One-dimensional transient numerical modeling of a micro-power generation system

Emre Tatli
West Virginia University

Follow this and additional works at: https://researchrepository.wvu.edu/etd

Recommended Citation
Tatli, Emre, "One-dimensional transient numerical modeling of a micro-power generation system" (2003). Graduate Theses, Dissertations, and Problem Reports. 1402.
https://researchrepository.wvu.edu/etd/1402

This Thesis is protected by copyright and/or related rights. It has been brought to you by the The Research Repository @ WVU with permission from the rights-holder(s). You are free to use this Thesis in any way that is permitted by the copyright and related rights legislation that applies to your use. For other uses you must obtain permission from the rights-holder(s) directly, unless additional rights are indicated by a Creative Commons license in the record and/or on the work itself. This Thesis has been accepted for inclusion in WVU Graduate Theses, Dissertations, and Problem Reports collection by an authorized administrator of The Research Repository @ WVU. For more information, please contact researchrepository@mail.wvu.edu.
ONE-DIMENSIONAL TRANSIENT NUMERICAL MODELING OF A MICRO-POWER GENERATION SYSTEM

Emre Tatli

Thesis submitted to the College of Engineering and Mineral Resources at West Virginia University in partial fulfillment of the requirements for the degree of

Master of Science in Mechanical Engineering

Ismail Celik, Ph.D., Chair
Randall S. Gemmen, Ph.D.
Eric Johnson, Ph.D.
Ibrahim Yavuz, Ph.D.

Mechanical and Aerospace Engineering Department

Morgantown, West Virginia 2003

Keywords: Computational Fluid Dynamics, Turbomachinery, Numerical Modeling

Copyright 2003 Emre Tatli
ABSTRACT

ONE-DIMENSIONAL TRANSIENT NUMERICAL MODELING OF A MICRO-POWER GENERATION SYSTEM

Emre Tatli

Two core modules of a more comprehensive model for analyzing the flow of fluids in a hybrid system including a fuel cell are developed. First part consists of a new procedure for scaling turbomachinery maps. In the last few decades, the sizes of turbomachinery have been getting smaller and smaller for various purposes. New designs have pushed the limits of power generation to be in more compact sizes, which lead to smaller compressors and turbines. For new designs, it is important to know how the turbomachinery in the system will behave at certain conditions. But it is not practical to build a turbomachinery during the design process since there might be changes in the configuration of the system afterwards and it is a very costly procedure. It is also a very time consuming process to obtain experimental data for off design conditions. What is usually done is to use similar sized turbomachinery maps to predict the behavior of the system with reasonable approximations. This method provides better estimates with much less cost.

Second part introduces a transient one-dimensional numerical model of the flow in pipes. The program uses the projection method to solve for the variables in a transient 1-D compressible flow system in a variable cross-sectional duct with friction and heat transfer. This model is verified through comparison with the analytical solutions of various flows. It gives very accurate results for any type of flow including unsteady flows with friction and heat transfer. A case study of flow through a combustion chamber is also presented.
ACKNOWLEDGEMENTS

I would like to gratefully acknowledge the supervision of Dr. Ismail Celik during my Master’s work. With his knowledge, enthusiasm, dedication to his work and educating the people around him, I have learned so much about Computational Fluid Dynamics, as well as many aspects of life.

I would like to thank my committee members Dr. Randall S. Gemmen, Dr. Eric Johnson and Dr. Ibrahim Yavuz for their valuable suggestions for improving my thesis.

I would like to thank the Department of Energy National Energy Technology Laboratory (DOE-NETL) for funding my research.

I thank the Celik family for being my family away from home. I don’t know what I would have done without them.

I thank Ibrahim, for being my friend, my brother. You are a unique person.

I thank all my friends at the CFD Lab for making this journey an enjoyable one.

I thank my family back home, for always being there when I needed them. I love you all.

Finally, I would like to thank Ezgi (Pisim), for never giving up in those tough two years spent away from each other. You are my angel.
# TABLE OF CONTENTS

ABSTRACT .............................................................................................................. ii
ACKNOWLEDGEMENTS ...................................................................................... iii
TABLE OF CONTENTS ......................................................................................... iv
LIST OF FIGURES ................................................................................................. vi
LIST OF TABLES .................................................................................................... ix
NOMENCLATURE .................................................................................................. x

## Chapter 1  Introduction ..................................................................................... 1

1.1  Objective ........................................................................................................ 1
1.2  Literature Survey .......................................................................................... 2
  1.2.1  Gas Turbines and Turbomachinery ......................................................... 2
  1.2.2  Numerical Methods / Computational Fluid Dynamics ......................... 4

## Chapter 2  Small Sized Turbomachinery Map Scaling ....................................... 7

2.1  Introduction ................................................................................................... 7
2.2  Dimensional Analysis in Turbomachines ..................................................... 7
  2.2.1  Method 1 ................................................................................................ 9
  2.2.2  Method 2 ................................................................................................ 9
  2.2.3  Method 3 ................................................................................................ 11
  2.2.4  Method 4 ................................................................................................ 13
2.3  Cordier Analysis .......................................................................................... 14
2.4  Procedure and Results ................................................................................ 18
  2.4.1  Compressor Scaling .............................................................................. 18
    2.4.1.1  Surge and Stall ................................................................................ 18
    2.4.1.2  Formulation ..................................................................................... 20
  2.4.2  Expander Scaling .................................................................................. 25
2.5  Comparison of Linear and Non-Linear Scaling Methods ......................... 29
2.6  Interpolation Routine .................................................................................. 30
2.7  Discussion .................................................................................................... 31
2.8  Recommendations ....................................................................................... 32

## Chapter 3  One-Dimensional Transient Compressible Flow Solver ................... 34

3.1  Introduction ................................................................................................... 34
3.2  Code Formulation ....................................................................................... 34
3.3  Verification of the One-Dimensional Transient Compressible Flow Solver .................................................................................................................. 39
  3.3.1  Steady State Flow Verification ............................................................... 39
  3.3.2  Unsteady Flow Verification .................................................................. 43
  3.3.3  Energy Equation Verification ................................................................ 48
    3.3.3.1  Discussion of the Energy Equation Solution .................................... 52
3.4  Case Study – Combustion Chamber Flow .................................................. 55
3.5  Conclusion ................................................................................................... 59

## Chapter 4  Conclusion ....................................................................................... 60

## Chapter 5  References ....................................................................................... 61

Appendix A Interpolation Program
Appendix B One-Dimensional Transient Compressible Flow Solver
Appendix C Second Order Runge Kutta Program
Appendix D Adams-Moulton Program
Appendix E Mass and Energy Balance Equations
Appendix F Improved Fluent Results
Appendix G Turbomachinery Map Data
LIST OF FIGURES

Figure 2.1: Non-dimensionalized Turbine Map ......................................................... 12
Figure 2.2: Cordier Diagram ..................................................................................... 16
Figure 2.3: Efficiency vs. Specific Diameter .............................................................. 17
Figure 2.4: Pressure Ratio vs. Non-Dimensional Volumetric Flow Rate of the large compressor ........................................................................................................ 23
Figure 2.5: Pressure Ratio vs. Non-Dimensional Volumetric Flow Rate of the small compressor ........................................................................................................ 23
Figure 2.6: Corrected Pressure Ratio vs. Corrected Non-Dimensional Volumetric Flow Rate of the large compressor ........................................................................ 24
Figure 2.7: Corrected Pressure Ratio vs. Corrected Non-Dimensional Volumetric Flow Rate of the small compressor ........................................................................ 24
Figure 2.8: Non-dimensional Volumetric Flow Rate vs. Pressure ratio of the small expander .............................................................................................................. 25
Figure 2.9: Non-dimensional Volumetric Flow Rate vs. Pressure ratio of the medium sized expander ..................................................................................................... 26
Figure 2.10: Non-dimensional Volumetric Flow Rate vs. Pressure ratio of the large expander ............................................................................................................. 26
Figure 2.11: Corrected Non-dimensional Volumetric Flow Rate vs. Corrected Pressure ratio of the medium sized expander ................................................. 27
Figure 2.12: Corrected Non-dimensional Volumetric Flow Rate vs. Corrected Pressure ratio of the small expander ........................................................................... 28
Figure 2.13: Corrected Non-dimensional Volumetric Flow Rate vs. Corrected Pressure ratio of the large expander ........................................................................... 28
Fig. 2.14: Corrected Pressure ratio vs. Corrected Non-dimensional Volumetric Flow rate of the large and small compressors using Linear Scaling ........................................... 29
Fig. 2.15: Corrected Pressure ratio vs. Corrected Non-dimensional Volumetric Flow rate of the large and small compressors using Non-Linear Scaling ........................................... 30
Figure 2.16: General form of mass flow rate and efficiency vs. pressure ratio maps of a typical expander ............................................................................................... 31
Figure 3.1: Duct of arbitrary cross section ................................................................. 34
Figure 3.2: Comparison of Pressure profiles (Code vs. Manufactured solution) ............. 41
Figure 3.3: Comparison of Velocity profiles (Code vs. Manufactured solution) ..........41
Figure 3.4: Comparison of Density profiles (Code vs. Manufactured solution) ...........42
Figure 3.5: Comparison of Temperature profiles (Code vs. Manufactured solution) .........42
Figure 3.6: Comparison of Flow rate profiles (Code vs. Manufactured solution) ...........43
Figure 3.7: Comparison of velocity wrt time at x = 0.5 m (Code vs. Analytical Soln.) ......45
Figure 3.8: Comparison of pressure wrt time at x = 0.5 m (Code vs. Analytical Soln.) ......45
Figure 3.9: Comparison of velocity wrt time at x = 0.5 m with different number of control volumes.................................................................................................................................46
Figure 3.10: Comparison of pressure wrt time at x = 0.5 m with different number of control volumes.................................................................................................................................47
Figure 3.11: Comparison of velocity wrt time at x = 0.5 m with different time steps........47
Figure 3.12: Comparison of pressure wrt time at x = 0.5 m with different time steps........48
Figure 3.13: Comparison of Temperature between Fluent and Present Program for Cf = 0.015...........................................................................................................................................49
Figure 3.14: Comparison of Velocity between Fluent and Present Program for Cf = 0.015..50
Figure 3.15: Comparison of Pressure between Fluent and Present Program for Cf = 0.015...50
Figure 3.16: Comparison of Density between Fluent and Present Program for Cf = 0.015…..51
Figure 3.17: Comparison of Mass Flow Rate between Fluent and Present Program for Cf = 0.015...........................................................................................................................................51
Figure 3.18: Variation of Cf wrt time and position in Fluent........................................52
Figure 3.19: Comparison of Temperature between Fluent and Present Program for Cf = 0.05..................................................................................................................................................53
Figure 3.20: Comparison of Velocity between Fluent and Present Program for Cf = 0.05…53
Figure 3.21: Comparison of Pressure between Fluent and Present Program for Cf = 0.05…54
Figure 3.22: Comparison of Density between Fluent and Present Program for Cf = 0.05…..54
Figure 3.23: Comparison of Mass Flow Rate between Fluent and Present Program for Cf = 0.05..................................................................................................................................................55
Figure 3.24: Radius profile of the combustion chamber.............................................55
Figure 3.25: Flowrate vs. x at time = 10 seconds.........................................................56
Figure 3.26: Velocity vs. x at time = 10 seconds.............................................................56
Figure 3.27: P/P_inlet vs. x at time = 10 seconds..............................................................57
Figure 3.28: Density vs. x at time = 10 seconds..............................................................58
Figure 3.29: Temperature vs. x at time = 10 seconds

Figure App. F.1: Comparison of Centerline Velocity at x = 0.5m for the Present Program and Fluent

Figure App. F.2: Comparison of Pressure Ratio at x = 0.5m for the Present Program and Fluent

Figure App. F.3: Comparison of Centerline Density at x = 0.5m for the Present Program and Fluent

Figure App. F.4: Comparison of Mass Flow Rate at x = 0.5m for the Present Program and Fluent

Figure App. F.5: Comparison of Temperature at t = 0.015s and t = 0.04s for Present Program and Fluent
LIST OF TABLES

Table 2.1: Design point values of the large and the small compressors………………….22
Table 2.2: Design point values for the large, medium and the small expander……………27
Table App. E.1: Data from the present program………………………………………………98
Table App. E.2: Data from Fluent…………………………………………………………………98
NOMENCLATURE

\( \phi \) - non-dimensional volumetric flow rate

\( \phi_c \) - corrected non-dimensional volumetric flow rate

\( Q \) – volumetric flow rate (m\(^3\)/s)

\( N \) – rotational speed (rev/s)

\( D \) – diameter

\( P_{\text{des}} \) - design pressure

\( P_{\text{ref}} \) – reference pressure

\( T_{\text{ref}} \) – reference temperature

\( \text{PR} \) – pressure ratio

\( \text{PR}^* \) - corrected pressure ratio

\( \beta \) - correction factor for pressure ratio

\( m \) - mass flow rate

\( T_{01} \) – Inlet stagnation temperature

\( P_{01} \) - Inlet stagnation pressure

\( \omega \) - angular velocity of the rotor

\( \eta \) - efficiency of the flow process

\( a \) - speed of sound

\( \mu \) - fluid viscosity

\( P \) - power, Watts

\( \Delta h_{0s} \) - stagnation enthalpy

\( R \) - gas constant

\( c_p \) - specific heat
Chapter 1:

INTRODUCTION

1.1) Objective

In this thesis, two core modules of a more comprehensive model for analyzing the flow of fluids in a hybrid system including a fuel cell are developed. First part consists of a new procedure for scaling turbomachinery maps. In the last few decades, the sizes of turbomachinery have been getting smaller and smaller for various purposes. New designs have pushed the limits of power generation to be in more compact sizes, which lead to smaller compressors and turbines. For new designs, it is important to know how the turbomachinery in the system will behave at certain conditions. But it is not practical to build a turbomachine during the design process since there might be changes in the configuration of the system afterwards and it is a very costly procedure. It is also a very time consuming process to obtain experimental data for off design conditions. What is usually done is to use similar sized turbomachinery maps to predict the behavior of the system with reasonable approximations. For new designs involving fuel cells, turbomachinery of very small sizes must be used to achieve the desired high efficiency. However, there isn’t much literature about the maps of such small turbomachinery. Here, a new procedure using non-dimensional analysis is introduced to obtain small sized turbomachinery maps from relatively large ones. It consists of newly defined terms to be added to the commonly used non-dimensional factors to obtain terms so that when applied to any compressor or turbine, they will give similar order of magnitude values, and these values can then be used to scale maps. A program named Locate-x was constructed to first non-dimensionalize the input terms and scale the initially discretized map to a desired one with a previously decided design point.
In the second part of this thesis, a transient one-dimensional numerical model of the flow in the pipes and the turbomachinery units is constructed. The program uses the projection method [1] to solve for the variables in a transient 1-D compressible flow system, with a variable cross-sectional duct, friction and heat transfer. This model is verified through comparison with the analytical solution of a flow in a converging diverging nozzle, comparison with a manufactured solution [12] and lastly a pulsating flow through a duct. This program will use the outputs obtained from the program Locate-x as initial/boundary conditions when used on a system with turbomachinery.

1.2) Literature Survey

In this section, some literature that is directly relevant to the present work is reviewed.

1.2.1) Gas Turbines and Turbomachinery

Greitzer (1976) [9] presents a paper about surge and rotating stall in axial flow compressors. This paper reports a theoretical study of axial compressor surge. A non-linear model is presented that predicts the transient response of compression system subsequent to perturbation from steady operating conditions. This paper has two parts, first one explaining the theoretical compression system model and the second part the experimental results and comparison with theory.

Greitzer (1980) [10] presents a paper about axial compressor stall phenomena. The paper explains the reasons for initiation of stall in compressors and assessments are made of the various methods of predicting the onset of compressor and/or compression system instability,
such as empirical correlations, linearized stability analyses, and numerical unsteady flow calculation procedures.

**Greitzer (1981)** [11] presents an analysis on the stability of pumping systems. This is a review paper which includes axial and centrifugal compression systems, pumping systems involving cavitation, systems with two phase flow, systems with combustion, hydraulic systems, and systems which have two or more pumping elements in parallel. All of the above exhibit instabilities under certain operating conditions, although the mechanism of instability, as well as the particular system element that is responsible for the instability, will be quite different in the different systems. However, several basic concepts, such as the idea of negative damping which is associated with dynamic instability, is shown to be common to different systems.

**Fink, Cumpsty and Greitzer (1992)** [8] investigated surge dynamics in a free spool centrifugal compressor system. Several different aspects are addressed. First, two very different compression systems, one with a large downstream volume and one with the smallest possible downstream volume are employed to examine stall initiation phenomena as well as the behaviour of the compressor characteristics when operating in surge. The measurements show impeller stall at the inducer tips to be a key phenomena in initiating surge. The inducer stall is stationary and asymmetric, due to the presence of the volute, and is most severe near the volute tongue angular position. A nonlinear model of the system is also presented. The model deviates from previous treatments of unsteady flow in compressor systems in that the assumption of constant rotor speed is relaxed.

