REDUCED-ORDER MODELS FOR COMPUTATIONAL AEROELASTICITY

A Dissertation

by

BRIAN ANDREW FRENO

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

DOCTOR OF PHILOSOPHY

Chair of Committee,	Paul G. A. Cizmas
Committee Members,	Raytcho D. Lazarov
	John C. Slattery
	Theofanis Strouboulis
Head of Department,	Rodney D. Bowersox

December 2013

Major Subject: Aerospace Engineering

ABSTRACT

This dissertation presents a proper orthogonal decomposition (POD) method that uses dynamic basis functions. The dynamic functions are of a prescribed form and do not explicitly depend on time but rather on parameters associated with flow unsteadiness. This POD method has been developed for modeling nonlinear flows with deforming meshes but can also be applied to fixed meshes. The method is illustrated for subsonic and transonic flows with fixed and deforming meshes. This method properly captured flow nonlinearities and shock motion for cases in which the classical POD method failed.

Additionally, this dissertation presents a novel approach for assessing the number of basis functions used in POD. POD results are compared between subsonic and transonic flows for several cases. It is demonstrated that in order to determine the number of basis functions, it is better to assess the variation of individual energy values, as opposed to the cumulative energy values. Finally, for off-reference flow conditions, interpolation is performed on a tangent space to a Grassmann manifold, and the effect of interpolation order is investigated.

ACKNOWLEDGMENTS

This dissertation marks the end of nearly a decade of study at Texas A&M. The work presented herein, as well as the valuable insight gained that led me to this point, arose from the generosity of several people.

To that end, I begin by thanking my advisor, Dr. Paul Cizmas. Dr. Cizmas made my graduate studies possible and served as my advisor for my Master of Science degree as well. In addition to sharing his time and insight, he continuously challenged me.

I additionally thank the members of my committee for their time and valuable feedback. I was fortunate to have Dr. Raytcho Lazarov as my professor for Calculus III when I was a freshman and for a numerical partial differential equations course when I was a doctoral student. He and Dr. Theofanis Strouboulis served as committee members for my Master of Science degree. Dr. Strouboulis's numerical simulation and finite element courses confirmed my interest in computational mechanics, and he subsequently provided me with opportunities to refine my teaching skills as a grader, teaching assistant, and occasional lecturer. Though I have not known him as long, Dr. John Slattery's knowledge and interest in my work has been much appreciated.

This work has been predominantly funded through a NASA Graduate Student Researchers Program fellowship and through the GUIde 4 Consortium. I would like to thank my mentors at the NASA Marshall Space Flight Center, Dr. Andrew Brown and Preston Schmauch, for their assistance, feedback, and support.

My fellow graduate students throughout the years have also played an important role in my graduate studies. I would like to thank my officemate Neil Matula for our instrumental discussions and debates. I am also grateful to Raymond Fontenot,

Robert Brown, Thomas Brenner, Forrest Carpenter, Michael Joly, Nathan Jones, Shalom Johnson, Will Carter, Greg Worley, David Liliedahl, and Allen Ream.

Additionally, I would like to extend my gratitude to Dr. Glenn Gebert in the Aerodynamics Department at Lockheed Martin Missiles and Fire Control in Orlando, Florida for my two summer internships. The work was rewarding and motivated me to continue my studies beyond the undergraduate level.

Finally, I would like to thank my family and friends for their unconditional and unending support. I am especially thankful to my parents for their many sacrifices and for teaching me the benefits of hard work. They are the reason for my success.

TABLE OF CONTENTS

Page

Page

LIST OF FIGURES

FIGURE Page

FIGURE Page

FIGURE Page

LIST OF TABLES

NOMENCLATURE

Latin Letters

a_j	Time coefficient for j^{th} basis function
$\mathbf{D}$	Rate-of-strain tensor
d	Number of parameters contained in $\mathbf{\Gamma}$
E	Mass-specific total energy
$\mathbf{F}$	Flux vector
$\mathbf{F}_c$	Convective flux tensor
$\mathbf{F}_v$	Viscous flux tensor
$\mathbf{I}$	Identity matrix
k	Thermal conductivity coefficient
M	Mach number
M	Number of snapshots
m	Number of basis functions
$\mathbf{n}$	Unit vector normal to volume boundary
n	Number of unknown variables at a given time
p	Pressure
$\mathbf{R}$	Residual
S	Area of volume boundary
$\mathbf{T}$	Viscous stress tensor
t	Time
$\mathbf{U}$	State vector
u	x-component of velocity
v	y-component of velocity

$\mathbf{v}$	Velocity vector
w	z-component of velocity
$\mathbf{x}$	Spatial coordinate (index based)

Greek Letters

$\boldsymbol{\Gamma}$	Parameters associated with flow unsteadiness / mesh deformation
Θ	Temperature
λ	Eigenvalue indicating time average of square of Euclidean norm of projection of $\mathbf{U} - \bar{\mathbf{U}}$ onto $\boldsymbol{\varphi}$
μ	Dynamic viscosity
ρ	Density
$\boldsymbol{\varphi}_j$	j^{th} basis function
Ω	Control volume

Operators

$(\cdot, \cdot)$	Inner product
$\langle \cdot \rangle$	Time average
$\lvert \cdot \rvert$	Determinant
$\lVert \cdot \rVert$	Euclidean norm
$\bar{\cdot}$	Time average

Acronyms

L-BFGS	Limited-memory Broyden–Fletcher–Goldfarb–Shanno
ROM	Full-order model
POD	Proper orthogonal decomposition
ROM	Reduced-order model

1. INTRODUCTION*

1.1 Motivation and Background

Despite continuous advances in computer hardware, the computational cost of high-fidelity computational fluid dynamics simulations remains a limiting factor for many science- and engineering-relevant problems. A typical example of numerical simulations that require large computational resources is aeroelasticity, where unsteadiness of the flow and temporal variation of the mesh can be a computational burden.

Reduced-order modeling based on proper orthogonal decomposition (POD) has proven to be a successful method for reducing the computational time, while providing high-fidelity results for a wide range of applications covering transport phenomena and structural dynamics [1]. Through model reduction, dominant spatial modes are used to describe the flow. The nonlinear partial differential equations can then be reduced to ordinary differential equations from which the time coefficients that weight the spatial modes are calculated.

Proper orthogonal decomposition is a method through which snapshots of the flow obtained from the full-order model (FOM) are used to extract the optimal set of spatially dependent basis functions [2]. The large set of partial differential equations is then projected onto the basis functions, resulting in a much smaller set of ordinary differential equations.

---

*Part of this section is reprinted with permission from "Using proper orthogonal decomposition to model off-reference flow conditions" by B. A. Freno, T. A. Brenner, P. G. A. Cizmas, 2013. *International Journal of Non-Linear Mechanics*, vol. 54, pp 76–84, Copyright 2013 by Elsevier.

1.2 Literature Review

Reviews of POD-based reduced-order models (ROMs) have been presented in [3, 4, 5]. In the last decade, three main research directions were explored for POD-based ROMs: (i) improving the prediction of off-reference conditions, (ii) improving performance, and (iii) modeling moving/deforming meshes.

1.2.1 Off-Reference Conditions

Early POD-based ROMs focused on computing basis functions directly from snapshots of a FOM for the same flow parameters as the ROM [6]. This approach renders the computational savings of the ROM moot. For practical applications, it is necessary to extend the ROM to off-reference parameter sets [7].

Proposed modifications to the POD basis functions to account for off-reference conditions include direct interpolation, enriching the snapshot database [8], interpolation using subspace angles [9, 10, 11] or a tangent space to a Grassmann manifold [12, 13, 14], sensitivity analysis using parametric derivatives [15, 16], and using actuation modes [17, 18]. Some of these methods are reviewed in [19].

1.2.2 Performance

To improve performance for compressible flows, the use of physically or numerically sensible inner products has been suggested to better account for dynamically significant variables [20] and to improve ROM stability [21]. For multiphase flows, Brenner et al. [22] showed that treating field variables separately when assembling the autocorrelation matrix, which yields the POD basis functions, produces greater error than using a coupled approach. To solve flows with discontinuities, an augmented POD method [23] was developed using mathematical morphology. Several acceleration techniques were proposed in [24].

1.2.3 Deforming Meshes

The modeling of moving/deforming meshes has been primarily motivated by aeroelastic applications, which are notorious for requiring large computational resources. POD has been used in linear [10, 12, 25, 26, 27] and nonlinear aeroelastic simulations [28, 29, 30, 31, 32]. One of the primary challenges associated with nonlinear aeroelastic simulations is the motion of the mesh, particularly when it is deformed. Spatial and temporal integration no longer commute when the mesh varies in time. However, if the mesh is deformed in a topologically consistent manner, the integrals can commute if a computational index-based domain is used.

Anttonen [28] and Anttonen et al. [30, 33] proposed using different sets of index-based basis functions associated with different deformations; however, discontinuously changing basis functions with respect to time reduces the solution fidelity. Additionally, several sets of basis functions are required to yield a robust model, and a matching algorithm is necessary to determine the most appropriate set.

Liberge and Hamdouni [34] used interpolation by treating the fluid–structure domain as a multiphase flow. In addition to requiring interpolation, modifications to the boundary conditions are required. Lewin and Haj-Hariri [29] modeled the incompressible Navier–Stokes equations by using the reference frame of the moving airfoil to exploit the simplified boundary conditions that arise from incompressible viscous flow. Placzek et al. [31] modeled compressible flow for rigid-body motions. These approaches do not address mesh deformation.

1.3 Objective and Scope

This dissertation presents a new, index-based method that uses a dynamic average and dynamic basis functions to model compressible flow using a deforming mesh. There is no need for interpolation or modification of the boundary conditions. These

dynamic functions vary continuously with respect to parameters associated with the flow unsteadiness and/or mesh deformation, and they are optimal, subject to the prescribed form. Furthermore, one set of basis functions is used, and a matching algorithm is unnecessary.

Additionally, this dissertation offers several valuable insights. POD results are generated for several cases, and a comparison is made between subsonic and transonic flows. The effects of interpolation order when using a Grassmann manifold are investigated for the different flow regimes. Finally, the energy spectrum is used to assess the necessary number of basis functions. It is demonstrated that in order to determine the number of basis functions, it is better to assess the variation of individual energy values, as opposed to the cumulative energy values.

1.4 Novel Aspects of this Dissertation

In POD, the average and basis functions are functions only of space. In this dissertation, a dynamic average and dynamic basis functions are introduced to model flow for which the standard approach fails. These dynamic functions vary continuously with respect to time-dependent parameters associated with the flow unsteadiness and/or mesh deformation, and they are optimal, subject to the prescribed form. The work presented herein marks the first known usage of dynamic basis functions that vary with time.

The dynamic functions are better suited for capturing unsteady, highly nonlinear phenomena, especially when the static functions fail. Consequently, fewer dynamic basis functions are needed. For cases in which static basis functions are suitable, the use of fewer dynamic basis functions avoids the need to resolve the higher frequencies associated with a larger amount of basis functions. Several cases simulating deforming meshes and compressible flow are presented in which the dynamic func-

tions perform better than the static counterparts. In addition, this dissertation offers several valuable insights concerning the modeling of different flow regimes, deforming meshes, and off-reference conditions.

1.5 Outline

In Section 2, POD is discussed, the dynamic average and the dynamic basis functions are derived, and static basis function interpolation for off-reference flow conditions is explained. The physical model, FOM, and ROM of the flow solver are described in Section 3. In Section 4, results are shown for subsonic and transonic flow through a channel with fixed and deforming meshes using static and dynamic functions. Comparisons are made between the FOM and the ROM, and the results are discussed. In Section 5, results are shown for transonic flow through a linear compressor cascade with plunging blades using static and dynamic functions. Comparisons are made between the FOM and the ROM using static and dynamic functions, and the results are discussed. Results for off-reference flow conditions are presented and discussed in Section 6. In Section 7, methods for reducing the computational time associated with the ROM are described. Finally conclusions are presented in Section 8.

2. PROPER ORTHOGONAL DECOMPOSITION*

Proper orthogonal decomposition is a method through which an optimal set of orthogonal spatial basis functions is extracted from a set of data, from which the mean has typically been subtracted. The spatial basis functions are linearly combined using time-dependent coefficients to form a reduced-order model:

$$\mathbf{U}(\mathbf{x},t) \approx \bar{\mathbf{U}}(\mathbf{x}) + \sum_{j=1}^{m} a_j(t)\boldsymbol{\varphi}_j(\mathbf{x}).$$

Through reduced-order modeling, the partial differential equations are reduced to a system of ordinary differential equations.

In this dissertation, proposed modifications to POD include replacing the static average and static basis functions with a dynamic average and dynamic basis functions. The dynamic average and dynamic basis functions do not explicitly depend on time but rather on parameters $\boldsymbol{\Gamma} \equiv \{\gamma_1, \ldots, \gamma_d\}^T$ associated with the flow unsteadiness and/or mesh deformation.

The dynamic functions take the form

$$f(\mathbf{x};\boldsymbol{\Gamma}) = f_0(\mathbf{x}) + \sum_{k=1}^{d} \gamma_k f_k(\mathbf{x}).$$

The elements of $\boldsymbol{\Gamma}$ can consist of time derivatives, powers, and products of the measured quantities, provided all elements are linearly independent.

The first subsection outlines the procedure for determining the static basis func-

*Part of this section is reprinted with permission from "Using proper orthogonal decomposition to model off-reference flow conditions" by B. A. Freno, T. A. Brenner, P. G. A. Cizmas, 2013. *International Journal of Non-Linear Mechanics*, vol. 54, pp 76–84, Copyright 2013 by Elsevier.

tions [2, 35], and the following two subsections show how the optimal dynamic average and dynamic basis functions of the prescribed form are computed. The final subsection discusses interpolating between sets of static basis functions for off-reference flow conditions.

2.1 Standard Approach

A more general framework for the traditional approach to POD is presented to facilitate the extensions proposed later in this section. Conventionally, after subtracting the time average, $\bar{\mathbf{U}}$, from the snapshots, $\mathbf{U}$, $\tilde{\mathbf{U}} \equiv \mathbf{U} - \bar{\mathbf{U}}$ is approximated by

$$\tilde{\mathbf{U}}(\mathbf{x}, t) \approx \sum_{j=1}^{m} a_j(t) \boldsymbol{\varphi}_j(\mathbf{x}),$$

where $a_j(t) = \left(\tilde{\mathbf{U}}(\mathbf{x}, t), \boldsymbol{\varphi}_j(\mathbf{x})\right) / \left(\boldsymbol{\varphi}_j(\mathbf{x}), \boldsymbol{\varphi}_j(\mathbf{x})\right)$, and $(\cdot, \cdot)$ is the inner product. The basis functions have been presumed mutually orthogonal to more efficiently span the subspace. $\tilde{\mathbf{U}}$ is equal to the sum of the approximation obtained from the projection onto the basis and the error:

$$\tilde{\mathbf{U}} = \sum_{j=1}^{m} \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_j\right)}{\left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j\right)} \boldsymbol{\varphi}_j + \left(\tilde{\mathbf{U}} - \sum_{j=1}^{m} \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_j\right)}{\left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j\right)} \boldsymbol{\varphi}_j \right).$$

Since the error is orthogonal to the approximation, the Pythagorean theorem holds, and

$$\left\| \tilde{\mathbf{U}} \right\|^2 = \left\| \sum_{j=1}^{m} \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_j\right)}{\left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j\right)} \boldsymbol{\varphi}_j \right\|^2 + \left\| \tilde{\mathbf{U}} - \sum_{j=1}^{m} \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_j\right)}{\left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j\right)} \boldsymbol{\varphi}_j \right\|^2,$$

where $\| \cdot \|$ is the L^2-norm. Consequently, minimizing the time-averaged error is equivalent to maximizing the time-averaged approximation. Due to the orthogonal-

ity assumption, the time-averaged square of the norm of the approximation can be simplified to

$$\left\langle \left\| \sum_{j=1}^{m} \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_j\right)}{\left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j\right)} \boldsymbol{\varphi}_j \right\|^2 \right\rangle = \left\langle \sum_{j=1}^{m} \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_j\right)^2}{\left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j\right)} \right\rangle,$$

as shown in Appendix A. $\langle \cdot \rangle$ denotes the time average.

The norm of the approximation is maximized by determining the optimal basis functions that maximize the functional

$$J[\boldsymbol{\varphi}] \equiv \left\langle \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}\right)^2}{(\boldsymbol{\varphi}, \boldsymbol{\varphi})} \right\rangle, \tag{2.1}$$

where the subscript j has been removed for convenience. Using the notation $\hat{\mathbf{A}}(t) \equiv \tilde{\mathbf{U}}(\mathbf{x}, t) \otimes \tilde{\mathbf{U}}(\mathbf{x}, t)$ yields $\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}\right)^2 \equiv \boldsymbol{\varphi}^T \hat{\mathbf{A}} \boldsymbol{\varphi}$, so that (2.1) becomes

$$J[\boldsymbol{\varphi}] \equiv \left\langle \frac{\boldsymbol{\varphi}^T \hat{\mathbf{A}}(t) \boldsymbol{\varphi}}{(\boldsymbol{\varphi}, \boldsymbol{\varphi})} \right\rangle. \tag{2.2}$$

As shown in Appendix B, (2.2) is extremized when

$$\left\langle \frac{\hat{\mathbf{A}}(t)\boldsymbol{\varphi}}{(\boldsymbol{\varphi}, \boldsymbol{\varphi})} - \frac{(\boldsymbol{\varphi}^T \hat{\mathbf{A}}(t) \boldsymbol{\varphi})\boldsymbol{\varphi}}{(\boldsymbol{\varphi}, \boldsymbol{\varphi})^2} \right\rangle = \mathbf{0}.$$

Additionally, for a non-trivial solution for $\boldsymbol{\varphi}$, it is necessary that

$$\left| \left\langle \frac{\hat{\mathbf{A}}(t)}{(\boldsymbol{\varphi}, \boldsymbol{\varphi})} - \frac{(\boldsymbol{\varphi}^T \hat{\mathbf{A}}(t) \boldsymbol{\varphi})\mathbf{I}}{(\boldsymbol{\varphi}, \boldsymbol{\varphi})^2} \right\rangle \right| = 0. \tag{2.3}$$

Equation (2.3) is an eigenvalue problem, in which

$$\lambda = \left\langle \frac{\boldsymbol{\varphi}^T \hat{\mathbf{A}}(t) \boldsymbol{\varphi}}{(\boldsymbol{\varphi}, \boldsymbol{\varphi})} \right\rangle = \left\langle \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}\right)^2}{(\boldsymbol{\varphi}, \boldsymbol{\varphi})} \right\rangle \tag{2.4}$$

are the eigenvalues and $\boldsymbol{\varphi}$ are the eigenvectors of $\langle \hat{\mathbf{A}} \rangle$. If the eigenvectors are normalized so that $\|\boldsymbol{\varphi}\| = 1$, $\lambda = \langle a(t)^2 \rangle$. Consequently, the eigenvectors with the largest eigenvalues are the most significant basis functions. Additionally, since $\hat{\mathbf{A}}$ is symmetric positive semidefinite, the eigenvectors are orthogonal.

Assuming the number of snapshots, M, is less than the number of unknown values in each snapshot, n, the eigenvalue problem can be further simplified using the method of snapshots [35]. The basis functions are expressed as linear combinations of the mean-subtracted snapshots, $\tilde{\mathbf{U}}$, and the matrix is reduced from $n \times n$ to $M \times M$.

2.2 Dynamic Average

Instead of subtracting a static average, $\bar{\mathbf{U}}(\mathbf{x})$, as is done in the standard approach, subtracting an optimal dynamic average of the form

$$\bar{\mathbf{U}}(\mathbf{x}; \boldsymbol{\Gamma}) \equiv \bar{\mathbf{U}}_0(\mathbf{x}) + \sum_{k=1}^{d} \gamma_k \bar{\mathbf{U}}_k(\mathbf{x}) \tag{2.5}$$

is proposed. In (2.5), $\boldsymbol{\Gamma} \equiv \{\gamma_1, \ldots, \gamma_d\}^T$ consists of time-dependent parameters associated with the flow unsteadiness and/or mesh deformation. For example, the elements of $\boldsymbol{\Gamma}$ can be associated with the pitching or plunging of an airfoil or the modal coefficients of a deformable structure. The elements can consist of time derivatives, powers, and products of the measured quantities, provided all elements are linearly independent.

The dynamic average is considered optimal when the functional

$$J\left[\bar{\mathbf{U}}_0,\ldots,\bar{\mathbf{U}}_d\right] \equiv \left\langle \left\| \mathbf{U} - \bar{\mathbf{U}}_0 - \sum_{k=1}^{d} \gamma_k \bar{\mathbf{U}}_k \right\|^2 \right\rangle, \tag{2.6}$$

which measures the time-averaged difference between $\mathbf{U}$ and the dynamic average, is minimized. Equation (2.6) is extremized by solving

$$\frac{\partial J}{\partial \delta}\left[\bar{\mathbf{U}}_0,\ldots,\bar{\mathbf{U}}_{k-1},\bar{\mathbf{U}}_k+\delta\boldsymbol{\phi},\bar{\mathbf{U}}_{k+1},\ldots,\bar{\mathbf{U}}_d\right]\Big|_{\delta=0} = 0, \qquad 0 \le k \le d. \tag{2.7}$$

The system of equations (2.7) reduces to

$$\left\langle \begin{Bmatrix} \gamma_0 \\ \vdots \\ \gamma_d \end{Bmatrix} \begin{Bmatrix} \gamma_0 \\ \vdots \\ \gamma_d \end{Bmatrix}^T \right\rangle \begin{Bmatrix} \bar{\mathbf{U}}_0 \\ \vdots \\ \bar{\mathbf{U}}_d \end{Bmatrix} = \begin{Bmatrix} \langle \gamma_0 \mathbf{U} \rangle \\ \vdots \\ \langle \gamma_d \mathbf{U} \rangle \end{Bmatrix},$$

where $\gamma_0 \equiv 1$.

2.3 Dynamic Basis Functions

In addition to employing a dynamic average, the use of dynamic basis functions is also proposed. Similarly, the basis functions take the form

$$\boldsymbol{\varphi}_j(\mathbf{x};\boldsymbol{\Gamma}) \equiv \tilde{\boldsymbol{\varphi}}_0^j(\mathbf{x}) + \sum_{k=1}^{d} \gamma_k \tilde{\boldsymbol{\varphi}}_k^j(\mathbf{x}). \tag{2.8}$$

Instead of one unknown spatial function for each basis function, there are now $d+1$: $\{\tilde{\boldsymbol{\varphi}}_0^j,\ldots,\tilde{\boldsymbol{\varphi}}_d^j\}$.

