2014-07-01

Analysis and Compression of Large CFD Data Sets Using Proper Orthogonal Decomposition

Trevor Jon Blanc
Brigham Young University - Provo

Follow this and additional works at: https://scholarsarchive.byu.edu/etd

Part of the Mechanical Engineering Commons

BYU ScholarsArchive Citation
Blanc, Trevor Jon, "Analysis and Compression of Large CFD Data Sets Using Proper Orthogonal Decomposition" (2014). All Theses and Dissertations. 5303.
https://scholarsarchive.byu.edu/etd/5303

This Thesis is brought to you for free and open access by BYU ScholarsArchive. It has been accepted for inclusion in All Theses and Dissertations by an authorized administrator of BYU ScholarsArchive. For more information, please contact scholarsarchive@byu.edu, ellen_amatangelo@byu.edu.
Analysis and Compression of Large CFD Data Sets
Using Proper Orthogonal Decomposition

Trevor J. Blanc

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

Steven E. Gorrell, Chair
Matthew R. Jones
Scott L. Thomson

Department of Mechanical Engineering
Brigham Young University
June 2014

Copyright © 2014 Trevor J. Blanc
All Rights Reserved
ABSTRACT

Analysis and Compression of Large CFD Data Sets Using Proper Orthogonal Decomposition

Trevor J. Blanc
Department of Mechanical Engineering, BYU
Master of Science

Efficient analysis and storage of data is an integral but often challenging task when working with computation fluid dynamics mainly due to the amount of data it can output. Methods centered around the proper orthogonal decomposition were used to analyze, compress, and model various simulation cases. Two different high-fidelity, time-accurate turbomachinery simulations were investigated to show various applications of the analysis techniques. The first turbomachinery example was used to illustrate the extraction of turbulent coherent structures such as traversing shocks, vortex shedding, and wake variation from deswirler and rotor blade passages. Using only the most dominant modes, flow fields were reconstructed and analyzed for error. The reconstructions reproduced the general dynamics within the flow well, but failed to fully resolve shock fronts and smaller vortices. By decomposing the domain into smaller, independent pieces, reconstruction error was reduced by up to 63 percent. A new method of data compression that combined an image compression algorithm and the proper orthogonal decomposition was used to store the reconstructions of the flow field, increasing data compression ratios by a factor of 40.

The second turbomachinery simulation studied was a three-stage fan with inlet total pressure distortion. Both the snapshot and repeating geometry methods were used to characterize structures of static pressure fluctuation within the blade passages of the third rotor blade row. Modal coefficients filtered by frequencies relating to the inlet distortion pattern were used to produce reconstructions of the pressure field solely dependent on the inlet boundary condition. A hybrid proper orthogonal decomposition method was proposed to limit burdens on computational resources while providing high temporal resolution analysis.

Parametric reduced order models were created from large databases of transient and steady conjugate heat transfer and airfoil simulations. Performance of the models were found to depend heavily on the range of the parameters varied as well as the number of simulations used to traverse that range. The heat transfer models gave excellent predictions for temperature profiles in heated solids for ambitious parameter ranges. Model development for the airfoil case showed that accuracy was highly dependent on modal truncation. The flow fields were predicted very well, especially outside the boundary layer region of the flow.

Keywords: proper orthogonal decomposition, reduced order models, reduced order reconstruction, data compression, computational fluid dynamics, post-processing, domain decomposition, computational fluid dynamics, coherent structures, turbomachinery, pressure distortion, conjugate heat transfer
ACKNOWLEDGMENTS

I would like to express much appreciation to my committee members for their valuable input and assistance in directing, performing and writing about this research. I would also like to recognize the other research assistants in the BYU Turbomachinery Research Laboratory for their collaboration. Additionally, much of this research would not have been able to be completed without the funding provided by the United States Air Force under Contract No. FA8650-12-M-2284. Finally, I would like to thank Earl Duque from Intelligent Light for his partnership and feedback in this project.
TABLE OF CONTENTS

LIST OF TABLES ......................................................... vi

LIST OF FIGURES ...................................................... vii

NOMENCLATURE ......................................................... x

Chapter 1 Introduction .................................................. 1
1.1 Background and Motivation ............................................. 1
1.2 Plan of Development ................................................... 3

Chapter 2 Literature Review .............................................. 5

Chapter 3 Methods ........................................................ 9
3.1 Proper Orthogonal Decomposition ..................................... 9
3.2 Reduced Order Reconstructions ..................................... 12
3.3 Data Compression ..................................................... 13
3.4 Reduced Order Modeling .............................................. 15
3.5 Domain Partitioning ................................................... 19

Chapter 4 Blade Row Interaction Rig Analysis and Compression ......... 22
4.1 Blade Row Interaction Rig ............................................. 22
4.2 Results ................................................................. 24
4.2.1 POD and ROR ....................................................... 25
4.2.2 Domain Partitioned ROR .......................................... 31
4.2.3 IPOD Compression ................................................ 34
4.3 Conclusions and Recommendations ................................... 36

Chapter 5 Full Annulus Multi-Stage Fan with Inlet Pressure Distortion ... 37
5.1 Computational Method .................................................. 37
5.2 POD Description ...................................................... 38
5.3 Analysis and Discussion ............................................... 41
5.4 Extension of POD Methods ......................................... 51

Chapter 6 Reduced Order Modeling of Conjugate Heat Transfer Simulations ... 53
6.1 Transient Conjugate Heat Transfer for Cylinder in Crossflow ............... 53
6.2 Steady-State Conjugate Heat Transfer for an Internally Cooled Turbine Blade ... 57

Chapter 7 Reduced Order Modeling of Flow Over an Airfoil .................... 62
7.1 Case Description ...................................................... 62
7.2 ROM Development ................................................... 63
7.3 Results and Discussion ............................................... 67
7.4 Concluding Remarks ................................................ 69
LIST OF TABLES

4.1 JPOD compression results with and without domain partitioning for the rotor pas-
sage. Note that the ASCII size of the data set was originally 74,687 KB. . . . . . . . . 34

6.1 Comparisons between the ROM and CFD solutions for the transient heating of a
cylinder in crossflow . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 56
6.2 Parameter sets used to compare CFD and ROM solutions . . . . . . . . . . . . . . 58
6.3 Maximum and RMS temperature error in ROM predictions for the steady-state
cooled turbine case . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 60

7.1 The number of modes required for each percent energy specification for the ROM . 64
7.2 Error calculations for ROMs created from various sets of simulations from the
original database compared with all the trial simulations. . . . . . . . . . . . . . . . . . . . . 65
7.3 ROM error calculations compared with CFD for nine trial parameter combinations. 67
# LIST OF FIGURES

1.1 A graphical outline for the relationship between POD-based methods and presented results. ......................................................... 3

2.1 A line of best fit determined by minimizing the $\sum (p_i)^2$ where $p_i$ is the orthogonal distance of each point from the line. .................. 6

3.1 Procedure for extracting POD modes from a set of CFD data. The rows of $D$ always correspond to the same location within the computational domain, and the columns of $D$ correspond to a specific extract. The entries in $D$ denoted by $f(x,y)$ are flow scalars from the CFD simulation. .............................. 10

3.2 This figure graphically shows how a three-dimensional block can be transformed into a two-dimensional surface for JPOD compression by unfolding each layer to make one long surface. ...................................................... 14

3.3 Conversion of a database of transient CFD solutions to $D_t$ in preparation for the POD. The value $m$ represents the total number of computational domain locations being modeled. ............................................. 17

3.4 An example of how a computational rotor domain is partitioned into $8 \times 8$ pieces. ............................................................... 20

4.1 Cross-section of the BRI test rig. ............................................................... 23

4.2 Computational domain for the BRI study consisting of 8 swirler, 8 deswirler, and 7 rotor passages ......................................................... 23

4.3 The first nine POD modes from the deswirler blade passage. ............... 27

4.4 The expansion coefficients as a function of the time step for the second and third POD modes of the deswirler data set. ......................... 28

4.5 Deswirler passage ROR of non-dimensionalized velocity snapshots 40, 44, 48 and 52 using the first 9 POD modes. In a, c, e, and g, the top profile is the original data set and the bottom profile is the ROR. The contours in b, d, f and h show the percent error from the respective snapshot ROR. ......................... 29

4.6 The first nine POD modes from the rotor blade passage. ....................... 30

4.7 Rotor passage ROR of non-dimensionalized velocity snapshots 72, 76, and 80 using the first 9 POD modes. In a, c, and e, the left profile is the original data set and the right profile is the ROR. The contours in b, d, and f show the percent error from the respective snapshot ROR. ......................... 32

4.8 The RMS and maximum error from the ROR (9 modes) of the rotor flow field at various degrees of partitioning. ................................. 33

4.9 The $8 \times 8$ domain partitioned ROR of snapshot 84 of the rotor passage flow field. In a, the left contour plot is the CFD solution and the right is the ROR. The percent error contours are shown in b. ................................. 35

5.1 The entire computational domain consisting of seven regions and 660 million nodes. ............................................................... 39

5.2 The inlet total pressure boundary condition giving a one-per-revolution distortion pattern. ............................................................ 39

5.3 Mass flow convergence plot. ............................................................... 40
5.4 The first POD mode from both the (a) Snapshot and (b) Repeated Geometry methods and their relative (c) normalized coefficients ........................................ 42
5.5 The second POD mode from both the (a) Snapshot and (b) Repeated Geometry methods and their relative (c) normalized coefficients ........................................ 44
5.6 The (a) third POD mode from the Snapshot method, the (b) fourth POD mode from the Repeated Geometry method, and their relative (c) normalized coefficients ........................................ 45
5.7 Frequency analysis of the normalized coefficients for mode 3 of the snapshot method and mode 4 of the RG method ................................................................. 46
5.8 The (a) fourth POD mode from the Snapshot method, the (b) third POD mode from the Repeated Geometry method, and their relative (c) normalized coefficients ........................................ 47
5.9 A graphical description of the (a) sample line through rotor passage and the (b) modal values from the RG POD scaled by their corresponding singular value ........................................ 48
5.10 The original and filtered coefficients from mode five of the RG POD ........................................ 49
5.11 A five-mode reconstruction of the RG non-dimensionalized static pressure data set using coefficients that have been fit to low-frequency sinusoidal signals ........................................ 50
5.12 Frequency analysis for snapshot POD mode coefficients from a data set containing 1000 computational extracts per revolution ........................................ 52

6.1 A diagram showing the basic setup for the cylinder in crossflow case. ........................................ 54
6.2 This is the grid used for all of the cylinder heat transfer simulations; it consisted of a 100,000 cell cylinder region and a 365,000 cell fluid region ........................................ 54
6.3 The plot shows the parameter space spanned by the database of solutions; therefore, the ROM should only be seen as valid for parameter values that fall within this space ........................................ 56
6.4 Comparison of the CFD and ROM maximum and mean cylinder temperatures for the random parameter set of 470.6 K and 29.5 m/s ........................................ 58
6.5 Sample solution at T = 470.6 K and V = 29.5 m/s temperature contours for the (a) CFD and (b) ROM solutions at t = 2 s along with respective (c) error contours. The highest ROM error occurs in the front of the cylinder at the symmetry plane ........................................ 59
6.6 The variation in maximum and minimum temperatures in the cooled turbine blade for each of the CFD simulations ........................................ 59
6.7 Temperature contours for the (a) CFD and (b) ROM sample solution using Set 4 parameters. The error contours are also shown in (c). ........................................ 61

7.1 The computational grid used to perform the CFD simulations ........................................ 63
7.2 Singular values for the interpolation matrix, F, to help determine a suitable cutoff value ........................................ 64
7.3 Graphical depiction of the parameter space that is spanned by the database of solutions as well as the trial points used to determine cutoff value and percent energy. ........................................ 66
7.4 Contour plot of Mach number RMS error between nine ROM-generated database solutions and their relative CFD solutions ........................................ 66
7.5 Contour plots at the Z = 0.025 m plane of the Mach number from the (a) CFD and (b) ROM solutions using the parameter values for Trial 1. The (c) absolute Mach number error is also shown. ........................................ 68
7.6 A plot of the iso-surface where the absolute Mach error between the CFD and ROM is equal to 0.01 for the Trial 1 parameter set
**NOMENCLATURE**

- \( a_j \): Modal coefficients from the proper orthogonal decomposition
- \( A \): Matrix of modal coefficients used to create a reduced order reconstruction
- \( A_{\text{dec}} \): Matrix of modal coefficients from decompressed modes
- \( b \): Bit length used in the JPEG compression
- \( C \): Matrix containing expansion coefficients for a reduced order model
- \( D \): A two-dimensional dataset containing information for one or more simulations
- \( D_{\text{ROR}} \): The reduced order reconstruction of a dataset
- \( \delta_j \): Parameter used to decompress each basis mode stored as a JPEG
- \( f(k) \): A vector of interpolating functions based on a set of input parameters
- \( F \): A matrix containing radial basis functions for interpolation in a reduced order model
- \( k \): A set of input parameters
- \( K \): Matrix containing all of the parameter values used in a ROM database
- \( \lambda_j \): Eigenvalues of \( D^T D \) arranged in decreasing order
- \( m \): Number of rows in \( D \) and corresponds to spatial locations in a domain
- \( n \): Number of columns in \( D \) and corresponds to various entities (ex. time, position)
- \( N_k \): Number of parameter sets used in the database of a reduced order model
- \( N_p \): Number of grid points
- \( N_m \): Number of modes used in a reduced order model
- \( P \): Percent energy parameter to determine how many basis modes are preserved
- \( \phi_j \): Basis vectors obtained from the proper orthogonal decomposition
- \( \Phi \): A matrix containing a selection of basis modes (usually the most dominant) from the proper orthogonal decomposition
- \( \Phi_{\text{dec}} \): Decompressed matrix of basis modes
- \( r \): The minimum number of modes to characterize a given percent energy of a system
- \( R(k) \): A reduced order model depending on parameter set, \( k \)
- \( \Sigma \): An \( m \times n \) matrix containing singular values from the singular value decomposition
- \( \sigma_j \): The singular values calculated from the singular value decomposition
- \( t \): Subscript denoting a specific point in time
- \( U \): An orthonormal \( m \times m \) matrix containing the basis modes from the singular value decomposition
- \( V \): An orthonormal \( n \times n \) matrix from the singular value decomposition
- \( v_j \): Eigenvectors of \( D^T D \) as well as the columns of \( V \)
- \( x \): Variable representing spatial dimensions
- \( y \): Variable representing various entities (ex. time, position)
CHAPTER 1. INTRODUCTION

1.1 Background and Motivation

Computational fluid dynamics (CFD) is an extremely useful tool for solving and analyzing both simple and complex flow problems. CFD involves numerically solving partial differential equations that are derived based on the principles of conservation of mass, momentum, and energy. To do this, a volume of material is discretized into very small pieces or cells that interact with each other and bounding surfaces based on those governing equations. Within each cell is stored flow information such as directional momentum, pressure, density, etc. specific to that region. High-fidelity simulations provide increased resolution in both space and time discretization to resolve complex and physically accurate flow features.

Increasingly capable and available computational resources have enabled high-fidelity simulations to be solved within a reasonable amount of time. However, with the development of more impressive supercomputing capabilities, and inherently larger and more complex problems, has come the challenge of post-processing these CFD solutions. For example, 660 million cell time-accurate simulations are currently being run to observe the effect of inlet pressure distortion on the performance of a three-stage fan. To save an instantaneous time step from a simulation of this size requires over 80 gigabytes of disk space. Sifting through the data from these high-fidelity simulations is daunting and time-consuming. Also, longterm storage of these data sets poses a significant challenge. Access to data analysis and compression methods then becomes a valuable asset for CFD users. The goal is to extract the most relevant and revealing flow information as quickly as possible and then store that data as efficiently as possible. One particularly useful method for data analysis and compression that can be used in a large variety of applications is the proper orthogonal decomposition (POD). Depending on the topic of application, the POD can go by a variety of names including Karhunen-Loève decomposition, principal component analysis, empirical eigenfunction decomposition, and singular value decomposition to name just a few.
The POD is a type of linear spectral method that produces a set of basis modes which represents the dynamics within a data set. A spectral method in this case is a technique that enables the representation of a field variable from a CFD solution as the summation of a set of basis functions or modes. These modes may be predefined to fit a specific form as it is with a Fourier series which uses a linear combination of sine and cosine functions of increasing frequency. In other spectral techniques the modes are defined by the solution to the partial differential equation itself. The POD falls under this type of spectral method. For example, solving the Navier-Stokes equations for a fluid domain through time using CFD provides an ensemble of flow scalars that may be analyzed using the POD to extract the modes describing the solution to these complex equations. These modes represent how the flow field scalar fluctuates through both time and space, and thus also present an excellent venue for analyzing turbulent flow characteristics. The most dominant modes may also be used to simplify or reduce the order of complexity required to estimate the flow field.