**Hansen et al. (1981)** [12] presented results from an experimental and theoretical study of surge in a small centrifugal compressor. Experimental results for deep surge are compared to
predictions based on the lumped parameter model of Greitzer (1976) for axial compressors. Both negative and positive flow branches of the steady characteristic, being essential for the model, were measured. Predictions are in fair agreement with data when using a relaxation time smaller than the one proposed for axial compressors. The stability limit of the model equations have been studied for finite amplitude disturbances.

**Stenning (1980)** [22] extended the surge analysis to rotating stall and surge. Safe off design operation of compressors is limited to the region in which the flow is stable. Flow instabilities can be of two types, rotating stall and surge. The first of these factors subjects the blading to high oscillating stresses, while the second may have a disastrous effect on the whole system of which the compressor is a component. In the paper, the properties of these two types of instability are discussed and some simple criteria for determining system stability are presented.

1.2.2) Numerical Methods / Computational Fluid Dynamics

**S.Majumdar (1988)** [16] wrote a paper on the role of underrelaxation in momentum interpolation for calculation of flow with nonstaggered grids. The well known drawback of the nonstaggered variable arrangement is the occurrence of nonphysical oscillations (wiggles) of the pressure and/or velocity field, leading to severe numerical instabilities in some cases. Such pressure wiggles are mainly to the de-linking of the pressure at the node in question while expressing the discrete analog of the pressure-gradient term in the momentum equations. This paper identifies the basic reason for the observed dependence of the results on the underrelaxation parameter and proposes how to implement the momentum
interpolation in an iterative algorithm to achieve a unique solution that is independent of the underrelaxation parameter used.

**J. McGuirk and G. J. Page (1990)** [17] wrote a paper on shock capturing using a pressure correction method. This method is suitable for the calculation of flow containing a wide range of Mach numbers such as transonic impinging jet. The method uses equations based on properties per unit volume so that momentum is retained as a basic dependent variable rather than velocity. This simplifies the discretization of the time dependent flow equations and allows a direct relationship to be determined between pressure and mass flux. The hyperbolic nature of the system of equations is obtained by using the retarded pressure approach. This is a transformation of the real pressure based on local Mach number and is used in the momentum and pressure correction equations. The shocked quasi one dimensional flow in a nozzle is used as a test of shock capturing properties and speed of computation.

**Schobeiri et al. (1994)** [21] wrote a paper on a generic, modularly structured computer code for simulation of dynamic behavior of aero-and power generation gas turbine engines. The program is called GETRAN. The design concept, the theoretical background essential for the development of the modularly structured simulation code GETRAN, and several critical simulation cases are presented in this paper. The code being developed is capable of simulating the nonlinear dynamic behavior of single and multispool core engines, turbofan engines, and power generation gas turbine engines under adverse dynamic operating conditions.
**Summary:**

All the above discussed papers have some relevance on the current research but none of them actually point out the main aspects of the topics in this thesis. For example, the work by Greitzer et al. require inputs from compressor maps. However, there is no original work for scaling down the results from large compressors. The most relevant work on computational aspects is the one by Schobeiri et al. (1994). But the equations and the procedures this paper uses are not suitable for the solution of 1-D compressible flow equations. They give oscillations and some diverging solutions because of the numerical methods which are not suitable to the type of flows presented in this thesis.
CHAPTER 2:
SMALL Sized TURBOMACHINERY MAP SCALING

2.1) Introduction:

Obtaining experimental data for the off-design conditions for any turbomachine is an expensive and a time consuming process. That is the reason for the difficulty in finding turbomachinery maps in literature, especially for small turbines. To overcome this problem, a scaling procedure is suggested. The aim is the following: without having to construct an experimental setup, a map that gives the off design operating conditions is targeted using any map in hand and the suggested scaling procedure. In this chapter, various non-dimensionalizing methods are compared and an alternative method is devised. The new method makes it possible to extrapolate to small scales from large scale maps.

For compressors, surge and stall phenomena is a very important concept and so this is also described briefly. The scaling of large compressor maps to smaller sizes should be able to give us accurate results about the range we are working, whether it is in the surge range or not.

To adapt the scaling procedure to the computer environment, an interpolation / extrapolation routine is written that helps the user to enter the data available and get the desired results. The routine also warns the user about any surge / data out of range events that may occur.

2.2) Dimensional Analysis in Turbomachines

There are 3 types of similarity, namely geometric, kinematic and dynamic. A model and prototype are geometrically similar if all physical or body dimensions in all three axes (for Cartesian coordinates) have the same linear ratio. The motions of two systems are kinematically similar if similar elements (i.e., parts of the system) lie at similar locations at
similar times. And dynamic similarity requires that the additional effects of force-scale or mass-scale be maintained between a model and a prototype, taking length scale and time scale.

The goal in similarity is to be able to accurately predict the performance or behavior of bodies or machines from another set of conditions or from another situation. Dimensional analysis is usually performed using Buckhingam-Pi Method. This method can be summarized as follows: If one has a set of performance variables based on a number of fundamental dimensions, the variables can be grouped into products that are dimensionless. There are a total of four basic dimensions, M (mass), L (length), T (time) and θ (temperature).

The steps in obtaining the number of dimensionless \( \pi \) terms is as follows:

- List and count the variables \( (N_v) \)
- List and count the basic units \( (N_u) \)
- Select a number of variables as primary (usually equal to the number in step 2)
- Form \( \pi \) products at the number of \( (N_v - N_u) \)

For turbomachinery, four methods have been introduced in order to scale down existing turbomachinery maps. In this chapter, after obtaining all the scaling methods, comparisons will be made and the best method is chosen in accordance with the best matching.
2.2.1) Method 1:

The basic non-dimensional parameters are common to all turbomachines and have many different expressions, depending on the working fluid (compressible or incompressible) and the application (compressor, pump, turbine). For turbines operating with compressible flows, one way to write the non-dimensional terms is as follows:

\[
\begin{align*}
\text{Mass flow rate} & \quad \frac{m\sqrt{RT_{01}/\gamma}}{D^2 P_{01}} & (2.1) \\
\text{Pressure ratio} & \quad \frac{P_{01}}{P_2} & (2.2) \\
\text{Speed} & \quad \frac{\pi ND}{\sqrt{\gamma RT_{01}}} & (2.3) \\
\text{Reynolds number} & \quad \frac{\rho_{01} ND^2}{\mu} & (2.4)
\end{align*}
\]

Symbols used are defined as follows:

- Mass flow rate : \( m \)
- Inlet stagnation temperature : \( T_{01} \)
- Inlet stagnation pressure : \( P_{01} \)
- Inlet diameter of the turbine : \( D \)

2.2.2) Method 2:

For a turbine, the primary objective is to design a machine to deliver a specified power output. The power available from a turbine is a function of:

- The inlet stagnation conditions \( P_{01} \) and \( T_{01} \)
- The mass flow rate \( m \)
- The size of the machine
- The shape of the machine
- The angular velocity of the rotor \( \omega \)
- The efficiency of the flow process \( \eta \)
- The gas viscosity (if the Reynolds number is high, its influence will be small over the normal operating range and can be ignored during design)

The functional relationship then can be written as: (after using the Buckingham-Pi Theorem and non-dimensionalizing the terms)

\[
f\left(\frac{W}{P_{01}(RT_{01})^{1/2}D^2}, \frac{m(RT_{01})^{1/2}}{P_{01}D^2}, \frac{ND}{(RT_{01})^{1/2}}, \eta\right) = 0
\]  

(2.5)

where \( W \) is the power output different from the previous approach.

For these set of parameters the inlet stagnation speed of sound \( (a_0 = \sqrt{RT_0}) \) [4] is used as a reference parameter. The turbines should be geometrically similar for this procedure. In scaling turbines from one size to another, or the same turbine from one operating condition to another, it is usually possible to scale all the dimensionless groups except the Reynolds number. This can not be scaled precisely unless the inlet density can be controlled independently of the other inlet conditions.

But fortunately, from experimental studies it can be concluded that Reynolds number corrections are usually small and they could be neglected. [12] The last step in scaling of turbines is to obtain turbine maps in some way, digitizing may be a solution.
2.2.3) Method 3:

The above mentioned dimensional analysis will not result in a unique set of dimensionless variables since there are many ways to choose variables, other variables are dependent on the following listed, i.e. $\Delta T$ (temperature), $\Delta P$ (pressure)

- gH-head variable (velocity scale)
- Q – volume flow rate
- P – power, Watts
- N - speed
- D - diameter
- $\rho$ - fluid density
- $\mu$ - fluid viscosity
- $d/D$ - dimensionless diameter ratio
- $L/D$ - dimensionless duct length
- $\eta$ - efficiency

There are three units involved in the analysis: mass, length and time. (M,L,T) So we can reduce the seven variables to four. Choosing $\rho$, N, D as primaries, the $\pi$ products are formed with the same method before. So:

$$\pi_1 = \frac{Q}{N.D^3} = \phi$$

is called the flow coefficient \hspace{1cm} (2.6)

$$\pi_2 = \frac{\mu}{N.D^2.\rho} = \frac{1}{Re}$$

is the inverse of Reynold’s number \hspace{1cm} (2.7)
\[ \pi_3 = \frac{g.H}{N^2.D^2} = \phi \] is called the head coefficient \hspace{1cm} (2.8)

\[ \pi_4 = \frac{P}{\rho.N^3.D^3} = \xi \] is called the power coefficient \hspace{1cm} (2.9)

Also it can be easily shown that:

\[ \eta = \frac{\phi \cdot \phi}{\xi} \] \hspace{1cm} (2.10)

At the characteristic curves relating \( P \) and \( \Delta p_T \) versus \( Q \), the known values of \( \rho, N, D \) are chosen and the data tables of \( P, \Delta p_T \) and \( Q \) are converted to \( \phi, \phi, \xi \) and \( \eta \). After plotting these, the dimensionless performance plots of the turbomachine is obtained.

The advantage is that the \( \phi - \phi, \xi - \phi, \eta - \phi \) curves (Fig. 2.1) can be used to generate infinite number of \( gH-Q, P-Q, \eta-Q \) curves by choosing any value of \( \rho, N, D \).

![Figure 2.1: Nondimensional Turbine Map](image-url)
2.2.4) Method 4:

When there are significant changes in density through the turbomachine, the use of volume flow rate and the head rise become inappropriate. [11] We should work with mass flow rate. The head rise must be replaced by the change in stagnation enthalpy $\Delta h_{0s}$, or a related variable. If the flow is adiabatic, $\Delta h_{0s}$ can be related to the change in stagnation temperature through the turbomachine:

$$\Delta h_{0s} = c_p \Delta T_0 \quad (2.11)$$

In addition, we can consider the relation of the total energy of the fluid with other variables. The aim is to write specific variables in terms of other variables.

$$\Delta h_{0s} = f_1(N, D, m, T_0, \rho_0, \mu, \gamma, \frac{d}{D}) \quad (2.12)$$

$$P = f_2(N, D, m, T_0, \rho_0, \mu, \gamma, \frac{d}{D}) \quad (2.13)$$

$$\eta = f_3(N, D, m, T_0, \rho_0, \mu, \gamma, \frac{d}{D}) \quad (2.14)$$

The new set of parameters needed to be considered in compressible effects are as follows:

- $R$, $c_p$: gas constant, specific heat
- $\Delta h_{0s} = c_p \Delta T_0$: stagnation enthalpy
- $m$: mass flow rate
- $\rho_{01} = p_{01} / RT_{01}$: density
- $a_{01} = (\gamma RT_{01})^{1/2}$: stagnation speed of sound

These are to be used with the previous dimensions

- $\mu$: viscosity (kg/m.s)
- $N$: rotational speed (radians/s)
- $D$: diameter (m)
The $\pi$ terms are formed as we did in the previous cases, and they come out to be the followings:

$$\pi_1 = \frac{m}{(\rho_0 N D^3)} : \text{mass flow coefficient} \tag{2.15}$$

$$\pi_2 = \frac{p}{(\rho_0 N^3 D^5)} : \text{power coefficient} \tag{2.16}$$

$$\pi_3 = \frac{\Delta h_{0s}}{(N^2 D^2)} : \text{work or head coefficient} \tag{2.17}$$

$$\pi_4 = \frac{N D}{a_0} : \text{Mach number} \tag{2.18}$$

$$\pi_5 = \frac{\rho_0 N D^3}{\mu} : \text{Reynolds number} \tag{2.19}$$

2.3) **Cordier Analysis:**

The scaling rules discussed above considered cases which the effects were examined in an isolated fashion. But we need a more systematic way of varying more than one variable at a time in order to change both head and flow performance a specified amount. At this point, we use the specific speed and the specific diameter definitions, which is the Cordier analysis. The specific speed and the specific diameter equations are obtained as follows:

$$\Delta p_r = \rho N^2 D^2 \psi => N = \left(\frac{\Delta p_r}{\rho D^2 \psi}\right)^{1/2} \tag{2.20}$$

$$Q = N D^3 \phi => N = \left(\frac{Q}{D^3 \phi}\right) \tag{2.21}$$
By equating through $N$:

$$N = \left( \frac{\Delta p_T}{\rho D^2 \psi} \right) = \left( \frac{Q}{D^3 \phi} \right) \quad (2.22)$$

This yields:

$$D^2 = \frac{\left( \frac{Q}{\phi} \right)}{\left( \frac{\Delta p_T}{\rho \psi} \right)^{1/2}} \Rightarrow D = \frac{Q^{1/2}}{\left( \frac{\Delta p_T}{\rho} \right)^{1/4}} \cdot \frac{\psi^{1/4}}{\phi^{1/2}} \quad (2.23)$$

Going back to $N = \frac{Q}{D^3 \phi}$:

$$N = \frac{\left( \frac{\Delta p_T}{\rho} \right)^{3/4}}{\frac{Q^{1/2}}{\phi^{1/2}}} \cdot \frac{\phi^{1/2}}{\psi^{3/4}} \quad (2.24)$$

So the definitions are:

$$D_s = \frac{D\left( \frac{\Delta p_T}{\rho} \right)^{1/4}}{Q^{1/2}} = \frac{\psi^{1/4}}{\phi^{1/2}} \quad (2.25)$$

$$N_s = \frac{NQ^{1/2}}{\left( \frac{\Delta p_T}{\rho} \right)^{3/4}} = \frac{\phi^{1/2}}{\psi^{3/4}} \quad (2.26)$$

Most importantly these two variables are specified at best efficiency.

The procedure is:

- scale to a performance specification: $\Delta p_T, Q, \rho$
- use $N_s$ and $D_s$
- arrive at the optimum $N$ and $D$

This requires geometric similarity.
In the 1950s, Cordier carried out an extensive empirical analysis using experimental data. He attempted to correlate the data in terms of $N_s, D_s, \eta_f$ using $\Delta p_f$. He found that turbomachines which had good to excellent efficiencies tended to group along a definable curve when plotted with their values of $N_s, D_s$.

He further found out that the efficiencies of these machines grouped into a definable curve as a function of $D_s$.

The derivation of the Cordier curve is as follows:

$$y = ax^b \Rightarrow \log y = \log a + b \log x \quad (or, N_s = aD_s^b)$$

Using the graph below (Fig. 2.2), we can derive the values for $a$ and $b$

$$x = D_s = 1, y = N_s = 9 : a = 9$$

$$x = 8.5, y = 0.1 : b = \frac{\log(0.1) - \log(9)}{\log(8.5)} = -2.103$$

![Figure 2.2: Cordier Diagram](image)
So in the upper box

\[ N_s \equiv 9D_s^{-2.103} \quad (\text{for } D_s \leq 2.8) \quad (2.27) \]

Similarly for the lower box

\[ N_s \equiv 3.25D_s^{-1.126} \quad (\text{for } D_s \geq 2.8) \quad (2.28) \]

The efficiency band shown below (Fig. 2.3) represents the best total efficiency that can reasonably be expected. This band can be converted into a piecewise set of simple equations as follows:

\[
\begin{align*}
\eta_T &= 0.149 + 0.625D_s - 0.125D_s^2 & \text{for } & D_s \leq 2.5 \\
\eta_T &= 0.864 + 0.0531D_s - 0.01106D_s^2 & \text{for } & 2.5 \leq D_s \leq 5.0 \\
\eta_T &= 1.1285 - 0.0529D_s & \text{for } & 5.0 \leq D_s \leq 10
\end{align*}
\]

Figure 2.3: Efficiency vs. Specific Diameter
2.4 ) Procedure and Results

2.4.1 ) Compressor Scaling:

2.4.1.1 ) Surge and Stall

An essential part of the design process for a compressor is being able to predict the condition at which instability will occur. In most literature the outcome of the instability will be referred to as stall, but either stall or surge can happen. For example, stall can be used to define fully developed rotating stall. Stall can also refer to the fluid dynamic transient which initiates a surge, or stall can be used to indicate that the flow in a blade row is separated. Compressors can often work quite satisfactorily with regions of separated flow and that the appearance of separation (often called stall) on one blade row is not necessarily an indicator of imminent compressor instability and stall.