2.3.1 Optimization

The basis functions will be determined individually, with the average and the projection of the previously obtained basis functions being subtracted from $\mathbf{U}$, such

that, for $\boldsymbol{\varphi}_j$,

$$\tilde{\mathbf{U}} \equiv \mathbf{U} - \bar{\mathbf{U}} - \sum_{i=1}^{j-1} a_i \boldsymbol{\varphi}_i. \tag{2.9}$$

In (2.9), $[\mathbf{a}]_k \equiv a_k$ is computed from $\mathbf{Pa} = \mathbf{b}$, where $[\mathbf{P}]_{ik} \equiv (\boldsymbol{\varphi}_i, \boldsymbol{\varphi}_k)$ and $[\mathbf{b}]_i \equiv \left(\boldsymbol{\varphi}_i, \mathbf{U} - \bar{\mathbf{U}}\right)$ for $1 \leq i, k \leq j-1$. Therefore, each subsequent basis function can be obtained by extremizing (2.1). Using (2.8) and the identities $\boldsymbol{\varphi}^T \hat{\mathbf{A}} \boldsymbol{\varphi} \equiv \tilde{\boldsymbol{\varphi}}^T \mathbf{A} \tilde{\boldsymbol{\varphi}}$ and $(\boldsymbol{\varphi}, \boldsymbol{\varphi}) \equiv \tilde{\boldsymbol{\varphi}}^T \mathbf{B} \tilde{\boldsymbol{\varphi}}$, where $\gamma_0 \equiv 1$,

$$\tilde{\boldsymbol{\varphi}} \equiv \begin{Bmatrix} \tilde{\boldsymbol{\varphi}}_0 \\ \vdots \\ \tilde{\boldsymbol{\varphi}}_d \end{Bmatrix}, \qquad \mathbf{A}(t) \equiv \begin{bmatrix} \gamma_0\gamma_0\hat{\mathbf{A}} & \cdots & \gamma_0\gamma_d\hat{\mathbf{A}} \\ \vdots & \ddots & \vdots \\ \gamma_d\gamma_0\hat{\mathbf{A}} & \cdots & \gamma_d\gamma_d\hat{\mathbf{A}} \end{bmatrix}, \qquad \mathbf{B}(t) \equiv \begin{bmatrix} \gamma_0\gamma_0\mathbf{I} & \cdots & \gamma_0\gamma_d\mathbf{I} \\ \vdots & \ddots & \vdots \\ \gamma_d\gamma_0\mathbf{I} & \cdots & \gamma_d\gamma_d\mathbf{I} \end{bmatrix},$$

Equation (2.2) becomes

$$J[\tilde{\boldsymbol{\varphi}}] = \left\langle \frac{\tilde{\boldsymbol{\varphi}}^T \mathbf{A}(t) \tilde{\boldsymbol{\varphi}}}{\tilde{\boldsymbol{\varphi}}^T \mathbf{B}(t) \tilde{\boldsymbol{\varphi}}} \right\rangle. \tag{2.10}$$

Equation (2.10) is extremized analogously to the manner used with the standard approach in Subsection 2.1, such that

$$\left| \left\langle \frac{\mathbf{A}(t)}{\tilde{\boldsymbol{\varphi}}^T \mathbf{B}(t) \tilde{\boldsymbol{\varphi}}} - \frac{\left(\tilde{\boldsymbol{\varphi}}^T \mathbf{A}(t) \tilde{\boldsymbol{\varphi}}\right) \mathbf{B}(t)}{\left(\tilde{\boldsymbol{\varphi}}^T \mathbf{B}(t) \tilde{\boldsymbol{\varphi}}\right)^2} \right\rangle \right| = 0. \tag{2.11}$$

Equation (2.11) is a generalized eigenvalue problem

$$\tilde{\mathbf{A}} \tilde{\boldsymbol{\varphi}} = \lambda \tilde{\mathbf{B}} \tilde{\boldsymbol{\varphi}}, \tag{2.12}$$

where

$$\tilde{\mathbf{A}} \equiv \left\langle \frac{\mathbf{A}(t)}{\tilde{\boldsymbol{\varphi}}^T \mathbf{B}(t) \tilde{\boldsymbol{\varphi}}} \right\rangle, \qquad \hat{\mathbf{B}} \equiv \left\langle \frac{\left(\tilde{\boldsymbol{\varphi}}^T \mathbf{A}(t) \tilde{\boldsymbol{\varphi}}\right) \mathbf{B}(t)}{\left(\tilde{\boldsymbol{\varphi}}^T \mathbf{B}(t) \tilde{\boldsymbol{\varphi}}\right)^2} \right\rangle,$$

$$\lambda \equiv \left\langle \frac{\tilde{\boldsymbol{\varphi}}^T \mathbf{A}(t) \tilde{\boldsymbol{\varphi}}}{\tilde{\boldsymbol{\varphi}}^T \mathbf{B}(t) \tilde{\boldsymbol{\varphi}}} \right\rangle, \qquad \tilde{\mathbf{B}} \equiv \frac{\hat{\mathbf{B}}}{\lambda}. \tag{2.13}$$

The eigenvectors can be determined by using an iterative eigenvalue algorithm that computes the eigenvector corresponding to the most dominant eigenvalue. The remaining basis functions can be determined by updating $\tilde{\mathbf{U}}$ and recomputing $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{B}}$.

2.3.2 Problem Size Reduction

The dimension of the matrices in (2.13) is $(d+1)n \times (d+1)n$ and therefore large. As previously mentioned, it is assumed the number of snapshots, M, is less than n.

Letting $\alpha(t) \equiv \tilde{\boldsymbol{\varphi}}^T \mathbf{A}(t) \tilde{\boldsymbol{\varphi}} \equiv \left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}\right)^2$, $\beta(t) \equiv \tilde{\boldsymbol{\varphi}}^T \mathbf{B}(t) \tilde{\boldsymbol{\varphi}} \equiv (\boldsymbol{\varphi}, \boldsymbol{\varphi})$, and

$$\mathbf{G} \equiv \begin{bmatrix} \tilde{\mathbf{G}}_0 \\ \vdots \\ \tilde{\mathbf{G}}_d \end{bmatrix} \in \mathbb{R}^{(d+1)n \times M}; \qquad \mathbf{H}(t) \equiv \begin{bmatrix} \tilde{\mathbf{H}}_0(t) \\ \vdots \\ \tilde{\mathbf{H}}_d(t) \end{bmatrix} \in \mathbb{R}^{(d+1)n \times n};$$

$$\left[\tilde{\mathbf{G}}_k\right]_{ij} \equiv \gamma_k(t_j) \frac{\left[\tilde{\mathbf{U}}(\mathbf{x}, t_j)\right]_i}{\sqrt{\beta(t_j)}}, \quad \begin{matrix} 1 \le i \le n \\ 1 \le j \le M \end{matrix}; \qquad \tilde{\mathbf{H}}_k(t) \equiv \gamma_k(t) \frac{\sqrt{\alpha(t)}}{\beta(t)} \mathbf{I} \in \mathbb{R}^{n \times n};$$

it holds that $\tilde{\mathbf{A}} = \frac{1}{M} \mathbf{G}\mathbf{G}^T$ and $\hat{\mathbf{B}} = \left\langle \mathbf{H}\mathbf{H}^T \right\rangle$. Furthermore, since $\hat{\mathbf{B}} = \left\langle \mathbf{H}\mathbf{H}^T \right\rangle$ and the requisite linear independence of $\{\gamma_0, \ldots, \gamma_d\}$ ensures the invertibility of $\hat{\mathbf{B}}$, $\hat{\mathbf{B}}$ is symmetric positive definite. Therefore, $\hat{\mathbf{B}}$ can be decomposed through a Cholesky

decomposition: $\hat{\mathbf{B}} = \frac{1}{M}\mathbf{L}\mathbf{L}^T$. Consequently, (2.12) can be written as

$$\mathbf{G}\mathbf{G}^T\tilde{\boldsymbol{\varphi}} = \mathbf{L}\mathbf{L}^T\tilde{\boldsymbol{\varphi}}, \tag{2.14}$$

where $\mathbf{L}$ takes the form

$$\mathbf{L} = \begin{bmatrix} L_{0,0}\mathbf{I} & & & \mathbf{0} \\ L_{1,0}\mathbf{I} & L_{1,1}\mathbf{I} & & \\ \vdots & \vdots & \ddots & \\ L_{d,0}\mathbf{I} & L_{d,1}\mathbf{I} & \cdots & L_{d,d}\mathbf{I} \end{bmatrix} \in \mathbb{R}^{(d+1)n\times(d+1)n}. \tag{2.15}$$

Since $\hat{\mathbf{B}}$ consists of $d+1$ rows and $d+1$ columns of identity submatrices, each of which multiplied by a constant, the cost of the Cholesky decomposition and inversion is inexpensive.

The standard approach to POD can be used to obtain a set of M orthonormal static basis functions $\boldsymbol{\Phi} \equiv [\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_M]$ that span all of the snapshots. Consequently, $\tilde{\mathbf{U}}$ can be expressed as a linear combination of the static basis functions:

$$\tilde{\mathbf{U}}(\mathbf{x}, t) = \sum_{i=1}^{M} \tilde{w}_i(t)\boldsymbol{\phi}_i(\mathbf{x}),$$

or $\tilde{\mathbf{U}} = \boldsymbol{\Phi}\tilde{\mathbf{W}}$, where

$$\tilde{\mathbf{W}} \equiv \left[\tilde{\mathbf{w}}(t_1), \ldots, \tilde{\mathbf{w}}(t_M)\right] \in \mathbb{R}^{M\times M}; \qquad \begin{aligned} \left[\tilde{\mathbf{U}}\right]_{ij} &\equiv \left[\tilde{\mathbf{U}}(\mathbf{x}, t_j)\right]_i, && 1\le i\le n, \quad 1\le j\le M; \\ \left[\boldsymbol{\Phi}\right]_{ij} &\equiv \left[\boldsymbol{\phi}_j(\mathbf{x})\right]_i, && 1\le i\le n, \quad 1\le j\le M; \\ \left[\tilde{\mathbf{W}}\right]_{ij} &\equiv \tilde{w}_i(t_j), && 1\le i\le M, \quad 1\le j\le M. \end{aligned}$$

Furthermore, the number of unknowns associated with the dynamic basis functions can be reduced by expressing each of the $\{\tilde{\boldsymbol{\varphi}}_0, \ldots, \tilde{\boldsymbol{\varphi}}_d\}$ as a linear combination

of the static basis functions:

$$\tilde{\boldsymbol{\varphi}}_k(\mathbf{x}) = \sum_{j=1}^{M} c_j^k \boldsymbol{\phi}_j(\mathbf{x}), \qquad 0 \le k \le d, \tag{2.16}$$

or $\tilde{\boldsymbol{\varphi}} = \hat{\boldsymbol{\Phi}}\mathbf{c}$, where

$$\hat{\boldsymbol{\Phi}} \equiv \begin{bmatrix} \boldsymbol{\Phi} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Phi} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \boldsymbol{\Phi} \end{bmatrix} \in \mathbb{R}^{(d+1)n \times (d+1)M}; \qquad \mathbf{c} \equiv \begin{Bmatrix} \tilde{\mathbf{c}}_0 \\ \vdots \\ \tilde{\mathbf{c}}_d \end{Bmatrix} \in \mathbb{R}^{(d+1)M};$$

$$\left[\tilde{\mathbf{c}}_k\right]_j \equiv c_j^k, \quad 1 \le j \le M.$$

Multiplying (2.14) by $\hat{\boldsymbol{\Phi}}^T$ and substituting (2.16) yields

$$\hat{\boldsymbol{\Phi}}^T \mathbf{G}\mathbf{G}^T \hat{\boldsymbol{\Phi}}\mathbf{c} = \hat{\boldsymbol{\Phi}}^T \mathbf{L}\mathbf{L}^T \hat{\boldsymbol{\Phi}}\mathbf{c}. \tag{2.17}$$

Exploiting the structure of $\hat{\boldsymbol{\Phi}}$ and $\mathbf{L}$ and the orthonormality of $\boldsymbol{\Phi}$: $\boldsymbol{\Phi}^T\boldsymbol{\Phi} = \mathbf{I} \in \mathbb{R}^{M \times M}$, the right-hand side of (2.17) can be reduced to $\tilde{\mathbf{L}}\tilde{\mathbf{L}}^T\mathbf{c}$, where $\tilde{\mathbf{L}}$ has the same form as $\mathbf{L}$ (2.15), except the identity submatrices are $M \times M$ instead of $n \times n$.

Equation (2.17) becomes

$$\hat{\boldsymbol{\Phi}}^T \mathbf{G}\mathbf{G}^T \hat{\boldsymbol{\Phi}}\mathbf{c} = \tilde{\mathbf{L}}\tilde{\mathbf{L}}^T\mathbf{c},$$

or

$$\tilde{\mathbf{L}}^{-1}\hat{\boldsymbol{\Phi}}^T \mathbf{G}\mathbf{G}^T \hat{\boldsymbol{\Phi}}\mathbf{c} = \tilde{\mathbf{L}}^T\mathbf{c}. \tag{2.18}$$

Introducing the transformation $\mathbf{b} \equiv \tilde{\mathbf{L}}^T \mathbf{c}$, (2.18) can be written as

$$\tilde{\mathbf{L}}^{-1} \hat{\mathbf{\Phi}}^T \mathbf{G} \mathbf{G}^T \hat{\mathbf{\Phi}} \tilde{\mathbf{L}}^{-T} \mathbf{b} = \mathbf{b},$$

or more compactly as

$$\mathbf{C}\mathbf{C}^T \mathbf{b} = \mathbf{b}, \tag{2.19}$$

where

$$\mathbf{C} \equiv \tilde{\mathbf{L}}^{-1} \hat{\mathbf{\Phi}}^T \mathbf{G} \equiv \begin{bmatrix} \tilde{\mathbf{C}}_0 \\ \vdots \\ \tilde{\mathbf{C}}_d \end{bmatrix} \in \mathbb{R}^{(d+1)M \times M}; \qquad \left[\tilde{\mathbf{C}}_k\right]_{ij} \equiv \sum_{\ell=0}^{k} L^{-1}_{k,\ell} \frac{\gamma_\ell(t_j) \left[\tilde{\mathbf{W}}\right]_{ij}}{\sqrt{\beta(t_j)}}, \quad \begin{matrix} 1 \le i \le M \\ 1 \le j \le M \end{matrix}.$$

Equation (2.19) can be solved using nonlinear iterative partial least squares [36, 37], thus avoiding the computation of $\mathbf{C}\mathbf{C}^T$. $\tilde{\mathbf{A}}$ in (2.12) is $(d+1)n \times (d+1)n$, whereas $\mathbf{C}$ in (2.19) is $(d+1)M \times M$. Additionally the number of unknowns associated with each basis function has been reduced from $(d+1)n$ to $(d+1)M$. $\tilde{\boldsymbol{\varphi}}$ is recovered through the inverse transformation:

$$\tilde{\boldsymbol{\varphi}} = \hat{\mathbf{\Phi}} \tilde{\mathbf{L}}^{-T} \mathbf{b}.$$

Because $\tilde{\mathbf{U}}$ and $\tilde{\boldsymbol{\varphi}}$ are expressed in terms of static basis functions, $\tilde{\mathbf{U}}$ does not need to be computed explicitly from (2.9), and $\tilde{\boldsymbol{\varphi}}$ only needs to be computed once $\mathbf{c}$ is known. Instead of computing $\tilde{\mathbf{U}}$, $\tilde{\mathbf{w}}$ is computed:

$$\tilde{\mathbf{w}} = \mathbf{w} - \sum_{i=1}^{j-1} a_i \hat{\mathbf{c}}_i,$$

where

$$\mathbf{w} \equiv \mathbf{\Phi}^T\Big(\mathbf{U} - \bar{\mathbf{U}}\Big), \qquad \hat{\mathbf{c}} \equiv \sum_{k=0}^{d} \gamma_k \tilde{\mathbf{c}}_k,$$

$$[\mathbf{P}]_{ik} \equiv (\boldsymbol{\varphi}_i, \boldsymbol{\varphi}_k) = (\hat{\mathbf{c}}_i, \hat{\mathbf{c}}_k)\,, \qquad [\mathbf{b}]_i \equiv \Big(\boldsymbol{\varphi}_i, \mathbf{U} - \bar{\mathbf{U}}\Big) = (\hat{\mathbf{c}}_i, \mathbf{w})\,.$$

Additionally, due to the orthonormality of $\mathbf{\Phi}$, α and β can be computed more efficiently from

$$\alpha(t) \equiv \Big(\tilde{\mathbf{U}}, \boldsymbol{\varphi}\Big)^2 = \Big(\tilde{\mathbf{w}}, \hat{\mathbf{c}}\Big)^2, \qquad \beta(t) \equiv (\boldsymbol{\varphi}, \boldsymbol{\varphi}) = \Big(\hat{\mathbf{c}}, \hat{\mathbf{c}}\Big).$$

2.3.3 Eigenvalue Algorithm

The eigenvalue algorithm is begun by computing $\mathbf{\Phi}$ and $\mathbf{w}$. Each basis function, $\boldsymbol{\varphi}_j$, is computed as follows:

- Compute $\tilde{\mathbf{w}} \equiv \mathbf{w} - \sum_{i=1}^{j-1} a_i \hat{\mathbf{c}}_i$
- Until convergence is achieved with $\lambda = \left\langle \dfrac{\alpha}{\beta} \right\rangle$,
  - Compute α and β
  - Compute $\tilde{\mathbf{L}}$ and $\mathbf{C}$
  - Transform $\mathbf{c}$ into $\mathbf{b}$: $\mathbf{b} \equiv \tilde{\mathbf{L}}^T\mathbf{c}$
  - Compute the score vector $\mathbf{t}$: $\mathbf{t} = \mathbf{C}^T\mathbf{b}$
  - Update the eigenvector: $\mathbf{b} = \mathbf{C}\mathbf{t}$
  - Transform $\mathbf{b}$ back to $\mathbf{c}$ and normalize: $\mathbf{c} \equiv \tilde{\mathbf{L}}^{-T}\mathbf{b} / \Big\|\tilde{\mathbf{L}}^{-T}\mathbf{b}\Big\|$
- Upon convergence, compute $\tilde{\boldsymbol{\varphi}} = \hat{\mathbf{\Phi}}\mathbf{c}$, normalize, and obtain $\boldsymbol{\varphi}_j$ from $\tilde{\boldsymbol{\varphi}}$

The solution to each optimal dynamic basis function that maximizes (2.1) is $\boldsymbol{\varphi} = \tilde{\mathbf{U}}$. Therefore, for the initial iteration in determining each basis function, $\alpha = \left(\tilde{\mathbf{U}}, \tilde{\mathbf{U}}\right)^2 = \left(\tilde{\mathbf{w}}, \tilde{\mathbf{w}}\right)^2$ and $\beta = \left(\tilde{\mathbf{U}}, \tilde{\mathbf{U}}\right) = \left(\tilde{\mathbf{w}}, \tilde{\mathbf{w}}\right)$. The $\mathbf{b}$ corresponding to the most dominant eigenvalue in (2.19) is computed, and the iteration process is initiated.

2.3.4 Limited-Memory Broyden–Fletcher–Goldfarb–Shanno Algorithm

The preceding two subsections discussed what will be referred to as the eigenvalue algorithm for extremizing (2.10). As an alternative to the eigenvalue algorithm, the limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm [38, 39] is considered.

The L-BFGS algorithm is a quasi-Newton optimization method that circumvents the need to explicitly store the large, $(d+1)n \times (d+1)n$ Hessian matrix, making it one of the few general optimization algorithms suitable for this large-scale optimization problem.

2.4 Static Basis Function Interpolation

The ROM requires spatial functions, which have thus far been the basis functions obtained from applying POD to the snapshots generated by the full-order model. Running the FOM for every ROM case is counter to the motivation behind ROMs. Therefore, as an alternative, functions can be generated by interpolating between static basis functions corresponding to flow parameters that bracket the conditions of interest. In this dissertation, the static basis functions for off-reference conditions are generated through interpolation on a tangent space to a Grassmann manifold [12, 13]. The resulting basis functions are orthogonal.

For a set of L simulations corresponding to different flow conditions parameterized by χ, a set of basis functions, $\boldsymbol{\Phi}_i \equiv [\boldsymbol{\varphi}_1, \ldots, \boldsymbol{\varphi}_m]_i$, is generated for each simulation i.

One of the $\mathbf{\Phi}_i$ is taken to be the reference point on the Grassmann manifold and the origin point for interpolation, $\mathbf{\Phi}_0$. Each of the remaining $\mathbf{\Phi}_i$ is logarithmically mapped using thin singular value decomposition,

$$\left(\mathbf{I} - \mathbf{\Phi}_0 \mathbf{\Phi}_0^H\right) \mathbf{\Phi}_i \left(\mathbf{\Phi}_0^H \mathbf{\Phi}_i\right)^{-1} = \mathbf{U}_i \mathbf{\Sigma}_i \mathbf{V}_i^H,$$

to the tangent space at the reference point:

$$\mathbf{\Gamma}_i = \mathbf{U}_i \tan^{-1} \mathbf{\Sigma}_i \mathbf{V}_i^H.$$

$\mathbf{V}_i^H$ denotes the conjugate transpose of $\mathbf{V}_i$. On the tangent space, the interpolation is performed for the condition of interest, χ_ℓ:

$$\mathbf{\Gamma}_\ell = \sum_{i=1}^{L} \alpha_i \mathbf{\Gamma}_i,$$

where α_i are the coefficients arising from Lagrangian interpolation with respect to χ. $\mathbf{\Phi}_\ell$ is then obtained through an exponential mapping of $\mathbf{\Gamma}_\ell$:

$$\mathbf{\Gamma}_\ell = \mathbf{U}_\ell \mathbf{\Sigma}_\ell \mathbf{V}_\ell^H,$$
$$\mathbf{\Phi}_\ell = \mathbf{\Phi}_0 \mathbf{V}_\ell \cos \mathbf{\Sigma}_\ell + \mathbf{U}_\ell \sin \mathbf{\Sigma}_\ell.$$

3. FLOW SOLVER

In this section, the physics and discretization of the flow model are discussed, and the reduced-order model is derived.

3.1 Physical Model

Fluid flow is governed by the conservation of mass, momentum, and energy. For three-dimensional viscous flow, in the absence of source terms, these axioms can be expressed in integral form as

$$\frac{\partial}{\partial t}\int_{\Omega(t)} \mathbf{U} d\Omega + \oint_{\partial\Omega(t)} (\mathbf{F}_c - \mathbf{F}_v)\,\mathbf{n}\, dS = \mathbf{0}, \tag{3.1}$$

where

$$\mathbf{U} \equiv \begin{Bmatrix} \rho \\ \rho\mathbf{v} \\ \rho E \end{Bmatrix}, \qquad \mathbf{F}_c \equiv \begin{bmatrix} \rho\left(\mathbf{v}-\mathbf{v}_g\right)^T \\ \rho\mathbf{v}\left(\mathbf{v}-\mathbf{v}_g\right)^T + p\mathbf{I} \\ \rho E\left(\mathbf{v}-\mathbf{v}_g\right)^T + p\mathbf{v}^T \end{bmatrix}, \qquad \mathbf{F}_v \equiv \begin{bmatrix} \mathbf{0}^T \\ \mathbf{T} \\ \left(\mathbf{T}\mathbf{v} + k\nabla\Theta\right)^T \end{bmatrix},$$

$$\mathbf{v} \equiv u\mathbf{i} + v\mathbf{j} + w\mathbf{k}, \qquad \mathbf{T} \equiv 2\mu\mathbf{D} - \frac{2}{3}\mu\left(\nabla\cdot\mathbf{v}\right)\mathbf{I},$$

and $\mathbf{v}_g$ is the velocity of the boundary of Ω, which satisfies the geometric conservation law [5, 40, 41]:

$$\mathbf{v}_g^T\mathbf{n} = \frac{\Delta\Omega}{S\Delta t}.$$

Using conservative variables for the POD basis functions allows the time derivative to be simplified due to the orthogonality of the basis functions [42]. As a result, $\mathbf{U}$ is captured through snapshots, and the basis functions are computed.