The basic principle behind the POD modes is that they maximize the projection of the data set from which they are extracted onto the mode while also enforcing orthonormality. While there is one mode that maximizes the projection for a given data set, additional modes can be extracted by removing first the maximum mode from the data set and then re-optimizing. This produces a set of POD modes that are ranked based on how much relevant information they contribute to the original data set. Since the importance of each mode is quantified, the order of the data set may be reduced by neglecting modes that contribute relatively little information. Reduced order reconstructions (ROR) use this selection of modes to produce a simplified or more generalized version of the original flow field. This simplification may also introduce error into the ROR since certain modes of variation are no longer included. At the same time, reducing the order of the data set also reduces the amount of data space required for storage. Combining methods for producing a ROR with image compression techniques, this data reduction quality of the POD can be increased.

While a single, large, high-fidelity CFD simulation can be analyzed using the POD, the technique can also be applied to sets of simulations as well. A set of simulations, or database, refers to the same computational space, but each simulation is solved using varying values for input parameters (i.e. fluid or solid properties, boundary conditions). This database may come as a result of a parametric study for a specific geometry at different operating conditions. POD modes
extracted from the database are coupled with interpolating functions to produce reduced order models (ROM). A ROM enables the prediction of CFD field variables for input parameters within the range of those used to produce the solution database. These solutions, while not perfectly accurate, can be produced very quickly compared to the actual CFD solution itself. The ability to rapidly generate flow field estimates will improve inverse design and optimization efforts.

1.2 Plan of Development

Since the fundamental concepts behind the POD have been around for over a century, there is an extensive set of literature devoted specifically to the POD and its application. In Chapter 2, a brief presentation of some of the key works defining the method through the years is given. A review will also be given of other analysis, modeling, and compression methods to compare with the methods in this thesis. Because the results are presented and analyzed in a case by case manner, the methods utilized throughout are described in Chapter 3. Following that explanation, the next four chapters expound upon and illustrate the variety of applications for POD-based techniques. Figure 1.1 gives a visual representation of how the four result chapters relate to the POD techniques described in Chapter 3.

Chapter 4 contains analysis of turbulent structures within turbomachinery flows along with applications of a new CFD compression method. This will be followed by a unique application of the POD for pressure distortion at the inlet of a highly complex three-stage fan in Chapter 5. The next two chapters focus particularly on developing and evaluating ROMs. In Chapter 6, both tran-

Figure 1.1: A graphical outline for the relationship between POD-based methods and presented results.
sient and steady heat transfer CFD cases are generated to assess the ability of a ROM to estimate temperature profiles in solids at various flow conditions. On the other hand, some of the weaknesses of ROM methods for predicting viscous flow are shown in Chapter 7 by modeling flow over an airfoil. Chapter 8 will give concluding remarks regarding the POD methods and its present applications while providing recommendations for future development and use. In addition, Matlab code is provided in Appendix A to show how the methods are applied in practice.
CHAPTER 2. LITERATURE REVIEW

The proper orthogonal decomposition (POD) is a method that owes its origins to ideas presented over a century ago. In 1901, Pearson [1] presented work on finding lines and planes of best fit to a system of points existing in multi-dimensional space. He observed that current methods for the best-fitting lines assumed a preferential dependence of one of the coordinates on the other. Pearson instead defined the line of best fit by minimizing the sum of the squared distances of each point to its perpendicular location on the line as shown in Figure 2.1. This is also the same way of saying that the line of best fit maximizes the projection of the data set onto the best fit line. This optimization method is the main concept behind determining optimal modes from a multi-dimensional set of data.

Over 30 years later, Hotelling [2, 3] independently developed an iterative method to determine what he called principal components. Hotelling’s focus was on analyzing trends in complex statistically based data sets and how the different variables correlated with one another. These principal components were the equivalent of the best fit slopes defined by Pearson’s method except in a greater number of dimensions. The principal components were found one at a time, and once one was found, the variation depicted by the component was removed from the correlation matrix and the next component was determined. Each component represented the optimal variation fit for the respective matrix. This continued until all of the principal components were found. He also ranked the components by the percent variance that each characterized, explaining that some components are “trivial” due to the relatively small amount of variance they contained.

Both Pearson and Hotelling made use of what in linear algebra is called the eigenvalue decomposition [4]. The same information may also be obtained using a very popular and useful matrix factorization technique called the singular value decomposition (SVD). The principal components are the equivalent to the eigenvectors of the correlation matrix and the eigenvalues corresponded to the amount of variance in each direction. Given an $m \times n$ matrix, $D$, where $m$ is
Figure 2.1: A line of best fit determined by minimizing the \( \sum (p_i)^2 \) where \( p_i \) is the orthogonal distance of each point from the line

greater and \( n \), the correlation matrix is defined as \( D^T D \), producing a symmetric \( n \times n \) matrix. The eigenvectors, or modal coefficients as they will be referred to in this thesis, relate the trends between the different dimensions of the data set. These dimensions may refer to different moments in a sequence of time, physical positions, independent parameter-based solutions, or countless other entities. However, this thesis will continue primarily with the three types of dimensions just specified.

This procedure goes by a variety of names including the Hotelling transform, principal component analysis, empirical orthogonal functions, Karhunen-Loève transform, and others already mentioned. This is an obvious indication of its versatility and utility in many categories of data analysis. While the working name of these equivalent methods may change depending on the area of application, the general technique will be referred to primarily as POD in this work. The POD is a linear spectral technique that enables a complex set of data to be formulated into summation of coefficients and modes.

\[
D(x,t) = \sum_{j=1}^{n} a_j(t) \phi_j(x) \tag{2.1}
\]

In Eqn. 2.1, the basic form of a spectral representation of a finite two-dimensional matrix is shown where \( a_j(t) \) are the modal coefficients, \( \phi_j(x) \) are the POD modes, \( x \) denotes a specific location, and \( t \) represents a discrete point in time, space, or other defining dimension. Because the data sets
encountered within computational fluid dynamics (CFD) are always finite, these POD modes also may be referred to as basis vectors.

The first researcher to apply the POD to turbulence in fluid flows was Lumley [5] in 1967. POD modes were extracted to portray turbulent coherent structures that vary within a flow. Sirovich [6] later published a series of papers further expanding these ideas as well as introducing the method of snapshots to examine unsteady flows. This method refers to constructing the data set from equally spaced snapshots or strobes in time of the domain of interest. He also recommends that only certain number of these POD modes are necessary to define the data set based on what he terms as the percent energy (see Chapter 3 for additional explanation).

The POD is extremely beneficial because it is able to extract both spatial and temporal patterns from a set of data without being biased toward a specific outcome and without significant interaction with the data itself. The modes may also reveal specific qualities of a simulation and sources of fluctuation that may not be readily apparent from the solution itself. For example, certain modes and their coefficients may be dominated by frequencies relating to other physical phenomena. Berger et al. [7,8] use the POD modes from high-resolution PIV to analyze a transonic jet to correlate with acoustic readings taken by far-field probes. By taking advantage of this modal identification, efforts may be made to control a specific type of flow variation to reduce jet noise. They also used reduced order reconstructions (ROR) of the PIV data in order to filter out the smaller and less dominant flow structures to simplify the turbulence. Additionally, POD modes can be used to develop controller models for various chaotic signals [9].

While the extraction of POD modes are generally well understood and can be performed using well-known algorithms, how the POD is used continues to expand. One such area of application is in data compression, due to the optimality and ranking of the POD modes. Andrew et al. [10] touted the POD for its optimality, but claimed it was “impractical” because of the lack of computational power at the time. Computational advances have made the POD a readily available option. CFD simulations can be compressed using the POD by truncating the set of modes to contain a certain number of modes or percentage of the variance of the original data set. If the CFD data set is structured, additional steps can be applied to further reduce the data required to store a simulation. For example, Schmalzl [11] examined the merit of applying image compression techniques to structured CFD solutions. He found that accurate and swift compression resulted
from the Joint Photographic Experts Group (JPEG) algorithm, which is based off the discrete cosine transform. Applying this image compression with the already truncated modes can greatly increase the data reduction.

A common application for the POD is model order reduction used to simplify solutions to partial differential equations by using the Galerkin projection. The method has been shown to give interesting results for a variety applications [12–16], including turbomachinery-related flows. For example, Hall et al. [12] used eigenmode analysis to successfully compute time-linearized potential flow in a rotating blade passage. More recently, Brenner et al. [15] successfully modeled high Mach number inviscid, compressible turbomachinery flow to predict the transient flow independently of the full-order model from which the modes were found. They found that efficiency difference between the two models was less than 0.01 points and the flow fields produced were very similar.

The modeling presented in this thesis consists of developing a parameter-based reduced order model (ROM) from a database of CFD solutions that were solved using different combinations of fluid boundary conditions. Ding et al. [17] used cubic spline interpolation to estimate heat transfer and fluid flow solutions while varying only a single key parameter. They found that simple CFD simulations could be predicted quickly and accurately. Combining the POD modes generated from the database and interpolation methods, the ideal ROM would predict solutions for boundary condition value combinations not used to create the database. These interpolating functions enable multiple input parameters for the ROM. Using this method, Larson and Jones [18] modeled the time-dependent reflectance profiles. Also, Ostrowski et al. [19, 20] created ROM networks to inversely determine heat conduction and convective boundary condition parameters for both steady and transient cases. Their study showed excellent application for quickly optimizing complicated heat conduction problems. However, the database solutions used defined the convective boundary conditions for the solid rather than solving the conjugate problem. This study extends the ROM strategy to multi-parameter conjugate heat transfer and fluid flow problems and evaluates their performance.
CHAPTER 3. METHODS

3.1 Proper Orthogonal Decomposition

The main purpose for using the proper orthogonal decomposition (POD) is that it establishes a set of basis modes or functions from a set of data which can be used for a wide variety of purposes. Because much has been written concerning the intricate mathematical details concerning its development, the reader is directed to Holmes et al. [16] for an excellent step-by-step derivation of the POD and its properties. The POD is essentially a linear spectral method, similar in concept to a Fourier series, that transforms a dataset into sum of products of basis modes and their respective modal coefficients. Figure 3.1 gives the basic flow for compiling data from a simulation to extract the POD modes and coefficients. The POD is shown as somewhat of a black box, but is discussed in more detail below.

For example, let $D(x, y)$ represent a dataset of flow scalars (denoted by $f(\ast, y)$ in Figure 3.1) that is function of both $x$ and $y$. Then the POD produces a representation of $D$ given by

$$D(x, y) = \sum_j a_j(y) \phi_j(x) \quad (3.1)$$

where $a_j(y)$ are modal coefficients and $\phi_j(x)$ are the basis modes. In all of the methods presented here, $x$ represents a discrete spatial location, such as a specific cell in a computational domain. However, $y$ will represent a range of discrete entities depending on the analysis technique, including time, position, and parameter set combinations. It should be noted that when $y$ represents equally spaced extracts in time, the specific application of the POD is called the method of snapshots (see [6]).

There are multiple ways to perform the POD and extract basis modes from a dataset. The first way to obtain the basis modes is by using the singular value decomposition (SVD). The SVD is an extremely useful and common matrix factorization in linear algebra [4] that can be applied to
Figure 3.1: Procedure for extracting POD modes from a set of CFD data. The rows of $D$ always correspond to the same location within the computational domain, and the columns of $D$ correspond to a specific extract. The entries in $D$ denoted by $f(\ast, y)$ are flow scalars from the CFD simulation.

A matrix of any size. It is assumed that the dataset, $D$, can be represented by an $m \times n$ matrix where the rows correspond with spatial locations, as in $x$, and the columns represent time or position, as in $y$. Then, using the SVD, $D$ can be factored into three matrices

$$D = U \Sigma V^T$$

such that $U$ and $V$ are $m \times m$ and $n \times n$ orthonormal matrices, respectively, and $\Sigma$ is an $m \times n$ containing the singular values, $\sigma_j$, of $D$ along its diagonal. The magnitudes of the singular values
give an idea of relative variation information content contained within their corresponding vectors in $U$ and $V$. Based on the definition of $D$, the columns of $U$ are the spatial basis modes, $\phi_j$, denoted in Eqn. 3.1.

The SVD is convenient since there are many mathematical programs with built-in algorithms that can be used to solve for the decomposed matrices. However, it may not always be desirable to extract all of the basis modes at one time, especially if there are memory constraints due to the size of the original dataset. Additionally, the POD is often used for order reduction, and only a handful of the modes extracted would be required. A second and more efficient way to procure $\phi_j$ is to solve directly for each spatial mode individually. This is done by calculating the eigenvalues, $\lambda_j$, and eigenvectors, $v_j$, of $D^TD$ which is a symmetric $n \times n$ matrix. Assuming that $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$, then $v_1$ is equivalent to the first column of $V$, $v_2$ to the second column, and so on. Also, the singular values of $D$ correspond to the square root of the eigenvalues of $D^TD$. Therefore, by modifying Eqn. 3.2 to solve for the basis modes or individual columns of $U$, each $\phi_j$ can be found independently using the relationship in Eqn. 3.3.

$$
\phi_j = \frac{Dv_j}{\sqrt{\lambda_j}}
$$  \hspace{1cm} (3.3)

Returning to the reduction in order concept, it is desired that a method be used to determine how many basis modes are generated or preserved from the POD. Sirovich [6] suggested a criterion for this decision by taking the minimum number of modes, $r$, that captures a given percentage of energy, $P$, satisfying

$$
\frac{\sum_{j=2}^{r} \lambda_j}{\sum_{j=2}^{n} \lambda_j} \times 100\% < P
$$  \hspace{1cm} (3.4)

The first mode is not included in the previous calculations because it contains the mean POD mode, and the energy is a measure of the fluctuation from the mean. In his work, the values being analyzed were flow velocities, and so the energy calculations were related to the average kinetic energy of a fluid system. Some of the examples considered here will not only observe velocities, but also temperatures and pressures. However, the concept of percent energy as shown in Eqn. 3.4 will be used regardless of the type of value being investigated.

The individual modes themselves can give a lot of valuable information about the dynamics of a simulation. The eigenvectors relating to each mode then gives frequency information which
may be correlated with other physical characteristics of a simulation. This is particularly evident when the columns of $D$ are representative of snapshots in time in a solution. If this is the case, the first POD mode for a given dataset represents a mean condition and remaining POD modes display structures of fluctuation from the mean condition through time. As a result, these modes contain coherent structures which correspond to fluid flow phenomena like vortices, shockwaves, wakes, and any other time- or position-dependent entity. The presence of multiple structures of variation in a basis mode can reveal correlations that may not readily be seen from the solution as a whole.

### 3.2 Reduced Order Reconstructions

By taking only a selection of the basis modes, a reduced order reconstruction (ROR) of the original dataset can be created. The basis modes do not necessarily have to be the most dominant modes of the POD, but the ROR is likely to be more accurate if high-energy modes are used. Therefore, one may base the number of modes for the ROR on a certain percent energy. Additionally, there may be specific modes that have frequencies or structures of interest that may be singled out to include in a reconstruction. Whatever the method, mode truncation is an essential part in producing the order reduction as the name implies. First, let $N_m$ be the total number of modes selected for use in the ROR. The truncated basis vector matrix, $\Phi$, can then be formed by extracting the desired basis modes. Next, a matrix of coefficients, $A$, is formed to satisfy the following relationship:

$$\Phi A = D$$

(3.5)

Because $\Phi$ is an orthonormal matrix, $A$ can easily be found by multiplying both sides of Eqn. (3.5) by $\Phi^T$. The ROR of the original dataset, $D_{ROR}$, is then calculated using

$$D_{ROR} = \Phi A$$

(3.6)

The truncation of the basis vectors introduces error into the ROR. The amount of error is roughly related to the percentage of the total energy of the modes not included in the ROR. The error is usually concentrated in areas of the domain where there are local extrema. Error is also especially apparent in areas where there are extreme gradients in the data set, such as a steep
change in velocity that occurs across a shock wave. As a result, the ROR generalizes the flow field, but it still provides an accurate depiction of the flow features that lie in the space spanned by the modes used in the reconstruction.

### 3.3 Data Compression

Assuming that a given ROR produces satisfactory results (in terms of reconstruction error or some other specification), the reduction in order naturally translates into a reduction in data. Given a $N_m$-mode ROR of a data set containing $N_t$ snapshots, the initial compression ratio can be estimated:

$$\text{Compression Ratio} = \frac{N_t}{N_m} : 1 \quad (3.7)$$

The actual compression ratio is a little less due to the storage of the coefficient matrix, $A$, for the ROR. The amount of compression in this portion is, therefore, heavily dependent on the ratio of the number of original snapshots to the number of modes in the reconstruction.