Centrifugal compressors can operate fairly satisfactorily with rotating stall present. Centrifugal compressors can also operate with axisymmetric stall near the inducer tips or with stationary non-axisymmetric stall produced by downstream asymmetry, usually the volute. The centrifugal compressors tolerance to the stalled regions is mostly because so much of the pressure rise is produced by centrifugal effects which will occur even in the presence of rotating stall cells or other forms of separated flow. Therefore, surge is usually the mode of low flow behavior which is most important in centrifugal compressors. [11]

If a diffusing flow is found against a surface, then the possibility for the flow to be retarded so severely that it no longer follows the surface exists. The streamlines adjacent to the surface will leave the surface and a region of reverse flow will develop from that point along the surface. Thus, the momentum in the streamlines adjacent to the surface is insufficient to overcome the viscous shear stresses and adverse pressure gradient along the
surface. If the adverse pressure gradient and viscous shear stresses are high enough to reduce the fluid velocity to zero, it is forced to deviate from the surface. Large viscous shear stresses predominate locally and the flow becomes reoriented or separated.

Several elements of a compressor stage can stall without the entire stage stalling. If a stage has a very strong stall in one of its elements, or a number of elements together collectively stall, so that the head versus flow characteristic is no longer stable (negatively sloped), then the stage has entered stage stall. Basically, stage stall is the condition where the basic flow characteristic of the stage alone is no longer stable and the head/flow characteristic becomes positive.

For a compressor operating at a given speed, as the flow is continuously reduced there is usually a condition where the system will no longer operate in a stable manner. Large oscillations in the impeller discharge and inlet conditions will result. The system and stage then interact together in a violent manner giving surge. A phenomenon in which a compressor interacts in an unstable manner with other components to give a strongly coupled, fluctuating flow in the network, with complete flow reversal throughout the stage on cyclic basis is known as surge.

The idea of rotating, or dynamic, stall is far more difficult to report in detail than static stall. Static stall conditions are stationary in nature and can be associated with a fixed location in the machine. Dynamic stall conditions are not fixed and rotate in the machine system at some fraction of the rotational speed of the machine. The blockage caused by the stall cell leads to a reduction in the blade incidence on one side and an increase on the other. The blade with the increased incidence will stall next. The stall tends to run in the direction in which the incidence is increased. For a rotor row it means that the cell moves backwards relative to the rotor, for a stator it moves in the direction of the rotor. If viewed by a stationary observer, the rotation of the stall cell is always in the same direction as the rotor.
Such dynamic conditions can exist in the inducer, in the impeller, or in the diffuser. In centrifugal compressors, the vaneless diffuser is by far the most common part to suffer problems of dynamic instability.

Diffuser rotating stall can be triggered by a local inversion of the radial velocity component in the vaneless diffuser theoretically and experimentally. So, the initiation of vaneless diffuser rotating stall depends on the boundary layer (local return flow and increased blockage), but the dynamic characteristics (number of stall cells and propagation speed) depend on the inviscid part of the flow.

Large amplitude velocity and pressure variations in the vaneless diffuser and upstream of the impeller are characteristics of abrupt impeller rotating stall. It can occur on either the negative or positive slope sides of the performance curve and results in a discontinuity in the pressure rise curve.

2.4.1.2 ) Formulation

The examples and development so far have shown how to use $\phi$ and $\psi$ with $\xi$ to change the level of performance of a given design through the changes in speed, diameter and density. This was done by first calculating $\phi$, $\psi$ and $\xi$ from a known machine, followed by recalculating $\Delta p$, Q and P by choosing new values of $\rho$, N and D.

A more common procedure for scaling involves rearranging the low-speed nondimensional parameters presented before into ratios of the physical parameters by remembering that, with a geometrically similar change, $\phi$, $\psi$ and $\xi$ will be invariant (subject to restrictions or changes in Mach and Reynolds numbers). But due to the small sizes of the turbomachinery maps used, the conventional linear scaling methods [23] used don’t give accurate results, so a correction factor which is obtained through a trial and error
procedure is added to the nondimensionalizing terms. The method presented is not obtained through experimental results but rather through the turbomachinery maps available from literature.

For a compressor, the non-dimensionalized volumetric flow rate is calculated as:

\[ \phi = \frac{Q}{ND^3} \]  

(2.29)

First a correction factor is necessary for Eq. (2.29) as indicated in the literature. (to get a result in a certain scale) The factor to non-dimensionalize the mass flow rate is \( \sqrt{RT / PA} \).

\[ \frac{m.a_0}{p.A} = \frac{m.\sqrt{RT_0}}{p.A} = \text{unitless} \]  

(2.30)

Starting from this point, several attempts are made to find a correction factor to bring all volumetric flow rates to a certain range. The factor which gave the best results is as follows:

\[ \phi_c = \phi\left(\frac{P_{\text{des}}^2}{\sqrt{T_{\text{des}}}^2}, \frac{\sqrt{T_{\text{ref}}}}{P_{\text{ref}}^2}\right) \]  

(2.31)

In this case, \( T_{\text{ref}}, P_{\text{ref}} \) are assumed to be constant for the pressure range in a certain compressor. As is the case in literature, the constant terms are taken out from the correction factor to simplify the formulation using the reference values. (\( T_{\text{ref}} = 300K, P_{\text{ref}} = 1 \text{atm} \))

The pressure ratio is corrected using linear scaling, which assumes:

\[ PR^* = \beta.PR \]  

(2.32)
where $PR^*$ is the corrected pressure ratio and $\beta$ is the scaling parameter with the desired design point.

We had two compressor maps available (not exactly geometrically similar) and the procedure outlined above is applied to these compressors. The design values are as follows:

<table>
<thead>
<tr>
<th>Design Values</th>
<th>LARGE</th>
<th>SMALL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diameter: (cm)</td>
<td>13</td>
<td>4.5</td>
</tr>
<tr>
<td>N: (krpm)</td>
<td>90</td>
<td>120</td>
</tr>
<tr>
<td>Efficiency: (%)</td>
<td>79</td>
<td>74</td>
</tr>
<tr>
<td>Mass Flow Rate: (kg/s)</td>
<td>0.4</td>
<td>0.1054</td>
</tr>
<tr>
<td>Density: (kg/m3)</td>
<td>1.2</td>
<td>1.2</td>
</tr>
<tr>
<td>Pressure Ratio:</td>
<td>4.5</td>
<td>2.10</td>
</tr>
</tbody>
</table>

Table 2.1: Design point values of the large and the small compressors

The maps for the large and the small compressors are given in figures (2.4) and (2.5), respectively. After applying the outlined procedure above, the scaled maps are generated, which are given in figures (2.6) and (2.7) for the scaled maps for the large and the small compressors, respectively.
Fig. 2.4: Pressure ratio vs. Non-dimensional Volumetric Flow rate of the large compressor

Fig. 2.5: Pressure ratio vs. Non-dimensional Volumetric Flow rate of the small compressor
Fig. 2.6: Corrected Pressure ratio vs. Corrected Non-dimensional Volumetric Flow rate of the large compressor

Fig. 2.7: Corrected Pressure ratio vs. Corrected Non-dimensional Volumetric Flow rate of the small compressor
2.4.2) Expander Scaling:

A similar procedure is followed for expander scaling, that is:

\[ \phi_c = \phi\left(\frac{P_{\text{des}}^2}{\sqrt{T_{\text{des}}}}, \frac{\sqrt{T_{\text{ref}}}}{P_{\text{ref}}}\right) \]  (2.33)

For the pressure, linear scaling is employed.

There were three expander maps available; small, medium and large, which are given in figures (2.8), (2.9) and (2.10), respectively. The design point values of the above parameters are listed in Table 2.2. Two different comparisons are made; between small and the medium, and between small and large to see the effect of size similarity.

Fig. 2.8: Non-dimensional Volumetric Flow rate vs Pressure ratio of the small expander
Fig. 2.9: Non-dimensional Volumetric Flow rate vs Pressure ratio of the medium sized expander

Fig. 2.10: Non-dimensional Volumetric Flow rate vs Pressure ratio of the large expander
The design points are:

<table>
<thead>
<tr>
<th>Design Values</th>
<th>LARGE (cm)</th>
<th>MEDIUM (cm)</th>
<th>SMALL (cm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diameter</td>
<td>16</td>
<td>10</td>
<td>8.5</td>
</tr>
<tr>
<td>N: (krpm)</td>
<td>130</td>
<td>98</td>
<td>100</td>
</tr>
<tr>
<td>Mass Flow Rate: (kg/s)</td>
<td>0.45</td>
<td>0.414</td>
<td>0.3</td>
</tr>
<tr>
<td>Pressure Ratio</td>
<td>3.5</td>
<td>2.91</td>
<td>2.6</td>
</tr>
</tbody>
</table>

Table 2.2: Design point values for the large, medium and the small expander

The analysis was performed for the expanders as follows:

(i) Between the medium and the small expanders. (The scaled maps for this case are shown in Figures 2.11 and 2.12, respectively)

(ii) Between the small and the large expanders. (The scaled maps for the small and the large expanders are shown in Figures 2.12 and 2.13, respectively)

Fig. 2.11: Corrected Non-dimensional Volumetric Flow rate vs Corrected Pressure ratio of the medium sized expander
Fig. 2.12: Corrected Non-dimensional Volumetric Flow rate vs Corrected Pressure ratio of the small expander

Fig. 2.13: Corrected Non-dimensional Volumetric Flow rate vs Corrected Pressure ratio of the large expander
2.5) Comparison of Linear and Non-Linear Scaling Methods

The figures below shows the scaling of the large and small compressor maps (Figures 2.4 and 2.5) using linear and non-linear scaling, respectively. As can be seen from the figures, non-linear scaling brings the maps much closer together and is better in collecting various maps to the same order of magnitude range.

![Diagram showing linear scaling comparison between large and small compressors.](image-url)

Fig. 2.14: Corrected Pressure ratio vs. Corrected Non-dimensional Volumetric Flow rate of the large and small compressors using Linear Scaling
Fig. 2.15: Corrected Pressure ratio vs. Corrected Non-dimensional Volumetric Flow rate of the large and small compressors using Non-Linear Scaling

2.6) Interpolation Routine

Working with various maps and requiring data which are not exactly on the curves on the maps, an interpolation/extrapolation routine is devised. (See Appendix A ) This program reads the data files for the turbomachinery maps. A turbomachinery map has two forms: mass flow rate vs. pressure ratio or efficiency vs. pressure ratio, where both mass flow rate and efficiency are also functions of rotational speed of the rotor shaft. (Fig. 2.16 ) So this routine makes an interpolation/extrapolation to find the desired variable, whether it is mass flow rate, pressure ratio or efficiency using already supplied data from a turbomachinery map. It can be utilized to use more than one map and interpolate between the them. The program can also be used to output the values from the original maps to give an approximate error estimate.
2.7) Discussion

Various non-dimensional parameters and many scaling procedures exist for scaling of the expander and compressor maps. Scaling of relatively large geometrically similar turbomachines can be done quite accurately. But for small scale units, some of the factors that are negligible for larger units become critical. For example, at large scales, Reynolds number effect is neglected. But at small scales, it plays a major role. Besides the Reynolds number effect, the tip clearance is also very important for small units. In large scale
turbomachinery, the gap between the blades and the casing, namely the tip clearance, is relatively small compared to the overall size of the machine. But again at small scales, the gap becomes an important factor. As a result, as can be seen from the corresponding maps, the accuracy of scaling is open to discussion. As an example, take the scaling between the large and the small compressors. For a corrected non-dimensional volumetric flow rate of 0.3 and a RPM/RPM_design of 1, the scaled large compressor map gives surge while the scaled small compressor map gives a finite pressure ratio, namely about 3.1. Similar cases exist for the expander maps. Overall, the geometry of the turbomachinery units is a very important factor for the accuracy of the scaling methods presented.

2.8) Recommendations

It is recommended that rather than using a simple map, multiple of maps ranging from large to small sizes be used for scaling. This can be accomplished by implementing an interpolation scheme based on say the actual size of the unit to be designed compared to the ones for which maps are available. Another parameter that can be used for interpolation is the power output of the unit being designed. If this procedure is to be followed an accurate interpolation routine needs to be developed by digitizing the maps and tabulating the data in discrete form.

To overcome the problem of uncertainty in scaling small sized turbomachines, Computational Fluid Dynamics (CFD) is the most practical and efficient solution. Besides giving accurate results, it can also predict unsteady flow consequences, such as surge and stall.

It is further recommended that a more complete map be generated in the range of interest using CFD. To overcome the problem of uncertainty in scaling of small sized
turbomachines, CFD approach will be the most practical and efficient solution. Besides more accurate results, CFD can also predict unsteady flow consequences, such as surge and stall.
3.1) Introduction

Numerical methods are good tools for the solution of differential equations which don’t have exact analytical solutions. This becomes especially very important in the area of fluid mechanics since the equations considered are non-linear differential equations.

This chapter introduces a code (see Appendix B) which uses the projection method [13] for the solution of flow variables for one-dimensional transient compressible duct flow with variable area, friction and heat transfer. Code formulation is described in detail and then the verification cases are introduced for different flows such as temperature specified converging diverging duct flow (manufactured solution), standing wave flow and constant area pipe flow with heat transfer.

3.2) Code Formulation

Consider compressible one-dimensional unsteady flow in a duct of arbitrary cross sectional area. (Fig. 3.1)
The conservation equations for such flows are:

**Continuity:**
\[
\frac{\partial (\rho A)}{\partial t} = - \frac{\partial (m)}{\partial x}
\]  
(3.1)

**Momentum:**
\[
\frac{\partial (m)}{\partial t} = - \frac{\partial}{\partial x} (u m + p A) + p \frac{dA}{dx} = - \frac{\partial}{\partial x} (u m) - A \frac{dp}{dx}
\]  
(3.2)

Energy Equation (neglecting the potential energy):
\[
\frac{\partial}{\partial t} (\rho A (e + \frac{u^2}{2})) = - \frac{\partial}{\partial x} (m (h + \frac{u^2}{2})) - q^* + u \frac{4A}{R} \tau_w
\]  
(3.3)

where \( q^* \) is the heat transfer per unit length, \( e \) is the internal energy and \( h \) is the enthalpy.

We multiply (3.2) by the velocity and rearrange the terms to obtain the mechanical energy equation:
\[
\frac{\partial}{\partial t} (\rho A \frac{u^2}{2}) = - \frac{\partial}{\partial x} (m \frac{u^2}{2}) - uP \frac{dA}{dx}
\]  
(3.4)

After subtracting (3.4) from (3.3) and using the definition \( h = e + \frac{P}{\rho} \) we obtain;
\[
\frac{\partial}{\partial t} (\rho A h) = - \frac{\partial}{\partial x} (m h) - q^* + \frac{\partial}{\partial t} (AP) + Au \frac{dP}{dx} + u \frac{4A}{R} \tau_w
\]  
(3.5)

Since \( \frac{\partial A}{\partial t} = 0 \) the last two terms amount to \( W_p = A DP / Dt \), which is the pressure work.

The heat transfer rate can be calculated from;
\[
q^* = P_{er} h_{conv} (T - T_a)
\]  
(3.6)

Here, \( P_{er} = \) perimeter, \( T_a = \) ambient temperature and \( h_{conv} \) is an average heat transfer coefficient at that location.

Finally, (3.6) can be written in terms of temperature by using the relation ;
\[
dh = C_p dT
\]  
(3.7)
Numerical Solution of the Momentum Equation:

We let \( F = \rho Au = m \) and write the momentum equation as;

\[
\frac{\partial F}{\partial t} = f_{rhs} - A \frac{\partial P}{\partial x} + S
\]  
(3.8)

where \( f_{rhs} \) represents all the terms involving convection, diffusion and friction etc. and \( S \) is the source term.

We first calculate an approximate value \( \hat{F} \) without the influence of pressure using the MacCormack’s Method [18], i.e.

Predictor : \( \hat{F} = F^n + \Delta t f^n_{rhs} \)  
(3.9)

Then apply a correction step

Corrector : \( F^* = F^n + \frac{\Delta t}{2} (f^n_{rhs} + \hat{f}_{rhs}) \)  
(3.10)

where \( \hat{f}_{rhs} \) is the rhs of eqn.(3.8) without the pressure and evaluated using \( \hat{F} \) and the source.