3.2 Full-Order Model

Letting $\mathbf{F} \equiv (\mathbf{F}_c - \mathbf{F}_v)\mathbf{n}$, (3.1) is discretized using finite volumes [43]:

$$\frac{\Delta\left(\Omega_k \mathbf{U}_k\right)}{\Delta t} = -\sum_{\ell=1}^{\text{faces}(k)} \mathbf{F}_{k\ell} S_\ell \equiv -\mathbf{R}_k. \tag{3.2}$$

Equation (3.2) is solved using a Runge–Kutta method with a Roe–Riemann flux-difference splitting scheme.

Mesh deformation is achieved through radial basis function interpolation within the updated boundaries [44].

3.3 Reduced-Order Model

The standard POD approximation for $\mathbf{U}$ is

$$\mathbf{U}(\mathbf{x}, t) \approx \bar{\mathbf{U}}(\mathbf{x}) + \sum_{j=1}^{m} a_j(t)\boldsymbol{\varphi}_j(\mathbf{x}).$$

When the average and/or basis functions are dynamic, a dependency on a set of parameters $\boldsymbol{\Gamma} \equiv \{\gamma_1, \ldots, \gamma_d\}^T$ is introduced. If the dynamic average is used, $\bar{\mathbf{U}}(\mathbf{x})$ is replaced with $\bar{\mathbf{U}}(\mathbf{x}; \boldsymbol{\Gamma})$, where

$$\bar{\mathbf{U}}(\mathbf{x}; \boldsymbol{\Gamma}) \equiv \bar{\mathbf{U}}_0(\mathbf{x}) + \sum_{k=1}^{d} \gamma_k \bar{\mathbf{U}}_k(\mathbf{x}).$$

Similarly, if the dynamic basis functions are used, $\boldsymbol{\varphi}_j(\mathbf{x})$ is replaced with $\boldsymbol{\varphi}_j(\mathbf{x}; \boldsymbol{\Gamma})$, where

$$\boldsymbol{\varphi}_j(\mathbf{x}; \boldsymbol{\Gamma}) \equiv \tilde{\boldsymbol{\varphi}}_0^j(\mathbf{x}) + \sum_{k=1}^{d} \gamma_k \tilde{\boldsymbol{\varphi}}_k^j(\mathbf{x}).$$

Equation (3.2) can be rearranged as follows:

$$\mathbf{U}_k^{n+1} = \frac{\Omega_k^n}{\Omega_k^{n+1}}\mathbf{U}_k^n - \frac{\Delta t}{\Omega_k^{n+1}}\mathbf{R}_k. \tag{3.3}$$

Projecting (3.3) onto the basis yields

$$\mathbf{P}^{n+1}\mathbf{a}^{n+1} = \mathbf{P}^n\mathbf{a}^n - \mathbf{r} - \mathbf{q}, \tag{3.4}$$

if $\Omega_k^n/\Omega_k^{n+1} \approx 1$. In (3.4),

$$[\mathbf{P}]_{ij}^{n+1} \equiv \begin{cases} \left(\boldsymbol{\varphi}_i^{n+1}, \boldsymbol{\varphi}_j^{n+1}\right), & \text{dynamic basis functions} \\ \delta_{ij}, & \text{static basis functions} \end{cases};$$

$$[\mathbf{P}]_{ij}^{n} \equiv \begin{cases} \left(\boldsymbol{\varphi}_i^{n+1}, \boldsymbol{\varphi}_j^{n}\right), & \text{dynamic basis functions} \\ \delta_{ij}, & \text{static basis functions} \end{cases};$$

$$\{\mathbf{a}\}_j \equiv a_j;$$

$$\{\mathbf{r}\}_i \equiv \left(\boldsymbol{\varphi}_i^{n+1}, \frac{\Delta t}{\Omega^{n+1}}\mathbf{R}\right);$$

$$\{\mathbf{q}\}_i \equiv \begin{cases} \left(\boldsymbol{\varphi}_i^{n+1}, \Delta\bar{\mathbf{U}}\right), & \text{dynamic average} \\ 0, & \text{static average} \end{cases}.$$

Equation (3.4) is a system of ordinary differential equations from which each successive $\mathbf{a}$ is computed.

Computation of $\mathbf{P}$ and $\mathbf{q}$ is expedited by computing the inner products of the spatially dependent contributions $\left(\tilde{\boldsymbol{\varphi}}_p^i, \tilde{\boldsymbol{\varphi}}_q^j\right)$ and $\left(\tilde{\boldsymbol{\varphi}}_p^j, \bar{\mathbf{U}}_q\right)$ once and linearly combining

them using the products of the elements of $\mathbf{\Gamma}$:

$$
\begin{aligned}
[\mathbf{P}]_{ij}^{n+1} &\equiv \left(\boldsymbol{\varphi}_i^{n+1}, \boldsymbol{\varphi}_j^{n+1}\right) = \sum_{p=0}^{d}\sum_{q=0}^{d} \gamma_p^{n+1} \left(\tilde{\boldsymbol{\varphi}}_p^i, \tilde{\boldsymbol{\varphi}}_q^j\right) \gamma_q^{n+1}, \\
[\mathbf{P}]_{ij}^{n} &\equiv \left(\boldsymbol{\varphi}_i^{n+1}, \boldsymbol{\varphi}_j^{n}\right) \quad = \sum_{p=0}^{d}\sum_{q=0}^{d} \gamma_p^{n+1} \left(\tilde{\boldsymbol{\varphi}}_p^i, \tilde{\boldsymbol{\varphi}}_q^j\right) \gamma_q^{n}, \\
\{\mathbf{q}\}_j &\equiv \left(\boldsymbol{\varphi}_j^{n+1}, \Delta\bar{\mathbf{U}}\right) \; = \sum_{p=0}^{d'}\sum_{q=1}^{d} \gamma_p^{n+1} \left(\tilde{\boldsymbol{\varphi}}_p^j, \bar{\mathbf{U}}_q\right) \left(\gamma_q^{n+1} - \gamma_q^{n}\right),
\end{aligned}
\tag{3.5}
$$

where $\gamma_0^{n+1} \equiv \gamma_0^n \equiv 1$, and d' is d for dynamic basis functions or 0 for static basis functions.

4. DYNAMIC FUNCTION RESULTS – CHANNEL

The reduced-order model resulting from the application of proper orthogonal decomposition to the governing equations was used to model flow through a channel with a bump, and the results are compared with those that arose from the full-order model.

For these cases, an inviscid flow field was modeled through the 5-meter-long channel with a 1-meter height shown in Figure 4.1. The middle meter of the channel contained a sinusoidal bump with a 0.1-meter height. The channel was discretized using 150 cells along the length and 30 cells along the height.

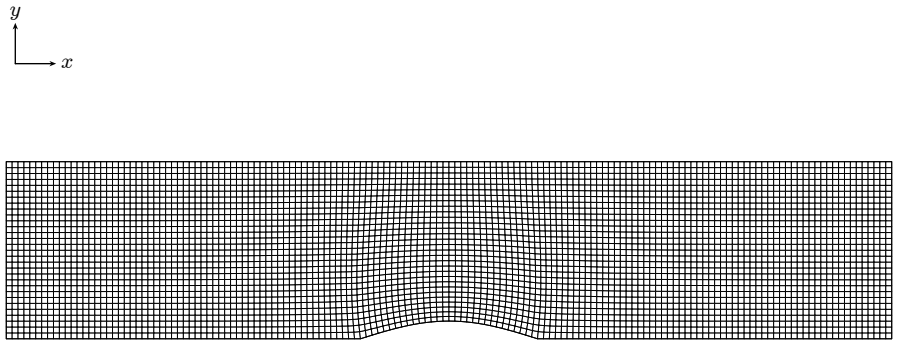

Figure 4.1: Channel mesh for dynamic functions.

Two flow regimes were simulated: a subsonic flow with an inlet Mach number of 0.5, and a transonic flow with an inlet Mach number of 0.75. Two approaches were used to generate unsteady flow: (1) sinusoidally varying the channel back pressure, and (2) sinusoidally varying the channel bump height. In the first case, the computational mesh was fixed, while in the second case, the mesh deformed. ROM results are shown using a static average with static basis functions, a dynamic average with

static basis functions, and a dynamic average with dynamic basis functions.

4.1 Fixed Mesh

For the fixed-mesh case, the static pressure at the channel outlet was varied sinusoidally to force unsteadiness of the flow. The back pressure was prescribed by

$$p_b = \bar{p}_b \left[1 + 0.05 \sin(\omega t)\right],$$

where $\bar{p}_b$ was 101,325 Pa and ω was 60 rad/s. Consequently, the reduced frequency based on half of the bump length was 0.176 for the subsonic case and 0.118 for the transonic case. The basis functions were obtained from 1000 snapshots during the second second, which spanned more than nine periods.

ROM results are shown using a static average with static basis functions, a dynamic average with static basis functions, and a dynamic average with dynamic basis functions. For the dynamic functions, $\mathbf{\Gamma} \equiv \{\gamma,\, \dot{\gamma}\}^T$, where γ and $\dot{\gamma}$ were respectively set equal to the dimensionless back pressure and its time derivative. The dynamic basis functions were computed using the eigenvalue algorithm and the L-BFGS algorithm. In this section, when the algorithm is omitted, the dynamic basis functions were obtained from the eigenvalue algorithm.

4.1.1 Subsonic Flow

Figure 4.2 shows the energy of each basis function,

$$\mathcal{E} \equiv \lambda_i / \sum_{j=1}^{M} \lambda_j, \tag{4.1}$$

as well as the cumulative energy, computed for the subsonic case. In (4.1), $\lambda_j \equiv \left\langle (\tilde{\mathbf{U}}, \boldsymbol{\varphi}_j)/(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j) \right\rangle$, and the average has been subtracted: $\tilde{\mathbf{U}} \equiv \mathbf{U} - \bar{\mathbf{U}}$. Note that $\tilde{\mathbf{U}}$

is different for the static and dynamic average cases since the static and dynamic averages are different. With the dynamic average, the energy of the static and dynamic basis functions obtained from the eigenvalue algorithm was comparable. However, the energy of the dynamic basis functions obtained from the L-BFGS algorithm accumulated energy less rapidly than those obtained from the eigenvalue algorithm.

Figure 4.3 shows the error in the force per unit length acting on the bump along the y-direction for different amounts of basis functions. The error measure for the force is defined to be $\varepsilon_f \equiv \sqrt{\left\langle \left(f_{\mathrm{FOM}} - f_{\mathrm{ROM}}\right)^2 \right\rangle} / \sqrt{\langle f_{\mathrm{FOM}}^2 \rangle}$. Using the dynamic average offered a generally higher fidelity result, often with an error that was an order of magnitude less than with the static average. With the dynamic average, the flow was not better modeled by the dynamic basis functions, and the dynamic basis functions obtained from the eigenvalue algorithm performed better than those obtained from the L-BFGS algorithm.

Additionally, an excerpt of the time history of the force is plotted in Figure 4.4, and contour plots of the Mach number and Mach number error for the subsonic case are presented in Figures 4.5 and 4.6. The Mach number error is defined to be $\varepsilon_M \equiv \left| M_{\mathrm{FOM}} - M_{\mathrm{ROM}} \right| / M_{\mathrm{FOM}}$. These contour plots correspond to the instance when the maximum Mach number occurred. Although the contour plots of the Mach number error shown in Figure 4.6 look differently, the magnitude of the error values was less than 6×10^{-4}. Consequently, Figures 4.4–4.6 show that all of the ROMs accurately predicted the flow field.

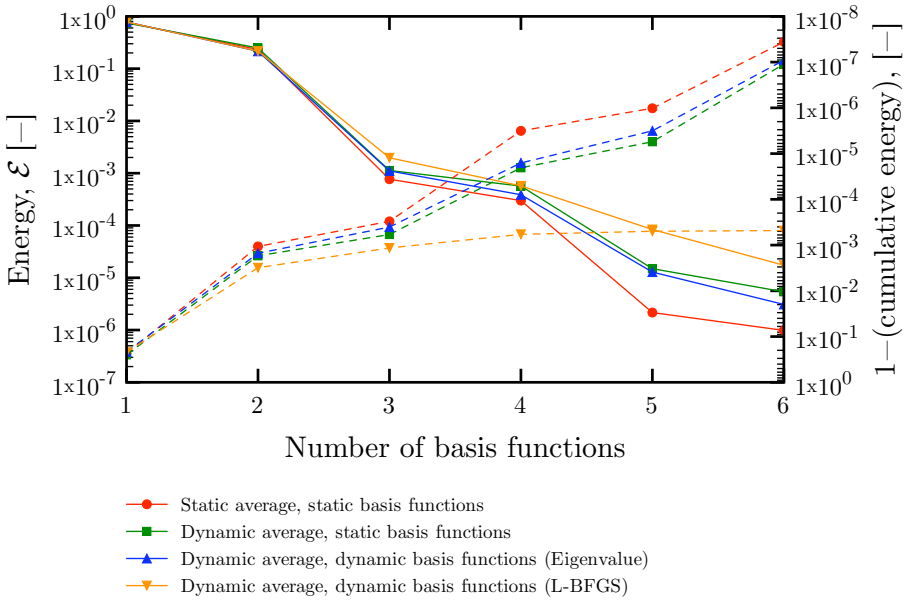

Figure 4.2: Fixed mesh, subsonic case: energy spectrum. Solid lines show energy of each basis function; dashed lines show 1 − (cumulative energy).

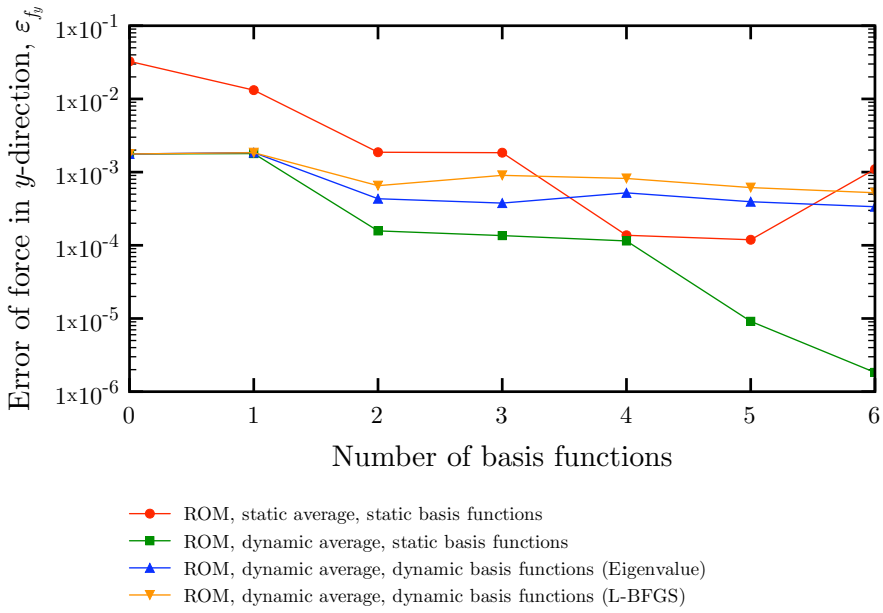

Figure 4.3: Fixed mesh, subsonic case: error in force per unit length acting on bump along y-direction.

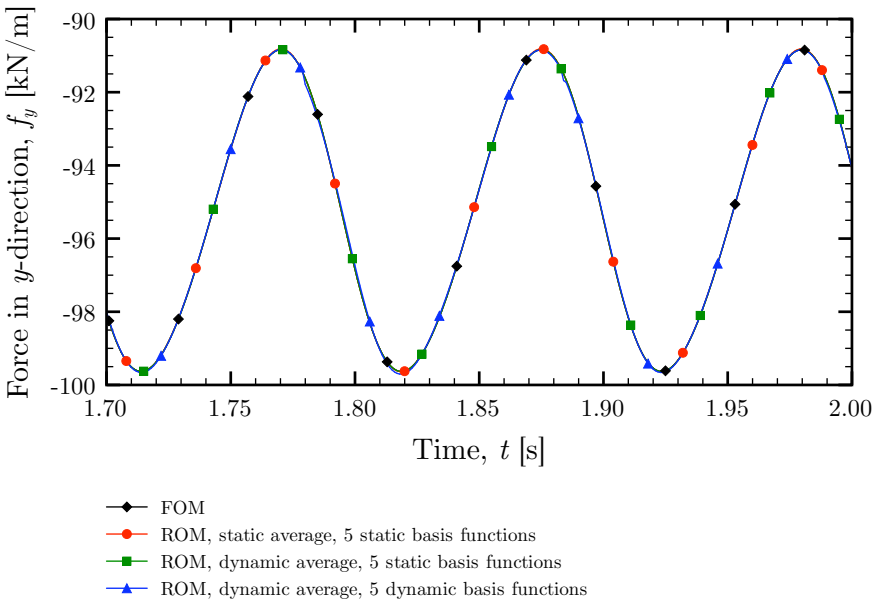

Figure 4.4: Fixed mesh, subsonic case: force per unit length acting on bump along y-direction.

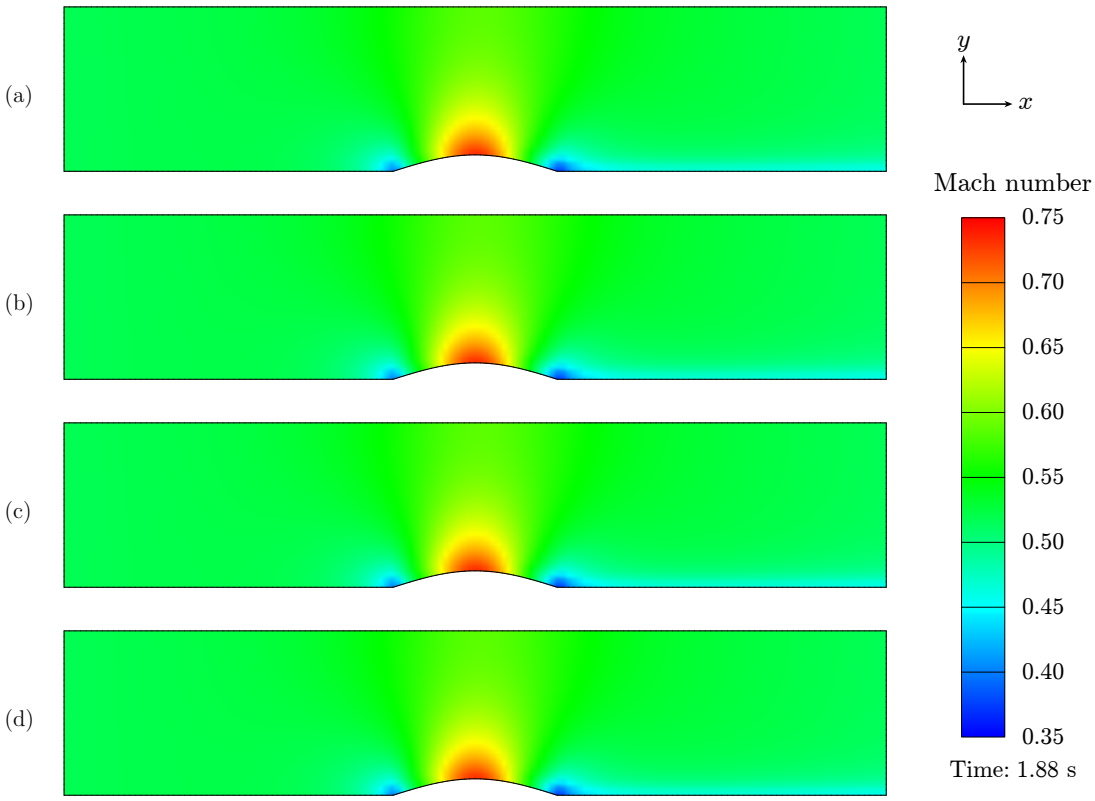

Figure 4.5: Fixed mesh, subsonic case: Mach number contour plots. (a) FOM; (b) ROM, static average, 5 static basis functions; (c) ROM, dynamic average, 5 static basis functions; and (d) ROM, dynamic average, 5 dynamic basis functions.

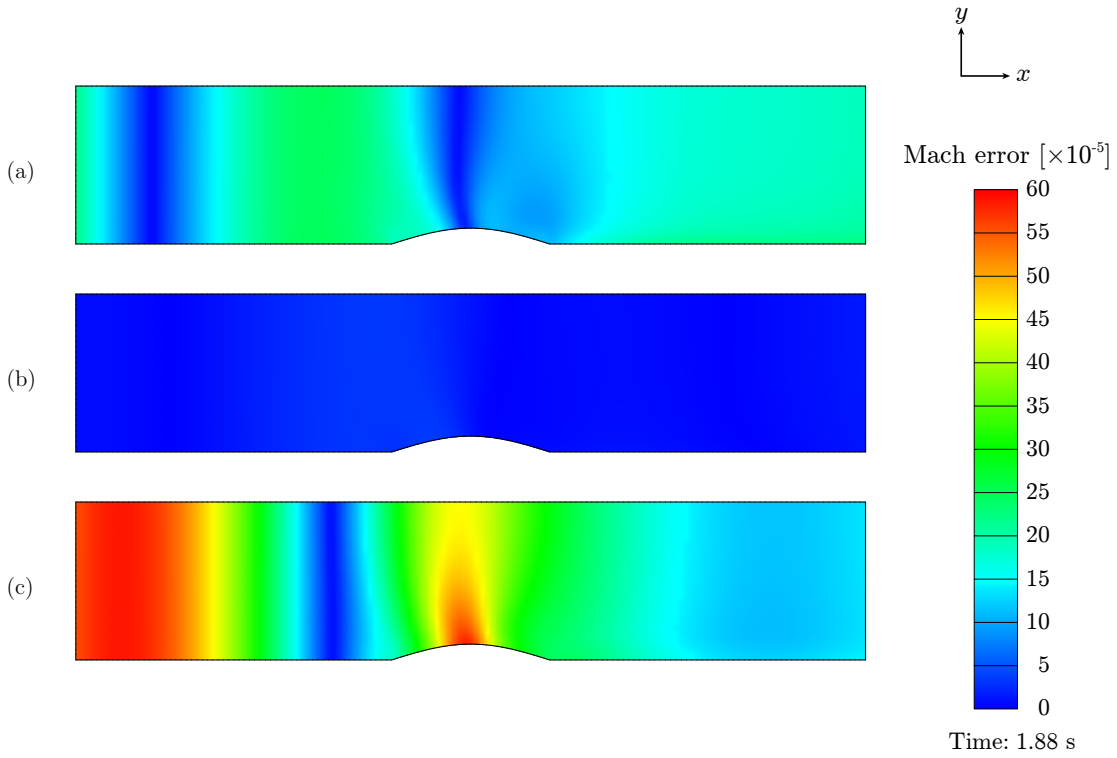

Figure 4.6: Fixed mesh, subsonic case: Mach number error contour plots. (a) ROM, static average, 5 static basis functions; (b) ROM, dynamic average, 5 static basis functions; and (c) ROM, dynamic average, 5 dynamic basis functions.