This compression ratio can be greatly increased by applying image compression techniques to the individual modes. Schmalzl [11] used a variety of image compression algorithms to store a CFD solution and showed that it can produce high compression ratios while maintaining accuracy. He found that using the Joint Photographic Experts Group (JPEG) image compression algorithm, CFD could be stored quickly and accurately. The JPEG standard, a lossy compression method, utilizes the discrete cosine transform to code pixel information. It is best applied where there are smooth transitions in pixel values, making it a good candidate for its application to CFD.

Combining both the JPEG image compression and POD methods, ultra high compression ratios can be obtained. This is a new compression technique that has not been developed previously according to the author’s knowledge. It is fairly straight-forward to apply since it is based off of two very well-known data processing procedures. The bridging of the gap between the JPEG and POD steps is what is most novel here. Thus, a provisional patent has been obtained for this new combinatorial method. The JPEG-POD (JPOD) compression procedure for a two-dimensional, structured grid CFD solution will be explained in more detail. The method was developed in Matlab, so there may be some nuances relative to that environment that may not be present in other settings.
The first step in the JPOD compression technique is to extract the desired POD modes. While every POD mode is initially a one-dimensional vector, each is converted to a 2-D array with the dimensions corresponding to the number of rows and columns in the original grid. For example, a grid with 50 rows and 50 columns in the computational domain would correspond to a $50 \times 50$ matrix. There are two important parameters from each mode that must be stored in order to decompress the data. The first parameter is the minimum value in the basis vector. The second specifies the resolution, $\delta_j$, of the values in the basis vector and is determined using the following formula:

$$
\delta_j = \frac{\max(\Phi_j) - \min(\Phi_j)}{2^b - 1} \quad (3.8)
$$

where $\Phi_j$ is the mode being compressed and $b$ represents the bit length that will be used in the JPEG compression. From Eqn. (3.8), we see that a larger bit length increases the resolution and possibly the accuracy at the expense of a decreased compression ratio.

To prepare the array for JPEG compression, the values are normalized such that each entry is between zero and one. The next step is to use the JPEG algorithm to compress the 2-D array as an image file with the same dimensions. In some cases, the quality of the JPEG compression can be altered using an additional input value specifying the quality of the compression. As the quality of the compression increases, the compression ratio decreases. Because this is not the primary focal point of this method, the quality used for JPOD compression was left at the Matlab default value of 75 (out of 100).

Note that the coefficient matrix for the ROR has not yet been found. In order to determine the best set of coefficients, we use the decompressed modes. This is done because the JPEG compression may introduce some variation into the modes themselves and fitting the decompressed modes with their own coefficients gives a better reconstruction. In the decompression process, each pixel represents a data point and contains an integer value between zero and $2^b - 1$. Using this

Figure 3.2: This figure graphically shows how a three-dimensional block can be transformed into a two-dimensional surface for JPOD compression by unfolding each layer to make one long surface.
integer in combination with the saved parameters, the decompressed value is obtained. Once this is
done for the entire mode, the magnitudes should be adjusted so that the $L^2$-norm is one, preserving
the orthonormality of the basis vector. Using the decompressed modes, $\Phi_{\text{dec}}$, the coefficients can
be found with Eqn. (3.9).

$$A_{\text{dec}} = \Phi_{\text{dec}}^T D$$ (3.9)

The coefficients and input parameters are stored in their own data files. After writing all of
the files, sending them to a ZIP folder provides an additional level of lossless compression. This is
most effective for the ASCII data files. The final compression ratio is dependent now on a variety
of factors, including the number of modes, the bit length in the JPEG compression, and the quality
of the JPEG compression. While the JPEG compression method is lossy and introduces some error
of its own in addition to the ROR error, it will be shown that this error is negligible for complex
data sets.

While the JPOD method is applied to two-dimensional slices of a simulation in this case, it
may also be adapted for the three-dimensional solution. This is done by converting the grid from
three dimensional space to two by unfolding the grid to create a single long surface. Figure 3.2
shows one method for how this can be accomplished to maintain continuity with adjacent grid
locations. For more complicated structured grids, such as those with multiple blocks, additional
steps for conversion to two dimensions may be necessary. Once the domain has been converted,
the JPOD compression method can be applied as previously described.

3.4 Reduced Order Modeling

In order to create a reduced order model (ROM), first a database of solutions is necessary.
The database should contain a number solutions of where each is the result of a unique set of input
parameters. The columns of the database, $D$, then represent data obtained for a specific governing
input parameter configuration. The rows of $D$ correspond to individual data point locations within
the computational domain. The combination of all the parameter sets is represented by $K$, a $N_p \times N_k$
matrix where $N_p$ is the total number of input parameters and $N_k$ is the number of input parameter
combinations. Note that this data matrix is representative of only a single step in time and that,
in order to model a transient data set, there must be a collection of data matrices in which each
matrix represents a specific moment in time. The data should be aligned at the same or relatively equivalent initial condition to make this process successful.

The end goal of the ROM is to be able to generate a steady-state or transient solution data set for an arbitrary set of input parameters, $k$, using some reduced order model, $R(k)$, at time step $t$. More specifically, a transient ROM can be understood as a set of independent ROMs corresponding to specific snapshots in time. This model may be represented by Eqn. (3.10)

$$R(k)_t = \Phi_tC_t f(k)$$  \hspace{1cm} (3.10)

where $k$ is an some set of input parameters, $\Phi_t$ is a set of basis vectors for a given time step, $C_t$ is an expansion coefficient matrix for a given time step, and $f(k)$ is a vector of interpolated values formed from $K$ and the input parameters.

The process for obtaining each component of the model will now be explained. Because the transient ROM requires POD modes for each moment in time modeled, the data set for each time step must also be defined. Figure 3.3 shows how each $D_t$ is formed from the original database of solutions. The entries in $D_t$ can represent any of the simulation scalar values such as Mach number or temperature. The modes extracted by the POD from these data sets represent how the database varies depending on the input parameters for a specific moment in the simulation time.

First, the percent energy is chosen to determine the number of basis vectors for $\Phi_t$ at every time step of the transient ROM. This is a difficult number to determine because the truncated collection of modes must be able to accurately depict the variation in the data set. However, including too many lower energy modes may try to force the ROM to fit the interpolation to structures that do not correspond well with the input parameters. Therefore, there is a balance that must be determined, possibly through an iterative approach. Using one of the POD methods described previously, the first $r$ basis vectors are calculated from the database for each time step of the model.

The next step is to determine the matrix $C_t$ which contains the expansion coefficients for each time step. To accomplish this, we assume that the original data set, $D_t$, can be defined by the following relation in Eqn. (3.11).

$$D_t = \Phi_tC_t F(K)$$  \hspace{1cm} (3.11)
Figure 3.3: Conversion of a database of transient CFD solutions to $D_t$ in preparation for the POD. The value $m$ represents the total number of computational domain locations being modeled.

The term $F(K)$ is a $N_k \times N_k$ set of radial basis interpolation functions evaluated using the matrix $K$. Each column of the interpolation matrix, $F_j$, can be found using Eqn. (3.12).

$$F_j = \begin{bmatrix}
\frac{1}{\sqrt{(K_{1,1} - K_{1,1})^2}} \left(\frac{K_{1,1} - K_{2,1}}{\max(K,1)}\right)^2 + \left(\frac{K_{2,1} - K_{3,1}}{\max(K,2)}\right)^2 + \ldots + \left(\frac{K_{Np,1} - K_{1,1}}{\max(K,Np)}\right)^2 + 1
\vdots
\frac{1}{\sqrt{(K_{1,2} - K_{1,2})^2}} \left(\frac{K_{1,2} - K_{2,2}}{\max(K,1)}\right)^2 + \left(\frac{K_{2,2} - K_{3,2}}{\max(K,2)}\right)^2 + \ldots + \left(\frac{K_{Np,2} - K_{1,2}}{\max(K,Np)}\right)^2 + 1
\vdots
\frac{1}{\sqrt{(K_{1,Nk} - K_{1,Nk})^2}} \left(\frac{K_{1,Nk} - K_{2,Nk}}{\max(K,1)}\right)^2 + \left(\frac{K_{2,Nk} - K_{3,Nk}}{\max(K,2)}\right)^2 + \ldots + \left(\frac{K_{Np,Nk} - K_{1,Nk}}{\max(K,Np)}\right)^2 + 1
\end{bmatrix} \quad (3.12)$$

It should be noted that $\max(K,m)$ is equal to the maximum value of all entries of row $m$ in the parameter matrix $K$. Having obtained $F(K)$, we must solve for $C_t$ in Eqn. (3.11). However, since $F(K)$ may be singular, the Moore-Penrose pseudoinverse [21] should be used. The
The pseudoinverse is also calculated using the SVD.

\[ F = U_F \Sigma_F V_F^T \]  

(3.13)

Because \( U_F \) and \( V_F \) in Eqn. (3.13) are orthonormal matrices, each has an inverse that is equal to the transpose of the respective matrix. While \( \Sigma_F \) is a diagonal matrix and its inverse may be found readily, if \( F \) was a singular matrix, the SVD will produce zero (or nearly zero) singular values. Therefore, the risk of it being singular necessitates the pseudoinverse, \( F^+ \), which satisfies Eqn. (3.14).

\[ FF^+ = FV_F S_F U_F^T = U_F \Sigma_F V_F^T V_F S_F U_F^T = I \]  

(3.14)

In order to form \( S_F \) in the above equation, we establish a minimum value for the singular values contained in \( \Sigma_F \). Like the percent energy decision, the optimal cutoff value may not be readily apparent from from the singular values. However, as a general rule of thumb, at least some of the singular values should be neglected because they can create interpolation noise that would affect the accuracy of the ROM. If the singular value is below some cutoff value, its respective matrix entry in \( S_F \) is set to zero. Otherwise, the reciprocal of the singular value is used as shown in Eqn. (3.15).

\[ S_F = \begin{cases} 
\text{diag}\{1/\sigma_{F,i}\}, & \text{if } \sigma_{F,i} \geq \sigma_{\text{min}} \\
0 & \text{otherwise}
\end{cases} \]  

(3.15)

Once the pseudoinverse of the interpolating function matrix is found, the expansion coefficients for each time step may be determined by solving for \( C_t \) in Eqn (3.11).

\[ C_t = \Phi_t^T D_t F^+ \quad \text{where} \quad F^+ = V_F S_F U_F^T \]  

(3.16)

The last piece to this ROM technique is the \( f(k) \) term which is a vector of interpolated values using an arbitrary parameter set that falls within the minimum and maximum parameter bounds set by \( K \). Therefore, let \( k \) be a one-dimensional vector of the length \( N_p \) whose entries correspond to the same input parameters as the rows of \( K \). Then the interpolating function vector
can be found using Eqn. (3.17).

\[
f(k) = \begin{bmatrix}
\sqrt{\left(\frac{k_1-K_{1,1}}{max(K,1)}\right)^2 + \left(\frac{k_2-K_{2,1}}{max(K,1)}\right)^2 + \cdots + \left(\frac{k_{Np}-K_{Np,1}}{max(K,Np)}\right)^2 + 1} \\
\vdots \\
\sqrt{\left(\frac{k_1-K_{1,2}}{max(K,1)}\right)^2 + \left(\frac{k_2-K_{2,2}}{max(K,2)}\right)^2 + \cdots + \left(\frac{k_{Np}-K_{Np,2}}{max(K,Np)}\right)^2 + 1} \\
\vdots \\
\sqrt{\left(\frac{k_1-K_{1,Nk}}{max(K,1)}\right)^2 + \left(\frac{k_2-K_{2,Nk}}{max(K,2)}\right)^2 + \cdots + \left(\frac{k_{Np}-K_{Np,Nk}}{max(K,Np)}\right)^2 + 1}
\end{bmatrix}
\]  

(3.17)

The matrices \( \Phi_t \) and \( C_t \) are constants once the transient ROM has been formulated, but a new \( f_k \) vector must be calculated for every arbitrary parameter set solution desired. This also requires that \( K \) be a necessary piece of information to include for the model.

For some applications, such as with simple computational heat transfer problems, the transient ROM lends itself very well since there may be a well-defined initial condition from which the time steps can be aligned. In other more complex cases, an alignment of time steps may be more difficult or not possible at all. Also, the defining parameter set \( K \) and the interpolating functions become integral elements in determining the accuracy of the model. Having a greater number of sample data sets is beneficial as it tends to increase the accuracy of the model. Once the ROM has been formulated, generating new solutions is much quicker than solving the problem using a simulation since the calculations required are simply a set of matrix multiplications.

### 3.5 Domain Partitioning

A technique known as domain partitioning can be particularly useful when applied to RORs and ROMs. Domain partitioning is simply dividing a computational domain into a collection of independent pieces. Those pieces are then independently analyzed using the POD producing basis modes specific to each piece. This method has been used especially for developing models for various fluid [22] and structural [23] phenomena.

The motivation for applying domain partitioning is similar to the motivation for refining a computational grid. In areas of the computational domain where more dynamic or high gradient
phenomena are located, it is beneficial to increase the resolution of the grid to obtain greater accuracy in modeling the phenomena. Likewise, partitioning the domain into smaller pieces focuses in on the local phenomena and obtains POD basis modes that better represent the modal variations specific to that area.

A second reason for partitioning the domain is that the POD is more effective when the grid points in the data set are more related. The term *related* can refer to points having similar dominant frequencies, mean values, phase, or other phenomena. For instance, consider the example of flow around a cylinder. It is unlikely that the flow through a point upstream of the cylinder has similar dynamic characteristics as the flow through a point located in the wake region. Therefore, these areas should be treated independently to hone in on the flow dynamics specific to the region in which the point resides.

While there may be a variety of techniques for partitioning the domain, a convenient method is to divide the grid into sections that contain an equal number of grid points. The actual physical size of these pieces would then vary due to changes in grid density. A sample $8 \times 8$ partitioning of a rotor blade domain can be seen in Figure 3.4. Each division contains the same number of grid points. Notice that the pieces directly above and below the blade surfaces are much
much larger in size compared with the pieces at its leading and trailing edges of the rotor due to grid density differences.

By combining domain partitioning with the ROR or ROM methods, increased accuracy while using the same number of modes can be obtained. The increased accuracy is due to fact that each the POD of each piece gives coefficients specific to that region’s basis modes. However, this does also result in a significant amount of additional data to store. More effective partitioning methods may be possible, however, they will not be discussed in this thesis.
CHAPTER 4. BLADE ROW INTERACTION RIG ANALYSIS AND COMPRESSION

This chapter presents the analysis and compression of a time-accurate, turbulent, turbomachinery simulation using the proper orthogonal decomposition (POD) and related methods. Coherent structures, such as shocks and vortices, extracted from the flow fields of both a deswirler and rotor blade passage are shown. Reduced order reconstructions (ROR) of the deswirler and rotor passages are produced by truncating the set of POD modes and then analyzed for accuracy in the reproduction of the time-varying flow field. Domain partitioning is applied to the rotor passage to increase ROR accuracy and resolve smaller turbulent structures in the flow. The combinatorial JPEG-POD (JPOD) compression method is evaluated using the original and domain-partitioned rotor data. Compression ratios using this combined method are increased by up to 45 times when compared to just the truncation of the basis functions for the reduced order reconstruction. The compression error introduced by the image compression step is shown to be negligible compared to the error introduced by the basis vector truncation.

4.1 Blade Row Interaction Rig

The data used for the POD investigation was obtained from simulations performed by Clark and Gorrell [24, 25] on a computational domain based on the Blade-Row Interaction (BRI) test rig (see Figure 4.1). The computational domain models the first three rows of the BRI test rig as a quarter annulus. A periodic grid (see Figure 4.2) with 8 swirler passages, 8 deswirler passages, and 7 rotor passages was generated in order to produce the high-fidelity, time-accurate simulations that demonstrated shock generation and propagation, vortex shedding, and boundary layer characteristics. The grid consisted of over 150 million nodes.

The design conditions of the BRI rig are such that the rotor blades are moving at transonic speeds from the hub to the tip. In the original study, simulations were run for different spacings of the swirler/deswirler vanes from the rotor blades. These solutions were validated using PIV and
Figure 4.1: Cross-section of the BRI test rig.