Now if we had included the pressure we would have had

\[
F^{n+1} = F^n + \frac{\Delta t}{2} (f^n_{rhs} + \hat{f}_{rhs}) - \Delta t A \frac{\partial P^{n+1}}{\partial x}
\]  
(3.11)

After subtracting (3.10) from (3.11) to obtain

\[
F^{n+1} - F^* = -\Delta t A \frac{\partial P^{n+1}}{\partial x}
\]  
(3.12)

We assume that if the pressure field is correct, \( F^{n+1} \) should satisfy the continuity equation (3.1), i.e.

\[
\frac{\partial}{\partial x} (F^{n+1}) = -\frac{\partial (\rho A)}{\partial t} = -A \frac{\partial \rho}{\partial t}
\]  
(3.13)

Taking the derivative of (3.12) w.r.t. \( x \) and using (3.13) yields an equation for pressure, i.e.

\[
\frac{\partial}{\partial x} (A \frac{\partial P^{n+1}}{\partial x}) = \frac{1}{\Delta t} \left[ \frac{\partial \hat{F}^*}{\partial x} + A \frac{\partial \rho}{\partial t} \right]
\]  
(3.14)

Equation (3.14) corresponds to the Poisson equation for pressure in one-dimensional flows.

The derivative of the density wrt time can be approximated explicitly. Eg,

\[
\frac{\partial \rho}{\partial t} = \left( \rho^n - \rho^{n-1} \right) / \Delta t
\]  
(3.15)

After the pressure field calculation (3.12) is used to calculate \( F^{n+1} \) from \( F^* \) field.
Equation (3.15) is discretized using finite volume method. Equations (3.9) and (3.10) constitute applications of MacCormack method. The energy equation is solved using the Finite Volume Method [18] which will be described next.

Solution of the Energy Equation:

The energy equation is written as: (k: thermal conductivity)

\[
\frac{\partial}{\partial t} (\rho A h) + \frac{\partial}{\partial x} (\rho A u h) = \frac{\partial}{\partial x} \left( \frac{A k}{C_p} \frac{dh}{dx} \right) - \frac{P h_{\text{com}}(T - T_o)}{C_p A} + A \frac{DP}{Dt} + u \frac{4A}{R} \tau_w
\]

(3.16)

Density is calculated from the ideal gas law:

\[
p = \rho RT
\]

(3.17a)

The continuity equation is:

\[
\frac{\partial}{\partial t} (\rho A) + \frac{\partial}{\partial x} (\rho A u) = 0
\]

(3.17b)

Integrating (3.16) over a control volume (see Fig. 3.1) we get;

\[
\frac{\Delta x_p (\rho A h)^{n+1}}{\Delta t} - \frac{\Delta x_p (\rho A h)^n}{\Delta t} + (\rho A u h)^n - (\rho A u h)_w = \left( \frac{A k}{C_p} \frac{dh}{dx} \right)^n - \left( \frac{A k}{C_p} \frac{dh}{dx} \right)_w - S_p Ah + S_c \Delta \forall
\]

(3.18a)

Collecting and rearranging the terms:

\[
\Rightarrow \frac{\Delta \forall}{\Delta t} (\rho h)^{n+1} - \frac{\Delta \forall}{\Delta t} (\rho h)^n = -(F_e - F_w) + D_e - D_w - S_p \Delta \forall h + S_c \Delta \forall
\]

From the continuity equation (3.17b) we obtain:

\[
(h^{n+1}) \frac{\Delta \forall}{\Delta t} \rho^{n+1} - (h^n) \frac{\Delta \forall}{\Delta t} \rho^n = -(C_e - C_w)h^{n+1}
\]

(3.18b)

After subtracting (3.18b) from (3.18a) we obtain:

\[
\Rightarrow \frac{\Delta \forall}{\Delta t} \rho^n(h^{n+1} - h^n) = -[(F_e - C_i h^{n+1}) - (F_w - C_i h^{n+1})] + D_e - D_w - S_p \Delta \forall h + S_c \Delta \forall
\]

(3.19)

where

\[
F_e = (\rho u A) h = C_e h_e
\]

\[
F_w = C_w h_w
\]
Using the above notation, the left hand side (LHS) of equation (3.19) can be written as:

\[
LHS = -[C_e(h_e - h_{e}^{n+1}) - C_w(h_w - h_{w}^{n+1})] + D_e - D_w - S_p\Delta\forall h_{e}^{n+1} + S_e\Delta\forall
\]

\[
D_e = \left(\frac{Ak}{C_p} \frac{dh}{dx}\right)_e = d_e(h_e - h_{e})
\]

\[
d_e = \left(\frac{Ak}{C_p\Delta x}\right)_e
\]

\[
D_w = d_w(h_p - h_{w})
\]

\[
d_w = \left(\frac{Ak}{C_p\Delta x}\right)_w
\]

After these are calculated in the program, they are plugged in the following;

\[
a_p^0 = \frac{\Delta\forall}{\Delta t} \rho^n
\]

\[
a_p^0(h_{e}^{n+1} - h_{e}^n) = -[F_e - D_e - C_e h_p] + [F_w - D_w - C_w h_p] - S_p\Delta\forall h + S_e\Delta\forall
\]

\[
a_p^0(h_{w}^{n+1} - h_{w}^n) = -[J_e - C_e h_p] + [J_w - C_w h_p] - S_p\Delta\forall h_p + S_e\Delta\forall
\]

\[
= -a_e h_e + a_w (h_w - h_p) - S_p\Delta\forall h_p + S_e\Delta\forall
\]

where

\[
J = F - D
\]

\[
C_e = (\rho u A)_e
\]

\[
P_e = \frac{C_e}{D_e} \quad \text{(Pecklet number)}
\]

\[
D_e = \left(\frac{Ak}{C_p\Delta x}\right)_e
\]

\[
S_e = \frac{P_{er} h_{con}}{A} + \frac{Dp}{Dt}; \quad S_p = \left(\frac{P_{er} h_{con}}{C_p A}\right)
\]

For linearization purposes, coefficients should be calculated at the old time level. Equation (3.23) constitute a set of linear system of equations that are solved using the tri-diagonal matrix algorithm (TDMA).
3.3) Verification of the One-Dimensional Transient Compressible Flow Solver

The code is verified through three different cases. For the first two cases, the energy equation is not solved (temperature profile is specified). In the third case, the solution of the energy equation is verified through a pipe flow with heat conduction/convection.

3.3.1) Steady state flow verification

Steady state flow solution of the code is verified through comparison with a manufactured solution for a converging-diverging nozzle profile. Manufactured solution is a solution of a problem that satisfies the equations but does not necessarily represent a real physical situation. The formulation of the manufactured solution is as follows:

\[ m = \text{given} \]
\[ \frac{A(x)}{A_0} = f(x) \quad ; \quad \frac{T(x)}{T_0} = g(x) \quad \text{(Area and temperature profiles are also given)} \]

Using the area and temperature profiles, the mass flow rate and velocity are calculated as:

\[ \dot{m} = \rho u A \quad \Rightarrow \quad u = \frac{m}{\rho A} = \frac{m g(x)}{p^* P_0 A_0 f(x)} \]

Define \( R(x) = \frac{g(x)}{f(x)} \quad \Rightarrow \quad u^* = \frac{R(x)}{p^*} \quad \text{(3.25)} \]

\[ \frac{\partial (m u)}{\partial x} = -A \frac{\partial P}{\partial x} - m \frac{\partial u^*}{\partial x} u_0 = -f(x) A_0 \frac{\partial P^*}{\partial x} P_0 \Rightarrow \frac{\partial P^*}{\partial x} = -\frac{\partial u^*}{\partial x} \frac{m u_0}{A_0 P_0} = \frac{1}{f(x)} \quad \text{(3.26)} \]

Taking the derivative of (3.25) w.r.t. x and substituting into (3.26) yields:

\[ \frac{\partial u^*}{\partial x} = \frac{dR(x)}{dx} \frac{1}{P^*} - R(x) \frac{1}{p^*^2} \frac{\partial P^*}{\partial x} \]

\[ \frac{\partial P^*}{\partial x} = -\left( \frac{dR(x)}{dx} \frac{1}{P^*} - R(x) \frac{1}{p^*^2} \frac{\partial P^*}{\partial x} \right) c \frac{\partial P^*}{\partial x} = -\frac{c dR(x)}{dx} \frac{P^*}{f(x) - cR(x)} \quad \text{(3.27)} \]

This last equation gives the pressure profile through the nozzle as a function of x.
Second order Runge-Kutta method (see Appendix C) is used to solve for the pressure profile, from Eqn. (3.27). Adams-Moulton scheme (see Appendix D) can also be used to determine this profile. After running the second order Runge Kutta scheme for a specified mass flow rate, 0.06 kg/s in this case, a pressure difference between the inlet and outlet of the nozzle is calculated. Then, this pressure difference is applied to the 1-D compressible flow solver and the mass flow rate obtained here is compared with the one previously specified for the Runge Kutta scheme.

The area and temperature profiles in the program are specified as follows:

\[
\frac{A(x)}{A_0} = f(x) = [0.51 + (x - 0.7)^2]^{1/2} \quad \text{where } A_0 = 10^{-3} \text{ m}^2
\]

\[
\frac{T(x)}{T_0} = g(x) = [0.51 + (x - 0.7)^2]^{1/2} \quad \text{where } T_0 = 300 \text{ K}
\]

The inlet pressure of the flow is set equal to atmospheric pressure, \( P_{\text{inlet}} = P_{\text{atm}} = 101325 \text{ Pa}, \) and the exit pressure is \( P_{\text{exit}} = (1 - \text{pres}_\text{fac}).(P_{\text{atm}}), \) where \( \text{pres}_\text{fac} \) is a factor in the program specified according to the pressure difference obtained in the Runge Kutta scheme. ( \( \text{pres}_\text{fac} = 0.02331 \) in this case)

Mass flow rate is specified for the manufactured solution and the pressure profile is obtained from eqn. (3.27). We also know the area and the temperature profiles so we can calculate the density profile from the ideal gas law, \( P = \rho RT \), and the velocity profile can be calculated using:

\[
\dot{m} = \rho Au
\]

The program run specifications are as follows:

- Number of control volumes (CV) = 40
- Time step = \( 1 \times 10^{-6} \)
- Total flow time = 3 seconds

The results obtained from the manufactured solution and the program agree very well and are shown below in Figures 3.2 – 3.6: (RK stand for Runge Kutta scheme, solution for the manufactured solution; NPM stands for Nozzle_Projection_Method, 1-D compressible flow solver)
Figure 3.2: Comparison of Pressure Profiles (Code vs. the Manufactured Solution)

Figure 3.3: Comparison of Velocity Profiles (Code vs. the Manufactured Solution)
Figure 3.4: Comparison of Density Profiles (Code vs. the Manufactured Solution)

Figure 3.5: Comparison of Temperature Profiles (Code vs. the Manufactured Solution)
3.3.2) Unsteady Flow verification

Unsteady flow results of the code are compared with the analytical solution of a standing wave profile. The velocity profile of a standing wave is:

\[ u = \cos(2\pi \lambda x) \sin(2\pi f t) \]  

(3.29)

where \( \lambda \) is the wave length and \( f \) is the frequency. For our case, the comparison is made for flow through a constant area pipe with a diameter \( d = 0.1 \) m and a length \( L = 1 \) m. The wave length, \( \lambda \), is chosen to be \( 1/100 \) m and frequency to be \( 10 \) Hz. (typical values for \( \lambda \) and \( f \) for standing waves) Since \( \lambda \) is very large compared with the maximum value of \( x \), which is \( L = 1 \) in this case, we can assume the velocity profile to be:

\[ u \approx \sin(2\pi f t) \]  

(3.30)

Writing the x-momentum equation:

\[
\frac{\partial}{\partial t} (\rho u A) + \frac{\partial}{\partial x} (\rho u^2 A + PA) = \frac{dA}{dx} - \frac{4A\tau_w}{R}
\]

(3.31)
where $\tau_w$ is the wall shear stress and is given by;

$$\tau_w = \frac{1}{2} C_f \rho |u|u$$

(3.32)

$C_f = $ friction coefficient

and assuming $\rho = $ constant and simplifying equation (3.31) yields;

$$\frac{\partial}{\partial t}(u) + \frac{\partial}{\partial x}(u^2) = -\frac{dP}{dx} \cdot \frac{1}{\rho} - \frac{2}{R}(C_f u |u|)$$

(3.33)

Inserting equation (3.30) into (3.33) and taking the integral from 0 to $x$ we get;

$$\frac{P}{\rho} - \frac{P_0}{\rho_0} = 2\pi f \cos(2\pi ft) + \frac{1}{R} C_f |\sin(2\pi ft)| |\sin(2\pi ft)| x$$

(3.34)

Equation (3.34) gives the analytical solution for the pressure as a function of $x$ and $t$. This equation is used to give the inlet and exit pressure boundary conditions for the computer program and the velocity profile obtained from the program is compared with the analytical solution.

Energy equation is not solved in this case and a temperature of 288K is specified throughout the duct. Temperature at the inlet and the outlet are also specified. The velocity and pressure profiles obtained from the computer program are compared with the analytical solution at the middle of the duct, $x = 0.5$ m.
Figure 3.7: Comparison of velocity wrt time at x = 0.5 (Code vs. Analytical Soln.)

Figure 3.8: Comparison of pressure wrt time at x = 0.5 (Code vs. Analytical Soln.)
The specifications for these runs were as follows:

Number of Control Volumes (CV) = 40

Time step = $1.0 \times 10^{-4}$

The effects of number of control volumes and the time step on the velocity and the pressure profiles are also analyzed. The figures below show the change of velocity and pressure with respect to time with different CV’s and time steps. It can be seen that the code is very stable and gives very close results with different parameters.

Figure 3.9: Comparison of velocity wrt time at $x = 0.5$ with different number of control volumes
Figure 3.10: Comparison of pressure wrt time at x = 0.5 with different number of control volumes

Figure 3.11: Comparison of velocity wrt time at x = 0.5 with different time steps
3.3.3) Energy Equation Verification

The energy equation was not solved for in the verification cases 3.3.1 and 3.3.2. This section analyzes a constant area pipe flow where the results from the commercial CFD program Fluent® is compared with the results obtained from the present program. The Fluent calculations were obtained from Li [15].

The constant area pipe has a diameter \( d = 0.1 \) m and length \( L = 1 \) m. Inlet pressure is specified as atmospheric pressure, \( P_{\text{inlet}} = P_{\text{atm}} = 101325 \) Pa, and the outlet pressure is defined as:

\[
P_{\text{outlet}} = (P_{\text{inlet}} - 400) + \Delta P \cdot \sin(2\pi \text{ ft})
\]

This is a pressure driven flow and the friction coefficient \( C_f \) is set equal to 0.015. The heat transfer coefficient is 200 W /m.K, which is a typical value. [12] The ambient temperature is 450 K and the inlet temperature of the flow is given by:

\[
T_{\text{inlet}} = T_{\text{ambient}} + \Delta T \cdot \sin(2\pi \text{ ft})
\]

The outlet temperature is obtained from the last two nodes of the domain.

The run specifications for Fluent® are as follows:
As the turbulence model, RNG k-ε is used. QUICK Scheme is used for the momentum equation. The time marching scheme is a 2nd order implicit scheme. Time step is $1 \times 10^{-3}$ seconds. Fluent results have been written to a file and then integrated to obtain area averaged quantities over the cross-section of the pipe. The comparison of temperature, velocity, pressure and density between the present program and Fluent at $x = 0.5$ m is given below:

![Temperature Comparison](image)

**Figure 3.13: Comparison of Temperature between Fluent and Present Program for Cf = 0.015**
Figure 3.14: Comparison of Velocity between Fluent and Present Program for $C_f = 0.015$

Figure 3.15: Comparison of Pressure between Fluent and Present Program for $C_f = 0.015$
Figure 3.16: Comparison of Density between Fluent and Present Program for Cf = 0.015

Figure 3.17: Comparison of Mass Flow Rate between Fluent and Present Program for Cf = 0.015
3.3.3.1) Discussion of the Energy Equation Solution

As can be seen from the above figures, there are differences between the results of Fluent and the present program. The reason for these differences is most probably the use of a constant friction coefficient. Fluent uses a time and position dependent friction factor (Fig.3.17) which adjusts itself dynamically according to flow conditions while the present program uses a constant value. The values obtained for temperature, velocity, pressure and density are highly dependent on the value of the friction factor. Figures 3.18 to 3.21 show the variation of the quantities for $C_f = 0.05$. It can be seen from these figures that there are considerable differences between the results obtained with $C_f = 0.015$. As $C_f$ increases pressure work against friction increases. This leaves less pressure available to drive the flow; hence $m$ decreases leading to much smaller velocities which in turn reduces the effective wall shear stress since $C_f$ is kept constant. The overall effect of reducing $C_f$ seems to be reasonable. But it is not clear why there are significant differences between two sets of runs.