4.1.2 Transonic Flow

For the transonic case, Figure 4.7 shows the energy of the basis functions. The static basis functions with a dynamic average contained less energy than the dynamic basis functions.

Figure 4.8 shows the error in the force with respect to the number of basis functions. For certain simulations, the ROMs diverged, and therefore the symbol is omitted on the plot. Using the dynamic average with the dynamic basis functions obtained from the eigenvalue algorithm generally performed better than with static basis functions, and using the dynamic average with static basis functions generally performed better than using the static average. Overall, the dynamic basis functions obtained from the eigenvalue algorithm performed better than those obtained from the L-BFGS algorithm.

The time history of the force is plotted in Figure 4.9, and contour plots of the Mach number and Mach number error for the transonic case are presented in Figures 4.10 and 4.11. The contour plots correspond to the instance when the maximum Mach number was achieved. The dynamic basis functions obtained from the eigenvalue algorithm were better able to address the flow nonlinearity and model the shock. Note that the error level for the transonic flow shown in Figure 4.11 is two to three orders of magnitude higher than that of the subsonic flow shown in Figure 4.6.

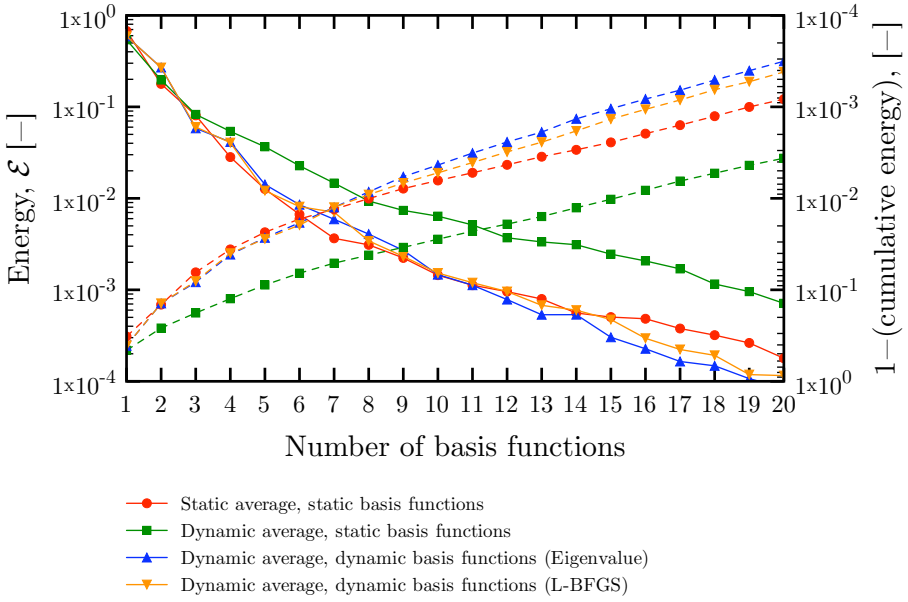

Figure 4.7: Fixed mesh, transonic case: energy spectrum. Solid lines show energy of each basis function; dashed lines show 1 − (cumulative energy).

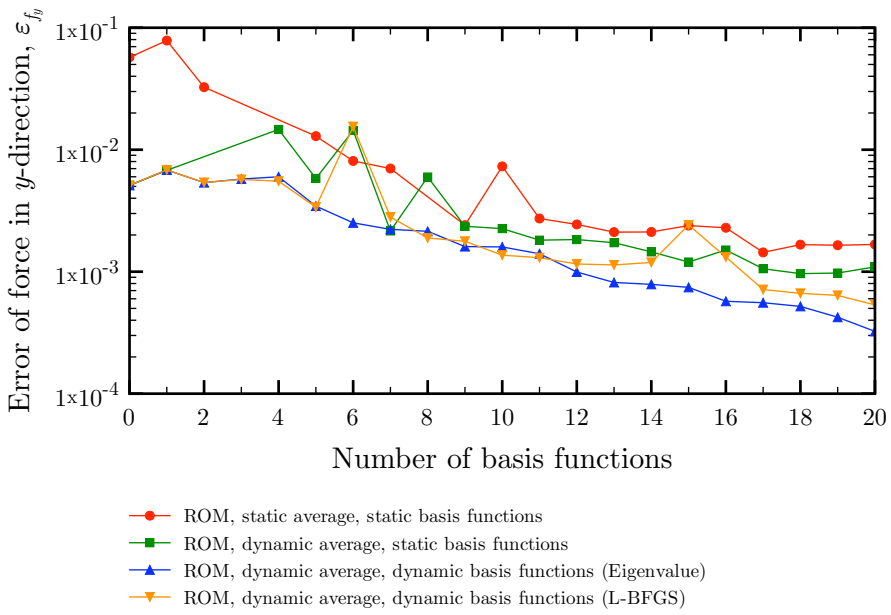

Figure 4.8: Fixed mesh, transonic case: error in force per unit length acting on bump along y-direction.

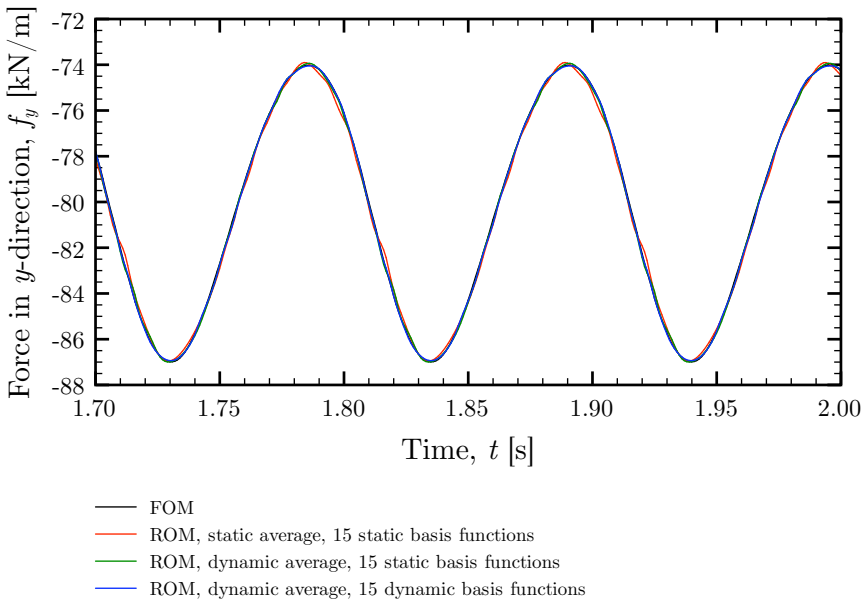

Figure 4.9: Fixed mesh, transonic case: force per unit length acting on bump along y-direction.

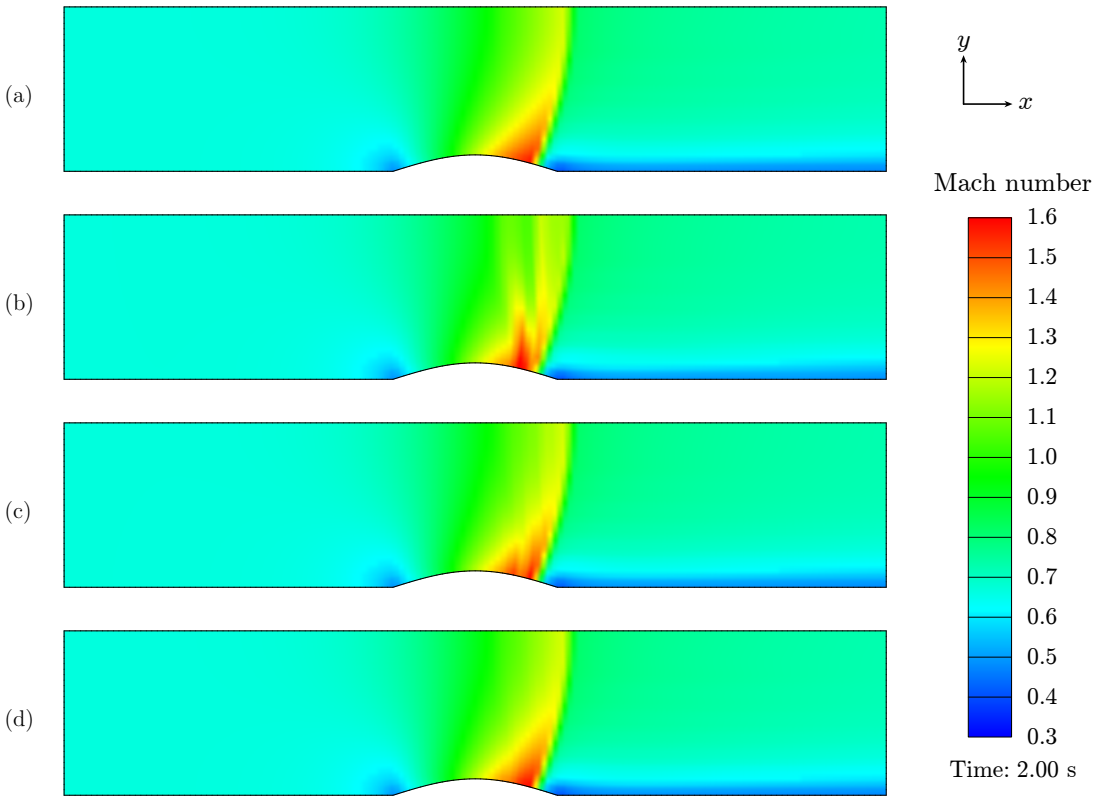

Figure 4.10: Fixed mesh, transonic case: Mach number contour plots. (a) FOM; (b) ROM, static average, 15 static basis functions; (c) ROM, dynamic average, 15 static basis functions; and (d) ROM, dynamic average, 15 dynamic basis functions.

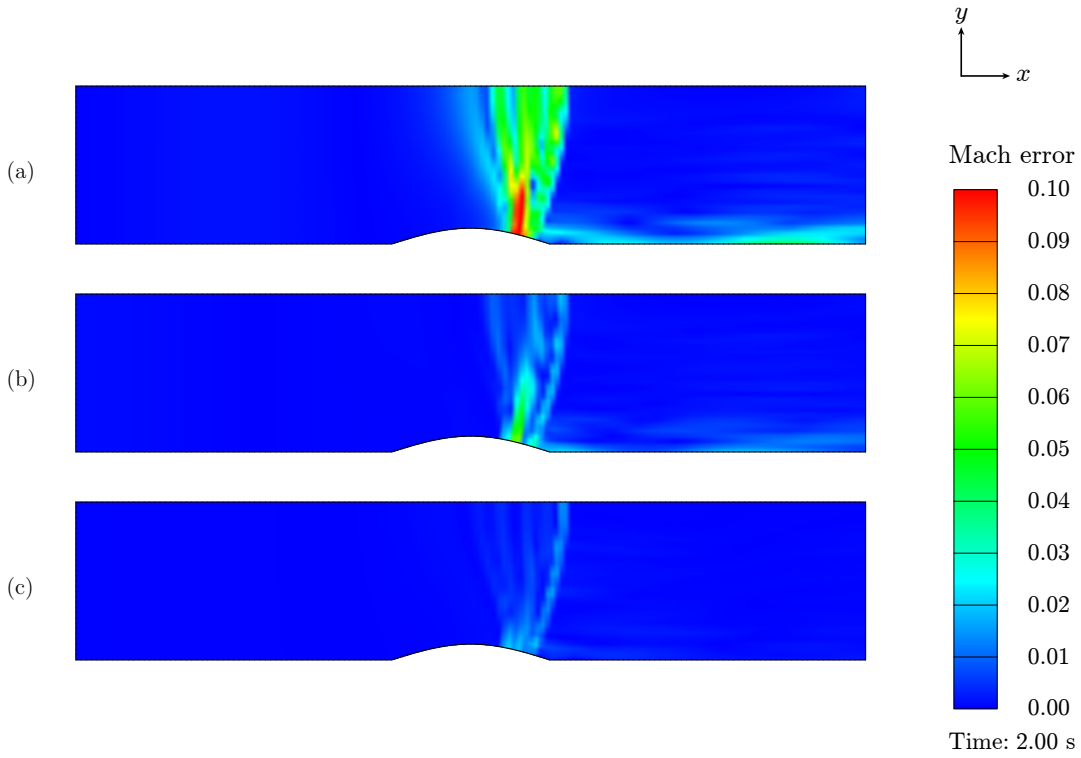

Figure 4.11: Fixed mesh, transonic case: Mach number error contour plots. (a) ROM, static average, 15 static basis functions; (b) ROM, dynamic average, 15 static basis functions; and (c) ROM, dynamic average, 15 dynamic basis functions.

4.2 Deforming Mesh

For the deforming-mesh case, the height of the bump was varied sinusoidally. The height was prescribed by

$$h = h_0 \left(1 + 0.4 \sin[\omega(t - t_0)]\right),$$

where h_0 is the original height distribution, ω is 60 rad/s, and t_0 is 0.01 s.

ROM results are shown arising from using the static average with static basis functions, the dynamic average with static basis functions, and the dynamic average with dynamic basis functions. For the dynamic functions, $\mathbf{\Gamma} \equiv \{\gamma,\, \dot{\gamma}\}^T$, where γ was set equal to $0.4 \sin[\omega(t - t_0)]$, and $\dot{\gamma}$ was the time derivative of γ.

4.2.1 Subsonic Flow

The energy of each basis function and the cumulative energy are shown in Figure 4.12 for the subsonic case. Using the dynamic average, there was little distinction between the energy of the static and dynamic basis functions obtained from the L-BFGS algorithm. Additionally, the energy of the dynamic basis functions obtained from the eigenvalue algorithm accumulated energy more rapidly than those obtained from the L-BFGS algorithm.

Figure 4.13 shows the error of the force per unit length acting on the bump along the y-direction with respect to basis function count. Increasing the number of basis functions did not modify the error level. With the dynamic average, there was little distinction between the use of static and dynamic basis functions. Using the static average with the static basis functions resulted in a better force prediction than using the dynamic average when more than one basis function was used.

An excerpt of the time history of the force is shown in Figure 4.14, and contour

plots of the Mach number and Mach number error for the subsonic case are presented in Figures 4.15 and 4.16. These contour plots correspond to the instance when the maximum Mach number occurred. Each of the ROMs provided reasonable results. Because of the error levels shown in Figure 4.13, only one basis function was used for the dynamic average cases.

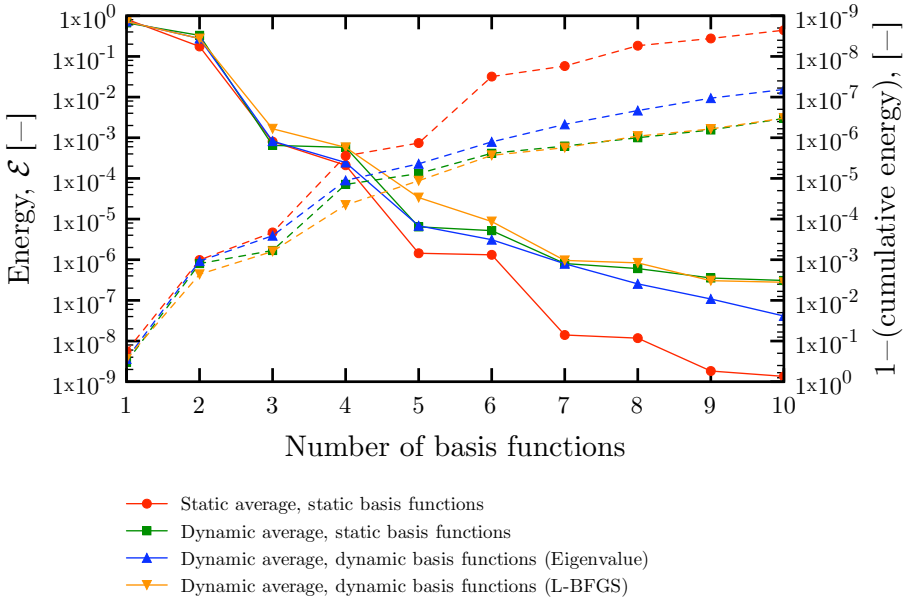

Figure 4.12: Deforming mesh, subsonic case: energy spectrum. Solid lines show energy of each basis function; dashed lines show 1 − (cumulative energy).

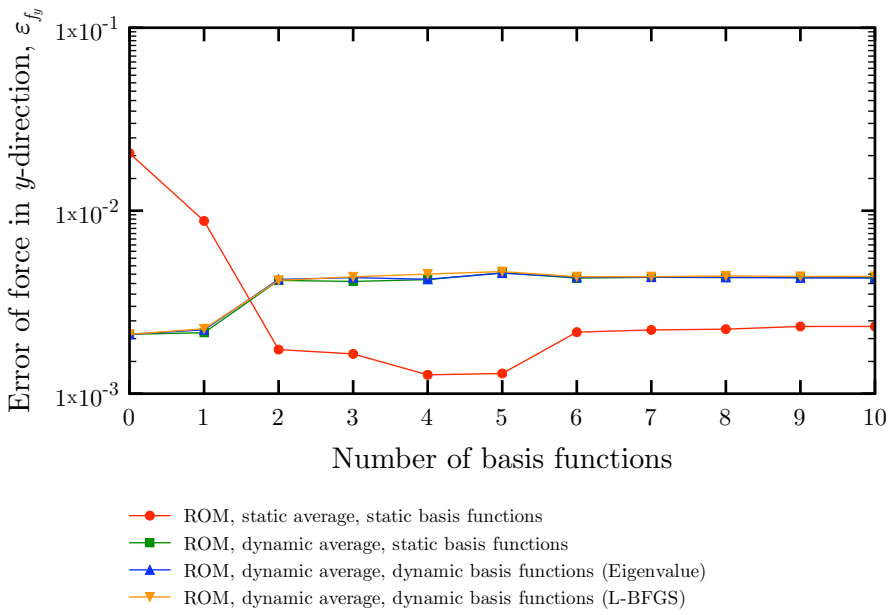

Figure 4.13: Deforming mesh, subsonic case: error in force per unit length acting on bump along y-direction.

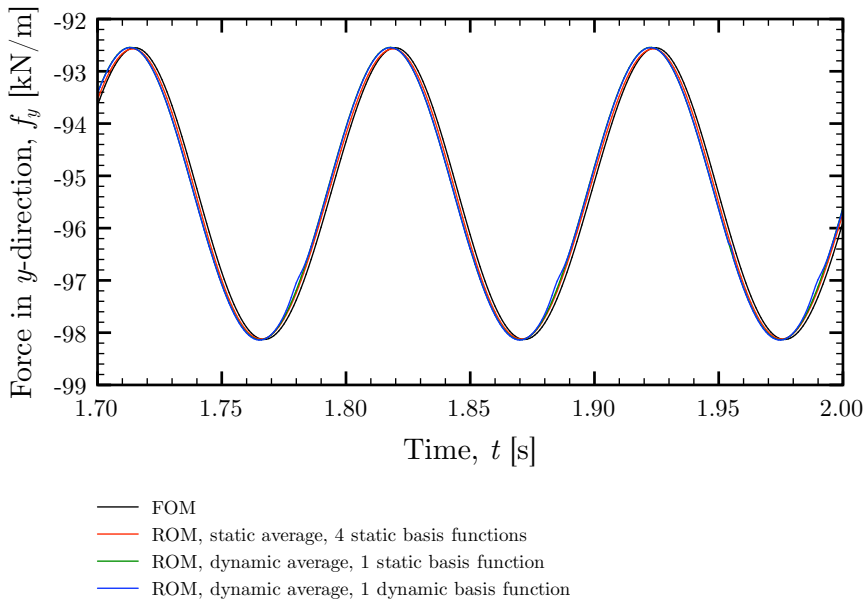

Figure 4.14: Deforming mesh, subsonic case: force per unit length acting on bump along y-direction.

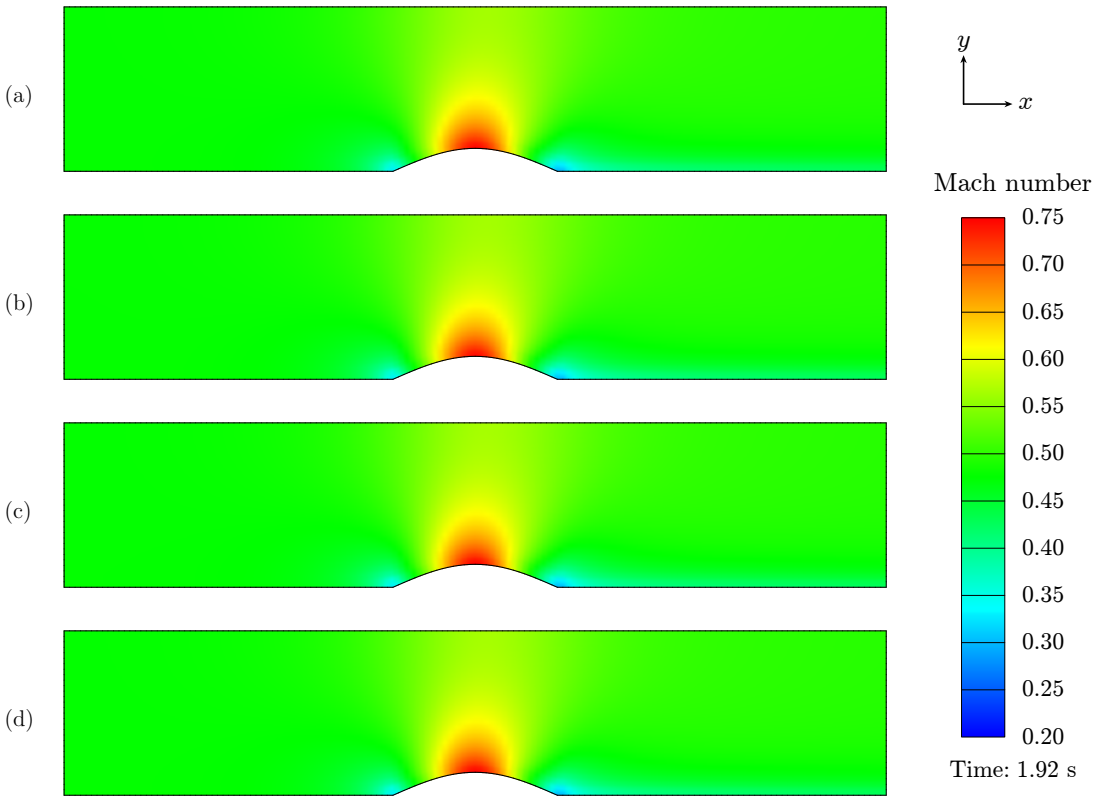

Figure 4.15: Deforming mesh, subsonic case: Mach number contour plots. (a) FOM; (b) ROM, static average, 4 static basis functions; (c) ROM, dynamic average, 1 static basis function; and (d) ROM, dynamic average, 1 dynamic basis function.