Figure 4.2: Computational domain for the BRI study consisting of 8 swirler, 8 deswirler, and 7 rotor passages.
performance data from the test rig [26,27]. The parallel flow solver TURBO [28] was used to solve the Unsteady Reynolds-Averaged Navier-Stokes (URANS) equations using a $\kappa - \varepsilon$ turbulence model for turbomachinery [29]. A quarter revolution of the rotor blade row was equivalent to 2240 time steps. The computations utilized hundreds of thousands of CPU hours in parallel on over 500 processors.

The inlet boundary condition was an isentropic inlet with the same temperature, pressure, and velocity profiles measured at the inlet of the experimental BRI rig with an inlet turbulence intensity of two percent. No-slip conditions were imposed on the hub, case, and each blade surface. A sliding interface was imposed at the interface between the deswirler and rotor computational sections. The exit mass flow rate corresponding to peak experimental efficiency of the BRI rig at mid-spacing was used at the rotor exit. Convergence of the simulation was defined to be when the inlet and exit mass flow rates converged to the same value.

For our purposes, a portion of the decreased loading, mid-spacing CFD solution was sufficient to show the benefits of the various POD methods. Two-dimensional slices from the fourth passage of the deswirler blade row and the third rotor blade passage were chosen for the POD and ROR analyses. Slices were extracted from the CFD solution at fifty percent span between the hub and tip of the respective blade passage. Each slice represented a snapshot in time of the solution and are referred to as snapshots in the remainder of this chapter. The dimensions of each snapshot were $201 \times 361$ for the deswirler passage and $226 \times 391$ for the rotor passage. A snapshot was saved every 20 time steps over one quarter rotor revolution resulting in 113 total snapshots. The actual time between each snapshot was $8.16 \times 10^{-6}$ seconds. The absolute velocity magnitude was the only solution value analyzed, but a similar procedure could be used for the individual velocity components, energy, or any other field variable.

4.2 Results

The results have been separated in three sections. In the first, the deswirler and the rotor blade POD and ROR analyses are presented. The second section portrays the effect of domain partitioning on ROR error for the rotor blade only. The final section contains results from the JPOD compression method for the rotor blade data both with and without domain partitioning.
4.2.1 POD and ROR

The deswirler and rotor blade passage data sets were analyzed using the POD and ROR. The values investigated were non-dimensionalized velocity magnitudes. Both data sets contained 113 total snapshots, which resulted in the same number of POD modes each. Neither of the two data sets were singular, which is to be expected due to the non-linear turbulent characteristics of the flow fields. The first POD mode is generally a representation of the mean flow conditions since the average flow field was not remove from the data sets. The reconstructions of the deswirler and rotor passages that will be presented utilize 9 of the 113 basis vectors, or about 8%. Therefore, the initial POD compression ratio is 12.5 : 1. The percent error of the reconstructions at each grid location was calculated using the following equation:

\[
\% \text{Error}_{ij} = \frac{|D_{\text{ROR},ij} - D_{ij}|}{\max(D)} \times 100\% \quad \text{for } i = 1 : N_g \text{ and } j = 1 : N_t \tag{4.1}
\]

The maximum non-dimensionalized velocity magnitudes for the deswirler and rotor sets were 1.23 and 1.95, respectively. The root mean square (RMS) of the reconstruction error is combined over the whole data set while the maximum reconstruction error is the highest local error in the ROR for all the snapshots.

Deswirler

The deswirler blade passage is just upstream of the rotor stage and downstream of the swirler blade passage with the axial flow going from the left to the right. The first nine POD modes (since they are ranked by percent energy) of velocity magnitude are portrayed in Figure 4.3. The modal values are not shown in the figure since they do not correspond directly to actual flow values by themselves. However, the relative magnitudes of the modal values depict specific flow structures that define the variation in the flow.

The percentage of the kinetic energy encompassed by these modes is just under 95 percent with the first mode characterizing 77 percent alone. Figure 4.3(a) displays the mean flow mode and shows the effect of the wake from the upstream swirler blade through the middle of the deswirler passage. The bow shocks traveling upstream from the rotor blades, however, are averaged out of the first mode. The second and third modes in Figures 4.3(b,c) depict the largest shock coherent
structures and look very similar. On closer inspection they show the shock structures spatially out of phase. The expansion coefficients charted against the time step in Figure 4.4 confirm this phase shift relationship. The phase shift, which is roughly 4 snapshots or $3.27 \times 10^{-5}$ seconds, allows the two modes to work together to define, roughly, the shockwave propagation upstream through the channel over time. Mode four (Figure 4.3(d)) shows an additional coherent structure, namely the separated flow region at the trailing edge of the deswirler blade. The POD modes return to characterizing the shock propagation with modes five and six (Figure 4.3(e,f)). However, they represent a frequency about twice that of modes two and three and have a reduced coefficient amplitude. Trailing wake and shock interactions seem to encompass the modes in Figures 4.3(g,h). The majority of the remaining modes display mixed characteristics as different flow phenomena combine and interact.

Four snapshots (40, 44, 48, and 52) of the deswirler passage velocity magnitudes, along with the nine-mode ROR and error contours, are shown in Figure 4.5. It is clear that the nine modes are unable to capture the sharp velocity changes present in the original data set. It is also unable to reconstruct some of the subtle gradients in the shock reflections further upstream. Within the shock and wake regions, the error ranged between 10-15% while the remainder of the passage was below 3%. Despite these shortcomings, the ROR gives a good general idea of the various flow dynamics of the system. While the shock fronts are definitely softened in the reconstructions, their propagation and position along the passage is reproduced relatively well. This can be seen especially in the error plots in Figures 4.5(b,d,f,h). Notice that there is a line of near zero error along the shock front showing that at least a portion of the shock is defined very accurately by the modes available. A large majority of the error is also centered around the top of the deswirler passage nearest to the blade where there are a lot of small variations at that boundary that are not resolved by the reconstruction. However, the general structure of the wake from the deswirler blade is still preserved. While the reconstructions lack accuracy in terms of specific velocity values in some areas of the domain, they still give an accurate portrayal of the main flow structures in the passage.
Figure 4.3: The first nine POD modes from the deswirler blade passage.
Figure 4.4: The expansion coefficients as a function of the time step for the second and third POD modes of the deswirler data set.

**Rotor**

The rotor blade is located just downstream of the deswirler and is rotating about the axis center at 16,404.7 rpm. The leading edge of the rotor is at the lower left portion of the domain. Figure 4.6 shows the first nine POD modes using the absolute velocity magnitude. The nine modes compose about 96.5 percent of the total energy of the system. The first mode once again represents the mean flow field and contains 93 percent of the energy alone. There is a general acceleration of the flow field in the axial direction. The highest velocities are observed on the suction side of the rotor blade. There is also a line of low velocity at the leading edge of the rotor due to the bow shock. The large vortical coherent structures are depicted in the trailing edge region of the second and third POD modes (Figures 4.6(b,c)). Like the second and third modes from the deswirler blade, these two basis vectors are out of phase from one another but characterize similar amounts of energy for the reconstruction. One unique structure in the second mode is the expansion wave located right off the leading edge of the rotor blade in the vertical direction. Smaller-scale vortical structures are also represented in Figures 4.6(d-f,i). The bow shock structure is most prevalent in mode seven (Figure 4.6(g)), but can also be seen in the fourth and sixth modes as well.

It is significant that the expansion wave variation and large vortical structures occur in the same modes. This suggests that the two turbulent structures are possibly dependent on the same dominant frequency inherent in the flow. Both structures occur in a mode that fluctuates at the same rate as the rotor blade-passing frequency. Every time the rotor blade passes through the wake of the upstream deswirler blade, the flow around the rotor blade is greatly affected, represented
Figure 4.5: Deswirler passage ROR of non-dimensionalized velocity snapshots 40, 44, 48 and 52 using the first 9 POD modes. In a, c, e, and g, the top profile is the original data set and the bottom profile is the ROR. The contours in b, d, f and h show the percent error from the respective snapshot ROR.
Figure 4.6: The first nine POD modes from the rotor blade passage.
by the shedding of the large vortical structures shown in modes two and three of Figure 4.6. This deduction can be made since the modal coefficients for these two modes only contain one dominant frequency similar to the signals seen in Figure 4.4 for the deswirler blade.

Reconstructions of the velocity magnitude in the rotor passage for snapshots 72, 76, and 80 using the nine POD modes are given in Figure 4.7(a,c,e). The biggest differences between the CFD solution (left side) and reconstructions (right side) are observed in the wake region of the rotor blade. This is to be expected due to the dynamic nature of the flow field at the trailing edge. A large portion of the error (Figures 4.7(b,d,f)) is also concentrated in the same region and generally ranges between 3-10% error. Midchord along the suction side of the blade, there is also a small section of higher error. As with the deswirler, the significance of these reconstructions is not necessarily with the individual velocity values, but rather with the relative values between neighboring domain locations to show the generalized flow field contours. The most distinguishable vortical features of the original flow fields are observable in the reconstructions as well. The ROR enables a reproduction of the flow field isolating only the most energy dominant modes leading to valuable analysis of the dynamic nature of critical coherent structures present in the flow. They also provide a convenient way to reduce the amount of data required to store the CFD solution.

4.2.2 Domain Partitioned ROR

The rotor blade domain was analyzed for the effect that partitioning has on reconstructing the flow field. In this case, the number of modes used in each domain partitioned ROR was the same as that used in the full domain ROR. The domain was partitioned based on the number of grid points alone with no other considerations. The number of divisions ranged from one to eight for both the rows and columns of the computational domain. The highest number of divisions observed for ROR accuracy was 64 from the $8 \times 8$ example (actual partitions of domain shown in Figure 3.4).

The domain partitioned RORs were observed for accuracy using both the RMS and maximum error. Both error quantifications were drastically improved by this technique allowing the POD to define modes specific to each partition. Therefore, the nine modes for the partitioned regions contain variation detail that was not able to be extracted from the unpartitioned data set in the same number of modes. Figure 4.8 charts the RMS and maximum percent ROR error based on the
Figure 4.7: Rotor passage ROR of non-dimensionalized velocity snapshots 72, 76, and 80 using the first 9 POD modes. In a, c, and e, the left profile is the original data set and the right profile is the ROR. The contours in b, d, and f show the percent error from the respective snapshot ROR.
total number of domain partitions. The whole domain ROR with nine modes resulted in an RMS error of 1.74% and maximum error of 33.01%. Compared to the full domain (1 × 1) ROR, the 8 × 8 case reduced the RMS and maximum errors by 63 and 50 percent, respectively. In contrast, to get the same RMS error reduction for a full domain ROR would require 59 modes, over six times the number of modes used in this analysis.

One interesting thing to notice from Figure 4.8 is that for the same number of partitions, the amount of error in the reconstruction can change. This is most likely due to how the domain is partitioned, what the individual pieces are shaped like, and which flow features each piece encompasses. Even though the domain has been partitioned and error reduced, the method for partitioning was not very elegant. Further investigation for intelligent domain partitioning may show increased accuracy and efficiency for the ROR.

Having observed in general the effects of domain partitioning on the ROR error, we turn to the specific example of the 8 × 8 reconstruction. Figure 4.9(a) shows the reconstruction of snapshot 84 with the original profile on the left and the reconstructed profile on the right. The quality of the reconstruction has been significantly increased when compared with those in Figure 4.7. Many of the detailed features of the flow field not captured with the full domain reconstructions are
now resolved with the partitioned ROR. The ROR captures not only the incoming wake from the deswirler blade but also the smaller vortical structures being shed from the rotor blade. The error contours in Figure 4.9(b) generally depict much better ROR accuracy over the domain than the unpartitioned reconstructions. This snapshot’s error is reduced by about 67 percent compared to the unpartitioned ROR snapshot.

One subtle effect resulting from partitioning the domain is that, in the reconstructed contours, there are slight artifacts that make the partition edges apparent. This is due to the fact that each piece of the domain is processed completely independently. Therefore, there may be small imposed discontinuities between neighboring partitions when the domain is reassembled. A possible solution to this issue may be to neglect the reconstructed values for the edges of each piece and instead use an interpolation method between the neighboring pieces to produce a smoother gradient between partitions.

4.2.3 JPOD Compression

The JPOD compression method was applied to the CFD solution of the rotor blade passage. It was also adapted to be able to handle the whole domain as well as the partitioned domain in order to observe trends in compression ratio as well as accuracy. The number of modes used in these results was once again chosen to be nine and the bit length used in the JPEG compression was twelve. Table 4.1 presents the JPOD compression results for $1 \times 1$ (unpartitioned), $4 \times 4$, and $8 \times 8$ domain partitioned cases. The compression was carried out using a computer with a dual-core Intel processor and 8 GB of memory.

As the number of partitions increased, the compression ratio tended to decrease. This is a caused by the requirement that each piece of a partitioned domain has its own expansion

<table>
<thead>
<tr>
<th>Partitions</th>
<th>Total Files</th>
<th>Size (KB)</th>
<th>Ratio</th>
<th>Error - RMS</th>
<th>Error - Max</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1 \times 1$</td>
<td>12</td>
<td>130</td>
<td>574:1</td>
<td>1.74%</td>
<td>33.05%</td>
<td>1.384</td>
</tr>
<tr>
<td>$4 \times 4$</td>
<td>177</td>
<td>315</td>
<td>237:1</td>
<td>1.08%</td>
<td>20.17%</td>
<td>2.764</td>
</tr>
<tr>
<td>$8 \times 8$</td>
<td>705</td>
<td>710</td>
<td>105:1</td>
<td>0.63%</td>
<td>16.35%</td>
<td>7.522</td>
</tr>
</tbody>
</table>
The 8 × 8 domain partitioned ROR of snapshot 84 of the rotor passage flow field. In a, the left contour plot is the CFD solution and the right is the ROR. The percent error contours are shown in b.

The RMS error of the ROR for all of the cases is the same as it was without the JPEG step addition. The maximum errors slightly increase for the 1 × 1 and 4 × 4 examples, but actually decrease for the 8 × 8 domain with the JPEG compression. The decrease in error is likely a random occurrence where the lossy JPEG compression algorithm happened to slightly correct the ROR value. This occurrence should not be considered a characteristic result for the JPOD method. Therefore, observing that the additional error from the JPEG compression is relatively minor or not present at all, we confirm that the error resulting from adding the JPEG compression step is negligible.

Table 4.1 also gives the amount of time required to perform these compression operations. As the number of partitions increased, the time required for compression also increased. However, because each piece is treated and processed independently, this issue may easily be solved using parallel computing.
4.3 Conclusions and Recommendations

Simulations of the Blade-Row Interaction (BRI) test rig were analyzed using the POD to identify turbulent coherent structures in the deswirler and rotor blade passages. POD modes were extracted from sets of two-dimensional snapshots of absolute velocity magnitude within each passage. Pairs of modes at similar frequencies but out of phase structures and coefficients characterized traversing shocks in the deswirler passage. Additional modes describing traversing shocks increased in both spatial and temporal frequency in order to capture the sharp gradients in the deswirler passage. Large rotor blade vortical structures were found to correlate with blade-passing frequency by comparing fluctuating structures within the modes.

Reduced order reconstructions (ROR) were created using the first nine modes from both passage data sets. While the ROR recreated the general instantaneous flow field well, shock fronts and vortices, especially smaller ones, were much less defined and there was higher error located in these regions of the flow. Domain partitioning was applied to the rotor passage to quantify the error reduction while using the same number of POD modes. Error was reduced by 63 percent when the domain was partitioned into 64 individual pieces. The smaller domain pieces allowed better definition of the smaller vortical structures shed from the rotor blade.

Finally, using the new combinatorial compression method, compression ratios for the nine-mode RORs topped out at 574:1 for the unpartitioned BRI rotor passage. The domain partitioned RORs had less compression ratio due to the storage of extra coefficients. Taking advantage of the smooth gradients of the POD modes and the structured domain grid, the JPOD compression method performed very well while adding negligible error to the RORs. Further work should be done to apply the JPOD method to three-dimensional data sets and different field scalars of the flow. This would provide better validation of the method and confirm its usefulness for longterm storage of CFD solutions.
CHAPTER 5. FULL ANNULUS MULTI-STAGE FAN WITH INLET PRESSURE DISTORTION

Currently, research is being performed using computational fluid dynamics (CFD) to study inlet pressure distortion on turbomachinery. Inlet distortion is not a trivial matter and can have a drastic effect on the performance of gas turbine engines. The distorted inlet conditions may originate from high angle of attack flight maneuvers, ingestion of weapon exhaust, and inlet design. Highly complex simulations can give a clearer understanding of how pressure distortion can affect flow dynamics within turbomachinery. Having these solutions available does not necessarily make the characterization of these effects easy to extract or analyze. Using the proper orthogonal decomposition (POD), dominant modes can be extracted that define the flow field dependence on the inlet pressure distortion.