![Figure 3.18: Variation of $C_f$ wrt time and position in Fluent](image)

Figure 3.18: Variation of $C_f$ wrt time and position in Fluent
Figure 3.19: Comparison of Temperature between Fluent and Present Program for Cf = 0.05

Figure 3.20: Comparison of Velocity between Fluent and Present Program for Cf = 0.05
Figure 3.21: Comparison of Pressure between Fluent and Present Program for $C_f = 0.05$

Figure 3.22: Comparison of Density between Fluent and Present Program for $C_f = 0.05$
3.4) Case Study – Diverging-Converging Nozzle

The steady flow through a diverging-converging nozzle is an interesting case study. The cross section of the nozzle in the flow direction is shown below;

Figure 3.24: Radius profile of the diverging-converging nozzle
For this case study, the heat transfer coefficient with the surroundings is taken to be an average value of 100 W/m.K. Ambient temperature is 300 K and the inlet temperature of the flow is 650 K. Friction is negligible. This is a pressure driven flow, inlet pressure is atmospheric pressure and the exit pressure is 800 Pa below atmospheric pressure.

The run specifications are;
- Number of control volumes = 40
- Time step = 1.\times10^{-4} \text{ seconds}
- Total run time = 10 \text{ seconds}

The results are as follows: (at t = 10 seconds)

Figure 3.25: Flowrate vs. x at time = 10 seconds
Figure 3.26: Velocity vs. x at time = 10 seconds

Figure 3.27: P/P_{inlet} vs. x at time = 10 seconds
Figure 3.28: Density vs. x at time = 10 seconds

Figure 3.29: Temperature vs. x at time = 10 seconds
The results are interesting in the sense that velocity and pressure are behaving the opposite way as they would be expected to behave, since for an adiabatic diverging-converging nozzle, velocity tends to decrease as the cross sectional area increases. [2] This is due to the high temperature difference with the surroundings, thus the heat transfer.

3.5) Conclusion

From the verification cases presented above, it can be concluded that the present pseudo-one dimensional transient model reproduces the analytical solutions. Thus it may be concluded that it is verified. However, comparison of the results from the present program to those from Fluent is not conclusive. As explained in Appendix E, after an integral analysis of mass and energy balance, a user mistake was discovered in the Fluent runs and it was corrected. The runs were repeated after correcting this mistake and the results are presented in Appendix F. After the correction, the agreement between the present calculations and those from Fluent has improved but unfortunately significant differences still remain.
Chapter 4:

CONCLUSIONS

Small sized turbomachinery scaling procedure presented gives satisfactory order of magnitude agreement between the scaled maps. The differences are due to the many factors concerning the mechanical design of the turbomachines. Finding two different maps with exactly the same physical design is a very difficult task considering that even finding any turbomachinery map is very difficult since these data are too expensive and time consuming to obtain. One other reason is that these maps are proprietary company information and is not distributed. With the procedure presented, one can construct small sized turbomachinery maps from the existing ones for relatively larger units, and get a fairly good estimation about the behavior of the turbomachine to be used in the design.

The one-dimensional transient compressible flow solver presented gives good agreement with exact solutions of various flow types, but as explained in Section 3.5, it is not clear why there are significant differences between the corresponding Fluent results and those from the present program. The use of constant friction and heat transfer coefficients in the present program is most probably the reasons behind the differences. The program can be used to solve for any type of duct flow with friction and heat transfer and can be incorporated within larger process control routines after validation.
Chapter 5:

REFERENCES


[15] Li, J., Private Communication, Mechanical and Aerospace Engineering Department, West Virginia University, Morgantown, WV (USA), 2003


Appendix A

c************************************************************************
c************************************************************************
c************************************************************************
c

--- Interpolation Routine for Turbomachinery Maps ---------------------
c ( Air Flow vs. Pressure Ratio & Efficiency vs. Pressure Ratio)
c

parameter (nmax=100)
common/emre/xdata(nmax,nmax),ydata(nmax,nmax),
& dummy1(nmax,nmax),dummy2(nmax,nmax),ndata(nmax),
& zdata(nmax)

Integer j,jl,jm,ju, selection,f
Character(len=30) d1,d2
Real x,yfinal,z_data,jd

open (unit=10, file='1ap.dat', status='unknown')
open (unit=11, file='2ap.dat', status='unknown')
  open (unit=12, file='3ap.dat', status='unknown')
open (unit=13, file='4ap.dat', status='unknown')
  open (unit=14, file='5ap.dat', status='unknown')
open (unit=15, file='6ap.dat', status='unknown')

  open (unit=16, file='1ep.dat', status='unknown')
open (unit=17, file='2ep.dat', status='unknown')
  open (unit=18, file='3ep.dat', status='unknown')
open (unit=19, file='4ep.dat', status='unknown')
  open (unit=20, file='5ep.dat', status='unknown')
open (unit=21, file='6ep.dat', status='unknown')

--- Air flow vs. Pressure Ratio --------------------------------------
f=6 ! Number of data files

Do i=1,f
    Read(9+i,*),ndata(i),zdata(i)
end do

Do k=1, f
    Do i=1,ndata(k)
        Read(9+k,*), xdata(k,i), ydata(k,i)
    end do
end do

Print*, 'Which of the two do you want to find; Air flow (1) &or Pressure ratio (2)?'
    Read*, Selection

    ********* The selection of the axes are made ****************

    If (Selection .eq. 1) then
        d1 = 'Pressure ratio'
        d2 = 'Air flow'
    Print*, 'Enter the value of Pressure ratio and & % (Rotational speed/Design):'
        Read*,x,z_data
    end if

    ******** Find the boundaries for rotational speed******
         Do i=1, f
            If ((z_data.gt.zdata(i)) .and. (z_data.le.zdata(i+1))) then
                g=i
            end if
         end do

    ******** Determine whether given data is out of range of the limits of the boundaries*****
         Do i=g,g+1
            If (x.gt.xdata(i,ndata(i)).or. (x.le.xdata(i,1))) then
                Print*, ' Data is out of range, we will extrapolate.'
            end if
       end if
end do
end if

If (Selection .eq. 2) then
d1 = 'Air flow'
d2 = 'Pressure ratio'

Print*, 'Enter the value of Air flow and \%(Rotational speed/Design)
&:'
Read*, x, z_data

c ****** Replace x with y and vice versa ********************

Do i=1, f
    Do j=1, ndata(i)
        dummy1(i,j) = xdata(i,j)
        dummy2(i,j) = ydata(i,j)
        ydata(i,j) = dummy1(i,j)
        xdata(i,j) = dummy2(i,j)
    end do
end do
c ******* Find the boundaries for rotational speed*******
Do i=1, f
    If ((z_data.gt.zdata(i)).and.(z_data.le.zdata(i+1))) then
        g=i
    end if
end do

C ******* Determine whether given data is out of range of the limits of the
boundaries*****

Do i=g, g+1
    If ((x.gt.xdata(i,ndata(i))).or.(x.le.xdata(i,1))) then
        Print*, ' Data is out of range, we will extrapolate.'
    end if
end do
end if

C ************ For N(g)
If (x.gt.xdata(g,ndata(g))) then
    slope = (ydata(g,ndata(g))-ydata(g,ndata(g)-1))/(xdata(g,ndata(g))-
    xdata(g,ndata(g)-1))
ydata_1 = ydata(g,ndata(g)) + slope*(x - xdata(g,ndata(g)))
goto 1020
end if

If (x.le.xdata(g,1)) then
  slope = (ydata(g,2)-ydata(g,1))/(xdata(g,2)-
  & xdata(g,1))
ydata_1 = ydata(g,1)-slope*(xdata(g,1) - x)
goto 1020
cend if

jl=0
ju=ndata(g)+1
10  If ((ju-jl).gt.1) then
    jm=(ju+jl)/2
If ((xdata(g,ndata(g)).ge.xdata(g,1)).eqv.(x.ge.xdata(g,jm)))then
    jl=jm
else
    ju=jm
endif
goto 10
dendif
If(x.eq.xdata(g,1))then
  j=ndata(g)-1
else
  j=jl
endif

slope = (ydata(g,j+1) - ydata(g,j))/(xdata(g,j+1)-xdata(g,j))
ydata_1 = ydata(g,j) + slope*(x - xdata(g,j))
1020  Print*
      Print*, 'The value of the function at the entered ', d1
      Print*, 'for N_',g,' is:'
      Print*, d2,'=',ydata_1
      Print*

  ************************** For N(g+1) **************************
If (x.gt.xdata(g+1,ndata(g+1))) then
  slope = (ydata(g+1,ndata(g+1))-ydata(g+1,ndata(g+1)-1))
  & /(xdata(g+1,ndata(g+1))-xdata(g+1,ndata(g+1)-1))
ydata_2=ydata(g+1,ndata(g+1)) + slope*(x - xdata(g+1,ndata(g+1)))
goto 1030
dend if
If (x.le.xdata(g+1,1)) then
   slope = (ydata(g+1,2)-ydata(g+1,1))/(xdata(g+1,2)-
   xdata(g+1,1))
   ydata_2 = ydata(g+1,1) - slope*(xdata(g+1,1) - x)
   goto 1030
end if

jd=0
jl=0
ju=ndata(g+1)+1
11  If ((ju-jl).gt.1.) then
   jm=(ju+jl)/2
   If ((xdata(g+1,ndata(g+1)).ge.xdata(g+1,1)).eqv.
   & (x.ge.xdata(g+1,jm)))then
      jl=jm
   else
      ju=jm
   endif
   goto 11
endif
If(x.eq.xdata(g+1,1))then
   j=ndata(g+1)-1
else
   j=jl
endif
slope=(ydata(g+1,j+1)-ydata(g+1,j))/(xdata(g+1,j+1)-xdata(g+1,j))
ydata_2 = ydata(g+1,j) + slope*(x - xdata(g+1,j))
1030 Print*
   Print*, 'The value of the function at the entered ', d1
   Print*, 'for N_','g+1', 'is:'
   Print*, d2,'=', ydata_2
   Print*
   c     *****Now we find the y value by interpolating the zdata values******
   yfinal=ydata_1+(z_data-zdata(g))*((ydata_2-ydata_1)
   & /(zdata(g+1)-zdata(g)))
   Print*, 'The value of the function for the entered ', d1
   Print*, d2,'=', yfinal
   Print*
c
**************************************************************************
c
************ Efficiency vs. Pressure Ratio ************************************
c
**************************************************************************

Do i=1,f
Read(15+i,*) ndata(i),zdata(i)
end do

Do k=1, f
Do i=1, ndata(k)
Read(15+k,*) xdata(k,i), ydata(k,i)
end do
end do

Print*, 'Which of the two do you want to find; Efficiency (1) &or Pressure ratio (2)? '
Read*, Selection

********* The selection of the axes are made ************

If (Selection .eq. 1) then
  d1 = 'Pressure ratio'
  d2 = 'Efficiency'

Print*, 'Enter the value of Pressure ratio and & % (Rotational speed/Design):'
Read*, x,z_data

******** Find the boundaries for rotational speed******
Do i=1, f
  If ((z_data.gt.zdata(i)).and.(z_data.le.zdata(i+1))) then
    g=i
  end if
end do

******** Determine whether given data is out of range of the limits of the boundaries****
Do i=g,g+1

    If (x.gt.xdata(i,ndata(i)).or.(x.lt.xdata(i,1))) then
        Print*, ' Data is out of range, we will extrapolate.'
    end if
end do
end if

If (Selection .eq. 2) then
    d1 = 'Efficiency'
    d2 = 'Pressure ratio'
    Print*, 'Enter the value of Efficiency and % (Rotational speed/Desi & gn):'
    Read*, x, z_data

    ******** Replace x with y and vice versa ********************

    Do i=1,f
        Do j=1,ndata(i)
            dummy1(i,j) = xdata(i,j)
            dummy2(i,j) = ydata(i,j)
            ydata(i,j) = dummy1(i,j)
            xdata(i,j) = dummy2(i,j)
        end do
    end do
end do

    ******** Find the boundaries for rotational speed ******
    Do i=1, f
        If ((z_data.gt.zdata(i)).and.(z_data.le.zdata(i+1))) then
            g = i
        end if
    end do
end do

    ******** Determine whether given data is out of range of the limits of the boundaries*****

    Do i=g,g+1
        If ((x.gt.xdata(i,ndata(i))).or.(x.le.xdata(i,1))) then
            Print*, ' Data is out of range, we will extrapolate.'
        end if
    end do

70
end do
end if

c ************ For N(g)
If (x.gt.xdata(g,ndata(g))) then
  slope = (ydata(g,ndata(g))-ydata(g,ndata(g)-1))/(xdata(g,ndata(g))- xdata(g,ndata(g)-1))
  ydata_1 = ydata(g,ndata(g)) + slope*(x - xdata(g,ndata(g)))
goto 1070
end if

If (x.le.xdata(g,1)) then
  slope = (ydata(g,2)-ydata(g,1))/(xdata(g,2) - xdata(g,1))
  ydata_1 = ydata(g,1)-slope*(xdata(g,1) - x)
goto 1070
end if

jl=0
ju=ndata(g)+1
70  If ((ju-jl).gt.1) then
    jm=(ju+jl)/2
    If ((xdata(g,ndata(g)).ge.xdata(g,1)).eqv.(x.ge.xdata(g,jm)))then
      jl=jm
    else
      ju=jm
    endif
    goto 70
  endif
  If(x.eq.xdata(g,1))then
    j=ndata(g)-1
  else
    j=jl
  endif
  slope = (ydata(g,j+1) - ydata(g,j))/(xdata(g,j+1)-xdata(g,j))
  ydata_1 = ydata(g,j) + slope*(x - xdata(g,j))
1070  Print*
      Print*, 'The value of the function at the entered ', d1
      Print*,'for N_ ',g,' is:'
      Print*, d2,='',ydata_1
      Print*

c *********************** For N(g+1) ***********************
If (x.gt.xdata(g+1,ndata(g+1))) then
slope = (ydata(g+1,ndata(g+1))-ydata(g+1,ndata(g+1)-1))
& /(xdata(g+1,ndata(g+1))-xdata(g+1,ndata(g+1)-1))

ydata_2 = ydata(g+1,ndata(g+1)) + slope*(x - xdata(g+1,ndata(g+1)))
goto 1080
end if

If (x.le.xdata(g+1,1)) then
  slope = (ydata(g+1,2)-ydata(g+1,1))/(xdata(g+1,2)-
& xdata(g+1,1))
ydata_2 = ydata(g+1,1) - slope*(xdata(g+1,1) - x)
goto 1080
end if

jd=0
jl=0
ju=ndata(g+1)+1
71  If ((ju-jl).gt.1.) then
  jm=(ju+jl)/2
  If ((xdata(g+1,ndata(g+1)).ge.xdata(g+1,1)).eqv.
& (x.ge.xdata(g+1,jm)))then
    jl=jm
  else
    ju=jm
  endif
  goto 71
endif
If(x.eq.xdata(g+1,1))then
  j=ndata(g+1)-1
else
  j=jl
endif

slope=(ydata(g+1,j+1)-ydata(g+1,j))/(xdata(g+1,j+1)-xdata(g+1,j))
ydata_2 = ydata(g+1,j) + slope*(x - xdata(g+1,j))
1080  Print*
  Print*, 'The value of the function at the entered ', d1
  Print*, 'for N_\text{1}', g+1, ' is:
  Print*, d2,'=', ydata_2

Print*
c    *****Now we find the y value by interpolating the zdata values******

  yfinal=ydata_1+(z_data-zdata(g))*((ydata_2-ydata_1)
& /(zdata(g+1)-zdata(g)))

Print*, 'The value of the function for the entered ', d1
Print*, d2, '=', yfinal

Print*

end
Appendix B

c******************************************************************
c******************************************************************
c******************************************************************
c****                                                        ******
c****                                                        ******
c****   One-dimensional transient compressible flow equations******
c****   in a variable cross sectional duct with friction ******
c****                       and heat transfer                ******
c****                                                        ******
c******************************************************************
c******************************************************************
c******************************************************************
c PROGRAM MAIN

INCLUDE "fcblock.for"

OPEN (unit=10, file='output.dat', status='unknown')
OPEN (unit=11, file='restart.dat', status='unknown')
OPEN (unit=12, file='radius.dat', status='unknown')
REWIND 10

c------ Constants

small = 1.0E-15
pi = 4.0*atan(1.0)
gasconst = 287.
Cp = 1007.
Cp_const = Cp/(Cp - gasconst)
spec_heat_ratio = 1.41
friction_coeff = 1.e-15
heat_trans_coeff = 1.e-15   ! Change h.