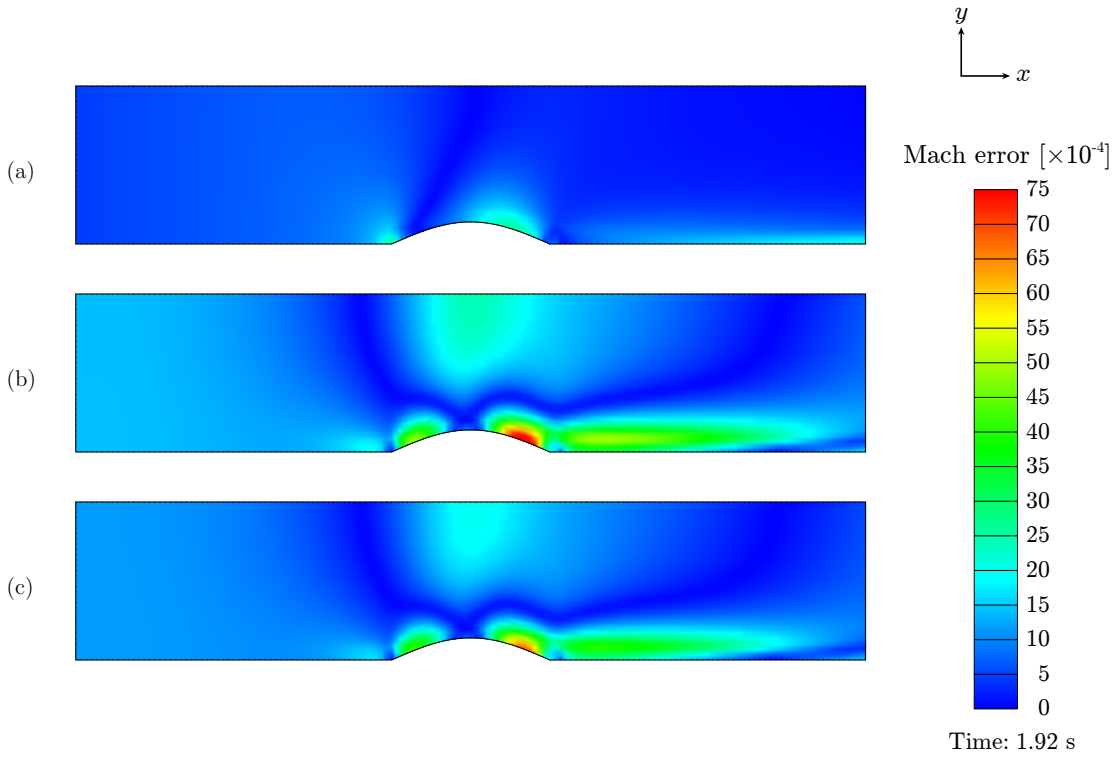

Figure 4.16: Deforming mesh, subsonic case: Mach number error contour plots. (a) ROM, static average, 4 static basis functions; (b) ROM, dynamic average, 1 static basis function; and (c) ROM, dynamic average, 1 dynamic basis function.

4.2.2 Transonic Flow

For the transonic case, the energy of the basis functions and the cumulative energy are plotted in Figure 4.17. The dynamic basis functions contained more energy than the static basis functions with the dynamic average.

Figure 4.18 shows the error of the force, and Figure 4.19 shows the time history of the force. The ROMs using the static average with static basis functions and the dynamic average with static basis functions diverged for several attempted amounts of basis functions. Table 4.1 shows the amounts of basis functions for which the ROMs converged or diverged. The ROMs using static basis functions diverged for several amounts of basis functions, whereas the ROMs using dynamic basis functions converged for most amounts of basis functions. Beyond forty static basis functions with the dynamic average, the error increased. Using the dynamic average and dynamic basis functions performed considerably better than using the dynamic average with static basis functions, which often failed.

	Average	Basis functions	Number of basis functions	Max. no. of basis functions
Converged	Static	Static	1 2 12	80
	Dynamic	Static	1 5 8 10 17 29 40 43 64 65	80
Diverged	Dynamic	Dynamic (Eig.)	2 3	20
	Dynamic	Dynamic (L-BFGS)	12 13 14	20

Table 4.1: Deforming mesh, transonic case: number of basis functions that led to ROM convergence or divergence.

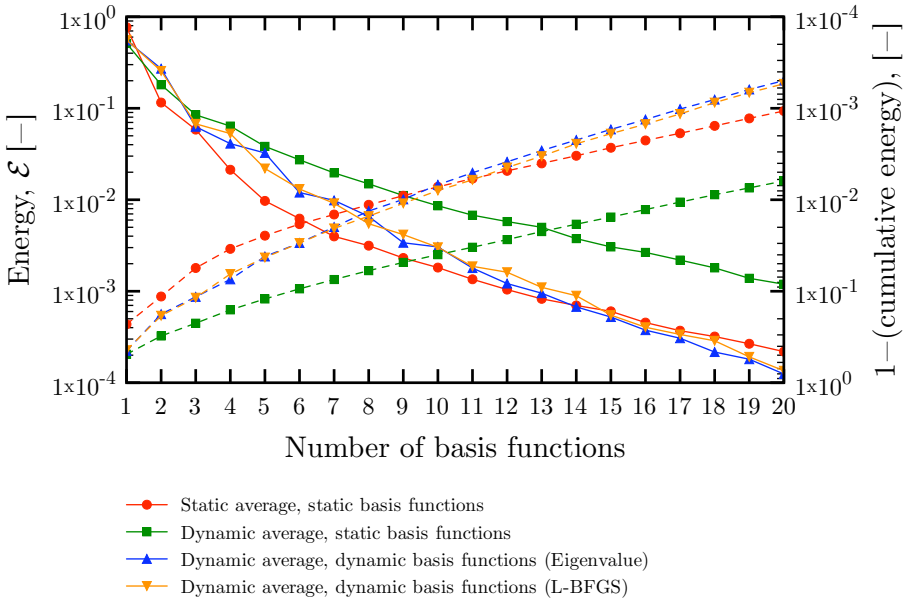

Figure 4.17: Deforming mesh, transonic case: energy spectrum. Solid lines show energy of each basis function; dashed lines show $1 -$ (cumulative energy).

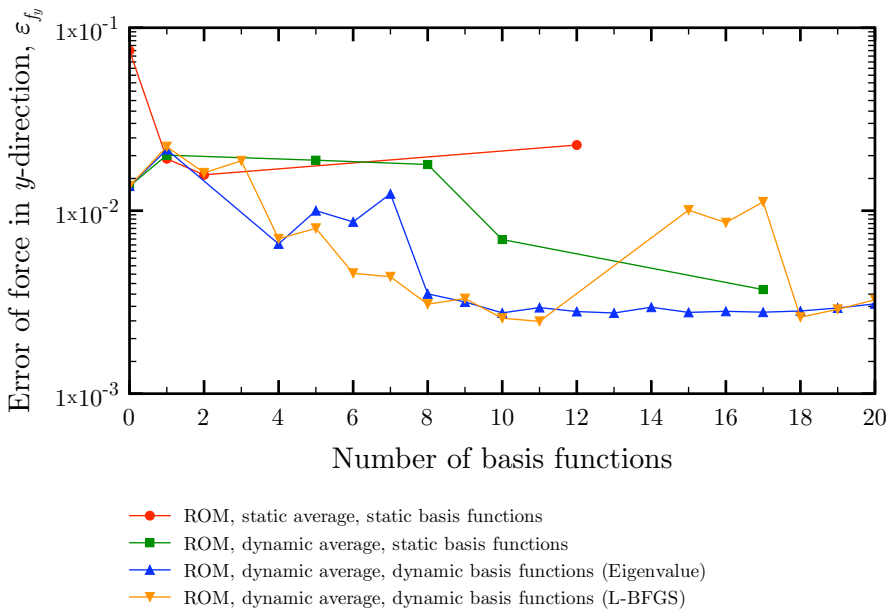

Figure 4.18: Deforming mesh, transonic case: error in force per unit length acting on bump along y-direction.

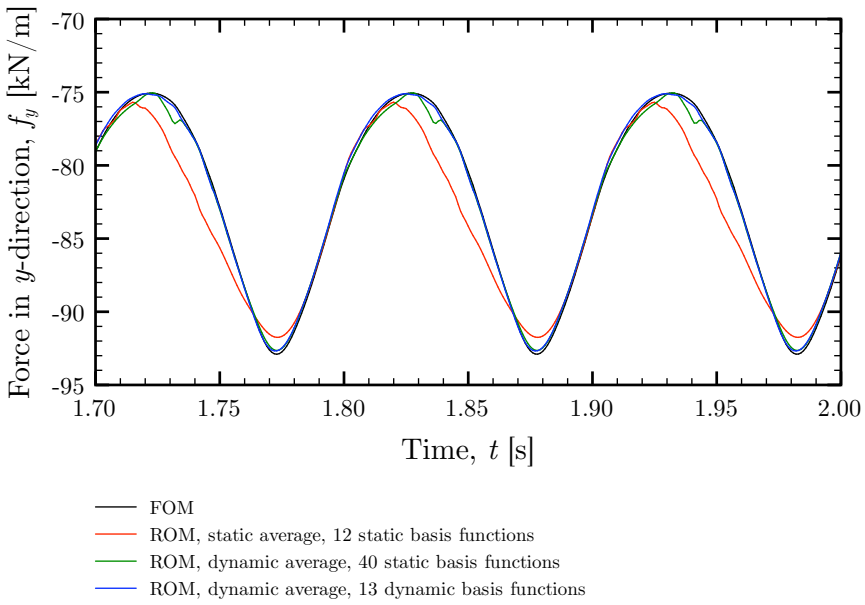

Figure 4.19: Deforming mesh, transonic case: force per unit length acting on bump along y-direction.

Contour plots of the Mach number and Mach number error for the transonic case are presented in Figures 4.20 and 4.21 when the maximum Mach number was achieved. Once more, the dynamic basis functions most closely matched the full-order model.

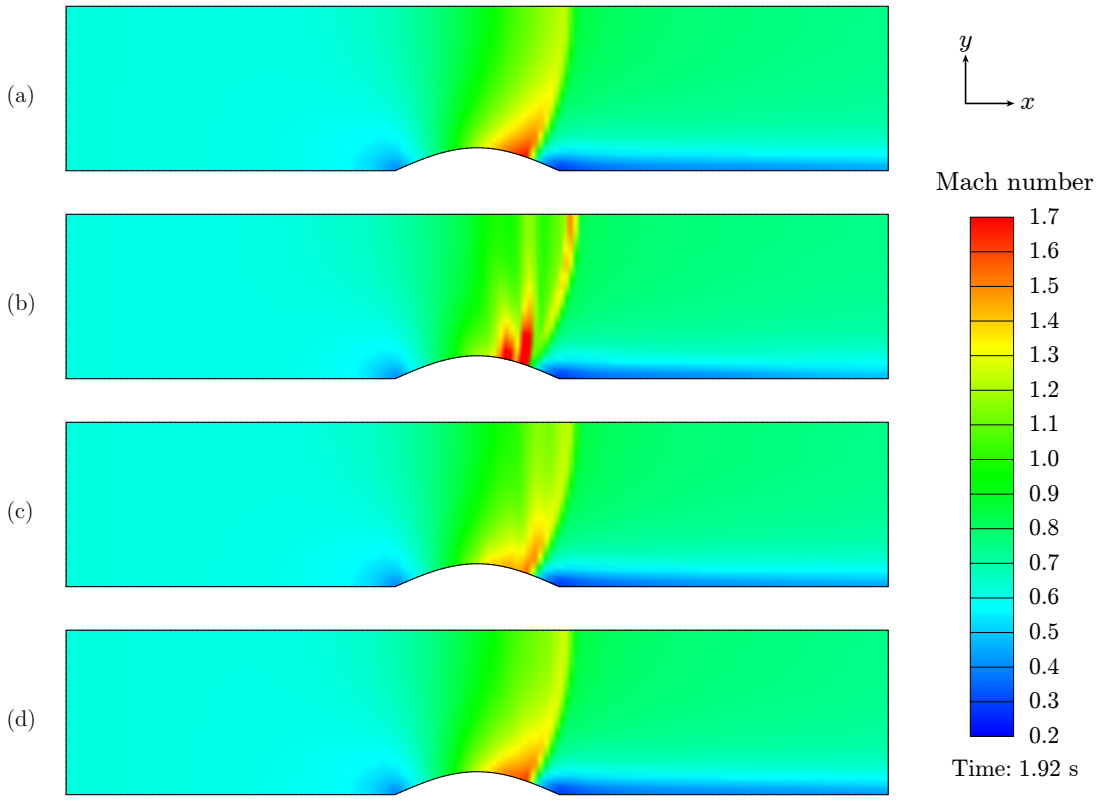

Figure 4.20: Deforming mesh, transonic case: Mach number contour plots. (a) FOM; (b) ROM, static average, 12 static basis functions; (c) ROM, dynamic average, 40 static basis functions; and (d) ROM, dynamic average, 13 dynamic basis functions.

Contour plots of the average and first two basis functions of the density are shown in Figure 4.22–4.25. The contour plots in Figure 4.22 are of the static average and static basis functions, and the contour plots in Figures 4.23–4.25 are of the dynamic average and dynamic basis functions. The dynamic functions were better equipped to model the shock.

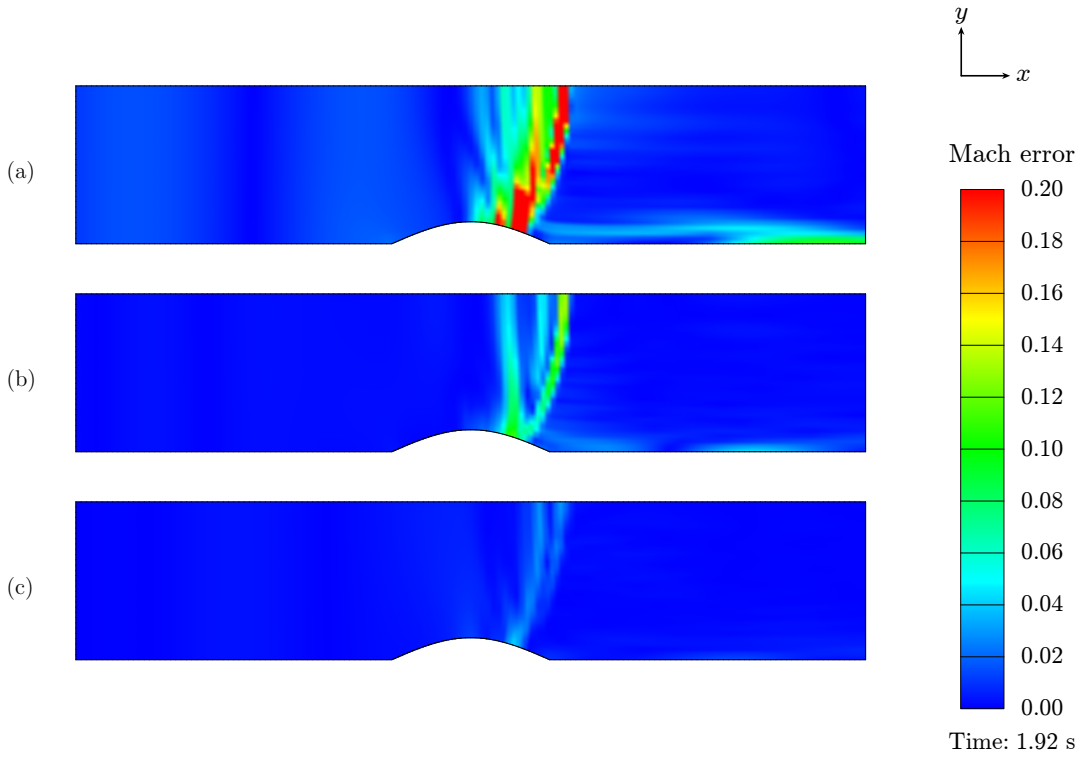

Figure 4.21: Deforming mesh, transonic case: Mach number error contour plots. (a) ROM, static average, 12 static basis functions; (b) ROM, dynamic average, 40 static basis functions; and (c) ROM, dynamic average, 13 dynamic basis functions.

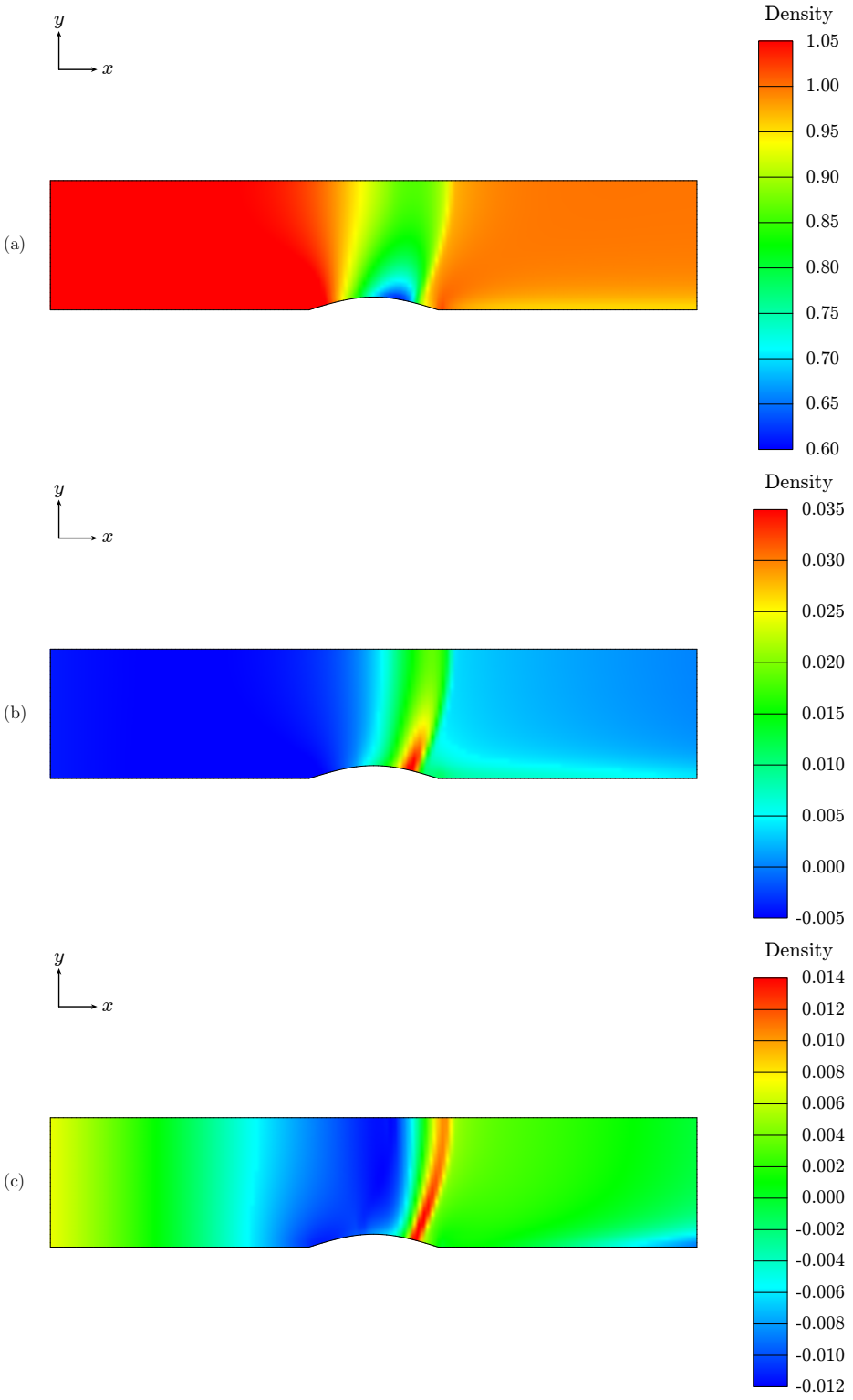

Figure 4.22: Density for deforming mesh, transonic case: (a) static average, (b) first static basis function, and (c) second static basis function.

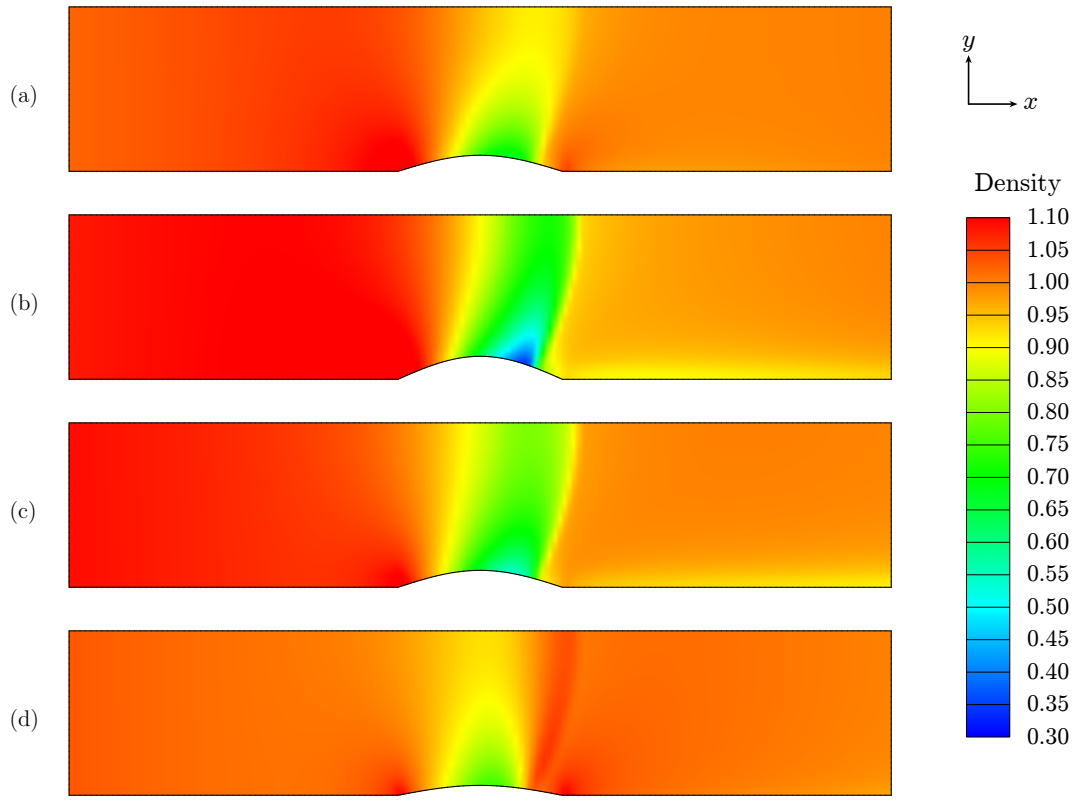

Figure 4.23: Deforming mesh, transonic case: dynamic average of density. (a) $\gamma = 0$, $\dot{\gamma} = \dot{\gamma}_{\max}$; (b) $\gamma = \gamma_{\max}$, $\dot{\gamma} = 0$; (c) $\gamma = 0$, $\dot{\gamma} = \dot{\gamma}_{\min}$; and (d) $\gamma = \gamma_{\min}$, $\dot{\gamma} = 0$.

