'(C)',AINP(2,2)
      WRITE(29,*) 'PS: ','(M)',AINP(1,3),'(C)',AINP(2,3)
      WRITE(29,*) 'PD: ','(M)',AINP(1,8),'(C)',AINP(2,8)
      WRITE(29,*) '****************************************************'
      WRITE(29,*) 'OUTLET PROFILE CORRECTION (PA)'
      WRITE(29,*)
      WRITE(29,*) 'PT: ','(M)',AOUTP(1,2),'(C)',AOUTP(2,2)
      WRITE(29,*) 'PS: ','(M)',AOUTP(1,3),'(C)',AOUTP(2,3)
      WRITE(29,*) 'PD: ','(M)',AOUTP(1,8),'(C)',AOUTP(2,8)
      WRITE(29,*) '****************************************************'
      WRITE(29,*) 'FAR PROFILE CORRECTION (PA)'
      WRITE(29,*)
      WRITE(29,*) 'PT: ','(M)',AFARP(1,2),'(C)',AFARP(2,2)
      WRITE(29,*) 'PS: ','(M)',AFARP(1,3),'(C)',AFARP(2,3)
      WRITE(29,*) 'PD: ','(M)',AFARP(1,8),'(C)',AFARP(2,8)
      WRITE(29,*) '****************************************************'
      WRITE(29,*) 'MASS FLOW (KG/SEC/M)'
      WRITE(29,*)
      WRITE(29,*) 'INLET PROFILE  :','(M)',MIMF,'(C)',CIMF
      WRITE(29,*) 'OUTLET PROFILE :','(M)',MOMF,'(C)',COMF
      WRITE(29,*) 'FAR PROFILE    :','(M)',MFMF,'(C)',CFMF
      WRITE(29,*) '****************************************************'
      WRITE(29,*) 'OUTLET PLANE LOSS COEFFICIENTS'
      WRITE(29,*)
      WRITE(29,*) 'MEASURED PROFILE      :',OLOSS(1)
      WRITE(29,*) 'CORRECTED PROFILE     :',OLOSS(2)
      WRITE(29,*) 'PFEIL CORRECTED       :',POLSS
      WRITE(29,*) 'PFEIL AXIAL-CORRECTED :',POLSSX
      WRITE(29,*) '****************************************************'
      WRITE(29,*) 'FAR PLANE LOSS COEFFICIENTS'
      WRITE(29,*)
      WRITE(29,*) 'MEASURED PROFILE      :',FLOSS(1)
      WRITE(29,*) 'CORRECTED PROFILE     :',FLOSS(2)
      WRITE(29,*) 'PFEIL CORRECTED       :',PFLSS
      WRITE(29,*) 'PFEIL AXIAL-CORRECTED :',PFLSSX
      WRITE(29,*) '****************************************************'
      WRITE(29,*) 'MIXING LOSS COEFFICIENTS'
      WRITE(29,*)
      WRITE(29,*) 'MEASURED PROFILE        :',MLOSS
      WRITE(29,*) 'PFEIL CORRECTED PROFILE :',PMLOSS
      WRITE(29,*) '****************************************************'
      WRITE(29,*) 'DRAG/LIFT RATIO'
      WRITE(29,*)
      WRITE(29,*) 'OUTLET MEASURED PROFILE  :',
     $            MOFN/SQRT(MOFT**2.0+MOFN**2.0)
      WRITE(29,*) 'FAR MEASURED PROFILE     :',
     $            MFFN/SQRT(MFFT**2.0+MFFN**2.0)
      WRITE(29,*) 'OUTLET CORRECTED PROFILE :',
     $            COFN/SQRT(COFT**2.0+COFN**2.0)
      WRITE(29,*) 'FAR CORRECTED PROFILE    :',
     $            CFFN/SQRT(CFFT**2.0+CFFN**2.0)
      WRITE(29,*) '****************************************************'
      WRITE(29,*) 'PRESSURE LIFT FORCE (N/M)'
      WRITE(29,*)
      WRITE(29,*) '(M)',BFT,'(C)',CBFT
      WRITE(29,*) '****************************************************'
C write all-summary out to output file
      WRITE(21,*) STGR,