5.1 Computational Method

For a more in depth discussion of the flow solver and computational approach for this analysis, the reader is directed to the work by Marshall et al. [30]. However, a brief overview of this particular simulation will be provided here. The flow solver used to solve the time-marching, three-dimensional Navier-Stokes equations is a modified version of OVERFLOW 2.2 [31–33], a NASA CFD code with origins dating back to the late 70’s. The full-annulus computational domain shown in Figure 5.1 consisted of 7 total blade rows: 1 inlet guide vane row, 3 rotor rows, and 3 stator rows. The number of nodes in the entire domain totaled 660 million. Overset meshing techniques between stationary and revolving blade rows were used. The simulation was fully unsteady, and a full revolution of the rotor blades required 7000 time steps. Being a time-accurate simulation, convergence was defined when the mass flow (measured at various axial locations) settled to a stable average value. The simulation was performed on supercomputers at both Brigham Young University’s Fulton Supercomputing Laboratory (FSL) and the Navy Department of De-
fense Supercomputing Resource Center (DSRC). At the Navy DSRC one rotor revolution required 33 hours running on 1216 processors. In comparison, the same number of iterations required around 70 hours using the FSL for the same number of processors.

The inlet boundary condition for the turbomachinery simulation was a total pressure distortion profile depicted in Figure 5.2. It shows a one-per-revolution sinusoidal pattern where the amplitude of variation is about 20% of the mean total pressure. A uniform total temperature was also specified at the inlet boundary. The exit boundary condition maintained the average static pressure at the exit surface to be equal to the inlet average total pressure. In order to throttle the mass flow through the fan, a nozzle was placed before the exit to choke the flow. By changing the nozzle area reduction, different operating points along a speed line could be simulated.

From the mass flow convergence plot in Figure 5.3, it is shown that the current simulation converged after approximately 39,000 time steps, or about five and a half rotor revolutions. The solution for this operating point was initialized from a converged solution at a nearby condition. This is why there are large deviations for mass flow at the beginning of the chart. Notice that even after the mass flow rates at each measurement location seem to settle to an average value, the rates still continue to fluctuate periodically. This solution represents a near-stall operating condition which can have increased fluctuations in mass flow rate.

5.2 POD Description

The POD is used on two fundamentally different data sets in order to analyze the effect of a one-per-revolution inlet total pressure distortion on the flow characteristics in turbomachinery. Both data sets contain non-dimensionalized static pressure values for two-dimensional computational surfaces within the third rotor blade cascade in a three-stage fan. The computational surfaces are located at about thirty percent immersion for both data sets. Immersion is defined as the distance of the computational surface from the outer casing divided by the total radial thickness of the blade. The specific grid dimensions for each computational surface used in the data sets are 74 in the circumferential direction and 169 in the axial direction.

The first data set utilizes only a single rotor blade passage and contains the pressure values at equally spaced time step intervals as it travels one full rotation around the annulus of the fan once the simulation has converged. This corresponds to the method of snapshots as described in
Figure 5.1: The entire computational domain consisting of seven regions and 660 million nodes

Figure 5.2: The inlet total pressure boundary condition giving a one-per-revolution distortion pattern
Figure 5.3: Mass flow convergence plot

Chapter 3. As the single passage revolves about the axis of the fan, the flow field changes due to a combination of both the inlet pressure distortion that has been transferred through fan and the turbulent flow dynamics from upstream and downstream blade rows.

The second data set is formed by using only a single instantaneous snapshot of the full annulus from the converged solution. Because the grid for each rotor passage in the stage is the same size and shape, geometrically identical computational surfaces (only offset by the pitch angle between each blade) can be extracted from the entire circumference of the fan. This data collection technique will be referred to as the method of repeating geometry (RG) and has not been specifically documented in the literature to the author’s knowledge. The main purpose of this technique is to isolate the inlet distortion-dependent POD modes first and foremost. While there are other flow dynamics relating to blade passing and wake interactions, these will be downplayed by the method because all the computational surfaces come from different passages. Therefore, using a combination of computational extracts or two-dimensional surfaces from around the rotor cascade from a single time step of the simulation, the effects of the pressure distortion on a rotor blade can be extracted. An important assumption here is that the effect of the pressure distortion on the flow field is relatively consistent through time since only one moment is actually analyzed.

There are number of benefits to using the RG method in place of the method of snapshots. The most notable benefit is the computational savings. In order to use the snapshot method, the simulation must solve 7000 additional time steps to gather the effect of the full-annulus pressure.
distortion after convergence, while the RG method does not. Additionally, the RG method may be used to analyze the effect of pressure distortion on stationary stator or IGV blade rows. The snapshot method would be unable to produce a similar analysis because the stationary passages only are affected by a specific portion of the pressure distortion. Another benefit of using the RG method is that no additional data must be output or stored because the data needed is readily available at each time step. This means that the method may be used at any point in a simulation to monitor and study flow dynamics. In order to identify possible shortcomings of the RG method, the POD modes generated from both methods are compared and contrasted to determine whether the results are of similar value. To make an equivalent comparison between the methods, the number of extracts used for the snapshot method is constrained to be the number of blades in the third rotor blade passage. Increasing the number of snapshots may produce additional useful information for analysis (see Section 5.4).

5.3 Analysis and Discussion

The first four POD modes from both the snapshot and RG data sets were extracted using techniques presented in Chapter 3. It is visually evident that there is excellent agreement between the POD modes for the two data sets, especially when comparing the first two modes from each. Figure 5.4 shows mode one for the snapshot and RG POD along with their normalized coefficients. In each figure representing the passage, it is important to note that the axial flow is left to right, and the rotor is moving upwards relative to the page.

The first mode shows the mean pressure POD mode and is characterized by a standing bow shock off of the leading edge of the rotor blade, denoted by the sharp change in static pressure. The mean POD mode gives a general depiction of the pressure structures present throughout the entire data set. Upon closer inspection, it appears that the shock structure portrayed in the RG POD mode in Figure 5.4(b) is more angled downstream than the shock in the Snapshot POD mode in Figure 5.4(a). This difference in shock structure may be due to differences in flow incidence or incoming velocity between the two data sets. The average static pressure in the RG data set is actually about 1.5% less than the average static pressure of the snapshot data set, providing evidence of velocity difference. Because the snapshot data has been obtained over time, unsteady fluctuations in mass flow are averaged out for a single passage. The RG data set is dependent only
Figure 5.4: The first POD mode from both the (a) Snapshot and (b) Repeated Geometry methods and their relative (c) normalized coefficients

upon the flow condition at a single point in time in the simulation. Therefore, the RG method represents a pseudo-unsteadiness because it does not account for the actual variations which occur through time. It is likely that this unsteadiness in mass flow rate over the simulation time is what causes this difference in the shape of the flow structures for the two POD methods.

The mode one coefficients in Figure 5.4(c) for both data sets show a very dominant one-per-revolution sinusoidal pattern with about a 6.9% deviation amplitude from the mean value (compared with the inlet deviation amplitude of 20%). This is obviously a direct result of the pressure
distortion inlet boundary condition that has been transferred through the first five blade rows of the fan. Also present in the mode one coefficients is a lower-magnitude higher-frequency signal. This component of the variation is most likely due to the rotor blades passing through the wakes from the upstream stator cascade. While the coefficients for the two POD methods do not match exactly, similar characteristics of the coefficients are maintained.

In Figure 5.5, the second POD mode is shown for both the snapshot and RG methods. While the mean POD mode in Figure 5.4 gives a clear general depiction of the pressure distribution within the passage, the second, and all other remaining modes, describe the pressure structures that show how the data set varies from the mean. Also, the values shown in the mode are less important than the structures they represent because they will vary depending on the magnitude and sign of the coefficients. Therefore, when the coefficient is near zero at a certain time or angle, the mode contributes little to no information to the original data set at that point. In the second mode, the main pressure structure is located at the bow shock. This specific structure will be discussed later in this section once the other modes have been shown. There is an additional structure on the pressure side of the rotor blade slightly downstream of the leading edge. While both POD methods capture the same structures in the second mode, the RG mode in Figure 5.5(b) shows less definition in the contours near the shock giving softer representation of the structure than the snapshot mode in Figure 5.4(a). The coefficients in Figure 5.5(c) once again show a dominant one-per-revolution signal with a weaker high-frequency overset. Using the same reasoning as with the first POD mode, these variations can be attributed to the pressure distortion and neighboring blade row interactions, respectively.

While the first two POD modes methods are matched in order within each method, modes three and four require a little more care to correlate between the two methods. Recall from Chapter 3 that the order of the modes depends on the magnitude of the relative singular value and that the singular value represents the magnitude of the projection of the mode onto the original data set. As a result, mode three of the snapshot POD most closely matches mode four of the RG POD, and the reverse is true as well. This is confirmed primarily by comparing the modal coefficients and structures and then determining qualitatively which coefficients matched the best.

Mode 3 from the Snapshot method and Mode 4 from the RG method are first compared in Figure 5.6. From the coefficient magnitude chart in Figure 5.6(c), it is clear that the two signals
Figure 5.5: The second POD mode from both the (a) Snapshot and (b) Repeated Geometry methods and their relative (c) normalized coefficients

share very similar characteristics. Not only is there a dominant low-frequency component, but there is also a high-frequency component with similar magnitude as the low-frequency signal. From the frequency analysis in Figure 5.7, which is coarse due to the small number of data sets, it is clear that there are multiple dominant frequencies other than the one-per-revolution expected from the pressure distortion. With the coarseness of the data set, it is difficult to point out specific causes for the higher-frequency fluctuations because they can not be fully defined due to Nyquist frequency limitations. This is one of the weaknesses of the RG method because it is limited by the
number of blade passages within the row being analyzed. The snapshot method does not have this limitation for the rotor blade row.

The bow shock pressure structure shown in Figures 5.6(a,b) is similar to the second mode, but the remainder of the passage shows much more variation than with the previous modes. As the flow proceeds through the rotor passage, there are points of fluctuating high and low pressure regions. The upstream wakes flowing periodically into the blade passage can explain this alternating pressure feature. It is clear that the snapshot POD mode Figure 5.6(a) contains more defined
high and low pressure regions than the RG mode in Figure 5.6(b), especially near the end of the passage. This is especially true near the trailing edge of the blade where there seems to be a varying wake structure more clearly depicted in Figure 5.6(a). This differences between the methods may be due to how the data sets were obtained. The snapshot data set used flow extracts from the same blade passage, increasing the coherence within the data set. The RG data set contains flow extracts from completely separate locations that are much less related in the sense that each location may contain independent flow dynamics unique to a particular blade passage. However, despite these differences, very similar information can be gathered from each method’s POD mode.

The final mode compared is snapshot POD mode four and RG POD mode three (see Figure 5.8). Once again, the coefficients are made up of a few dominant frequencies. However, unlike the previous modes, Figure 5.8(c) shows that the lower-frequency portion goes through two cycles per revolution instead of just one. At first glance, it is unclear why this type of signal is extracted using the POD. One explanation for this may be that the bow shock in the passage is not stationary, but moves upstream or downstream depending on the blade location relative to the distortion pressure profile. Because the shock is a discontinuity in the flow, its movement is very difficult to describe by POD modes. In Chapter 4, it is shown that traversing shocks are represented by sinusoidal structures that increase in both spatial and coefficient frequency as the POD modes decrease.
Figure 5.8: The (a) fourth POD mode from the Snapshot method, the (b) third POD mode from the Repeated Geometry method, and their relative (c) normalized coefficients in energy. In Figures 5.8(a,b) there are two alternate-sign pressure structures near the bow shock, whereas in Figures 5.4-5.6 there is only one pressure structure.

Figure 5.9 gives a closer look at the bow shock structures for the first three RG POD modes by charting modal values along a sample line across the shock. Each modal sample is scaled by its relative singular value so that magnitudes can be compared. Mode one in Figure 5.9(b) shows always positive and mostly increasing values representative of the mean condition. The second mode’s values vary about zero and show depict half of a sine wave across the shock region. The
third mode shows a full sine wave in the shock region. Therefore, the structures increase in spatial as well as coefficient frequency, confirming the shock movement idea. Additionally, the third RG POD mode could also represent fluctuations in the shock strength around the circumference of the fan due to the full sine wave shape.

After the first four POD modes, the two methods diverge and the lower-energy modes do not correlate very well between the methods. Despite this divergence, the RG method is still able to extract the most dominant POD modes related to the pressure distortion inlet condition. The effects of just the pressure distortion on the flow field can be further isolated by filtering out the high-frequency content of the modal coefficients and producing a reduced order reconstruction (ROR) of the original RG data set. The ROR was made using the first five RG POD modes only. Filtering was accomplished by fitting the coefficients to either a one-per-revolution (modes 1, 2, and 4) or two-per-revolution (modes 3 and 5) sinusoidal wave by minimizing the squared error. For example, in Figure 5.10 both the original normalized coefficients and the filtered coefficients are shown for the RG POD mode 5.

Eight circumferential extracts are presented from the five-mode ROR of the RG data set in Figure 5.11 using the filtered coefficients. It is very apparent that the non-dimensionalized static
Figure 5.10: The original and filtered coefficients from mode five of the RG POD

pressure contours change significantly around the circumference of the fan. Figure 5.11(d) show that the bow shock of the rotor blade is significantly detached from the leading edge. However, on the opposite side of the fan in Figure 5.11(h), the bow shock has almost been completely swallowed into the blade passage. This shows that the movement of the shock structure is primarily caused by the distorted inlet pressure to the fan since other high-frequency turbulent variations have been filtered out. Observing the actual RG data set may show similar trends, but using the RG POD the trends can be confirmed and quantified based on the filtered coefficients. While the same type of filtering and reconstruction can be done using the snapshot POD method, little additional information would be gained. Therefore, for isolating flow field changes due to pressure distortion, the RG method is more efficient than the snapshot method.

This technique has some similarity to the harmonic balance method for solving unsteady flows [34, 35], but it is in fact very different in application. In turbomachinery flows, the harmonic balance utilizes fundamental frequencies like blade passing or flutter vibration frequencies to fit the unsteady flow to a Fourier series instead of marching through time. This has been shown to be very effective for a variety of applications dominated by periodic fluctuations [36]. The POD, however, is not necessarily biased toward any frequency from the start because the modes are derived from optimizing the projection onto the original data set. If there are highly periodic elements of the unsteady simulation, the POD will extract those features as well. The benefit here is that we are not limited by a single fundamental frequency and its harmonics. Because of the complexity of this simulation (seven blade rows, pressure distortion, etc.), there are a variety of dominant frequencies, depending on the POD mode being observed. Just as the pressure distortion effect
Figure 5.11: A five-mode reconstruction of the RG non-dimensionalized static pressure data set using coefficients that have been fit to low-frequency sinusoidal signals.
is isolated from this time-marching simulation, it can also be neglected to focus on the remaining dominant frequencies of variation.

### 5.4 Extension of POD Methods

As mentioned in Section 5.2, the RG data set is limited by the number of passages in a given blade row while the snapshot data set is not. The snapshot method is only limited by the actual simulation time resolution, which in this case is 7000 time steps per rotor revolution. Increasing the resolution between extracts in the snapshot POD can produce very interesting and revealing results. For example, using 1000 snapshots through one revolution of a rotor passage, POD modes can be correlated directly with specific blade rows based on the dominant frequencies in the coefficients. Figure 5.12 shows a frequency analysis of the modal coefficients from one of the snapshot POD modes using this finer resolution. Increasing the sampling frequency allows the ability to distinctly identify and quantify the contributions of various sections of the fan to the variations within a given mode. The specific blade rows are identified based on the total number of blades in each row. While this particular mode contains a dominant pressure distortion component, the upstream and downstream blade rows also add significant variation. Identification of these components may help to determine causes of certain pressure structures in the mode.

In order to gain both the computational savings of the RG method as well as the resolution of the snapshot method, a hybrid technique can be used. For instance, if a rotor blade row has 20 passages and takes 7000 time steps to complete a full revolution, it only takes 350 time steps for a given passage to move to the initial position of the neighboring blade. Using the RG method of extracting computational surfaces for every passage at intervals of seven time steps, the 1000 total extracts can be obtained in a fraction of the time while simultaneously gathering data from the entire annulus. Because there are multiple rotor blade rows, the actual number of time steps would be dependent on the rotor row with the fewest number of blades.