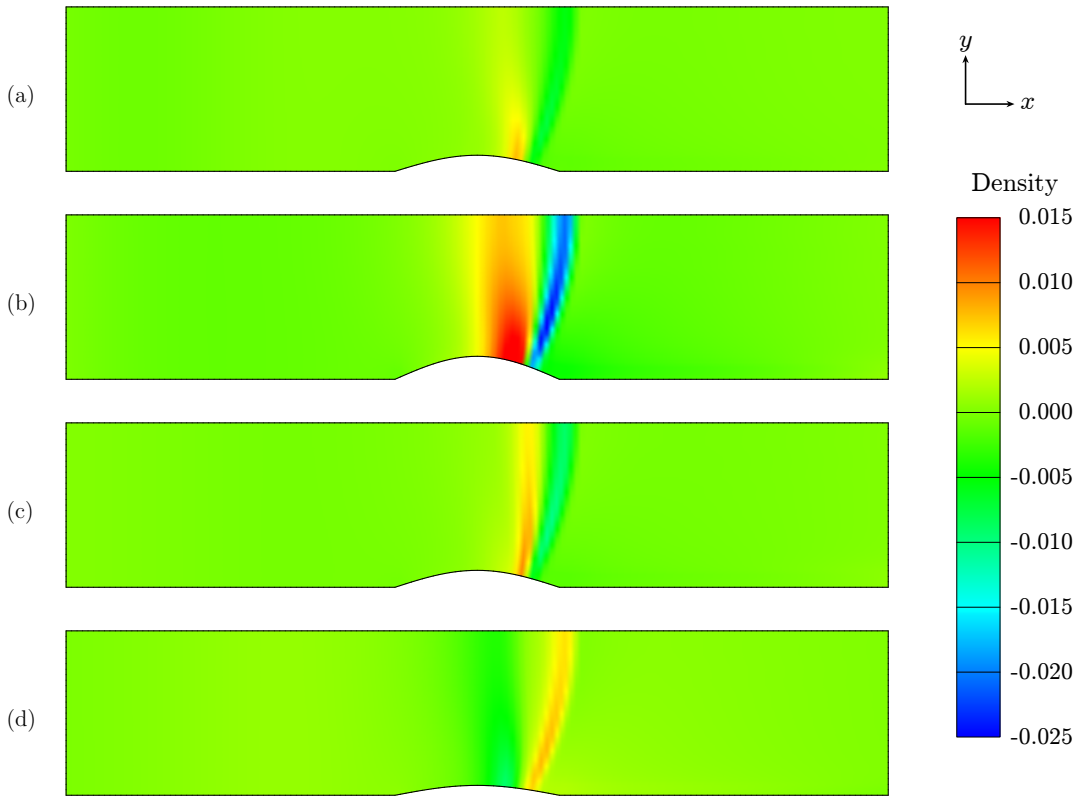

Figure 4.24: Deforming mesh, transonic case: first dynamic basis function of density. (a) $\gamma = 0$, $\dot{\gamma} = \dot{\gamma}_{\max}$; (b) $\gamma = \gamma_{\max}$, $\dot{\gamma} = 0$; (c) $\gamma = 0$, $\dot{\gamma} = \dot{\gamma}_{\min}$; and (d) $\gamma = \gamma_{\min}$, $\dot{\gamma} = 0$.

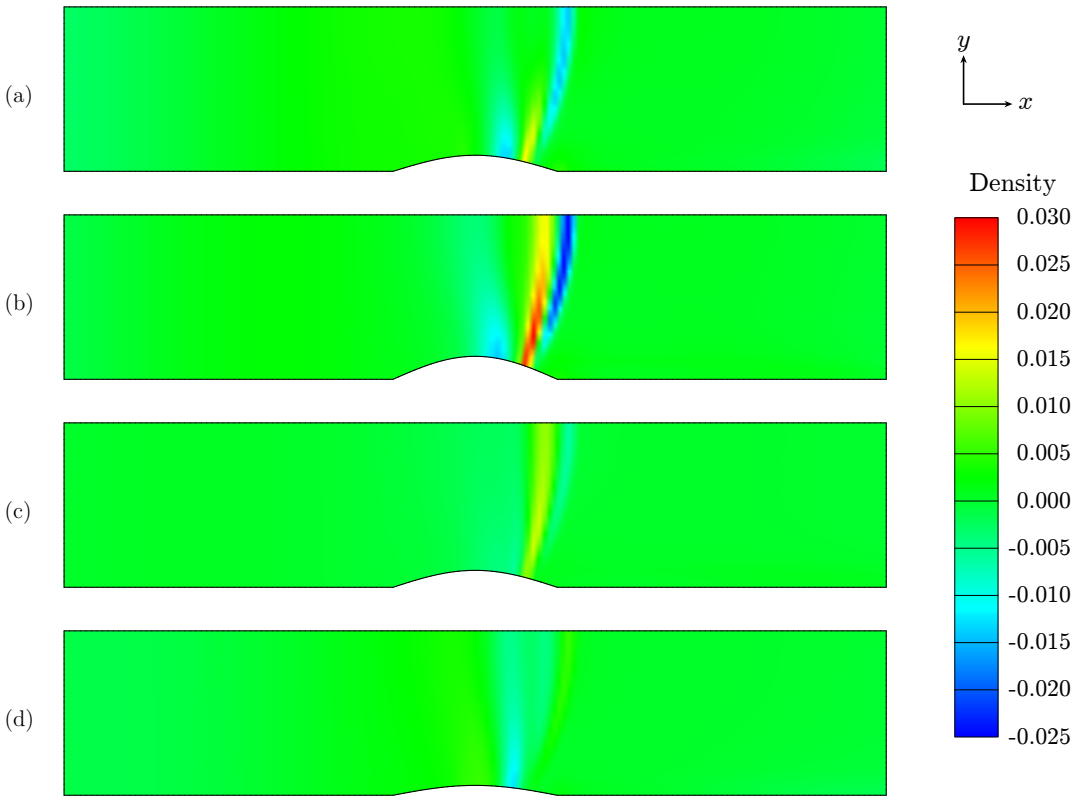

Figure 4.25: Deforming mesh, transonic case: second dynamic basis function of density. (a) $\gamma = 0$, $\dot{\gamma} = \dot{\gamma}_{\max}$; (b) $\gamma = \gamma_{\max}$, $\dot{\gamma} = 0$; (c) $\gamma = 0$, $\dot{\gamma} = \dot{\gamma}_{\min}$; and (d) $\gamma = \gamma_{\min}$, $\dot{\gamma} = 0$.

4.3 Discussion

4.3.1 Fixed Mesh – Subsonic Flow

For the subsonic case with the fixed mesh, the ROMs accurately predicted the flow field, as shown in Figures 4.3, 4.4, 4.5, and 4.6. Using the dynamic average offered a generally higher fidelity result than the static average, often with an error that was an order of magnitude less. With the dynamic average, the flow was better modeled by the static basis functions, as opposed to the dynamic basis functions. Using the dynamic average, the energy of the static and dynamic basis functions obtained from the eigenvalue algorithm was comparable, as shown in Figure 4.2. Furthermore, using dynamic basis functions obtained from the eigenvalue algorithm yielded better results than those obtained from the L-BFGS algorithm.

4.3.2 Fixed Mesh – Transonic Flow

The ROMs reasonably simulated the transonic case with the fixed mesh, as shown in Figures 4.8, 4.9, 4.10, and 4.11; however, there was a greater disparity in fidelity between the static and dynamic functions. When using the dynamic average, the dynamic basis functions obtained from the eigenvalue algorithm generally performed better than the static basis functions. Additionally, using the dynamic average with static basis functions generally performed better than using the static average with static basis functions. As shown in Figure 4.7, when using the dynamic average, the static basis functions contained less energy than the dynamic basis functions. Overall, the dynamic basis functions obtained from the eigenvalue algorithm performed better than those obtained from the L-BFGS algorithm. The dynamic basis functions obtained from the eigenvalue algorithm were better able to capture the shock movement than the static basis functions, as shown in Figures 4.10 and 4.11.

4.3.3 Deforming Mesh – Subsonic Flow

For the subsonic case with the deforming mesh, the ROM error shown in Figure 4.13 was noticeably higher than for the subsonic case with the fixed mesh shown in Figure 4.3. Additionally, increasing the number of basis functions did not improve the result. Using the dynamic average, there was little distinction between the use of static and dynamic basis functions, which had comparable energy, as shown in Figure 4.12. Using the static average with the static basis functions resulted in a better force prediction than using the dynamic average when more than one basis function was used. Nonetheless, each of the cases provided reasonable results, as presented in Figures 4.14, 4.15, and 4.16.

4.3.4 Deforming Mesh – Transonic Flow

For the transonic case with the deforming mesh, the ROM using the static average and static basis functions was unsuitable for modeling the flow. Furthermore, as shown in Figures 4.18, 4.19, 4.20, and 4.21, using the dynamic average and dynamic basis functions performed considerably better than using the dynamic average with static basis functions, which often failed. Additionally, fewer dynamic basis functions were needed, and the dynamic basis functions contained more energy, as shown in Figure 4.17.

From the Mach number and Mach number error plots shown in Figures 4.20 and 4.21, the dynamic basis functions, shown in Figures 4.23–4.25, better captured the flow nonlinearity and shock movement than the static functions shown in Figure 4.22. Generally, the dynamic basis functions obtained from the eigenvalue algorithm performed better than those obtained from the L-BFGS algorithm.

4.3.5 Summary

Static functions are suitable for modeling subsonic flows and flows with fixed meshes, but fail for transonic flows with deforming meshes. Dynamic functions are better able to overcome the shortcomings of the static functions when the flow is nonlinear and the mesh is deforming.

Additionally, the energy spectrum provides initial insight, but it is not a flawless indicator of ROM performance.

Generally, using too few basis functions produces a low-fidelity result; however, the fidelity can also deteriorate when there is an excessive amount of basis functions. In the latter case, the errors increase since the higher frequencies cannot be resolved without reducing the time step. Due to the dynamic nature, fewer dynamic basis functions are required, thereby justifying the additional storage requirements. Using fewer basis functions overcomes the need to resolve higher frequencies and evaluate additional time coefficients.

Two optimization algorithms were considered for computing the dynamic basis functions: the eigenvalue algorithm and the L-BFGS algorithm. The dynamic basis functions computed from the eigenvalue algorithm generally performed better than those obtained from the L-BFGS algorithm.

5. DYNAMIC FUNCTION RESULTS – TENTH STANDARD CONFIGURATION

The reduced-order model resulting from the application of proper orthogonal decomposition to the governing equations was used to model flow in the Tenth Standard Configuration, and the results are compared with those that arose from the full-order model.

The first subsection describes the simulations and the FOM implementation. In the second subsection, the ROM results are presented and compared with those that arose from the FOM. The final subsection provides a discussion of the results.

5.1 Full-Order Model

The Tenth Standard Configuration is a two-dimensional, linear compressor cascade consisting of modified NACA 0006 profiles at a 45-degree stagger angle [45, 46]. The modified airfoils have camber, the chord length is one meter, and the gap-to-chord ratio is one.

Figures 5.1 and 5.2 show the mesh used to discretize the domain. The mesh was generated using a Poisson solver and contained 16,500 nodes on each of the two layers parallel to the xy-plane. The inlet was positioned 3.5 chord lengths before the leading edge, as measured in the x-direction. The outlet was positioned 5.3 chord lengths beyond the trailing edge, as measured in the x-direction. There were 70 nodes from the inlet to the leading edge, 138 nodes around the airfoil, 70 nodes from the trailing edge and the outlet, and 40 nodes from airfoil to airfoil.

An inviscid transonic flow was simulated with an inlet Mach number of 0.8 and an inlet flow angle of 58 degrees. The blades experienced a forced plunging motion in the direction perpendicular to the chord. The motion for each blade was prescribed

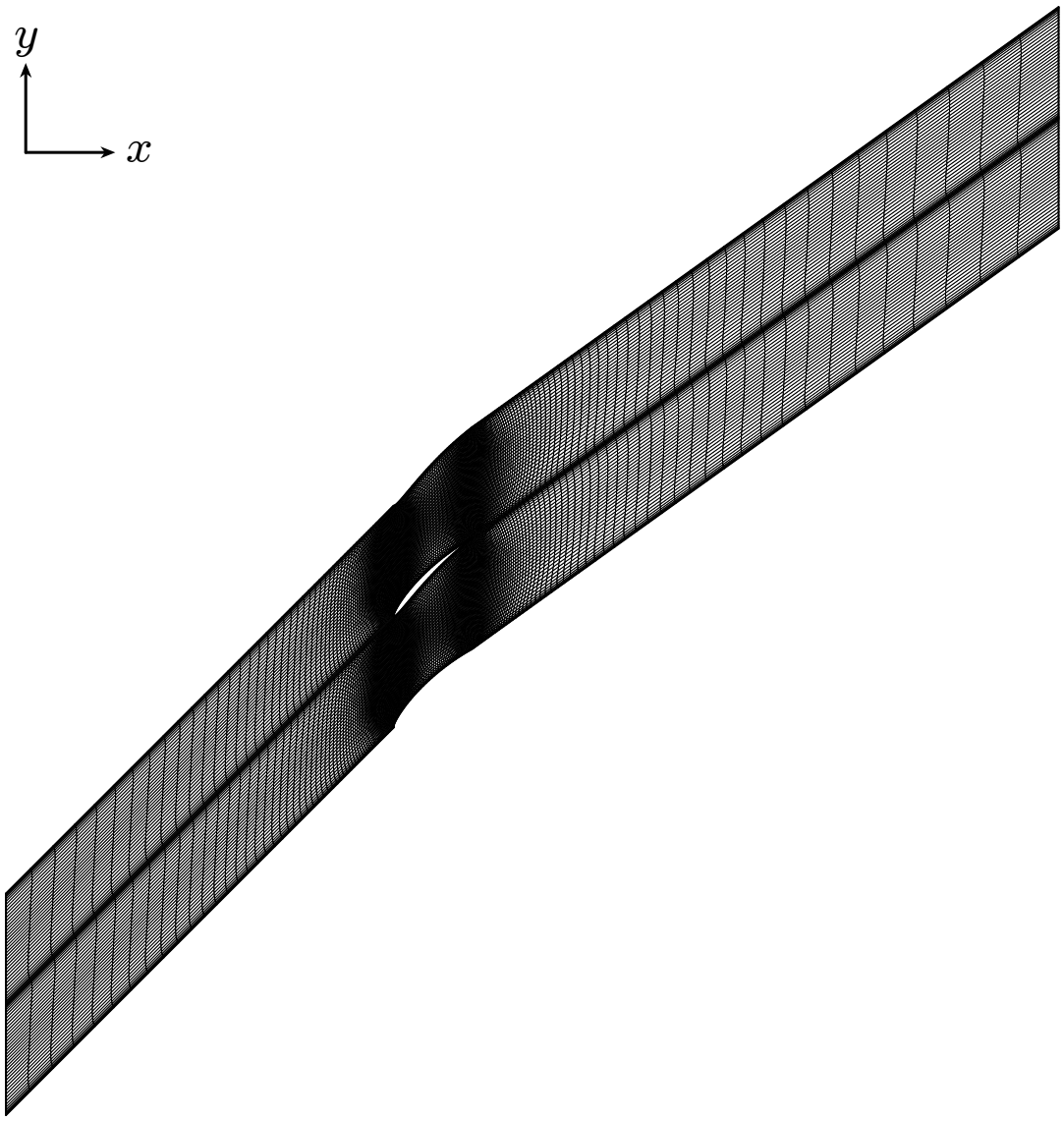

Figure 5.1: Tenth Standard Configuration mesh.

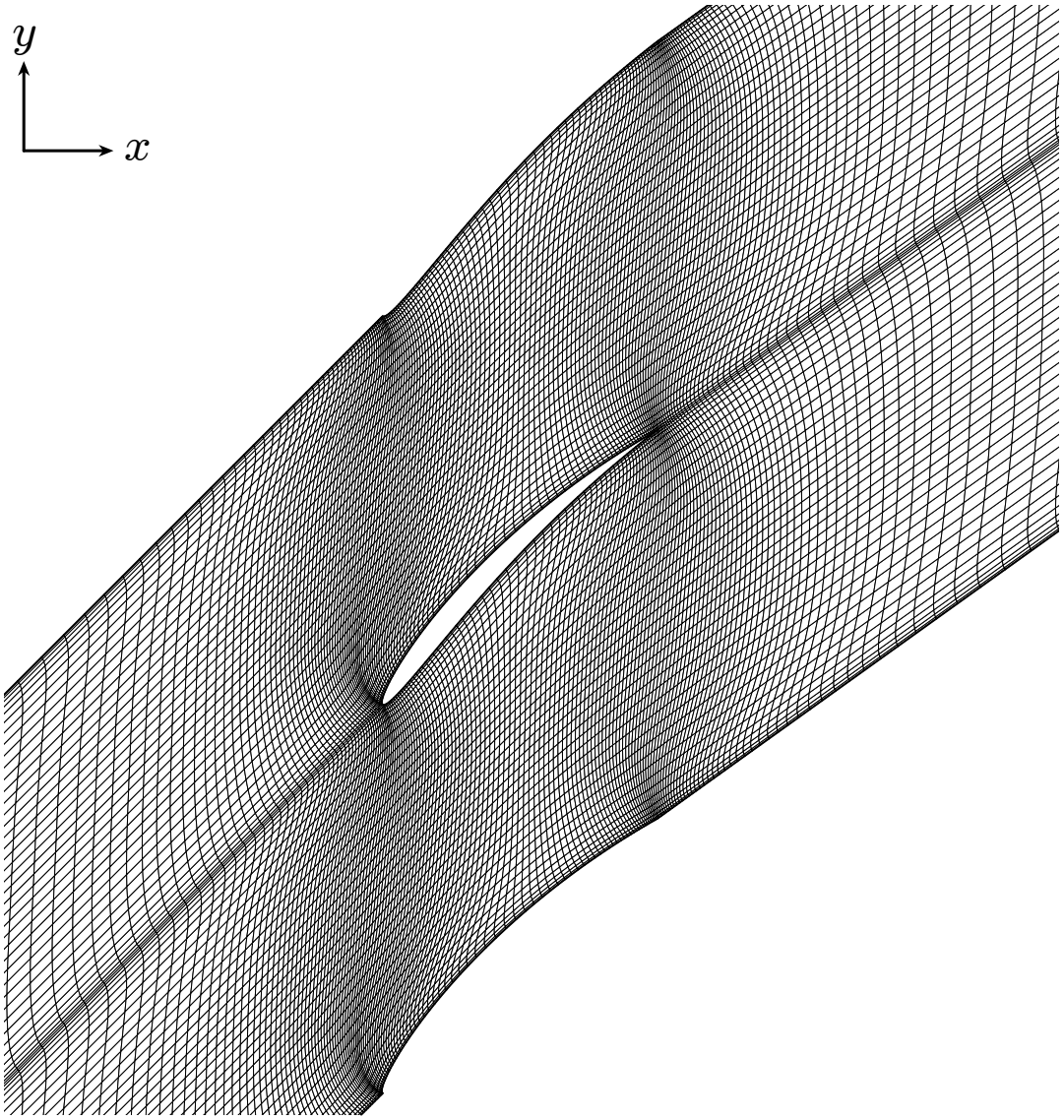

Figure 5.2: Detailed Tenth Standard Configuration mesh.

by

$$h = h_0 \cos\left(\omega t + n_b \sigma\right), \tag{5.1}$$

where n_b was the blade number, beginning with zero, and σ was the inter-blade phase angle.

For the simulations presented in this section, the angular frequency ω was such that the reduced frequency based on half of the chord was 0.5. Consequently, the period was 0.023 seconds, and the frequency was 43.3 Hz. The inter-blade phase angle was 180 degrees. The plunging amplitude h_0 was 5% of the chord, and the peak-to-peak amplitude was 10%.

The FOM used a dual-time stepping scheme. For each real-time step, up to 500 pseudo-time steps were used. The pseudo-time stepping ended if one of two conditions were reached: (1) the average residual for each variable was reduced to 10^{-9} or (2) the ratio of the average residual at the beginning of the real-time step to that at the current pseudo-time step exceeded 10^5. The FOM used a second-order accurate spatial discretization.

5.2 Reduced-Order Model

ROM results are shown using a static average with static basis functions, a dynamic average with static basis functions, and a dynamic average with dynamic basis functions. The basis functions were obtained from 1084 snapshots, which spanned more than ten periods between 0.50 and 0.75 seconds. The dynamic basis functions were computed using the eigenvalue algorithm discussed in Subsection 2.3.3.

Equation (5.1) and its time derivative can be alternatively written as

$$h = \alpha_1\gamma_1 + \alpha_2\gamma_2,$$
$$\dot{h} = \alpha_2\gamma_1 - \alpha_1\gamma_2,$$

where $\gamma_1 \equiv \cos\omega t$, $\gamma_2 \equiv \sin\omega t$, $\alpha_1 \equiv h_0 \cos n_b\sigma$, and $\alpha_2 \equiv -h_0 \sin n_b\sigma$. Because α_1 and α_2 do not vary with time, γ_1 and γ_2 provided suitable parameters for the dynamic functions since the plunging motion and its time derivatives could be expressed linearly in terms of these parameters without additional time dependencies.

Figure 5.3 shows the energy of each basis function (4.1), as well as the cumulative energy. In (4.1), $\lambda_j \equiv \left\langle (\tilde{\mathbf{U}}, \boldsymbol{\varphi}_j)/(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j) \right\rangle$, and the average has been subtracted: $\tilde{\mathbf{U}} \equiv \mathbf{U} - \bar{\mathbf{U}}$. Note that $\tilde{\mathbf{U}}$ is different for the static and dynamic average cases since the static and dynamic averages are different. The static basis functions with a dynamic average accumulated energy less rapidly than the dynamic basis functions, suggesting that the dynamic basis functions more efficiently captured the flow features.

The ROM using the static average with static basis functions diverged for several attempted amounts of basis functions. Table 5.1 shows the amounts of basis functions for which the ROMs converged or diverged. The ROMs using the dynamic average converged for all cases.