idebug = 0
i_restart = 0
print*, 'Enter i_restart (=1 use restart file, =0 do not)'
read*, i_restart
print*, 'Enter idegug (=1 print field vairable, =0 do not)'
read*, idebug

c------ reference quantities
vis_ref = 1.e-05
thermal_k = 1.0
pres_ref = 1.01325E+05
temp_ref = 300.
T_ambient = 300.
rho_ref = pres_ref/(gasconst*temp_ref)
pres_fac_old = 0.005
temp_fac = 1.0
period = 1.0

c
----- Geometry

c
r_middle = 0.10
eta_inlet = 1.250
eta_outlet = 1.10
r_inlet = eta_inlet*r_middle
r_outlet = eta_outlet*r_middle
duct_length = 1.0

c
IF(i_restart.ne. 1) then

c
------ Grid parameters

c
print*, 'input no. of cells, ncells =? '
read*, ncells
nx = ncells + 2
nxm1 = nx - 1
nxm2 = nx - 2
dx_cons = duct_length/float(ncells)
x(1) = -dx_cons/2.

dx(1) = dx_cons
do i = 2, nx
   dx(i) = dx_cons
   x(i) = x(i-1) + dx_cons
end do

c
------ Calculate cell size for the u-velocity

do i = 2, nx
   dxu(i) = x(i) - x(i-1)
end do
dxu(1) = dxu(2)

c
------ Calculate duct geometry

c
c
const_a = 0.5*(eta_inlet - 1.0)
const_b = 0.5*(eta_outlet - 1.0)

c
do i = 1, nx
c   eps = 2.0*x(i)/duct_length - 1.0
eta = 1. + (const_a*eps*(eps - 1.0) + const_b*eps*(eps + 1.0))
& *1.e-2 ! Remove comment for ~constant cross section
radius(i) = eta*r_middle
area(i) = pi*radius(i)*radius(i)
write(12,*) radius(i)
end do

-------------Comparison with Ref. Analytical Solution---------------------
Do i = 1, nx

Area(i)=sqrt(0.51+(x(i)-0.7)**2)*1.e-3
radius(i) = sqrt(area(i)/pi)
End do

print*, ' radius: i=1, nx=',nx
print*, (area(i), i= 1,nx)
pause

c------ Initial conditions
c
do i = 1, nx
  gamma(i) = small
  rho(i) = rho_ref*(1.0-0.023*x(i))
  pres(i) = pres_ref
  temp(i) = 300/sqrt(0.51+(x(i)-0.7)**2)
  flowrate(i) = small
  uvel(i) =sqrt(1.41*287*temp_ref)*(0.05+0.11*x(i))
  uvel(i) = 30.0
  enthalpy(i) = Cp*temp(i)
  phi(i) = small
end do

print*, 'input dt, ntmax, and max_iter'
read*, dt, ntmax, max_iter

print*, 'input iprint=?'
read*, iprint

c print*, 'Under relax. factors urf_rho, urf_pres, urf_F'
c read*, urf_rho, urf_pres, urf_F

urf_rho = 1.
urf_pres = 1.
urf_F = 1.
c
c------ Time loop
c
time = 0.
press_fac = press_fac_old

ELSE

CALL READ_IN

c-------- Change some parameters if desired
print*, 'old dt=',dt, ' ntmax=',ntmax, ' max_iter=',max_iter
print*, 'input new dt, ntmax, max_iter:'
read*, dt, ntmax,max_iter
print*, 'input iprint=?'  
read*, iprint

c
print*, 'old pres_fac=', pres_fac
print*, 'Input inlet and outlet pressures:
read*, pin, pout
pres_fac = (pin-pout)/pin

c
print*, 'input new pres_fac'
read*, pres_fac

c
print*, 'flowrate at time=',time
print*, (flowrate(i), i=1,nx)

print*, 'uvel at time=',time
print*, (uvel(i), i=1,nx)

print*, 'pres at time=',time
print*, (pres(i), i=1,nx)

END IF

do itime = 1, ntmax

c
---------- save values at the old time level

do i = 1, nx
  rho_old(i) = rho(i)
  pres_old(i) = pres(i)
  temp_old(i) = temp(i)
  uvel_old(i) = uvel(i)
  flowrate_old(i) = flowrate(i)
  enthalpy_old(i) = enthalpy(i)
  phi_old(i) = phi(i)
end do
DO iter = 1, max_iter

c--------- set boundary conditions

time = time + dt

pres(1) = pres_ref
cc pres(nx) = pres_ref*(1.0 + pres_fac*sin(2.*pi*time/period) ) ! Sinusoidal Pressure Change

pres(nx) = (1.0 - pres_fac)*pres_ref

ccc temp(1) = temp_ref
rho(1) = pres(1)/gasconst/temp(1)
enthalpy(1) = Cp*temp(1)

ccc temp(nx) = temp_fac*temp_ref
rho(nx) = pres(nx)/gasconst/temp(nx)
enthalpy(nx) = Cp*temp(nx)

c------------------ extrapolate from inner nodes

area_wb = 0.5*(area(1)+area(2))
rho_wb = 0.5*(rho(1)+rho(2))
slope_wb = (flowrate(3) - flowrate(2))/dx(2)
flow_wwb = flowrate(2) - slope_wb*dxu(2)/2.
uvel_wwb = flow_wwb/rho(1)/area(1)
flow_ewb = 0.5*(flowrate(3)+flowrate(2))
uvel_ewb = flow_ewb/rho(2)/area(2)
dconvdx = (uvel_ewb*flow_ewb - uvel_wwb*flow_wwb)/dxu(2)
rhs_F = - dconvdx - area_wb*(pres(2)-pres(1))/dxu(2)
flowrate(2) = flowrate_old(2) + dt*rhs_F
uvel(2) = flowrate(2)/area_wb/rho_wb

c
c---------------- Predictor step

if(idebug .eq. 1) then
    print*, 'Flowrate before calling CALC_UF'
    print*, (flowrate(i), i=2,nx)
end if

CALL CALC_UF

c
if(idebug .eq. 1) then
    print*, 'Flowrate after calling CALC_UF'
    print*, (flowrate(i), i=2,nx)
end if

---------- Calculate pressure from Poisson equation
CALL CALC_PRES

if(debug .eq. 1) then
   print*, 'After calling CALC_PRES: pres(i=1,nx)'
   print*, (pres(i), i=1,nx)
end if

---------- Some ideas about the implicit handling of d(rho)/dt
   (1) try a combination of continuity and isentropic relation
   (2) use ideal gas law to update rho and iterate
   (3) Use semi implicitness by including the diagonal elements
when calculating rho from continuity equation

END DO

c
c------ End of inner iteration

c
---------- Correction step: Correct flow field

do i = 2, nx

   areau = 0.5*(area(i)+area(i-1))
   pres_grad = (pres(i)-pres(i-1))/dxu(i)
   flowrate(i) = flowrate(i) - urf_f*dt*areau*pres_grad
   rhou = 0.5*(rho(i)+rho(i-1))
   uvel(i) = flowrate(i)/rhou/areau
end do

c----------Outlet: extrapolate from inner nodes
   area_eb = 0.5*(area(nx)+area(nxm1))
   rho_eb = 0.5*(rho(nx)+rho(nxm1))

   slope_eb = (flowrate(nx) - flowrate(nxm1))/dxu(nx)
   flow_eeb = flowrate(nx) + slope_eb*dxu(nx)/2.
   uvel_eeb = flow_web/rho(nx)/area(nx)
   flow_web = 0.5*(flowrate(nx)+flowrate(nxm2))
   uvel_web = flow_web/rho(nxm1)/area(nxm1)
   dconvdx = (uvel_eeb*flow_eeb - uvel_web*flow_web)/dxu(nx)
   rhs_F = - dconvdx - area_eb*(pres(nx)-pres(nxm1))/dxu(nx)
   flowrate(nx) = flowrate_old(nx) + dt*rhs_F
uvel(nx) = flowrate(nx)/area_eb/rho_eb

c

c
if(iddebug .eq. 1) then
    print*, 'Flowrate after correcting for pressure gradient'
    print*, (flowrate(i), i=2,nx)
end if

c
-------------- Set Boundary conditions after correction

c of the velocity field

c
----------- Solve energy equation
cc CALL CALC_ENT

c
CALL CALC_PHI(3)

c
----------- Update boundary conditions after solver
cc-------------- West Boundary
cc-------- Dirichlet condition
cc
enthalpy(1) = Cp*temp(1)
ccc temp(1) = enthalpy(1)/Cp
cc
ccc if (uvel_old(2) .lt. 0.) then
ccc  enthalpy(1) = enthalpy(2)
ccc  temp(1) = temp(2)
ccc end if

c-------------- East Boundary
cc-------- Dirichlet condition
cc
enthalpy(nx) = Cp*temp(nx)
ccc temp(nx) = enthalpy(nx)/Cp
cc
ccc if (uvel_old(nx) .gt. 0.) then
ccc  enthalpy(nx) = enthalpy(nxm1)
ccc  temp(nx) = temp(nxm1)
ccc end if
c---------------- Update density from Ideal-Gas law

do i = 1, nx
   rho(i) = pres(i)/(gasconst*temp(i))
end do

c
if (idebug .eq. 1) then

   print*, ''
   print*, '----- U-field: uvel i=2,nx'
   print*, (uvel(i), i=2,nx)
   pause

c
   print*, 'itime=', itime, '  iter=', iter
   print*, '----- Density field'
   print*, (rho(i), i=1,nx)
   pause

   print*, ''
   print*, '------ pressure field'
   print*, (pres(i), i=1,nx)
   pause

   print*, ''
   print*, '----- Temperature: i=1,nx'
   print*, (temp(i), i=1,nx)
   pause

c
end if

c
   print*, ''
   print*, '------ pressure field'
   print*, (pres(i), i=1,nx)
   pause

c
end if

c
------------------- Print results to a file

c
if(mod(itime,iprint) .eq. 0) then
   write(10,*) 'time =', time
   write(10,*) '  x      flowrate     velocity        P/P_inlet',
   & '           rho     temperature    enthalpy(J/m)
   rho_wb = 0.5*(rho(1) + rho(2))
   pres_wb = 0.5*(pres(1) + pres(2))
   tot_inlet_pres = pres_wb + rho_wb*uvel(2)*uvel(2)/2.

81
do i = 1, nx
    delp = pres(i) - pres_ref
    pres_ratio = pres(i)/tot_inlet_pres
    write(10,1010) x(i),flowrate(i),uvel(i),pres_ratio,rho(i),
    &      temp(i),area(i)! area/enthalpy(i)
end do

print*, 'itime=', itime, '  iter=', iter, ' time=', time
print*, '----- Density field'
print*, (rho(i), i=1,nx)

print*, '----- Temperature field'
print*, (temp(i), i=1,nx)

print*, ' *******   ++++   ******* '
print*, 'Flowrate after correcting for pressure gradient'
print*, (flowrate(i), i=2,nx)

end if

------ end of time loop

end do

------- write out restart data file

CALL WRITE_OUT

CLOSE (11)

1010 format(1x,7(1pE12.4,2x))

stop
end

+++

+++

SUBROUTINE CALC_UF

SUBROUTINE CALC_UF
INCLUDE "fcblock.for"

--- initialize the generic field variable
    do i = 1, nx
        phi(i) = 0.
    end do

--- Use McCormack method for the predictor step; backward diff.
    do i = 3, nxm1
        Conv_old = uvel_old(i)*flowrate_old(i)
        & - uvel_old(i-1)*flowrate_old(i-1)
        Conv_old = Conv_old/dx(i-1)
        rhou = 0.5*(rho(i) + rho(i-1))
        areau = 0.5*(area(i) + area(i-1))
        radiusu = sqrt(areau/pi)
        perimeter = 2.*pi*radiusu
        friction_force_old = friction_coeff*perimeter
        & *rhou*abs(uvel_old(i))*uvel_old(i)
        phi(i) = flowrate_old(i)
        & - dt*(Conv_old + friction_force_old)
    end do

--- inlet and outlet
--- Note that phi(2) and phi(nx) are not used again
--- phi(2) = flowrate(2)
--- slope = (phi(nxm1)-phi(nxm2))/dxu(nxm1)
--- phi(nx) = phi(nxm1) + dxu(nx)*slope

cc print*, 'After predictor step: gen_var (i=2,nx)'
c
cc print*, (phi(i), i=2,nx)
c
cc pause

c
--- Corrector step; forward difference
    do i = 3, nxm1
        Conv_old = uvel_old(i)*flowrate_old(i)
        & - uvel_old(i-1)*flowrate_old(i-1)
        Conv_old = Conv_old/dx(i-1)
        rhou = 0.5*(rho(i) + rho(i-1))
        areau = 0.5*(area(i) + area(i-1))
        radiusu = sqrt(areau/pi)
        perimeter = 2.*pi*radiusu
        friction_force_old = friction_coeff*perimeter
        & *rhou*abs(uvel_old(i))*uvel_old(i)
rhow = 0.5*(rho(i)+rho(i-1))
areaw = 0.5*(area(i)+area(i-1))
rhoe = 0.5*(rho(i)+rho(i+1))
areae = 0.5*(area(i)+area(i+1))

\[ uvel_i = \frac{\phi(i)}{rhow/areaw} \]
\[ uvel_ip1 = \frac{\phi(i+1)}{rhoe/areae} \]

\[ \text{Conv}_{\text{new}} = uvel_ip1*\phi(i+1) - uvel_i*\phi(i) \]
\[ \text{Conv}_{\text{new}} = \frac{\text{Conv}_{\text{new}}}{dx(i)} \]

\[ \text{friction}_{\text{force}}_{\text{new}} = \text{friction}_{\text{coeff}}*\text{perimeter} \]
& \[ *rhou*abs(uvel_i)*uvel_i \]

\[ \text{rhs}_{\text{old}} = \text{Conv}_{\text{old}} + \text{friction}_{\text{force}}_{\text{old}} \]
\[ \text{rhs}_{\text{new}} = \text{Conv}_{\text{new}} + \text{friction}_{\text{force}}_{\text{new}} \]
\[ \text{rhs}_{\text{avr}} = 0.5*(\text{rhs}_{\text{old}} + \text{rhs}_{\text{new}}) \]

\[ \text{flowrate}(i) = \text{flowrate}_{old}(i) - dt*\text{rhs}_{\text{avr}} \]
\[ uvel(i) = \frac{\text{flowrate}(i)}{rhow/areaw} \]

c end do
c

c------------- Extrapolate at the outlet boundary

c slope = (flowrate(nxm1)-flowrate(nxm2))/dxu(nxm1)
flowrate(nx) = flowrate(nxm1) + dxu(nx)*slope
c

\[ \text{rho}_{\text{eb}} = 0.5*(\text{rho}(nx)+\text{rho}(nxm1)) \]
\[ \text{area}_{\text{eb}} = 0.5*(\text{area}(nx)+\text{area}(nxm1)) \]
\[ uvel(nx) = \frac{\text{flowrate}(nx)}{\text{rho}_{\text{eb}}/\text{area}_{\text{eb}}} \]
c

RETURN
c

c END

c SUBROUTINE CALC_PRES
INCLUDE "fcblock.for"

c--------- Calculate coefficients
c
do i=2,nxm1
   areae = 0.5*(area(i+1)+area(i))
   areaw = 0.5*(area(i-1)+area(i))
   AE(i) = areae/dxu(i+1)
   AW(i) = areaw/dxu(i)
   AP(i) = AW(i) + AE(i)
   del_flow = flowrate(i+1) - flowrate(i)
cc
drhodt = small
   drhodt = (rho(i) - rho_old(i))/dt
   volume = area(i)*dx(i)
   QP(i) = -urf_pres*(del_flow + volume*drhodt)/dt
end do

c-------- Modify coefficients to impose boundary conditions
c
c------------- Inlet: Pressure is specified
   i = 2
   QP(i) = QP(i) + AW(i)*pres(i-1)
   AW(i) = 0.

c------------- Outlet: pressure is specified
   i = nxm1
   QP(i) = QP(i) + AE(i)*pres(i+1)
   AE(i) = 0.

c
+-+----------------------------------------------------------+
| c|++---------------------------------------------------------------++
+-+----------------------------------------------------------+

c----- use TDMA solver for the linear system of equations

c-------- Substitute pres into phi
   do i = 1,nx
      phi(i) = pres(i)
      phi_old(i) = pres_old(i)
   end do

   CALL SOLVE

c
-------- Substitute phi back into actual variable
do i = 1, nx
    pres(i) = phi(i)
    phi(i) = 0.
    phi_old(i) = 0.
end do