     $            OLOSS(1),POLSS,POLSSX,
     $            FLOSS(1),PFLSS,PFLSSX,
     $            MLOSS,PMLOSS,
     $            MOFN/SQRT(MOFT**2.0+MOFN**2.0),
     $            MFFN/SQRT(MFFT**2.0+MFFN**2.0),
     $            COFN/SQRT(COFT**2.0+COFN**2.0),
     $            CFFN/SQRT(CFFT**2.0+CFFN**2.0),
     $            REINX,REREFX,
     $            RHOIN,RHOREF,
     $            MIMF,CIMF,
     $            BFT,CBFT
      WRITE(24,*) STGR,POLSSX,0.0
C write data to output file plt.dat
      WRITE(26,*) 'VARIABLES="Y",',
     $            '"P_t [Pa]",',
     $            '"P_s [Pa]",',
     $            '"P_d [Pa]",',
     $            '"V [m/sec]"',
     $            '"V_x [m/sec]"',
     $            '"V_y [m/sec]"'
C write inlet data
      WRITE(26,*) 'ZONE T="cfd-in-m"'
      DO 28 N=IPE/3,2*IPE/3
         WRITE(26,*) INP(1,N,1),INP(1,N,2),
     $               INP(1,N,3),INP(1,N,8),
     $               INP(1,N,4),INP(1,N,9),INP(1,N,10)
   28 CONTINUE
      WRITE(26,*) 'ZONE T="cfd-in-c"'
      DO 29 N=IPE/3,2*IPE/3
         WRITE(26,*) INP(2,N,1),INP(2,N,2),
     $               INP(2,N,3),INP(2,N,8),
     $               INP(2,N,4),INP(2,N,9),INP(2,N,10)
   29 CONTINUE
C write outlet data
      WRITE(26,*) 'ZONE T="cfd-out-m"'
      DO 34 N=OS,OE
         WRITE(26,*) OUTP(1,N,1),OUTP(1,N,2),
     $               OUTP(1,N,3),OUTP(1,N,8),
     $               OUTP(1,N,4),OUTP(1,N,9),OUTP(1,N,10)
   34 CONTINUE
```

```
      WRITE(26,*) 'ZONE T="cfd-out-c"'
      DO 35 N=OS,OE
         WRITE(26,*) OUTP(2,N,1),OUTP(2,N,2),
     $               OUTP(2,N,3),OUTP(2,N,8),
     $               OUTP(2,N,4),OUTP(2,N,9),OUTP(2,N,10)
  35 CONTINUE
C write far data
      WRITE(26,*) 'ZONE T="cfd-far-m"'
      DO 40 N=FS,FE
         WRITE(26,*) FARP(1,N,1),FARP(1,N,2),
     $               FARP(1,N,3),FARP(1,N,8),
     $               FARP(1,N,4),FARP(1,N,9),FARP(1,N,10)
  40 CONTINUE
      WRITE(26,*) 'ZONE T="cfd-far-c"'
      DO 41 N=FS,FE
         WRITE(26,*) FARP(2,N,1),FARP(2,N,2),
     $               FARP(2,N,3),FARP(2,N,8),
     $               FARP(2,N,4),FARP(2,N,9),FARP(2,N,10)
  41 CONTINUE
C write data to output file nplt.dat and all-summary output file
      WRITE(27,*) 'VARIABLES="Y",',
     $            '"P_t_n",',
     $            '"P_s_n",',
     $            '"P_d_n",',
     $            '"V_n"',
     $            '"V_x_n"',
     $            '"V_y_n"'
C write inlet data
      WRITE(27,*) 'ZONE T="cfd-in-n"'
      DO 42 N=IPE/3,2*IPE/3
         WRITE(27,*) INP(3,N,1),INP(3,N,2),
     $               INP(3,N,3),INP(3,N,8),
     $               INP(3,N,4),INP(3,N,9),INP(3,N,10)
  42 CONTINUE
C write outlet data
      WRITE(27,*) 'ZONE T="cfd-out-n"'
      DO 43 N=OS,OE
         WRITE(27,*) OUTP(3,N,1),OUTP(3,N,2),
     $               OUTP(3,N,3),OUTP(3,N,8),