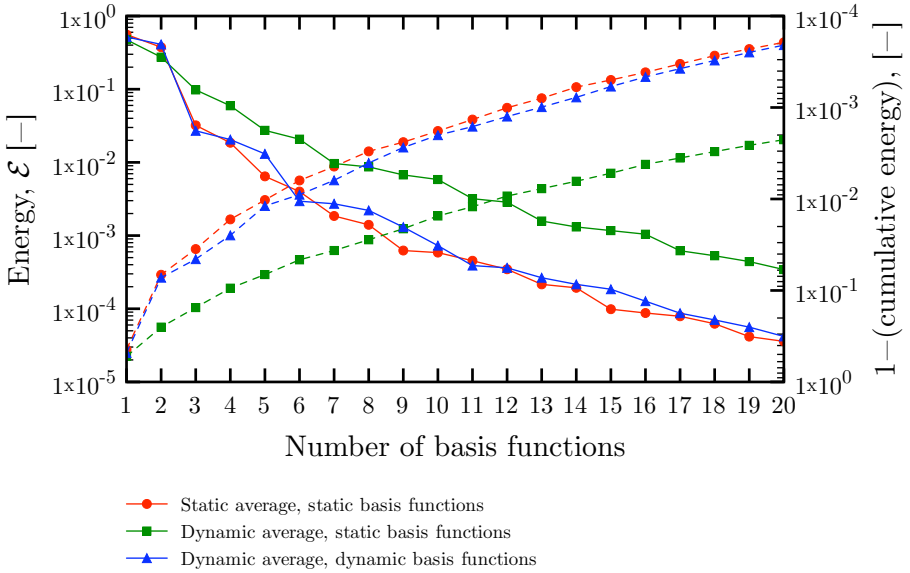

Figure 5.3: Tenth Standard Configuration: energy spectrum. Solid lines show energy of each basis function; dashed lines show 1 − (cumulative energy).

	Average	Basis functions	Number of basis functions	Max. no. of basis functions
Converged	Static	Static	1–2 8–11 15–17	20
Diverged	Dynamic	Dynamic	–	20
	Dynamic	Static	–	20

Table 5.1: Tenth Standard Configuration: number of basis functions that led to ROM convergence or divergence.

Figures 5.4 and 5.5 show the error in the force per unit length acting on a blade along the x- and y-directions for different amounts of basis functions. The error measure for the force is defined to be $\varepsilon_f \equiv \sqrt{\left\langle \left(f_{\mathrm{FOM}} - f_{\mathrm{ROM}}\right)^2 \right\rangle} / \sqrt{\langle f_{\mathrm{FOM}}^2 \rangle}$. For the simulations wherein the ROM diverged, the symbol is omitted on the plot. Although the static average with static basis functions and the dynamic average with static basis functions occasionally performed well, they did not perform as consistently as the dynamic average with dynamic basis functions. Additionally, the static average with static basis functions failed for several attempted cases.

An excerpt of the time history of the force is shown in Figures 5.6 and 5.7. As shown in these figures, the ROMs reasonably predicted the force acting on a blade.

For different amounts of dynamic basis functions, Figure 5.8 shows the time history of the difference in the force per unit length in the x-direction between the ROM and the FOM, $\Delta_{f_x} \equiv f_{x\mathrm{ROM}} - f_{x\mathrm{FOM}}$. The results are consistent with Figure 5.4, indicating that using too few or too many dynamic basis functions yielded a greater error.

Contour plots of the Mach number and Mach number error at time $t = 0.69$ seconds are presented in Figures 5.9 and 5.10. The Mach number error is defined to be $\varepsilon_M \equiv \left| M_{\mathrm{FOM}} - M_{\mathrm{ROM}} \right| / M_{\mathrm{FOM}}$. Figures 5.9 and 5.10 show that all of the ROMs reasonably predicted the flow field when the models did not diverge. For each of the ROMs, the greatest error occurred in the wake. The dynamic basis functions best modeled the shock when it was the strongest.

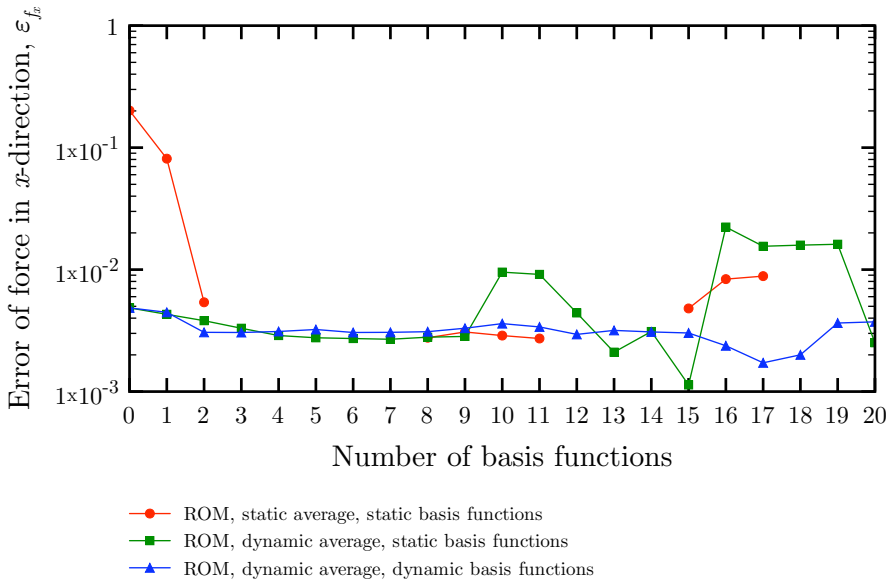

Figure 5.4: Tenth Standard Configuration: error in force per unit length acting on a blade along x-direction.

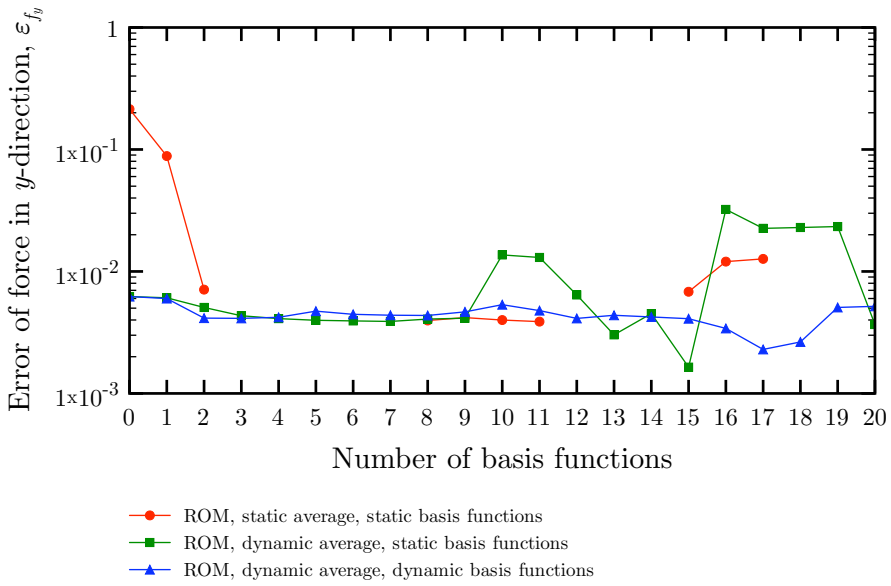

Figure 5.5: Tenth Standard Configuration: error in force per unit length acting on a blade along y-direction.

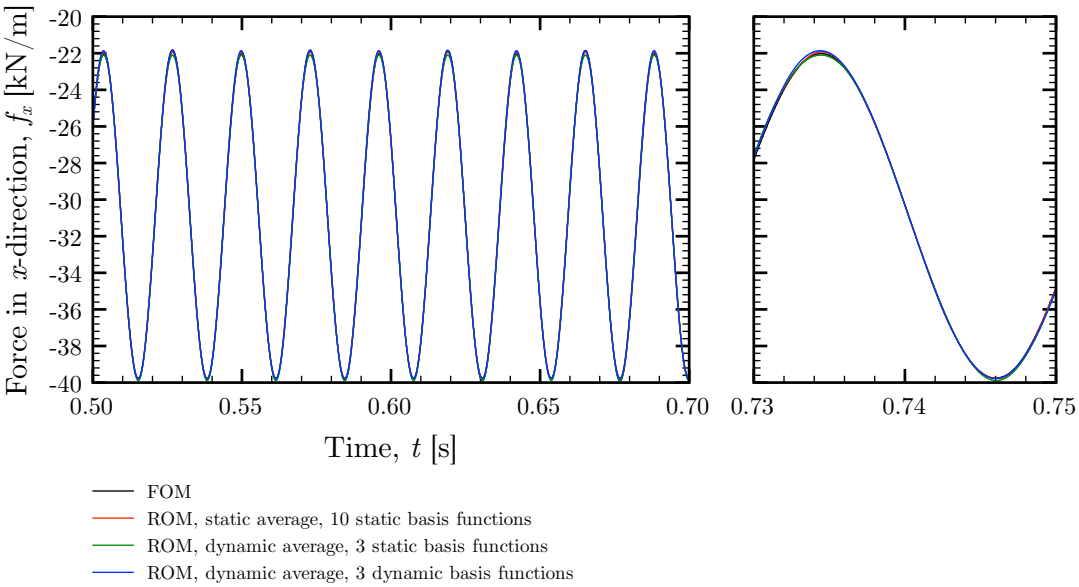

Figure 5.6: Tenth Standard Configuration: force per unit length acting on a blade along x-direction.

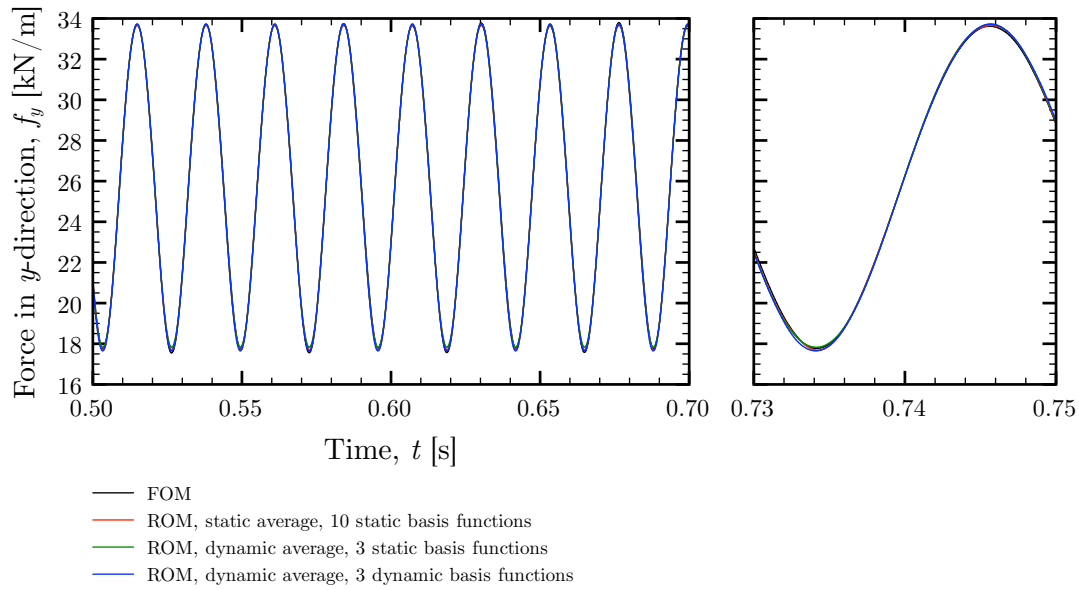

Figure 5.7: Tenth Standard Configuration: force per unit length acting on a blade along y-direction.

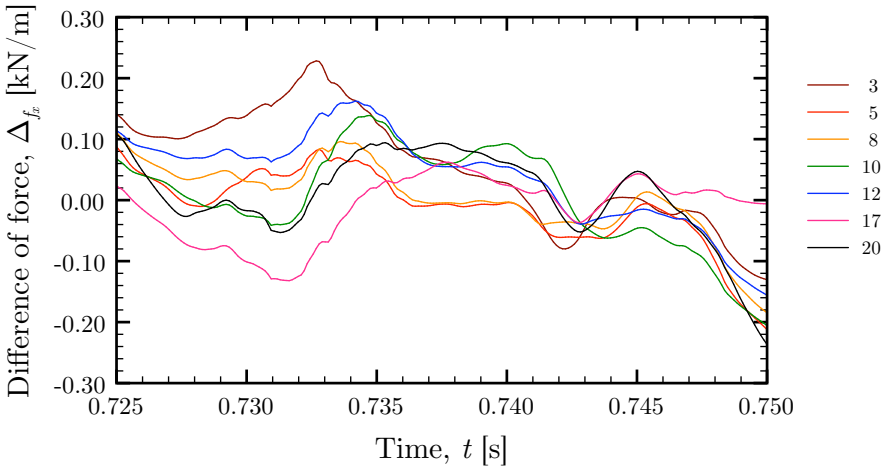

Figure 5.8: Tenth Standard Configuration: difference in force per unit length acting on a blade along x-direction.

Contour plots of the average and first two basis functions of the density are shown in Figures 5.11–5.16. The contour plots in Figures 5.11–5.13 are of the static average and static basis functions, and the contour plots in Figures 5.14–5.16 are of the dynamic average and dynamic basis functions at the instances listed in Table 5.2.

	Middle Blade		Outer Blade		$\mathbf{\Gamma}$	
Identifier	$h/h_{\max}$	$\dot{h}/\dot{h}_{\max}$	$h/h_{\max}$	$\dot{h}/\dot{h}_{\max}$	γ_1	γ_2
(a)	0	1	0	-1	0	1
(b)	1	0	-1	0	-1	0
(c)	0	-1	0	1	0	-1
(d)	-1	0	1	0	1	0

Table 5.2: Tenth Standard Configuration: dynamic function reference points.

The static average shown in Figure 5.11 provided a nearly identical flow field for each blade and accounted for some of the transonic region on the upper surfaces. The first and second static basis functions shown in Figures 5.12 and 5.13 emphasized the contrasting flow fields near the blades, for each blade.

Figure 5.11 shows the differences in the dynamic average on the upper surfaces, near the leading edge, between the two blades. These differences accounted for some of the information that was alternatively contained in the first two static basis functions. The first and second dynamic basis functions shown in Figures 5.15 and 5.16 primarily modeled the shock motion on the upper surfaces.

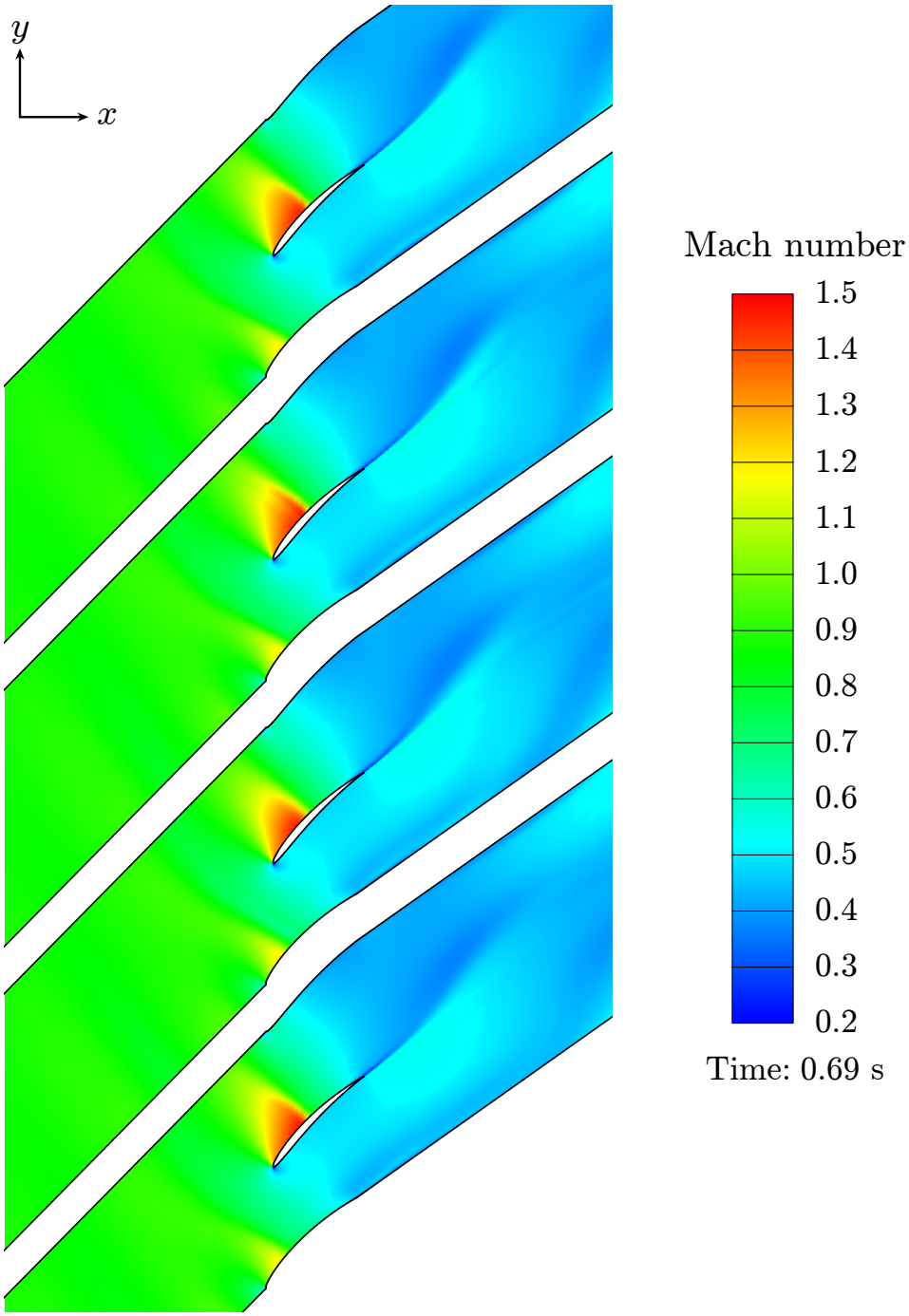

Figure 5.9: Tenth Standard Configuration: Mach number contour plots. From top to bottom: (a) FOM; (b) ROM, static average, 10 static basis functions; (c) ROM, dynamic average, 3 static basis functions; and (d) ROM, dynamic average, 3 dynamic basis functions.

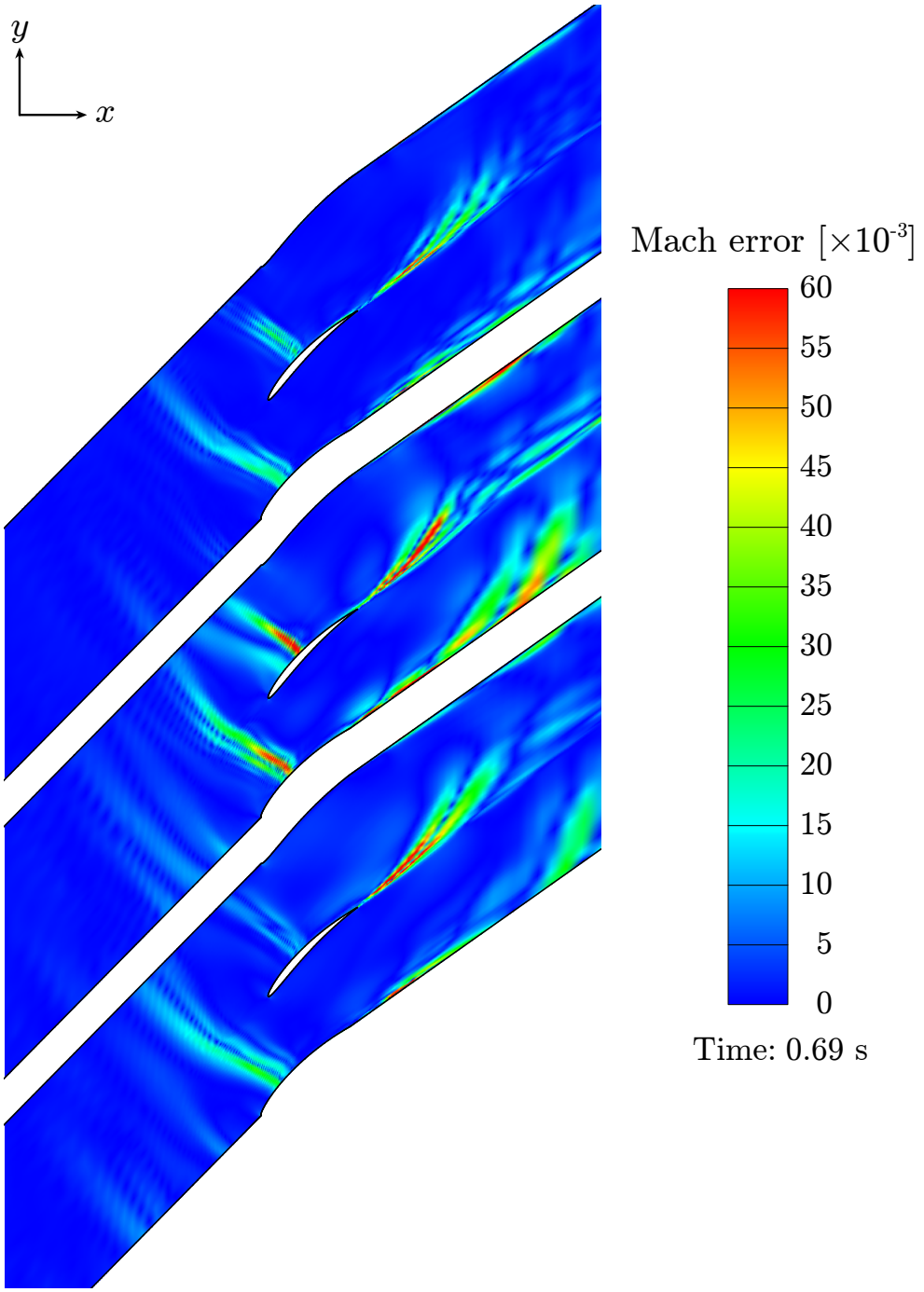

Figure 5.10: Tenth Standard Configuration: Mach number error contour plots. From top to bottom: (a) ROM, static average, 10 static basis functions; (b) ROM, dynamic average, 3 static basis functions; and (c) ROM, dynamic average, 3 dynamic basis functions.

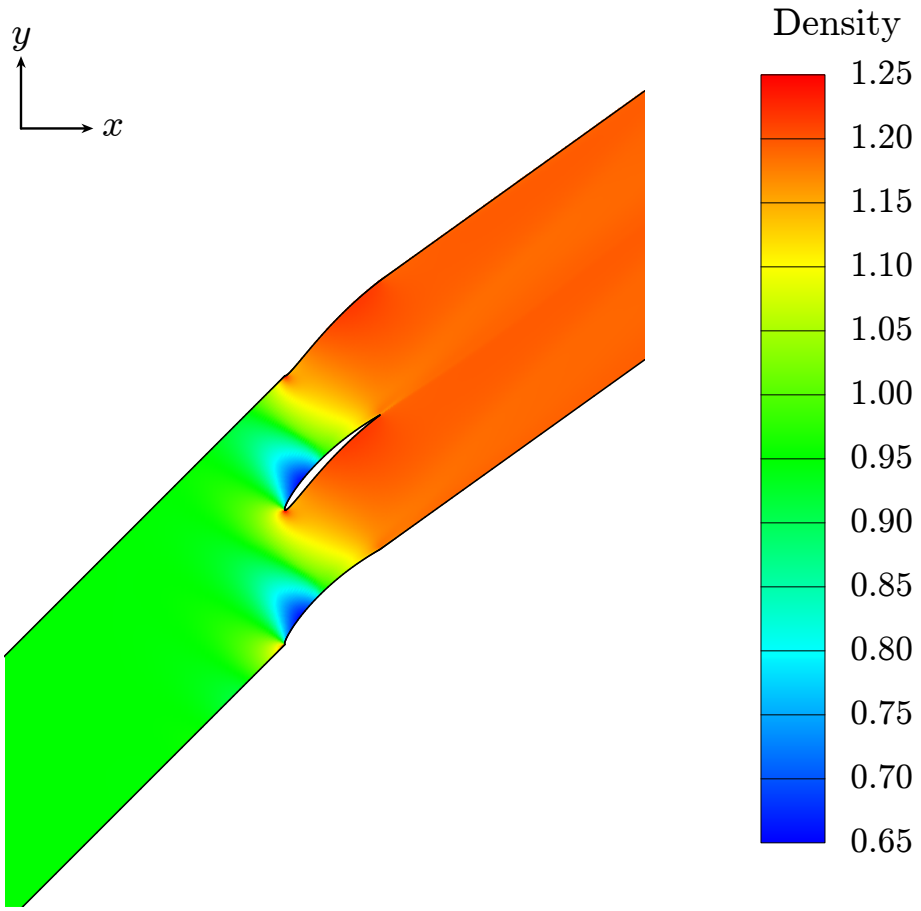

Figure 5.11: Tenth Standard Configuration: static average of density.