RETURN

END

SUBROUTINE CALC_PHI(id_var)

INCLUDE "fcblock.for"

do i = 1, nx
    if(id_var .eq. 3) then
        phi_old(i) = enthalpy_old(i)
        phi(i) = enthalpy(i)
    end if
end do

CALL COEFF_ENT

CALL BCOND_ENT

cc print*,'          i    AW    AE     AP      QP:'
cc do i = 2, nxm1
cc     print*, i, AW(i),AE(i),Ap(i),QP(i)
cc     pause
cc end do

CALL SOLVE

do i = 1, nx
    if(id_var .eq. 3) then
enthalpy(i) = phi(i)
temp(i) = enthalpy(i)/Cp
phi(i) = 0.
phi_old(i) = 0.
end if
end do

c
RETURN
c
END
c
+++
c********************************************************************
c----------------- ************************** -----------------------
cc
c********************************************************************

SUBROUTINE COEFF

INCLUDE "fcblock.for"

A_Peck(Peck) = amax1(0., 1. - 0.5*abs(Peck))

---- Generalized scheme

------- Calculate coefficient matrix for a particular scheme

ENTRY COEFF_ENT

do i = 2, nx-1

areaw = 0.5*(area(i)+area(i-1))
    areae = 0.5*(area(i)+area(i+1))
    volume = area(i)*dx(i)

------- Staggered grid is used, velocity lags to left half grid size
    uwest = uvel_old(i)
    ueast = uvel_old(i+1)
    uvelc = 0.5*(uvel(i+1) + uvel(i))

rhow = 0.5*(rho_old(i)+rho_old(i-1))
rhoe = 0.5*(rho_old(i)+rho_old(i+1))
gammaw = 0.5*(gamma(i)+gamma(i-1))

\[ \gamma_e = 0.5 \times (\gamma(i) + \gamma(i+1)) \]
\[ \text{acent}_{\text{old}} = \text{volume} \times \rho_{\text{old}}(i) / \text{dt} \]
\[ \text{conv}_{\text{w}} = \rho_{\text{how}} \times \text{uwest} \times \text{area}_w \]
\[ \text{conv}_{\text{e}} = \rho_{\text{hoe}} \times \text{ueast} \times \text{area}_e \]
\[ \text{diff}_{\text{w}} = \text{area}_w \times \gamma_{\text{maw}} / dx_u(i) \]
\[ \text{diff}_{\text{e}} = \text{area}_e \times \gamma_{\text{mwe}} / dx_u(i+1) \]
\[ \text{Peck}_{\text{e}} = \text{conv}_{\text{e}} / (\text{diff}_{\text{e}} + \text{small}) \]
\[ \text{Peck}_{\text{w}} = \text{conv}_{\text{w}} / (\text{diff}_{\text{w}} + \text{small}) \]
\[ \text{c} \]
\[ \text{c------------ Use hybrid scheme} \]
\[ \text{AE}(i) = \text{diff}_e \times A_{\text{Peck}}(\text{Peck}_{\text{e}}) + \text{amax1}(-\text{conv}_e, 0.0) \]
\[ \text{AW}(i) = \text{diff}_w \times A_{\text{Peck}}(\text{Peck}_{\text{w}}) + \text{amax1}(\text{conv}_w, 0.0) \]
\[ \text{c} \]
\[ \text{c------ Make sure that center coefficient is the sum of the neighboring coefficients.} \]
\[ \text{c} \] when there is no source term
\[ \text{AP}(i) = \text{acent}_{\text{old}} + \text{AW}(i) + \text{AE}(i) \]
\[ \text{c} \]
\[ \text{QP}(i) = \text{acent}_{\text{old}} \times \phi_{\text{old}}(i) \]
\[ \text{c} \]
\[ \text{c----- Add source term} \]
\[ \text{perimeter} = 2 \times \pi \times \text{radius}(i) \]
\[ \text{Sp} = - \text{Perimeter} \times \text{heat_trans_coeff} / \text{Cp} / \text{area}(i) \]
\[ \text{Sc} = \text{Perimeter} \times \text{heat_trans_coeff} \times T_{\text{ambient}} / \text{area}(i) \]
\[ \text{c------------ Calculate pressure work term} \]
\[ \text{dp}dt = (\text{pres}(i) - \text{pres}_{\text{old}}(i)) / \text{dt} \]
\[ \text{dpdx}_p = (\text{pres}(i) - \text{pres}(i-1)) / dx_u(i) \]
\[ \text{if}(\text{uvelc} \lt \text{0}) \]
\[ \text{dpdx}_p = (\text{pres}(i+1) - \text{pres}(i)) / dx_u(i+1) \]
\[ \text{conv}_p = \text{uvel}(i) \times \text{dpdx}_p \]
\[ \text{W}_p_{\text{res}} = \text{dp}dt \times \text{conv}_p \]
\[ \text{AP}(i) = \text{AP}(i) - \text{Sp} \times \text{volume} \]
\[ \text{QP}(i) = \text{QP}(i) + (\text{Sc} + \text{W}_p_{\text{res}}) \times \text{volume} \]
\[ \text{end do} \]
\[ \text{c} \]
\[ \text{RETURN} \]
\[ \text{END} \]
\[ \text{c} \]
\[ \text{c************************************************************************} \]
SUBROUTINE SOLVE

INCLUDE "fcblock.for"

dimension aa(nxmax), bb(nxmax), cc(nxmax), rr(nxmax), phi_tdma(nxmax)

ifir = 2
ilas = nx-1

!---- For tdm-algorithm shift places of coefficients

   do i = ifir, ilas
      ii = i-ifir + 1
      aa(ii) = -AW(i)
      bb(ii) = AP(i)
      cc(ii) = -AE(i)
      rr(ii) = QP(i)
   end do

   aa(1) = 0.
   cc(nx-1) = 0.
   ntdma = ilas-ifir+1

! print*, 'Calcu, Coefficients for t DMA:'
! print*, 'aa:',(aa(i), i=1,ntdma)
! print*, 'bb:',(bb(i), i=1,ntdma)
! print*, 'cc:',(cc(i), i=1,ntdma)
! print*, 'rr:',(rr(i), i=1,ntdma)
! pause

CALL TDMA(aa, bb, cc, rr, phi_tdma, ntdma)

!------- shift phi_tdma values to actual locations

   do i = 1, ntdma
      iphi = i + ifir -1
      phi(iphi) = phi_tdma(i)
   end do

RETURN

C
END

C

C******************************************************************************
SUBROUTINE BCOND(id_var)

INCLUDE "fcblock.for"

ENTRY BCOND_ENT

-------------- West Boundary

--------- Dirichlet condition

phi(1) = temp_fac*Cp*temp_ref
QP(2) = QP(2) + AW(2)*phi(1)
AW(2) = 0.

--------- Apply neuman condition if uvel_wb is less than zero

if (uvel_old(2) .lt. 0.) then
    phi(1) = phi(2)
    AP(2) = AP(2) - AW(2)
    AW(2) = 0.
end if

-------------- East Boundary

--------- Dirichlet condition

phi(nx) = Cp*temp_ref
QP(nxm1) = QP(nxm1) + AE(nxm1)*phi(nx)
AE(nx) = 0.

--------- Apply neuman condition if uvel_wb is gt. than zero

if (uvel_old(nx) .gt. 0.) then
    phi(nx) = phi(nxm1)
    AP(nxm1) = AP(nxm1) - AE(nxm1)
    AE(nxm1) = 0.
end if
SUBROUTINE TDMA(a,b,c,r,phi,n)
INTEGER n, nmax
REAL a(n),b(n),c(n),r(n),phi(n)
PARAMETER (nmax=201)

INTEGER j
REAL bet, gam(nmax)

if(b(1) .eq. 0.) pause 'TDMA: b(1)=0, rewrite equations'
bet = b(1)
phi(1) = r(1)/bet

c---- Decomposition and forward substitution
  do j=2,n
    gam(j) = c(j-1)/bet
    bet = b(j) - a(j)*gam(j)
    if(bet .eq. 0.) pause 'TDMA failed'
    phi(j) = (r(j) - a(j)*phi(j-1))/bet
  end do

c---- Back substitution
  do j = n-1, 1, -1
    phi(j) = phi(j) - gam(j+1)*phi(j+1)
  end do

SUBROUTINE RESTART_OUT

INCLUDE "fcblock.for"

ENTRY WRITE_OUT

print*,'In ENTRY WRITE_OUT, pres_fac=',pres_fac,' time=',time

------ Constants
write(11,*),urh_rho,urh_pres,urh_F,
& pi,small,gasconst,Cp,Cp_const,spec_heat_ratio,
& friction_coeff,heat_trans_coeff,T_ambient,
& pres_ref,temp_ref,rho_ref,pres_fac,temp_fac,time,
& duct_length,dt, nx,nxm1,nxm2,max_iter

------ Arrays

do i = 1,nx

write(11,*),flowrate(i),pres(i),rho(i),temp(i),
& uvel(i),enthalpy(i),
& flowrate_old(i),pres_old(i),rho_old(i),
& temp_old(i),uvel_old(i),enthalpy_old(i),
& phi_old(i),phi(i),
& x(i),area(i),radius(i),
& dx(i),dxu(i),
& viscosity(i),gamma(i)

end do

REWIND (11)
ENTRY READ_IN

Constants
read(11,*) urf_rho, urf_pres, urf_F,
& pi, small, gasconst, Cp, Cp_const, spec_heat_ratio,
& friction_coeff, heat_trans_coeff, T_ambient,
& pres_ref, temp_ref, rho_ref, pres_fac, temp_fac, time,
& duct_length, dt, nx, nxm1, nxm2, max_iter

print*, 'In ENTRY READ_IN, pres_fac=', pres_fac, ' time=', time

Arrays

do i = 1, nx

read(11,*) flowrate(i), pres(i), rho(i), temp(i),
& uvel(i), enthalpy(i),
& flowrate_old(i), pres_old(i), rho_old(i),
& temp_old(i), uvel_old(i), enthalpy_old(i),
& phi_old(i), phi(i),
& x(i), area(i), radius(i),
& dx(i), dxu(i),
& viscosity(i), gamma(i)
end do

REWIND (11)

RETURN

END
Appendix C

c
************************************************************************
c
This program solves an ordinary differential equation of the form
dy/dx = f(x,y)
using the 2nd order Runge-Kutta method
c
************************************************************************
c
f(x)=sqrt(0.51+(x-0.7)**2)  ! f(x)=A(x)/Ao  
g(x)=1/sqrt(0.51+(x-0.7)**2)  ! g(x)=T(x)/To
OPEN (unit=10, file='output_m.dat', status='unknown')
c
OPEN (unit=11, file='output_derivative.dat', status='unknown')
c
c------- Initial conditions and the step size
c
deltax=0.01 ! Step size for central difference applied to R(x)=g(x)/f(x)
c  c=6.078 ! c=PoAo/uo: Ao= 3.e-3, uo=50, Po=1.013e5
c  cmdot=0.035283 ! mdot=Rho0.uo.Ao=Po.uo.Ao/(RTo)
c  cmdot = 0.058389  
   A0=1.e-3  
   T0=300  
   P0=1.013*1.e5
c  u0=30.0  
  u0 = 49.6450  
c  c=cmdot*u0/(A0*P0)  
gasconst=287.1
print*, 'Did you change the function?'
print*, 'If not change it now!'  
pause
print*, 'Input initial conditions, x(0), P/Po(0)=?'  
read*, x0, y0
x = x0
y = y0  
print*, 'Input step size, h=?'  
read*, h
print*, 'Input the maximum value of x, xlast=?'  
read*, xlast
   write(10,*) 'x     P/Po     Rho     Velocity & Temperature     Area'
c   write(11,*) 'x     Rprime'
c
do while ( x .lt. xlast)

The derivative of R(x) is computed using central differencing
x1 = x - deltax
   R1 = g(x1) / f(x1)

x2 = x + deltax
   R2 = g(x2) / f(x2)
   Rprime = (R2 - R1) / (2 * deltax)
   c write(11, *) x, Rprime

   R = g(x) / f(x)
   s1 = -(c * Rprime * y) / (y**2 * f(x) - c * R)
   ytemp = y + h * s1
   x = x + h
   s2 = -((c * Rprime * ytemp) / (ytemp**2 * f(x) - c * R))
   savr = 0.5 * (s1 + s2)
   y = y + h * savr
   c
   print*, 'x=', x, ', y=', y
   rho = y * P0 / gasconst / g(x) / T0
   u_vel = c * dot / (rho * f(x) * A0)
   Temp = g(x) * T0
   Area = f(x) * A0
   write(10, 1000) x, y, rho, u_vel, Temp, Area
1000 format(6(f10.5, 2x))
   c
   end do
   c
   stop
   end
Appendix D

c**************************************************************************
*           This program solves first order ODE's using Adams-Moulton formula       *
*c**************************************************************************

parameter( nxmax = 200)
common/vars/ x(nxmax),y(nxmax), dx(nxmax), f(nxmax),
 &yslope(nxmax)
common/varss/n(nxmax,nxmax)

OPEN (unit=10, file='output.dat', status='unknown')

n(x,y) = y - x + 1 ; =e^(x^2/2)
print*, 'Input ncells, duct length and maxiter =? '
read*, ncells, duct_length, itermax
nx = ncells + 2
nxm1 = nx - 1
nxm2 = nx - 2
dx_cons = duct_length/float(ncells)
x(1) = -dx_cons/2.

dx(1) = dx_cons
do i = 2, nx
   dx(i) = dx_cons
   x(i) = x(i-1) + dx_cons
end do

do i=1, nx
   y(i) = 1.0
end do

write(10,*) ' x                  y'
do iter=1, itermax

   do i=2, nxm1
      f(i-1) = y(i-1)-x(i-1)+1
      f(i) = y(i)-x(i)+1
      f(i+1) = y(i+1)-x(i+1)+1
      yslope(i) = (5./12.)*f(i+1)+(8./12.)*f(i)-(1./12.)*f(i-1)
y(i+1) = y(i) + dx_cons*yslope(i)
   end do
end do

do i=1, nx
    write(10,*), x(i), y(i)
    end do
end
Appendix E

The mass and energy balance equations are checked for both Fluent and the present program. Two points, an inlet and an exit are chosen and data is taken at two different time steps for both programs.

**Present Program**

<table>
<thead>
<tr>
<th>Inlet</th>
<th>Velocity (m/s)</th>
<th>Density (kg/m3)</th>
<th>Temperature(K)</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1</td>
<td>19.79</td>
<td>0.90543</td>
<td>389.95</td>
</tr>
<tr>
<td>t2</td>
<td>20.059</td>
<td>0.89595</td>
<td>394.08</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Outlet</th>
<th>Velocity (m/s)</th>
<th>Density (kg/m3)</th>
<th>Temperature(K)</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1</td>
<td>23.733</td>
<td>0.86966</td>
<td>405.13</td>
</tr>
<tr>
<td>t2</td>
<td>23.737</td>
<td>0.87165</td>
<td>404.22</td>
</tr>
</tbody>
</table>

Table App. E.1: Data from the present program

<table>
<thead>
<tr>
<th>Inlet</th>
<th>Velocity (m/s)</th>
<th>Density (kg/m3)</th>
<th>Temperature(K)</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1</td>
<td>25.6</td>
<td>0.611</td>
<td>577</td>
</tr>
<tr>
<td>t2</td>
<td>25.9</td>
<td>0.555</td>
<td>635</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Outlet</th>
<th>Velocity (m/s)</th>
<th>Density (kg/m3)</th>
<th>Temperature(K)</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1</td>
<td>27.6</td>
<td>0.78</td>
<td>452</td>
</tr>
<tr>
<td>t2</td>
<td>28.0</td>
<td>0.695</td>
<td>507</td>
</tr>
</tbody>
</table>

Table App. E.2: Data from Fluent

The following equations are used, namely the continuity and the energy equation:

\[
\frac{\partial}{\partial t}(\rho A) + \frac{\partial}{\partial x}(\rho Au) = 0
\]

\[
\frac{\partial}{\partial t}(\rho Ah) + \frac{\partial}{\partial x}(\rho Auh) = \frac{\partial}{\partial x}\left(\frac{Ah}{C_p} dx\right) - \frac{P h_{\text{conv}}(T - T_a)}{C_p A} + A \frac{DP}{Dt} + u \frac{4A}{R} \tau_w
\]

The continuity equation is satisfied for both programs, but when we insert the above values from the tables into the energy equation, we see that Fluent doesn’t satisfy the equation as accurately as the present program.
Present Program:

\[
\frac{(8*10^{-4})*(0.625-0.275)}{0.3} \cdot \left[ (0.92748*1007*380.35) - (0.92702*1007*380.52) \right] \\
- \left[ (0.90543*8*10^{-4})*19.79*1007*389.95) - (0.86966*8*10^{-4})*23.733*1007*405.13) \right] \\
+ \frac{(0.625-0.275)\pi(200)(384-450)(2*0.016)}{1007*8*10^{-4}} \\
- \frac{4*2\pi*0.016*0.015*0.5*0.9*20}{0.016} \cdot (0.625-0.275) \\
=> 0 \equiv 8
\]

Fluent:

\[
\frac{(8*10^{-4})*(0.364-0.192)}{0.03} \cdot \left[ (0.555*1007*635) - (0.611*1007*577) \right] \\
- \left[ (0.611*8*10^{-4})*25.6*1007*577) - (0.78*8*10^{-4})*27.6*1007*452) \right] \\
+ \frac{(0.364-0.192)\pi(200)(550-450)(2*0.016)}{1007*8*10^{-4}} \\
- \frac{4*(2\pi*0.016)*0.01*0.5*0.6*25}{0.016} \cdot (0.364-0.192) \\
=> 0 \equiv 70
\]

As can be clearly seen, the present program gives a much more accurate energy balance compared to the Fluent results. The significant level of imbalance in the energy equation indicates that there must be some problem with the Fluent simulations.
Appendix F :

When the conservation equations were checked for both Fluent and the present program, the results showed that there was a user mistake in the Fluent runs. The results were repeated after correcting the mistake (which was related to geometry). The below figures show the improved Fluent runs compared with the present program.