     $               OUTP(3,N,4),OUTP(3,N,9),OUTP(3,N,10)
         WRITE(22,*) OUTP(3,N,1),OUTP(3,N,2),
     $               OUTP(3,N,3),OUTP(3,N,8),
     $               OUTP(3,N,4),OUTP(3,N,9),OUTP(3,N,10)
  43 CONTINUE
C write far data
      WRITE(27,*) 'ZONE T="cfd-far-n"'
      DO 44 N=FS,FE
         WRITE(27,*) FARP(3,N,1),FARP(3,N,2),
     $               FARP(3,N,3),FARP(3,N,8),
     $               FARP(3,N,4),FARP(3,N,9),FARP(3,N,10)
C         WRITE(22,*) FARP(3,N,1),FARP(3,N,2),
C     $               FARP(3,N,3),FARP(3,N,8),
C     $               FARP(3,N,4),FARP(3,N,9),FARP(3,N,10)
  44 CONTINUE
C write cp data to output file cp.dat and all-summary output file
      WRITE(25,*) 'VARIABLES="X",','"C_p"'
      DO 46 N=1,BDE
         WRITE(25,*) CP(N,1),CP(N,2)
         WRITE(23,*) CP(N,1),CP(N,2)
  46 CONTINUE
C terminate files for current case
      CLOSE(15)
      CLOSE(16)
      CLOSE(17)
      CLOSE(18)
      CLOSE(25)
      CLOSE(26)
      CLOSE(27)
      CLOSE(28)
      CLOSE(29)
C loop
      GO TO 30
   31 PRINT*, '================================='
      PRINT*, 'TOTAL CASES STUDIED: ',NUM
      PRINT*, '================================='
      END
************************************************************************
* This subroutine performs mass averaging using Trap. integration.     *
************************************************************************
* Hicham A Chibli - Texas A&M University - 12/11/09                    *
************************************************************************
      SUBROUTINE INTG(FULL,C,L,W,SET,K,CMN,MA,MINT,MFLOW)
      INTEGER C,L,W,SET,K,CMN,I
      REAL FULL(C,L,W),NX(K),NY(K),NZ(K),MA,MFLOW,MINT
C populate integration arrays
      DO 2 I=1,K
         NX(I)=FULL(SET,I,1)
         NZ(I)=FULL(SET,I,5)*FULL(SET,I,9)
         NY(I)=FULL(SET,I,CMN)*NZ(I)
    2 CONTINUE
C initialize integral
      MFLOW=0.0
      MINT=0.0
C calculate mass flow rate
      DO 3 I=1,K-1
         MFLOW=MFLOW+0.5*(NX(I+1)-NX(I))*(NZ(I)+NZ(I+1))
    3 CONTINUE
C integrate
      DO 1 I=1,K-1
         MINT=MINT+0.5*(NX(I+1)-NX(I))*(NY(I)+NY(I+1))
    1 CONTINUE
      MA=MINT/MFLOW
      MFLOW=ABS(MFLOW)
      RETURN
      END
************************************************************************
* This subroutine returns the non-blank component of a string.         *
************************************************************************
* Hicham A Chibli - Texas A&M University - 12/01/09                    *
************************************************************************
      SUBROUTINE CLEAN(TITLE,N)
C Define variables and constants
      INTEGER N
      CHARACTER*(*) TITLE
C Find the blank part of file name
      N=INDEX(TITLE,' ')
C Get the size of the non-blank part of file name
      N=N-1
      RETURN
      END
```