While in the present study only the non-dimensionalized static pressure values have been analyzed using snapshot and RG POD, additional flow information can be gathered using the same process on other scalar or derived quantities within the flow domain. The repeating geometry method has been show to be a viable and efficient way to extract and isolate the variation of flow features in a rotor passage caused by pressure distortion. The snapshot method, while capable
of producing more defined POD modes, may provide more information than desired and require much more time in producing a suitable data set. Furthermore, the same type of pressure distortion analysis can be applied to stationary blade rows using the RG method. The snapshot POD method is unable to give pressure distortion analysis for stationary blade rows, but may still give excellent information about turbulent flow characteristics. Thus, the decision to apply one or the other method is fully dependent on the type of analysis desired by the researcher.
CHAPTER 6. REDUCED ORDER MODELING OF CONJUGATE HEAT TRANSFER SIMULATIONS

As mentioned in Chapter 2, ROM techniques have been used to model heat conduction and convection, but the conjugate problem is being explored for the first time here. The accuracy and computational efficiency of ROM simulations are demonstrated by considering both a steady and a transient conjugate heat transfer problem. The first case is the transient simulation of a cylinder heated in crossflow. The second is a steady-state case of a rotating turbine blade with internal cooling. In both cases, input parameter sets are the boundary conditions in the fluid regions (i.e. inlet velocity, inlet temperature, etc), and the scalar being modeled is the temperature in the solid portions of the computational domains. All the simulations were run using the commercial CFD code Star-CCM+ [37], in parallel on the Fulton Supercomputing Lab (FSL) at Brigham Young University in Provo, Utah.

6.1 Transient Conjugate Heat Transfer for Cylinder in Crossflow

In order to show the effectiveness of this ROM technique in a transient environment, a simple flow case of a cylinder in heated crossflow with bounding walls was selected (see Fig. 6.1). The working fluid was air and the cylinder was modeled using the physical properties of aluminum. The cylinder diameter was 2 cm and the length to diameter ratio was 4:1. The varied parameters for this case were inlet velocity and inlet temperature. Air properties were also determined based on the inlet temperature values. The cylinder end at the wall was kept at a constant temperature for all the simulations.

The structured grid used in the simulations is shown in Fig. 6.2 and consisted of two separate regions, the surrounding fluid \(1 \times 10^5\) cells) and the solid cylinder \(3.65 \times 10^5\) cells). As seen from the grid, only a quarter of the cylinder is modeled by using symmetry boundary conditions at the center and middle planes of the cylinder. The top surface boundary also has symmetry condi-
Figure 6.1: A diagram showing the basic setup for the cylinder in crossflow case.

...tion applied. No-slip conditions were specified at the wall and cylinder surfaces. Energy coupling was specified at the interface between the air and solid cylinder regions. A k-ε turbulence model was also used in the simulations. The fluid region initial temperature and velocity was set to the same value as the inlet.

A variable time step was used for the simulations to decrease computation time even further. Also, since the cylinder approaches a steady-state condition, the rate of heat transfer into the cylinder from the surrounding fluid decreases with time. The time step initially is set to 0.001 s, but by the end of the simulation increases by two orders of magnitude. Equation 6.1 shows the

Figure 6.2: This is the grid used for all of the cylinder heat transfer simulations; it consisted of a 100,000 cell cylinder region and a 365,000 cell fluid region
variation of the time step with respect to the simulation time.

$$\Delta t(t) = \begin{cases} 
0.001 \text{ s}, & t \leq 0.2 \text{ s} \\
0.005 \text{ s}, & 0.2 < t \leq 1 \text{ s} \\
0.01 \text{ s}, & 1 < t \leq 2 \text{ s} \\
0.025 \text{ s}, & 2 < t \leq 5 \text{ s} \\
0.05 \text{ s}, & 5 < t \leq 10 \text{ s} \\
0.1 \text{ s}, & 10 < t \leq 20 \text{ s} 
\end{cases} \quad (6.1)$$

The transient ROM for the cylinder in crossflow case was created using 20 total simulations in which both the inlet velocity and inlet temperature were varied. Each simulation used a specified initial temperature of 300 K for the cylinder region. The maximum temperature reached in the transient cylinder simulations ranged from 305.9 to 326.8 K and was very dependent on the magnitude of the parameter values. Figure 6.3 shows all of the parameter combination solutions for the database as well as those parameter sets that were used in evaluating the ROM. Notice that the database solutions are evenly spaced. This is not essential when creating a ROM, but it does help to give a balanced coverage to the parameter space. A total of 68 snapshots in time were used to created the transient ROM, and spacing between snapshots increased with simulation time.

The number of basis vectors for $\Phi$ at each snapshot in time was determined using 99.99% of the total energy. With this energy percentage, the number of modes per snapshot ranged from two to six. The snapshots requiring the most modes were those that were nearest to the beginning of the simulation. The ROM generated required only about 20.3% of the memory space used by the original database of solutions. However, this data reduction is only important if the ROM can accurately predict solutions at random parameter sets.

In order to determine the accuracy of the ROM, three random parameter sets were selected to compare the ROM prediction with the CFD solution. Table 6.1 presents the general results for the maximum and root mean square (RMS) error in Kelvin for the three test parameter sets as well as all of the database solutions. The maximum error is the largest temperature discrepancy between the ROM and CFD solutions over the entire grid and simulation time. The RMS error
Figure 6.3: The plot shows the parameter space spanned by the database of solutions; therefore, the ROM should only be seen as valid for parameter values that fall within this space.

takes into account all temperature differences throughout the grid and simulation time and is a measure of the average error in the ROM predictions. Compared to the total temperature change in each simulation, the ROM error is very minimal. Both the maximum and RMS errors tend to increase with increasing inlet temperature and velocity due to higher heat transfer rates in those conditions. The ROM shows very good results overall when compared with the CFD database solutions, which is expected.

Looking more closely at the \{470.6 K, 29.5 m/s\} parameter set CFD and ROM solutions we can get a better idea of how well the temperature profiles in the cylinder are predicted. We first

Table 6.1: Comparisons between the ROM and CFD solutions for the transient heating of a cylinder in crossflow

<table>
<thead>
<tr>
<th>Sample Solution</th>
<th>Max ROM Error (K)</th>
<th>RMS ROM Error (K)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T = 470.6 K and V = 29.5 m/s</td>
<td>0.111</td>
<td>0.026</td>
</tr>
<tr>
<td>T = 441.9 K and V = 38.9 m/s</td>
<td>0.019</td>
<td>0.004</td>
</tr>
<tr>
<td>T = 411.4 K and V = 73.1 m/s</td>
<td>0.058</td>
<td>0.013</td>
</tr>
<tr>
<td>All Database Solutions</td>
<td>0.089</td>
<td>0.010</td>
</tr>
</tbody>
</table>
look at the accuracy of the ROM for the temporal development of the temperature in the cylinder. The graph in Fig. 6.4 shows a comparison between maximum and average cylinder temperatures in the CFD and ROM solutions through time. The two solutions match very well. While it is evident that near end of the solution time the ROM slightly underpredicts the maximum temperature, the deviation is inconsequential and does not appreciably degrade the overall ROM prediction.

Finally, to observe the temperature profiles themselves, we focus on the ROM and CFD solutions at a specific instant in time. Figure 6.5 depicts both the CFD and ROM solutions at two seconds into the simulation as well as the absolute temperature error. It is very difficult to discern any significant differences between the CFD and its ROM prediction. The temperature contours are almost identical, showing excellent accuracy. The error contours show that the maximum error in the prediction occurs at the corner of the two symmetry planes, or the front middle of the cylinder. This is also the position where the maximum temperature occurs as well.

These results show that transient temperature profiles can be very accurately predicted for a cylinder in crossflow in which the heat transfer is solved conjugately with the fluid flow. The computational efficiency of the ROM is significant. When run on 16 processors, each CFD simulation required approximately 30 minutes to complete. Simulations obtained using the ROM required less than one second to compute.

### 6.2 Steady-State Conjugate Heat Transfer for an Internally Cooled Turbine Blade

Each boundary condition was varied between four options shown in Eqs. 6.2, 6.3, 6.4. Therefore, 64 simulations the cooled turbine were run to include all possible parameter combinations with these values.

\[
T_{primary\ inlet} = [600, 800, 1000, 1200] K \tag{6.2}
\]

\[
\left(\frac{P_T}{P}\right)_{primary\ inlet} = [1.1, 1.15, 1.20, 1.25] \tag{6.3}
\]

\[
V_{cooling} = [5, 10, 15, 20] m/s \tag{6.4}
\]

The cooling fluid was kept at a constant inlet temperature of 285 K. The temperature profiles in the turbine blade showed much variation with the different parameter sets. This is evident...
Figure 6.4: Comparison of the CFD and ROM maximum and mean cylinder temperatures for the random parameter set of 470.6 K and 29.5 m/s.

from Fig. 6.6 which shows the range of temperatures that occurs in each of the 64 simulations run to produce the ROM database. The largest changes in temperature are caused by the variation in the primary fluid inlet temperature and the cooling fluid inlet flow velocity.

For the steady-state ROM basis vectors, the percent energy was chosen to be 99.99%. A total of 16 out of the possible 64 basis vectors were used according this energy percentage requirement. The ROM for the cooled turbine case only required 25% of the memory taken up by the original database CFD solutions. Five parameter sets were used to evaluate the accuracy of the temperature profiles predicted by the ROM (see Table 6.2). The parameters were chosen to give solutions from a variety of locations within the parameter space.

Table 6.2: Parameter sets used to compare CFD and ROM solutions

<table>
<thead>
<tr>
<th>Set</th>
<th>T (K)</th>
<th>V (m/s)</th>
<th>P_T/P</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1140</td>
<td>16.53</td>
<td>1.187</td>
</tr>
<tr>
<td>2</td>
<td>721</td>
<td>7.23</td>
<td>1.230</td>
</tr>
<tr>
<td>3</td>
<td>1065</td>
<td>5.72</td>
<td>1.216</td>
</tr>
<tr>
<td>4</td>
<td>980</td>
<td>9.34</td>
<td>1.163</td>
</tr>
<tr>
<td>5</td>
<td>865</td>
<td>12.45</td>
<td>1.131</td>
</tr>
</tbody>
</table>
Figure 6.5: Sample solution at T = 470.6 K and V = 29.5 m/s temperature contours for the (a) CFD and (b) ROM solutions at t = 2 s along with respective (c) error contours. The highest ROM error occurs in the front of the cylinder at the symmetry plane.

Figure 6.6: The variation in maximum and minimum temperatures in the cooled turbine blade for each of the CFD simulations.
As in the transient case, the steady-state ROM temperature error was calculated. The ROM was able to generate temperature profiles in a few seconds, while the CFD simulations required over 11 hours to reach convergence when run using 32 processors on the FSL. In Table 6.3, the maximum and RMS temperature error from the ROM predictions are presented. It is difficult to identify clearly any trends in the ROM error based on this small selection of examples, but there are some interesting points nonetheless. The most apparent is that at higher cooling fluid inlet velocities there are higher temperature prediction errors. Even when the primary fluid inlet temperature is high like in Set 3, the low cooling fluid velocity seems to be the dominant driver in prediction error.

Again, direct comparison of the temperature profiles generated using the ROM with the corresponding CFD simulation demonstrates the accuracy of the ROM. Temperature profiles for the 4th set of input parameters are shown in Fig. 6.7(a,b). A mapping of the absolute value of the difference between these temperature profiles is also shown in Fig. 6.7(c). Note that the shapes of the contours are very similar, and that the RMS temperature error is less than 2.5%. For heat transfer applications where there is already much variability and error inherent in many correlations, this level of error is very acceptable for the ROM. The largest errors occur near the center of the suction surface of the turbine blade, where the irregular shape of the internal cooling channel results in complex flow and heat transfer patterns. The base of the turbine blade shows error values very similar to the RMS error. It is anticipated that increasing the resolution in the database would

<table>
<thead>
<tr>
<th>Sample Solution</th>
<th>Max ROM Error (K)</th>
<th>RMS ROM Error (K)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set 1</td>
<td>21.16</td>
<td>12.59</td>
</tr>
<tr>
<td>Set 2</td>
<td>14.49</td>
<td>5.15</td>
</tr>
<tr>
<td>Set 3</td>
<td>16.10</td>
<td>7.01</td>
</tr>
<tr>
<td>Set 4</td>
<td>9.44</td>
<td>5.70</td>
</tr>
<tr>
<td>Set 5</td>
<td>10.78</td>
<td>2.29</td>
</tr>
<tr>
<td>Database Solutions</td>
<td>6.35</td>
<td>1.09</td>
</tr>
</tbody>
</table>
Figure 6.7: Temperature contours for the (a) CFD and (b) ROM sample solution using Set 4 parameters. The error contours are also shown in (c).

increase the accuracy of the ROM, and, as mentioned previously, an investigation of the size of the database needed to create a ROM with a specified level of accuracy is ongoing.
CHAPTER 7. REDUCED ORDER MODELING OF FLOW OVER AN AIRFOIL

This chapter describes the generation, results, and evaluation of a reduced order model (ROM) for steady-state computational fluid dynamics (CFD) simulations of flow over an airfoil. The ROM presented here spans two independent flow parameters: angle of attack and inlet Mach number. While the solid temperature distribution was modeled in Chapter 6, this chapter uses the local Mach number as the modeled scalar. The results presented here show excellent modeling of the flow field where viscous effects are less important. Locations in the computational domain where molecular viscosity dominates (boundary layers, flow separation regions, wakes) are modeled less accurately. These areas also produced modeling instabilities that are magnified by the ROM database resolution of the chosen parameter spaces.

7.1 Case Description

The specific case studied here is the subsonic flow of air over an RAE2822 airfoil. The Reynolds numbers for this flow range from $5.52 \times 10^5$ to $6.62 \times 10^5$ depending on the inlet flow Mach number and the airfoil geometry. The chord length and aspect ratio for this airfoil were 0.10 meters and one, respectively. Star-CCM+ [37], a commercial CFD suite, was used to solve the coupled, steady-state, three-dimensional Reynolds-Averaged Navier-Stokes governing equations. The Spalart-Allmaras turbulence model was also used and the $y^+$ values for all the database simulations was under five.

A graphical depiction of the computational grid used for the simulations, which consisted of about 2.18 million cells, is shown in Figure 7.1. At the inlet a free stream boundary at one chord upstream was specified to change the Mach number and angle of attack for the airfoil flow. Periodic boundary conditions were applied at the domain surfaces above and below the airfoil effectively simulating a cascade of airfoils with a pitch length of 1.5 chords. The pressure outlet boundary
Figure 7.1: The computational grid used to perform the CFD simulations was located three chords downstream of the airfoil. The boundary located at far end of the airfoil span was a no-slip wall, while the near end boundary utilized a symmetry condition.

Each database simulation was run to 4000 iterations in order to converge lift and drag coefficients as well as flow residuals. The Courant number used for the coupled solver was 5. All of the simulations were performed using the Fulton Supercomputing Lab (FSL) at BYU. Using 16 processors on the FSL, each simulation required about 3 hours to be completed.

7.2 ROM Development

The scalar of interest for this ROM was the local Mach number at the center of each cell in the computational domain. The parameters varied were the inlet Mach number and the angle of attack relative to the airfoil. The angle of attack modestly ranged from $-2.0^\circ$ to $7.5^\circ$ with a resolution of $0.5^\circ$ between database simulations. Inlet Mach number varied between $0.25$ and $0.30$ at an interval of $0.01$. In all, 120 CFD simulations were performed to generate the database from which the ROM could be created.

In Chapter 3, the ROM method mentions that there are two model criteria that the user may choose that affects both the accuracy and data required by the ROM. The first criterion is the percent energy for the proper orthogonal decomposition (POD) modes of the database matrix. Recall that the POD modes specify how the data set varies from simulation to simulation. In gen-
Table 7.1: The number of modes required for each percent energy specification for the ROM

<table>
<thead>
<tr>
<th>Percent Energy</th>
<th>Number of Modes</th>
<th>Percent Energy per Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>4</td>
<td>20.0</td>
</tr>
<tr>
<td>85</td>
<td>6</td>
<td>14.2</td>
</tr>
<tr>
<td>90</td>
<td>9</td>
<td>10.0</td>
</tr>
<tr>
<td>95</td>
<td>19</td>
<td>5.0</td>
</tr>
<tr>
<td>97.5</td>
<td>37</td>
<td>2.6</td>
</tr>
<tr>
<td>99</td>
<td>65</td>
<td>1.5</td>
</tr>
<tr>
<td>99.9</td>
<td>108</td>
<td>0.9</td>
</tr>
</tbody>
</table>

eral, the greater the number of modes, the greater the ability of the ROM to describe lower-energy fluctuations between the simulations. These lower energy flow structures are spatially smaller (like a flow separation bubble or wake-freestream boundary) but may contain important information in describing the data set dynamics. Table 7.1 shows how the number of POD modes is dependent on the percent energy choice. It is expected that the energy contribution per mode decreases greatly as more percent energy is specified.