Fig. App. F.1: Comparison of Centerline Velocity at x = 0.5m for the Present Program and Fluent
Fig. App. F.2: Comparison of Pressure Ratio at x = 0.5m for the Present Program and Fluent

Fig. App. F.3: Comparison of Centerline Density at x = 0.5m for the Present Program and Fluent
Fig. App. F.4: Comparison of Mass Flow Rate at x = 0.5m for the Present Program and Fluent

Fig. App. F.5: Comparison of Temperature at t = 0.015s and t = 0.04s for Present Program and Fluent
### Appendix G

**ORIGINAL MAPS (Non-dimensionalized)**

<table>
<thead>
<tr>
<th>Flow Rate PR</th>
<th>Flow Rate PR</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00664 PR 2.2</td>
<td>0.013736 PR 1.22</td>
</tr>
<tr>
<td>0.009255 PR 2.6</td>
<td>0.041208 PR 1.15</td>
</tr>
<tr>
<td>0.01006 PR 2.55</td>
<td>0.054944 PR 1.11</td>
</tr>
<tr>
<td>0.01106 PR 2.5</td>
<td>zone</td>
</tr>
<tr>
<td>0.01167 PR 2.4</td>
<td>0.024725 PR 1.44</td>
</tr>
<tr>
<td>0.01179 PR 1.9</td>
<td>0.027472 PR 1.45</td>
</tr>
<tr>
<td>0.01183 PR 2.5</td>
<td>0.041208 PR 1.42</td>
</tr>
<tr>
<td>zone</td>
<td>0.054944 PR 1.38</td>
</tr>
<tr>
<td>0.012273 PR 3.45</td>
<td>0.06868 PR 1.3</td>
</tr>
<tr>
<td>0.012877 PR 3.4</td>
<td>0.074174 PR 1.27</td>
</tr>
<tr>
<td>0.014084 PR 3.3</td>
<td>zone</td>
</tr>
<tr>
<td>0.014285 PR 3.25</td>
<td>0.030219 PR 1.7</td>
</tr>
<tr>
<td>0.014486 PR 3.05</td>
<td>0.041208 PR 1.69</td>
</tr>
<tr>
<td>0.014607 PR 2.15</td>
<td>0.054944 PR 1.65</td>
</tr>
<tr>
<td>0.014647 PR 1.4</td>
<td>0.06868 PR 1.6</td>
</tr>
<tr>
<td>zone</td>
<td>0.082415 PR 1.52</td>
</tr>
<tr>
<td>0.015694 PR 4.55</td>
<td>0.096151 PR 1.38</td>
</tr>
<tr>
<td>0.016498 PR 4.5</td>
<td>zone</td>
</tr>
<tr>
<td>0.017303 PR 4.45</td>
<td>0.041208 PR 1.92</td>
</tr>
<tr>
<td>0.018108 PR 4.35</td>
<td>0.054944 PR 1.91</td>
</tr>
<tr>
<td>0.018229 PR 3.9</td>
<td>0.06868 PR 1.9</td>
</tr>
<tr>
<td>0.018269 PR 2.63</td>
<td>0.082415 PR 1.84</td>
</tr>
<tr>
<td>0.018309 PR 1.6</td>
<td>0.096151 PR 1.7</td>
</tr>
<tr>
<td>zone</td>
<td>0.101646 PR 1.6</td>
</tr>
<tr>
<td>0.019718 PR 5.7</td>
<td>zone</td>
</tr>
<tr>
<td>0.020321 PR 5.6</td>
<td>0.046702 PR 2.16</td>
</tr>
<tr>
<td>0.021126 PR 5.5</td>
<td>0.054944 PR 2.16</td>
</tr>
<tr>
<td>0.021528 PR 5.2</td>
<td>0.06868 PR 2.14</td>
</tr>
<tr>
<td>0.021609 PR 3.3</td>
<td>0.082415 PR 2.1</td>
</tr>
<tr>
<td>0.021689 PR 1.8</td>
<td>0.096151 PR 2.02</td>
</tr>
<tr>
<td>zone</td>
<td>0.109887 PR 1.84</td>
</tr>
<tr>
<td>0.115382 PR 1.72</td>
<td>zone</td>
</tr>
<tr>
<td>0.054944 PR 2.44</td>
<td>0.06868 PR 2.42</td>
</tr>
<tr>
<td>0.082415 PR 2.4</td>
<td>0.096151 PR 2.34</td>
</tr>
</tbody>
</table>
## Scaled Maps (Corrected Non-dimensionalized)

<table>
<thead>
<tr>
<th>Large Compressor</th>
<th>Small Compressor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flow Rate</td>
<td>PR</td>
</tr>
<tr>
<td>0.134452</td>
<td>1.466667</td>
</tr>
<tr>
<td>zone</td>
<td>0.121151</td>
</tr>
<tr>
<td>0.187418</td>
<td>1.733333</td>
</tr>
<tr>
<td>0.203715</td>
<td>1.7</td>
</tr>
<tr>
<td>0.224087</td>
<td>1.666667</td>
</tr>
<tr>
<td>0.236309</td>
<td>1.6</td>
</tr>
<tr>
<td>0.238754</td>
<td>1.266667</td>
</tr>
<tr>
<td>0.239569</td>
<td>0.8</td>
</tr>
<tr>
<td>zone</td>
<td>0.242301</td>
</tr>
<tr>
<td>0.248532</td>
<td>2.3</td>
</tr>
<tr>
<td>0.260755</td>
<td>2.266667</td>
</tr>
<tr>
<td>0.285201</td>
<td>2.2</td>
</tr>
<tr>
<td>0.289275</td>
<td>2.166667</td>
</tr>
<tr>
<td>0.29335</td>
<td>2.033333</td>
</tr>
<tr>
<td>0.295794</td>
<td>1.433333</td>
</tr>
<tr>
<td>0.296609</td>
<td>0.933333</td>
</tr>
<tr>
<td>zone</td>
<td>0.363452</td>
</tr>
<tr>
<td>0.317795</td>
<td>3.033333</td>
</tr>
<tr>
<td>0.334093</td>
<td>3</td>
</tr>
<tr>
<td>0.35039</td>
<td>2.966667</td>
</tr>
<tr>
<td>0.366687</td>
<td>2.9</td>
</tr>
<tr>
<td>0.369132</td>
<td>2.6</td>
</tr>
<tr>
<td>0.369946</td>
<td>1.753333</td>
</tr>
<tr>
<td>0.370761</td>
<td>1.066667</td>
</tr>
<tr>
<td>zone</td>
<td>0.448257</td>
</tr>
<tr>
<td>0.399281</td>
<td>3.8</td>
</tr>
<tr>
<td>0.411504</td>
<td>3.733333</td>
</tr>
<tr>
<td>0.427802</td>
<td>3.666667</td>
</tr>
<tr>
<td>0.43595</td>
<td>3.466667</td>
</tr>
<tr>
<td>0.43758</td>
<td>2.2</td>
</tr>
<tr>
<td>0.43921</td>
<td>1.2</td>
</tr>
<tr>
<td>zone</td>
<td>0.484603</td>
</tr>
<tr>
<td>zone</td>
<td>0.508833</td>
</tr>
<tr>
<td>zone</td>
<td>0.242301</td>
</tr>
<tr>
<td>zone</td>
<td>0.302877</td>
</tr>
<tr>
<td>zone</td>
<td>0.363452</td>
</tr>
<tr>
<td>zone</td>
<td>0.424027</td>
</tr>
<tr>
<td>zone</td>
<td>0.484603</td>
</tr>
</tbody>
</table>
**ORIGINAL MAPS (Non-dimensionalized)**

<table>
<thead>
<tr>
<th>PR</th>
<th>phi</th>
<th>PR</th>
<th>phi</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.430057</td>
<td>9.91E-05</td>
<td>1.2</td>
<td>3.69E-05</td>
</tr>
<tr>
<td>1.560062</td>
<td>0.000111</td>
<td>1.3</td>
<td>4.10E-05</td>
</tr>
<tr>
<td>1.690068</td>
<td>0.000123</td>
<td>1.4</td>
<td>4.57E-05</td>
</tr>
<tr>
<td>1.75507</td>
<td>0.000127</td>
<td>1.5</td>
<td>4.74E-05</td>
</tr>
</tbody>
</table>

| zone                | 1.75507 | 0.000124 | 1.5 | 4.64E-05 |
| zone                | 1.885075 | 0.000136 | 1.6 | 5.09E-05 |
| zone                | 2.015081 | 0.000141 | 1.7 | 5.26E-05 |

| zone                | 1.885075 | 0.00013 | 1.6 | 4.88E-05 |
| zone                | 2.015081 | 0.000138 | 1.7 | 5.15E-05 |
| zone                | 2.145086 | 0.000142 | 1.8 | 5.26E-05 |
| zone                | 2.275091 | 0.000145 | 1.9 | 5.43E-05 |

| zone                | 2.145086 | 0.000138 | 1.8 | 5.12E-05 |
| zone                | 2.275091 | 0.000142 | 1.9 | 5.29E-05 |
| zone                | 2.340094 | 0.000145 | 2   | 5.39E-05 |
| zone                | 2.470099 | 0.000146 | 2.1 | 5.46E-05 |
| zone                | 2.600104 | 0.000148 | 2.2 | 5.53E-05 |
| zone                | 2.730109 | 0.000149 | 2.3 | 5.60E-05 |

| zone                | 2.340094 | 0.000139 | 2   | 5.22E-05 |
| zone                | 2.470099 | 0.000142 | 2.1 | 5.33E-05 |
| zone                | 2.600104 | 0.000145 | 2.2 | 5.43E-05 |
| zone                | 2.730109 | 0.000146 | 2.3 | 5.50E-05 |
| zone                | 2.860114 | 0.000148 | 2.4 | 5.53E-05 |

| zone                | 2.730109 | 0.000142 | 2.3 | 5.33E-05 |
| zone                | 2.860114 | 0.000143 | 2.4 | 5.39E-05 |
| zone                | 2.99012  | 0.000145 | 2.5 | 5.45E-05 |
| zone                | 3.055122 | 0.000146 | 2.6 | 5.48E-05 |
| zone                | 3.185127 | 0.000146 | 2.7 | 5.48E-05 |
**SCALED MAPS (Corrected Non-dimensionalized)**

<table>
<thead>
<tr>
<th>PRcorr</th>
<th>phicorr</th>
<th>PRcorr</th>
<th>phicorr</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2</td>
<td>0.00067</td>
<td>1.908</td>
<td>3.69E-05</td>
</tr>
<tr>
<td>2.4</td>
<td>0.00075</td>
<td>2.067</td>
<td>4.10E-05</td>
</tr>
<tr>
<td>2.6</td>
<td>0.00083</td>
<td>2.226</td>
<td>4.57E-05</td>
</tr>
<tr>
<td>2.7</td>
<td>0.00086</td>
<td>2.385</td>
<td>4.74E-05</td>
</tr>
<tr>
<td>zone</td>
<td></td>
<td>2.385</td>
<td>4.64E-05</td>
</tr>
<tr>
<td>2.7</td>
<td>0.00084</td>
<td>2.385</td>
<td>4.64E-05</td>
</tr>
<tr>
<td>2.9</td>
<td>0.00092</td>
<td>2.544</td>
<td>5.09E-05</td>
</tr>
<tr>
<td>3.1</td>
<td>0.00095</td>
<td>2.703</td>
<td>5.26E-05</td>
</tr>
<tr>
<td>zone</td>
<td></td>
<td>2.544</td>
<td>4.88E-05</td>
</tr>
<tr>
<td>2.9</td>
<td>0.00088</td>
<td>2.544</td>
<td>4.88E-05</td>
</tr>
<tr>
<td>3.1</td>
<td>0.00093</td>
<td>2.703</td>
<td>5.15E-05</td>
</tr>
<tr>
<td>3.3</td>
<td>0.00096</td>
<td>2.862</td>
<td>5.26E-05</td>
</tr>
<tr>
<td>3.5</td>
<td>0.00098</td>
<td>3.021</td>
<td>5.43E-05</td>
</tr>
<tr>
<td>zone</td>
<td></td>
<td>2.862</td>
<td>5.12E-05</td>
</tr>
<tr>
<td>3.3</td>
<td>0.00093</td>
<td>3.021</td>
<td>5.29E-05</td>
</tr>
<tr>
<td>3.5</td>
<td>0.00096</td>
<td>3.18</td>
<td>5.39E-05</td>
</tr>
<tr>
<td>3.6</td>
<td>0.00098</td>
<td>3.339</td>
<td>5.46E-05</td>
</tr>
<tr>
<td>3.8</td>
<td>0.00099</td>
<td>3.498</td>
<td>5.53E-05</td>
</tr>
<tr>
<td>4</td>
<td>0.001</td>
<td>3.657</td>
<td>5.60E-05</td>
</tr>
<tr>
<td>zone</td>
<td></td>
<td>3.657</td>
<td>5.60E-05</td>
</tr>
<tr>
<td>3.6</td>
<td>0.00094</td>
<td>3.18</td>
<td>5.22E-05</td>
</tr>
<tr>
<td>3.8</td>
<td>0.00096</td>
<td>3.339</td>
<td>5.33E-05</td>
</tr>
<tr>
<td>4</td>
<td>0.00098</td>
<td>3.498</td>
<td>5.43E-05</td>
</tr>
<tr>
<td>4.2</td>
<td>0.00099</td>
<td>3.657</td>
<td>5.50E-05</td>
</tr>
<tr>
<td>4.4</td>
<td>0.001</td>
<td>3.816</td>
<td>5.53E-05</td>
</tr>
<tr>
<td>zone</td>
<td></td>
<td>3.657</td>
<td>5.33E-05</td>
</tr>
<tr>
<td>4.2</td>
<td>0.00096</td>
<td>3.657</td>
<td>5.33E-05</td>
</tr>
<tr>
<td>4.4</td>
<td>0.00097</td>
<td>3.816</td>
<td>5.39E-05</td>
</tr>
<tr>
<td>4.6</td>
<td>0.00098</td>
<td>3.975</td>
<td>5.45E-05</td>
</tr>
<tr>
<td>4.7</td>
<td>0.00098</td>
<td>4.134</td>
<td>5.48E-05</td>
</tr>
<tr>
<td>4.9</td>
<td>0.00098</td>
<td>4.293</td>
<td>5.48E-05</td>
</tr>
</tbody>
</table>

As explained in Section 2.4.1.2, the non-dimensional compressor maps are corrected using the following procedure:
\[ \phi_c = \phi \left( \frac{P_{des}^2}{\sqrt{T_{des}^2}} \cdot \frac{\sqrt{T_{ref}}}{P_{ref}^2} \right) \]

The \( \phi \) term is the value we use directly from the original non-dimensional compressor map. The correction term,

\[ \text{Correction term} = \left( \frac{P_{des}^2}{\sqrt{T_{des}^2}} \cdot \frac{\sqrt{T_{ref}}}{P_{ref}^2} \right) \]

is calculated as follows. Design pressure is 4.5 times the reference pressure for the large compressor and 2.1 times for the small compressor as shown in Table 2.1. Reference pressure is taken to be the inlet pressure for the compressor in this case. The design temperature is equal to the inlet temperature, which is also the reference temperature. So, the correction factor becomes 20.25 for the large compressor and 4.41 for the small compressor, which can also be seen from the previous Excel files. These values bring both maps to same order of magnitude and the same procedure is followed for the expander maps.