The second criterion that must be chosen is a cutoff singular value for the interpolation matrix formed from the input parameters used in the database. Recall from Chapter 3 that this cutoff value must be applied in order to form the pseudo-inverse using the singular value decomposition (SVD). Figure 7.2 shows a plot of all of the singular values of the interpolation matrix for the airfoil parameter sets. It is difficult to determine from the plot itself what value should be chosen, although it is apparent that the matrix is indeed singular. This is signified by the portion of the plot.
Figure 7.2 where the singular values level off at an extremely low value from mode 84 and onward.

The method presented here to determine the cutoff value is to generate a variety of ROMs, each with a different percent energy and cutoff singular value. Then, using a set of trial input parameter combinations (see Section 7.3 for specific values), the ROM error is quantified by comparing the ROM-generated and CFD-generated solutions at those trial parameter sets. Figure 7.3 shows a chart of all of the parameter combinations used for the database and the trial combinations that helped to determine cutoff and energy values.

A contour plot depicting the changes in the root mean square (RMS) of the absolute Mach number over all the sample trial sets is shown in Figure 7.4. Notice that there is an obvious global minimum RMS error at 90% energy and a cutoff value of $10^{-10}$. This means that only 9 out of the 120 available POD modes are needed for the ROM, an over 92% reduction in data. What is interesting is that there is a general increase in error as the percent energy is increased. That error seems to also be magnified for lower cutoff values. This may be due to the fact that the lower energy POD modes contain flow structures that vary dramatically with input parameter and do not follow a pattern easily modeled by the interpolation functions. These errors are then amplified by including more and less important modes in the interpolation pseudo-inverse.

An additional aspect of the ROM that has not been observed yet is how the database itself affects the accuracy of the ROM. It is obvious that there is a computational trade-off for producing a more accurate ROM. In this case, 120 CFD simulations were used to span the parameter space for angle of attack and inlet Mach number. Table 7.2 gives RMS error calculations for ROMS

<table>
<thead>
<tr>
<th>Database Description</th>
<th># Simulations</th>
<th>RMS Error ($\times 10^{-3}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Whole Database</td>
<td>120</td>
<td>5.525</td>
</tr>
<tr>
<td>Half Database</td>
<td>60</td>
<td>7.512</td>
</tr>
<tr>
<td>Half of Angles</td>
<td>60</td>
<td>5.538</td>
</tr>
<tr>
<td>Quarter of Angles</td>
<td>30</td>
<td>34.33</td>
</tr>
<tr>
<td>Half of Mach Number</td>
<td>60</td>
<td>8.273</td>
</tr>
<tr>
<td>Third of Mach Number</td>
<td>40</td>
<td>10.95</td>
</tr>
</tbody>
</table>
Figure 7.3: Graphical depiction of the parameter space that is spanned by the database of solutions as well as the trial points used to determine cutoff value and percent energy.

Figure 7.4: Contour plot of Mach number RMS error between nine ROM-generated database solutions and their relative CFD solutions formed from various portions of the original database to determine whether 120 simulations actually provides a noticeable accuracy benefit. While the other database formulations all increase the error of the ROM predictions, the database with only half of the angles tested shows ROM results on par with the original. Obviously, this would have been good to know in hindsight. One way to determine this in the midst of a parametric CFD study would be to take an iterative approach when building the database. Starting with a coarse set of simulations, creating a new ROM after each additional simulation and assessing its accuracy in predicting the next simulation may enable a better monitoring of ROM error while being computationally efficient.
7.3 Results and Discussion

The criteria found to produce the minimum error in Figure 7.4 will be used for the ROM results for the remainder of this chapter. Table 7.3 shows the ROM Mach error over the entire computational domain for each parameter set trial. Also presented in the table is the percentage of cells in the computational domain over an absolute Mach error above 0.01. The trials were chosen so that a large portion of the parameter space would be tested. There does not seem to be any correlation between either of the parameter values and the RMS error calculations.

To get a closer look at where the error is concentrated in the computation domain, the Trial 1 parameter set CFD and ROM solutions are portrayed in Figure 7.5(a,b). The contour plot is at the midpoint of the airfoil span between the wall and the symmetry plane. In general, it is difficult to discern visual differences between the CFD and ROM contour plots. The error contours shown in Figure 7.5 give the absolute Mach number error for the surface. Most of the error is concentrated near the surface of the airfoil where molecular viscosity dominates. There are also separation bubbles along both the top and bottom airfoil surfaces where there are larger magnitude errors. Additional error can be found where the free stream transitions to the wake region.

To get a better depiction of the error through the three-dimensional domain, an iso-surface for the absolute Mach number of 0.01 is demonstrated in Figure 7.6. The iso-surface shows that most of the error is on the bottom surface near the leading and trailing edges of the airfoil. There is also some wake-boundary layer interactions near the wall that are very difficult to predict by the ROM. Along span of the airfoil, there is some error in the prediction of the stagnation point.

<table>
<thead>
<tr>
<th>Trial</th>
<th>Mach Number</th>
<th>Angle of Attack</th>
<th>RMS Error ($\times 10^{-3}$)</th>
<th>% Cells Above 0.01 Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.286</td>
<td>5.81</td>
<td>4.823</td>
<td>1.861</td>
</tr>
<tr>
<td>2</td>
<td>0.261</td>
<td>6.56</td>
<td>4.711</td>
<td>1.182</td>
</tr>
<tr>
<td>3</td>
<td>0.292</td>
<td>-0.44</td>
<td>4.744</td>
<td>1.377</td>
</tr>
<tr>
<td>4</td>
<td>0.273</td>
<td>3.81</td>
<td>6.728</td>
<td>3.753</td>
</tr>
<tr>
<td>5</td>
<td>0.258</td>
<td>-1.45</td>
<td>4.635</td>
<td>1.508</td>
</tr>
<tr>
<td>6</td>
<td>0.255</td>
<td>3.12</td>
<td>5.307</td>
<td>1.576</td>
</tr>
<tr>
<td>7</td>
<td>0.266</td>
<td>0.53</td>
<td>7.281</td>
<td>3.694</td>
</tr>
<tr>
<td>8</td>
<td>0.278</td>
<td>1.03</td>
<td>4.251</td>
<td>1.915</td>
</tr>
<tr>
<td>9</td>
<td>0.296</td>
<td>2.86</td>
<td>6.388</td>
<td>1.924</td>
</tr>
</tbody>
</table>
Figure 7.5: Contour plots at the $Z = 0.025$ m plane of the Mach number from the (a) CFD and (b) ROM solutions using the parameter values for Trial 1. The (c) absolute Mach number error is also shown.

Figure 7.6: A plot of the iso-surface where the absolute Mach error between the CFD and ROM is equal to 0.01 for the Trial 1 parameter set.
One aspect of the error not fully apparent from Figures 7.5-7.6 is that there are Mach numbers predicted that do not make physical sense. For example, in the ROM solution for the Trial 1 parameter set there are cells that have negative Mach numbers. This is true even though there are no zero values anywhere within the database. The reason why this occurs also relates to one of the major limitations of this ROM method, which is that ROM predictions are solely based off of fitting the POD modes using the interpolation functions. If there are large discrepancies between different simulation cell values, such as a separation bubble in one and not in another, the interpolating functions still have to fit those regions. This produces an instability in the ROM. One way to mitigate this effect is to somehow filter cells where there are extreme variations throughout the database that may produce these incoherent predictions. This filter may be replacing bad values with the closest valid value or simply by blanking the trouble cells. Another solution would be to limit the ROM to be valid for areas only beyond a certain distance from no-slip walls. A final method for reducing error may be to create separate ROMs for the different regions of the flow using domain partitioning.

7.4 Concluding Remarks

This case of modeling the flow over an airfoil has illustrated both the strengths and the weaknesses inherent in this ROM method. The ROM performs extremely well in areas of the flow field where flow separation, wake, and boundary layer effects are minimal. This being said, the modeling of inviscid type flows may enable very accurate ROM solutions because they do not have any of these effects. However, this is also a limitation since many flows of interest cannot neglect these flow features.

Another difficulty is determining where these ROMs may be used besides merely just evaluating their accuracy. If the type of flow being studied is an integral part of an optimization scheme that will be used repeatedly, the ROM technique would be very advantageous for traversing the parameter because solutions can be created and analyzed so rapidly. Parameter-based ROMs may also be a by-product of a parametric flow study and may be an excellent use of already available data that may aid in future research. If a ROM is to be built from the initial stages of a study, iteratively increasing the parameter space resolution and quantifying error from each new prediction may help to identify when no more simulations are needed for a database.
Additionally, in this work a model has been created for only local Mach number for the airfoil flow. This was done to show the ability to predict derived quantities using a ROM. To be able to estimate all of the characteristics of the flow field and generate derived quantities, other ROMs using remaining field scalars would have to be created. This is less of a burden however since all the data is available from the CFD solutions, and creating the ROM takes relatively much less time than generating its database.
CHAPTER 8. CONCLUSION

While computational fluid dynamics (CFD) tools are very useful for understanding fluid flow and heat transfer, they also have the capability of generating large sets of data. These large data sets are cumbersome to analyze and store, motivating the production of methods that both extract the most information in a small amount of time and reduce the amount data needed for storage. This work shows how the proper orthogonal decomposition (POD) can be applied to efficiently and effectively analyze large sets of CFD data. The POD was used on single large simulations for analysis and compression as well as on collections of smaller simulations for modeling. POD methods were applied to complex turbomachinery simulations to analyze turbulent structures as well as the effects of pressure distortion in turbomachinery flow for the first time. Additionally, a new method for CFD data storage was devised to take advantage of the optimality of the POD and effectiveness of image compression techniques. Regardless of the application, the versatility and utility of the POD is overwhelmingly emphasized throughout the results presented in this thesis. It is an invaluable tool for post-processing and should not be neglected when seeking to understand findings from complex CFD simulations.

Simulations of the Blade-Row Interaction (BRI) test rig were analyzed using the POD to identify turbulent coherent structures in the deswirler and rotor blade passages. POD modes were extracted from sets of two-dimensional snapshots of velocity magnitude within each passage. Large rotor blade vortical structures were found to correlate with blade-passing frequency by comparing fluctuating structures within the modes. Modes describing traversing shocks increased in both spatial and temporal frequency in order to capture the sharp gradients in the deswirler passage. Reduced order reconstructions (ROR) were created using the first nine modes from both passage data sets. While the ROR recreated the general instantaneous flow field well, shock fronts and vortices, especially smaller ones, were much less defined and there was higher error located in these regions of the flow. Domain partitioning was applied to the rotor passage to quantify the
error reduction while using the same number of POD modes. Error was reduced by 63 percent when the domain was partitioned into 64 individual pieces. Finally, using the new combinatorial compression method (JPOD), compression ratios for the nine-mode RORs topped out at 574:1 for the BRI rotor passage. The domain partitioned RORs had less compression ratio due to the storage of extra coefficients.

Another complex turbomachinery simulation was used to illustrate additional characteristics and applications of the POD for very large data sets. Static pressure data in the third rotor blade row of a three-stage fan was extracted to analyze flow structures varying primarily from the total pressure distortion pattern defined at the inlet boundary. The method of snapshots was compared with the method of repeating geometry to show how the same analysis can be done while using less computational resources. Both methods provided very closely related POD modes highlighting distorted pressure-dependent flow structures, such as the shifting of the bow shock to different positions in the rotor passage. The repeated geometry method was limited by the number of rotor blades in the row, while the snapshot method was not. The snapshot method using a higher temporal resolution also enabled identification of blade rows contributing to the POD modes.

Using the POD as a source for modes depicting the variation within a set of parameter-dependent simulations, both transient and steady reduced order models (ROM) were created to predict temperature profiles in solids. The databases consisted of conjugate heat transfer CFD solutions. The two-parameter transient ROM showed excellent results for predicting the temperature of a cylinder in heated crossflow, with the root mean square (RMS) of the prediction error below 0.03 K for all the sample parameter sets. The temporal evolution of the temperature within the cylinder was also modeled very well. In the steady ROM, three parameters were varied for a rotating turbine blade with internal cooling. The parameter space was ambitious with the primary flow ranging 600 K. However, despite the ambitious range, ROM predictions of the temperature profiles were within 2.5 percent of the CFD, which is a respectable approximation for heat transfer.

The final POD application presented in this work was creating a ROM to predict the flow field Mach number for flow over an airfoil while varying angle of attack and inlet Mach number. It was found that this ROM was very sensitive to the percent energy and cutoff value criteria required for building the ROM. Using a set of trial simulations at random input parameters, the percent energy and cutoff value were chosen to minimize the prediction RMS error. As a result, only 9
out of 120 possible POD modes were necessary for the ROM. The ROM predicted the flow field outside of the airfoil wake and boundary layer very well, but produced errant results near the airfoil surface, especially in regions where separated flow occurred. The RMS of the absolute Mach error was about 0.0055 over the set of trial parameter set solutions. Under some conditions, interpolation instabilities arose due to poor fitting of the interpolation functions to the original database.

8.1 Recommendations for Further Work

While this work provided a good base for POD applications for large data sets, there is still much to be explored. For example, the method is given for the JPOD compression for three-dimensional structured computational domains but it has yet to be applied. In order to fully validate that this compression method is acceptable, this should be done. For the analysis of pressure distortion and its effect on the performance of turbomachinery, this study has only scratched the surface. Here, only the static pressure for one stage of a single simulation was analyzed. There are many other field variables to observe and stages to focus on. Additionally, it would be advantageous to compare the POD modes from a distorted inlet simulation to a simulation of one with a uniform inlet. Also, within multistage turbomachinery simulations, data sets with temporal resolutions high enough can extract POD modes that capture high frequency fluctuations. The coefficients can then be filtered to represent only the fluctuations due to a specific blade row, just at the low frequency pressure distortion was isolated in this study. The idea for a hybrid snapshot-repeating geometry data set construction can enable these types of analyses without requiring the simulation to run for a whole revolution of the rotor blade. Finally, the ROMs built for the conjugate heat transfer and airfoil flow came from databases solely created for this purpose based on parameter sets designed ahead of time. However, as was shown in the airfoil case, this may not be computationally efficient. Therefore, further work to study iterative approaches to developing a ROM from scratch by providing gradually greater resolution in the input parameters may be advantageous. The airfoil ROM showed that instabilities may arise in trouble areas of the flow domain. Filtering and domain partitioning may be ways to correct for the weaknesses in the ROM method for predicting flow fields.
REFERENCES


APPENDIX A. MATLAB CODE

A.1 POD Methods

%% POD_AllModes Function
% Author: Trevor Blanc
% Affiliation: ME Department, BYU-Provo
% Date: 3 February 2014
% Description: This method is an alternative to the singular value
decomposition (SVD) method. The resulting outputs are essentially the
t same magnitude, but they may differ in sign. The method accepts a 2-D, m
x n matrix, D, assuming that m is larger than n. The function returns U, a
m x n matrix, V, a n x n matrix, and S, a vector of length n containing
singular values. This method is computationally faster than the SVD function
in matlab.

function [U S V] = POD_AllModes(D)

    % Determine the m and n values for D
    nRows = size(D,1);
    nCols = size(D,2);

    % Calculate the eigenvalues and eigenvectors for D'*D
    [Vec Val] = eig(D'*D);
    % The eigenvalues are the diagonals
    Val = diag(Val);
    % Sort the eigenvalues and eigenvectors in descending order
    EigVal = flipud(Val);
    EigVec = fliplr(Vec);
% The singular values are the square root of the eigenvalues
S = sqrt(EigVal);
% V is the matrix of eigenvectors
V = EigVec;
% Allocate memory for U
U = zeros(nRows,nCols);
% Loop through each eigenvalue and eigenvector to calculate each
% respective basis mode in U
for i = 1:nCols
    U(:,i) = D*V(:,i)/S(i);
end

%% POD

% POD_OneMode Function
% Author: Trevor Blanc
% Affiliation: ME Department, BYU-Provo
% Date: 3 February 2014
% Description: This method is used to produce just a single mode from the
% POD based on the inputs. The method accepts argument D, which is a 2-D,
% m x n matrix where m is greater than or equal to n. modeNum is an integer
% corresponding to the mode desired. u is a vector of length m, v is a
% vector of length n, and s is the corresponding singular value.

function [u s v] = POD_OneMode(D,modeNum)

% First determine the n value to check against modeNum
nCols = size(D,2);
if modeNum > nCols
    u = 0;
    v = 0;
    s = 0;
    return
end
% Find the eigenvalues and eigenvectors of D'*D
[Vec Val] = eig(D'*D);
% Redefine Val to be just the eigenvalues
Val = diag(Val);
% Arrange the eigenvalues and eigenvectors in decreasing value
EigVal = flipud(Val);
EigVec = fliplr(Vec);

% The singular values are the square root of the eigenvalues
s = sqrt(EigVal(modeNum));
% v is the eigenvector matrix
v = EigVec(:,modeNum);
% Calculate the specified basis mode
u = D*v/s;

end

%% POD
SelModes Function
% Author: Trevor Blanc
% Affiliation: ME Department, BYU-Provo
% Date: 4 February 2014
% Description: This method is used to produce just a selection of POD modes
% based on the inputs. The method accepts argument D, which is a 2-D,
% m x n matrix where m is greater than or equal to n. modeNum is a vector
% corresponding to the mode numbers desired. U is a matrix of modes, V is a
% matrix of coefficients, and S is the corresponding singular values.

function [U S V] = POD_SelModes(D,modeNums)

% First determine the m and n value to check against modeNums
nRows = size(D,1);
nCols = size(D,2);
% Determine the number of elements in the modeNums input
nModes = length(modeNums);
% Make sure all of modeNums entries exist and there are no duplicate
% entries. Order the modes numbers.
Mode = sort(modeNums,'ascend');
for i = nModes:-1:1
    if Mode(i) > nCols || Mode(i) < 1
        Mode(i) = [];
        nModes = nModes-1;
    else
        if i < nModes
            if Mode(i) == Mode(i+1)
                Mode(i) = [];
                nModes = nModes-1;
            end
        end
    end
end
end

% Find the eigenvalues and eigenvectors of D'*D
[Vec Val] = eig(D'*D);
% Redefine Val to be just the eigenvalues
Val = diag(Val);
% Arrange the eigenvalues and eigenvectors in decreasing value
EigVal = flipud(Val);
EigVec = fliplr(Vec);

% v is the eigenvector matrix
V = EigVec(:,Mode);
% The singular values are the square root of the eigenvalues
S = sqrt(EigVal(Mode));
% Calculate the specified basis modes by looping through all of V
U = zeros(nRows,nModes);
for i = 1:size(V,2)
    U(:,i) = D*V(:,i)/S(i);
end
end
function [U S V] = POD_EnergyModes(D,percentEnergy)

% Determine the m and n values for D
nRows = size(D,1);

% Calculate the eigenvalues and eigenvectors for D'*D
[Vec Val] = eig(D'*D);
% The eigenvalues are the diagonals
Val = diag(Val);
% Sort the eigenvalues and eigenvectors in descending order
EigVal = flipud(Val);
EigVec = fliplr(Vec);
% The total energy will not include the first/mean mode
Total = sum(EigVal(2:end));
Perc = EigVal(1:end)/Total;
PE = 0;
Modes = 1;
% Find how many modes are necessary to encompass the desired percent
% energy. The first mode is included, but not calculated.
while PE < percentEnergy/100
    Modes = Modes + 1;
    PE = PE + Perc(Modes);
end

% The singular values are the square root of the eigenvalues
S = sqrt(EigVal(1:Modes));
% V is the matrix of eigenvectors
V = EigVec(:,1:Modes);

% Allocate memory for U
U = zeros(nRows,Modes);

% Loop through each eigenvalue and eigenvector to calculate each
% respective basis mode in U
for i = 1: Modes
    U(:,i) = D*V(:,i)/S(i);
end

A.2 ROR Methods

%%% ROR

function ROR = ROR_Dominant(D,numModes)
% Create an array of integers in order from 1 to numModes
Modes = 1:numModes;
% Extract the first numModes POD modes using POD_SelModes function
[U S V] = POD_SelModes(D,Modes);
% Calculate the expansion coefficients required for the ROR
C = U'*D;
% Using the modes and coefficients, calculate the ROR
ROR = U*C;
end

%%% ROR_Selection Function

82
function ROR = ROR_Selection(D,modeNums)
    % Extract the POD modes specified by the modeNums array
    [U S V] = POD_SelModes(D,modeNums);
    % Calculate the expansion coefficients required for the ROR
    C = U'*D;
    % Using the modes and coefficients, calculate the ROR
    ROR = U*C;
end

function ROR = ROR_Energy(D,percentEnergy)
    % Extract the POD modes specified by the modeNums array
    [U S V] = POD_EnergyModes(D,percentEnergy);
    % Calculate the expansion coefficients required for the ROR
    C = U'*D;
    % Using the modes and coefficients, calculate the ROR
    ROR = U*C;
end
A.3 ROM Methods

%% ROM_SteadyState Function
% Author: Trevor Blanc
% Affiliation: ME Department, BYU-Provo
% Date: 4 February 2014
% Description: This is a method to produce a reduced order model (ROM) for
% a data base of steady state of time averaged solutions. The number
% of modes for the model is based on an energy percentage value. The
% database is a m x n set, where m is the number of gridpoints, n is
% the number of parameter sets. KMatrix is a k x n matrix where k is the
% number of input parameters, and n is the number of parameter sets.

function ROM = ROM_SteadyState(DataBase,KMatrix,EnergyPerc)

% ROM is a structure that contains Phi, C, K, Energy
ROM = struct([]);

% Rename a few objects for easier reference later in method
K = KMatrix;

% Define Interpolation Matrix that is used for all the timesteps
F = zeros(size(K,2),size(K,2));
for i=1:size(K,2)
    for j=1:size(K,2)
        Sum = 0;
        for m=1:size(K,1)
            Sum = Sum + ((K(m,i)−K(m,j))/max(K(m,:)))^2;
        end
        F(j,i) = 1/(Sum+1)^(1/2);
    end
end

% A SVD is performed on the interpolation matrix F and the factoring
% matrices are stored
[U_F Sig_F V_F] = svd(F);

% The singular values of F are then plotted and a minimum value is % requested from the user to determine the cutoff when forming the % pseudoinverse matrix
semilogy(1:1:length(diag(Sig_F)),diag(Sig_F),'o');
grid on;

% Using the LowLimit set by the user, the pseudoinverse is found
LowLimit = input('What is the cutoff singular value for F? ');
S_F = zeros(length(diag(Sig_F)),length(diag(Sig_F)));
for i=1:1:length(diag(Sig_F))
    if Sig_F(i,i) < LowLimit
        S_F(i,i) = 0;
    else
        S_F(i,i) = 1/Sig_F(i,i);
    end
end

% Get the basis modes corresponding to the EnergyPerc desired
[Phi S V] = POD_EnergyModes(DataBase,EnergyPerc);

% The coefficient matrix is then found using PHI, Piece, and the % pseudoinverse of F
C = Phi'*DataBase*V_F*S_F*transpose(U_F);

% Return the ROM structure
ROM(1).Phi = Phi;
ROM(1).C = C;
ROM(1).K = K;
ROM(1).Energy = EnergyPerc;
ROM(1).LowLimit = LowLimit;
end
function Solution = ROM_SteadyStateEval(parameterVector, ROM)

    % Define the Interpolated vector, f, using k and K
    f = zeros(1, size(ROM.K, 2));
    for i = 1:size(ROM.K, 2)
        sum = 0;
        for m = 1:size(ROM.K, 1)
            sum = sum + ((parameterVector(m) - ROM.K(m, i))/max(REM.K(m,:)))^2;
        end
        f(i) = 1/(sum+1)^(1/2);
    end
    f = transpose(f);

    % Solve the estimated solution
    Solution = ROM.Phi*ROM.C*f;
end
% of modes for each timestep is based on an energy percentage value. The database is a m x n x t set, where m is the number of gridpoints, n is the number of parameter sets, and t is the number of timesteps. KMatrix is a k x n matrix where k is the number of input parameters, and n is the number of parameter sets.

function ROM = ROM_Transient(DataBase,KMatrix,EnergyPerc)

% ROM is a structure that contains Phi, C, K, PieceSize, Energy, and NumTimesteps
ROM = struct([ ]); % Rename a few objects for easier reference later in method
NumTimesteps = size(DataBase,3); K = KMatrix;

% Define Interpolation Matrix that is used for all the timesteps
F = zeros(size(K,2),size(K,2));
for i=1:1:size(K,2)
    for j=1:1:size(K,2)
        Sum = 0;
        for m=1:1:size(K,1)
            Sum = Sum + ((K(m,i)-K(m,j))/max(K(m,:)))^2;
        end
        F(j,i) = 1/(Sum+1)^(1/2);
    end
end

% A SVD is performed on the interpolation matrix F and the factoring matrices are stored
[UF SigF VF] = svd(F);

% The singular values of F are then plotted and a minimum value is requested from the user to determine the cutoff when forming the pseudoinverse matrix
semilogy(1:1:length(diag(SigF)),diag(SigF),'o');
grid on;

% Using the LowLimit set by the user, the pseudoinverse is found
LowLimit = input('What is the cutoff singular value for F? ');
S_F = zeros(length(diag(Sig_F)),length(diag(Sig_F)));
for i=1:length(diag(Sig_F))
    if Sig_F(i,i) < LowLimit
        S_F(i,i) = 0;
    else
        S_F(i,i) = 1/Sig_F(i,i);
    end
end

% Allocate Space for the ROM
Phi = cell(NumTimesteps,1);
C = cell(NumTimesteps,1);

% Loop through all of the timesteps and create a ROM for each timestep
for i = 1:NumTimesteps

    % Isolate the Database to the specific timestep section
    Piece = DataBase(:,:,i);

    % Get the basis modes corresponding to the EnergyPerc desired
    [tempPhi S V] = POD_EnergyModes(Piece,EnergyPerc);

    % The coefficient matrix is then found using PHI, Piece, and the % pseudoinverse of F
    tempC = tempPhi'*Piece*V_F*S_F*transpose(U_F);

    % Store the phi and c cell structures in the larger time cell % structures
    Phi{i,1} = tempPhi;
    C{i,1} = tempC;
end
% Return the ROM structure
ROM(1).Phi = Phi;
ROM(1).C = C;
ROM(1).K = K;
ROM(1).Energy = EnergyPerc;
ROM(1).NumTimesteps = NumTimesteps;
ROM(1).LowLimit = LowLimit;

end

%% ROM_TransientEval Function
% Author: Trevor Blanc
% Affiliation: ME Department, BYU-Provo
% Date: 5 February 2014
% Description: This method produces a sample solution for Transient ROM
% created from the ROM_Transient function at an arbitrary set of input
% parameters. The parameters should fall within the minimum to maximum
% range of the parameter in the database from which the ROM was developed.
% The solution is a n x t matrix where n is the number of gridpoints, and t
% is the number of timesteps.

function Solution = ROM_TransientEval (parameterVector, ROM)

% Determine the total number of gridpoints in the domain
numG = size (ROM.Phi{1,1},1);

% Initialize the Solution
Solution = zeros (numG, ROM.NumTimesteps);

% Define the Interpolated vector, f, using k and K
f = zeros (1, size (ROM.K,2));
for i = 1:1:size (ROM.K,2)
    sum = 0;
    for m = 1:1:size (ROM.K,1)
        sum = sum + ((parameterVector(m) - ROM.K(m,i)) / max (ROM.K(m,:)))^2;
    end
    f(i) = sum;
end

% Compute the solution for each gridpoint and timestep
for i = 1:1:size (ROM.K,2)
    for j = 1:1:ROM.NumTimesteps
        Solution(:,j) = Phi' * (f(i) - C * j);
    end
end
\[ f(i) = \frac{1}{(\text{sum}+1)^{1/2}}; \]

end

\[ f = \text{transpose}(f); \]

% Loop through all the timesteps and solve each estimate
for k = 1:ROM.NumTimesteps

    Solution(:,k) = ROM.Phi\{k,1\}*ROM.C\{k,1\}*f;

end

end

A.4 JPOD Methods

%%% JPOD_Dominant Function
% Author: Trevor Blanc
% Affiliation: ME Department, BYU-Provo
% Date: 10 February 2014
% Description: This method compresses a data set, D, that may be
% represented by multiple iDim x jDim 2-D arrays. dataName specifies the
% name of the file that will result from the compression. The numModes
% input is the total number of POD modes used in the compression. These
% are the first modes of the POD. The method returns the size of the
% compressed file in kB.

function datasize = JPOD_Dominant(D,dataName,iDim,jDim,numModes)

    % Create the folder into which the JPEG and .dat files will go
    mkdir(dataName);

    % Extract the first numModes modes of the POD for D
    Modes = 1:numModes;
    [Phi S V] = POD_SelModes(D,Modes);
function datasize = JPOD_Energy(D, dataName, iDim, jDim, percEnergy)

    % Create the folder into which the JPEG and .dat files will go
    mkdir(dataName);

    % Extract the number of modes that contain a certain percentage of
    % energy of POD
    [Phi S V] = POD_EnergyModes(D, percEnergy);

    % Compress and store the JPOD modes
    JPOD_CompressModes(D, Phi, iDim, jDim, dataName);

    % Return the data size of the file in kB
    s = dir([dataName ' .zip']);
    datasize = s.bytes/1024;
function compressed = JPOD.CompressModes(D,Phi,iDim,jDim,dataName)

    numModes = size(Phi,2);

    % Initialize bit length for the JPEG compression
    numBit = 12;

    % Convert Phi into a iDim x yDim x numModes 3-D Array (this method may
    % differ depending on the dataset configuration)
    Phi_p = Two2ThreeDim(Phi,iDim,jDim);

    % Create matrix to store jpeg information
    IP = zeros(numModes,2);

    % Cycle through all modes in Phi_p and convert each to a .jpg
    for mode = 1:numModes
        % Store minimum value
        IP(mode,1) = min(min(Phi_p(:,:,mode)));
        % Store delta value depending on numBit
        IP(mode,2) = (max(max(Phi_p(:,:,mode))) ... 
                     -min(min(Phi_p(:,:,mode))))/(2^numBit-1);
        % Define name for mode
        name = [dataName '_M' int2str(mode) '.jpg'];
        
    end

end
% Convert and write mode to a jpeg image
plane = mat2gray(Phi_p(:,:,mode));
imwrite(plane,[dataName ' ' name],'Bitdepth',numBit);
end

% Decompress the modes to determine the adjusted expansion
% coefficients for the jpeg compression error
Phi_p = zeros(iDim,jDim,numModes);
for mode = 1:numModes
    Image = imread([dataName ' ' dataName '_M' int2str(mode) '.jpg']);
    minVal = IP(mode,1);
    delVal = IP(mode,2);
    Phi_p(:,:,mode) = minVal+delVal*double(Image);
end
Phi = Three2TwoDim(Phi_p);
% Make sure that the decompressed modes are normalized
for mode = 1:numModes
    Phi(:,mode) = Phi(:,mode)/norm(Phi(:,mode));
end
C = Phi'*D;
% Write the InputParameters matrix to a data file
fid = fopen([dataName ' ' 'InpPar.dat'], 'wt');
fclose(fid);
dlmwrite([dataName ' ' 'InpPar.dat'], IP, '-append', 'delimiter',' ');

% Write the C matrix to a data file
fid = fopen([dataName ' ' 'C.dat'], 'wt');
fclose(fid);
dlmwrite([dataName ' ' 'C.dat'], C, '-append', 'delimiter',' ');

% Zip the file
zip(dataName,dataName);
rmdir(dataName,'s');
compressed = 1;
function DataMatrix = JPOD_Decompress(dataName)

    % Unzip the main file
    unzip([dataName '.zip']);

    % The number of modes is the same as the number of jpeg images
    numModes = length(dir([dataName '.*.jpg']));

    % Import the coefficient matrix
    C = importdata([dataName '\C.dat']);

    % Read in the first mode image to get dimension values
    Image = imread([dataName '\' dataName 'M1.jpg']);
    iDim = size(Image,1);
    jDim = size(Image,2);

    % Initialize the mode containing array
    Phi_p = zeros(iDim,jDim,numModes);

    % Import the input parameters that are necessary for decompression
    InpPar = importdata([dataName '\InpPar.dat']);

    % Loop through and decompress all the images to modes
    for mode = 1:numModes
        if mode == 1
minVal = InpPar(mode,1);
delVal = InpPar(mode,2);
Phi(:,:,mode) = minVal+delVal*double(Image);
else
    Image = imread([dataName ' dataName '_M' int2str(mode) '.jpg']);
    minVal = InpPar(mode,1);
    delVal = InpPar(mode,2);
    Phi(:,:,mode) = minVal+delVal*double(Image);
end
end

% Convert Phi back to a 2-D matrix and normalize each column
Phi = Three2TwoDim(Phi);
for mode = 1:numModes
    Phi(:,mode) = Phi(:,mode)/norm(Phi(:,mode));
end

% Remove the unzipped directory from memory
rmdir(dataName,'s');

% Calculate the data matrix like in a reduced order reconstruction
DataMatrix = Phi*C;
end