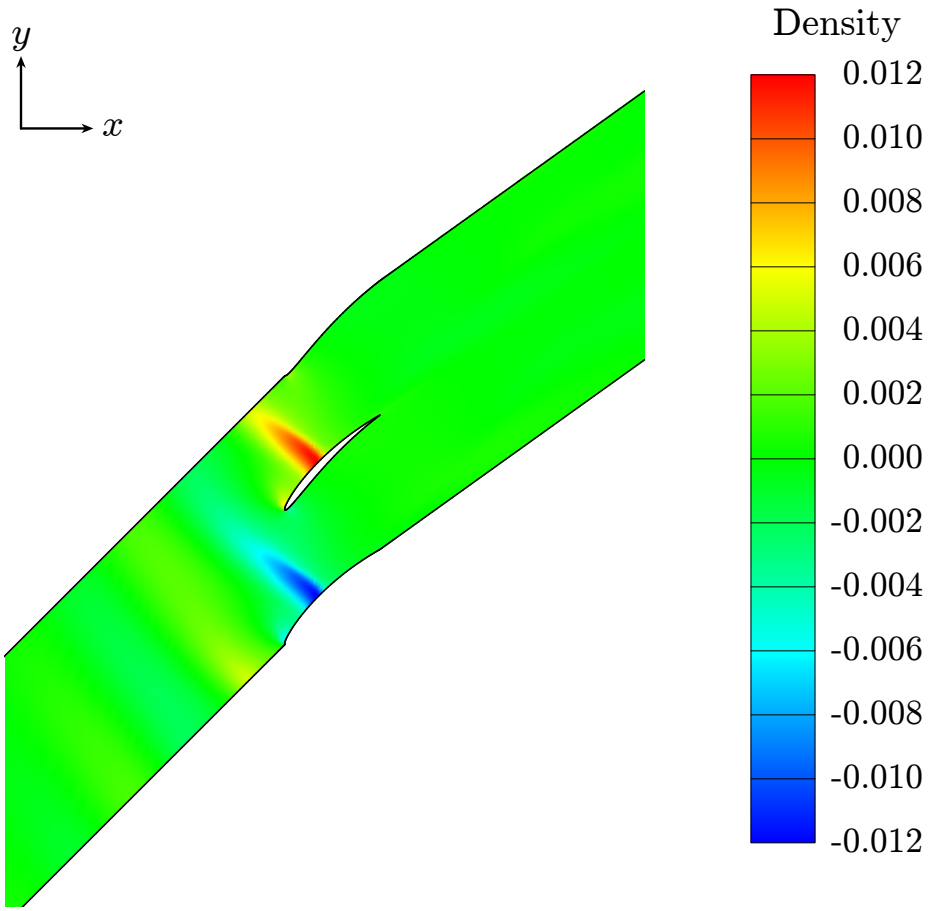

Figure 5.12: Tenth Standard Configuration: first static basis function of density.

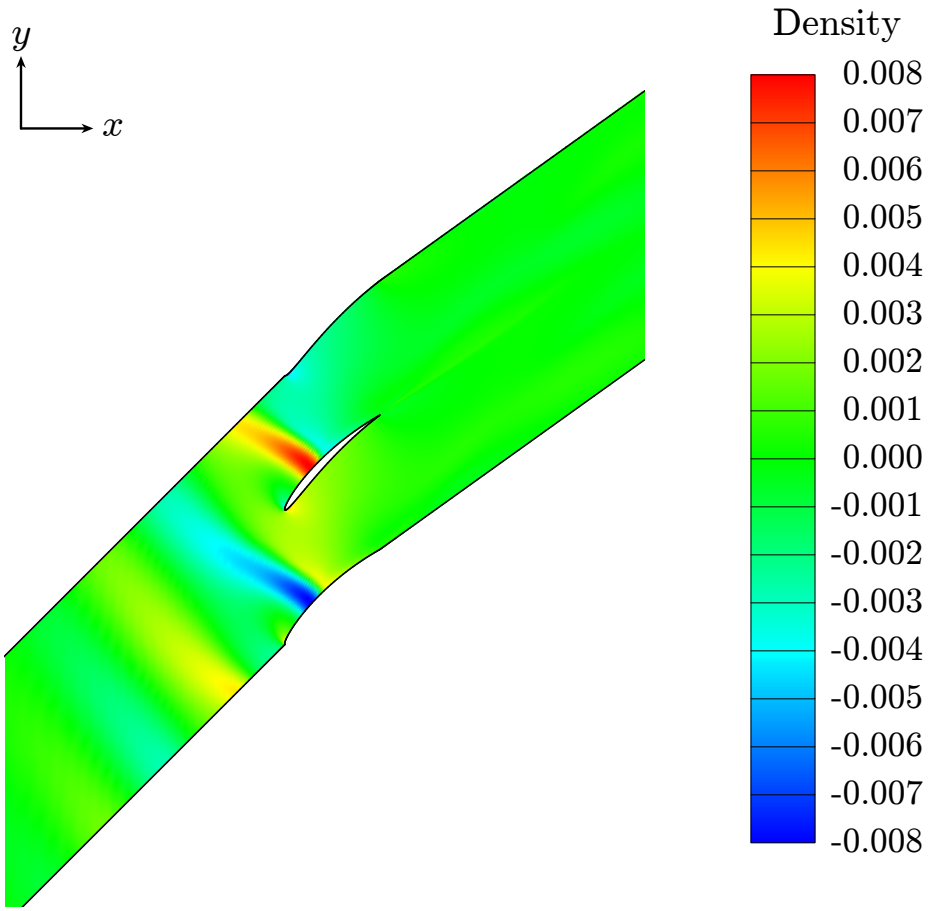

Figure 5.13: Tenth Standard Configuration: second static basis function of density.

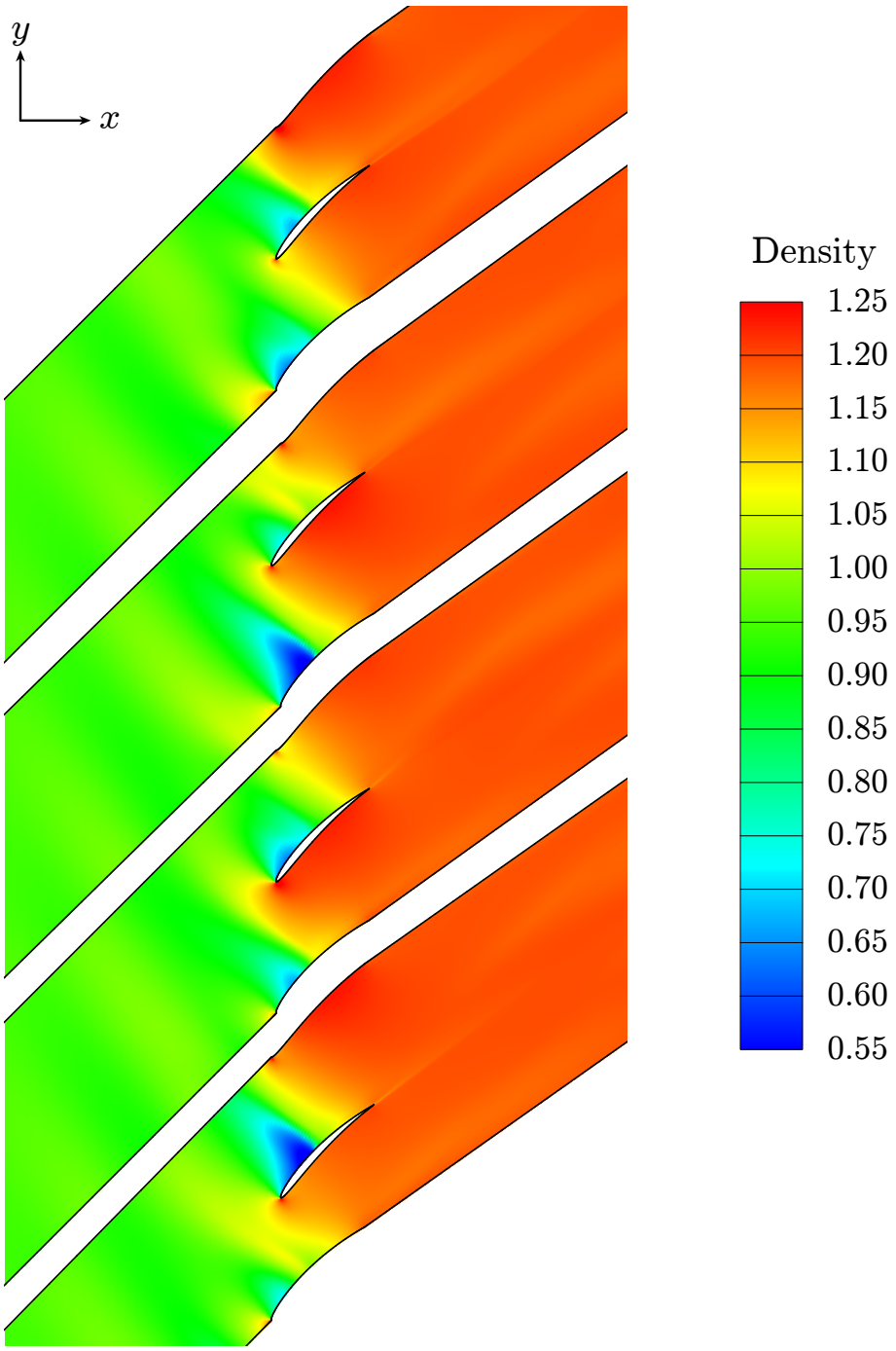

Figure 5.14: Tenth Standard Configuration: dynamic average of density. From top to bottom: (a), (b), (c), and (d), as indicated in Table 5.2.

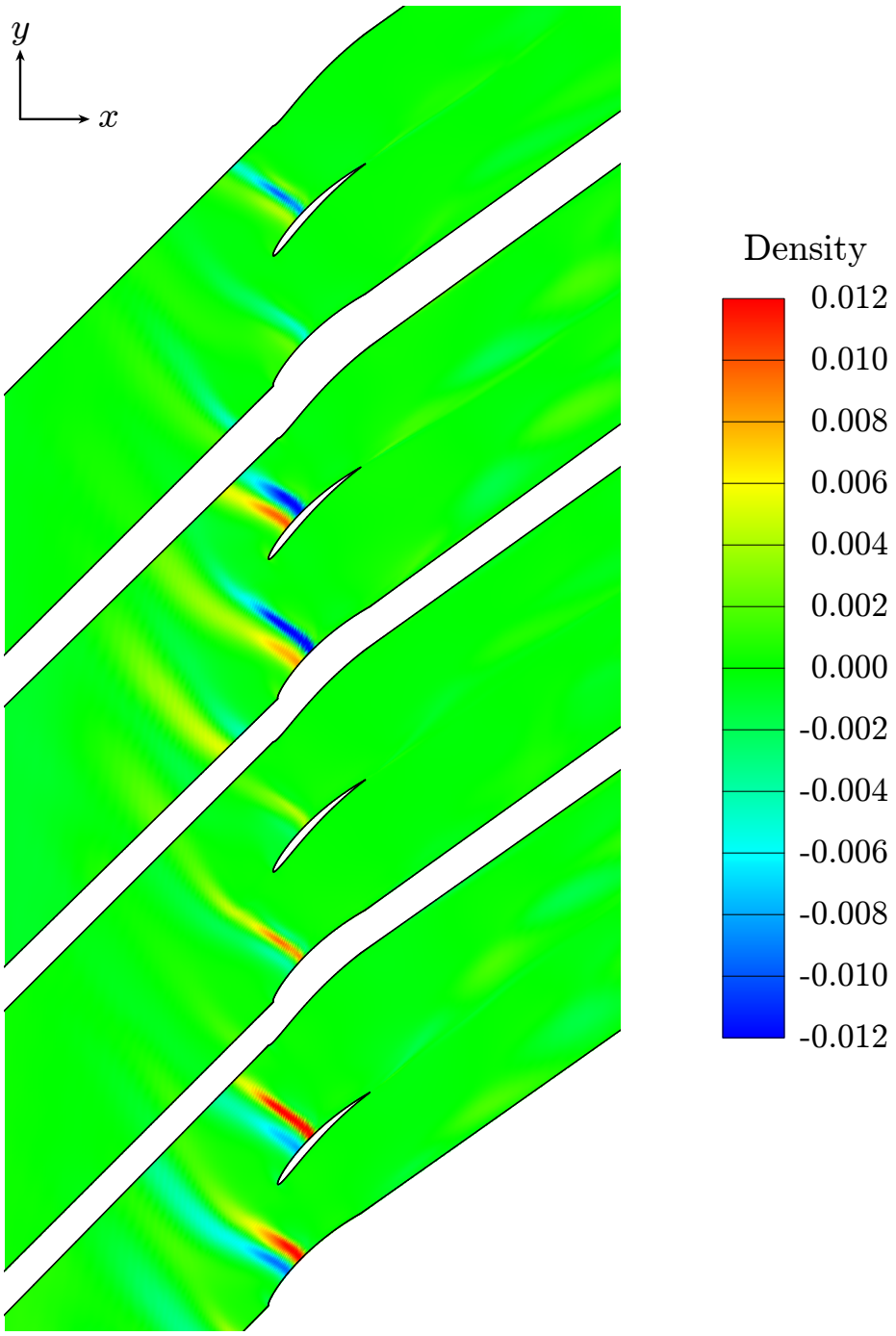

Figure 5.15: Tenth Standard Configuration: first dynamic basis function of density. From top to bottom: (a), (b), (c), and (d), as indicated in Table 5.2.

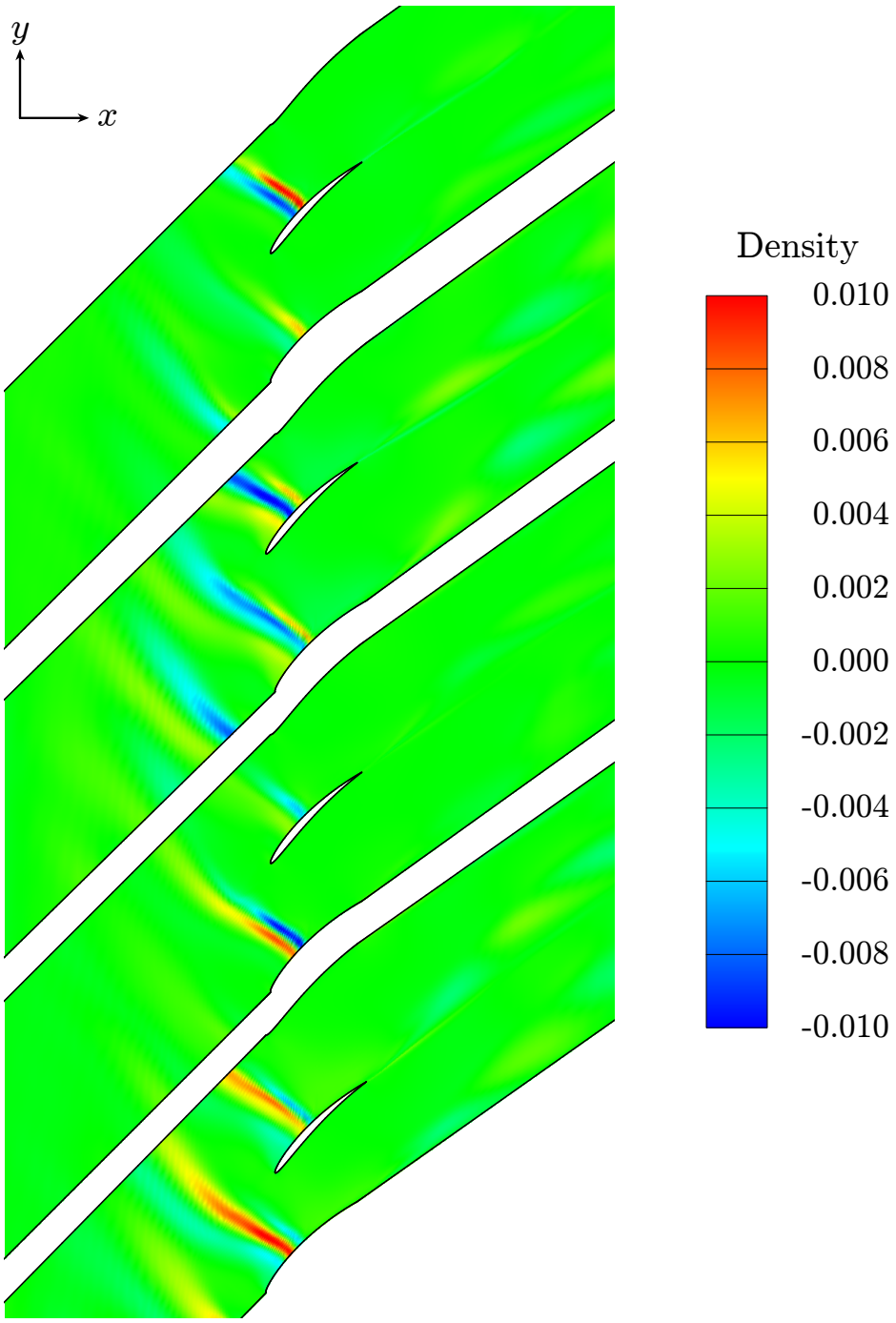

Figure 5.16: Tenth Standard Configuration: second dynamic basis function of density. From top to bottom: (a), (b), (c), and (d), as indicated in Table 5.2.

5.3 Discussion

Compared to the results shown for the channel with the bump in Subsection 4.2.2, the static average with static basis functions and the dynamic average with static basis functions were able to more effectively model the Tenth Standard Configuration without failing as frequently or producing such low-fidelity results. Nonetheless, the static average with static basis functions often failed, and the dynamic average with static basis functions produced inconsistent results.

The dynamic basis functions performed more consistently than the static and dynamic averages with static basis functions, with regard to solution fidelity, as shown in Figures 5.4–5.10, and stability, as shown in Table 5.1 and Figures 5.4 and 5.5. Occasionally, the static and dynamic averages with static basis functions performed better than the dynamic basis functions. However, using the static average with static basis functions often failed, and the dynamic average with static basis functions performed inconsistently.

As with the channel with the bump, the dynamic basis functions more stably modeled the flow in the Tenth Standard Configuration while providing an accurate result.

6. OFF-REFERENCE CONDITION RESULTS*

The reduced-order model resulting from the application of proper orthogonal decomposition to the governing equations was used to model flow through a channel with a sinusoidal bump, and the results are compared with those that arose from the full-order model.

For these cases, an inviscid flow field was modeled through the 5-meter-long channel with a 1-meter height shown in Figure 6.1. The middle meter of the channel contained a sinusoidal bump with a 0.1-meter height. The channel was discretized using 150 cells along the length and 30 cells along the height.

The static pressure at the channel outlet was varied sinusoidally to force unsteadiness of the flow. The back pressure was prescribed by

$$p_b = \bar{p}_b \left[1 + \varepsilon \sin(\omega t)\right], \tag{6.1}$$

where $\bar{p}_b$ was 101,325 Pa and ω was 68.0585 rad/s. Consequently, the reduced frequency based on half of the bump length was $0.1/M_{\text{inlet}}$. Two values were used for ε: 0.01 and 0.05. The FOM was used to simulate fourteen flows associated with inlet Mach numbers of 0.3–0.8 at sea level. Of the cases simulated, supersonic flow was first achieved using an inlet Mach number of 0.65 for both values of ε. The simulations spanned one second, which included at least ten periods, and 1000 snapshots were taken for each case. This section investigates the number of basis functions used, as well as the order of interpolation.

*Part of this section is reprinted with permission from "Using proper orthogonal decomposition to model off-reference flow conditions" by B. A. Freno, T. A. Brenner, P. G. A. Cizmas, 2013. *International Journal of Non-Linear Mechanics*, vol. 54, pp 76–84, Copyright 2013 by Elsevier.

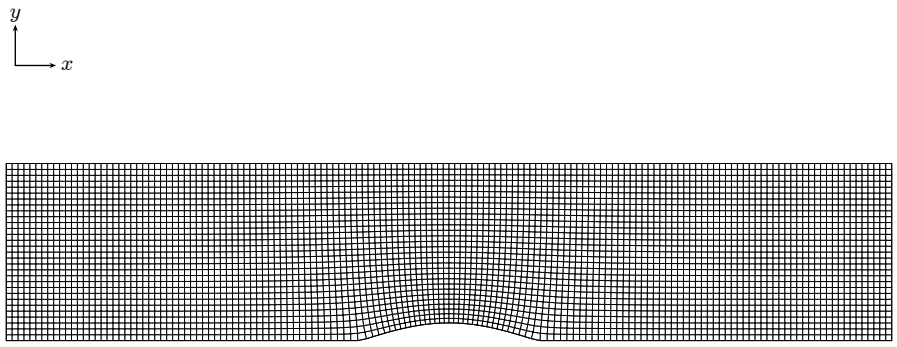

Figure 6.1: Channel mesh for off-reference conditions.

6.1 Number of Basis Functions

Using the snapshots obtained from the FOM simulation, basis functions were computed for use in the ROM. Figures 6.2 and 6.3 respectively show the energy of each basis function (4.1) for $\varepsilon = 0.05$ and 0.01. For $\varepsilon = 0.05$, Figure 6.4 shows the energy for all available basis functions, and Figure 6.5 alternatively shows the cumulative energy as a function of the number of basis functions.

Figure 6.2 shows that, for a larger ε, the energy decrease with basis function number increase was noticeably more gradual than for the case of a small ε in Figure 6.3. A smaller ε reduced the diversity of the flow field with respect to time and enabled the motion to be modeled with fewer basis functions.

Cases with inlet Mach numbers of 0.3–0.6 were purely subsonic. As shown in Figures 6.2 and 6.4 for $\varepsilon = 0.05$, the energy was largely accounted for within the first ten basis functions for the subsonic simulations. Conversely, for the transonic simulations, energy decrease with basis function number increase was considerably more gradual. Additionally, increasing the Mach number, and therefore increasing the flow nonlinearity, reduced the energy accumulation. The disparity in energy

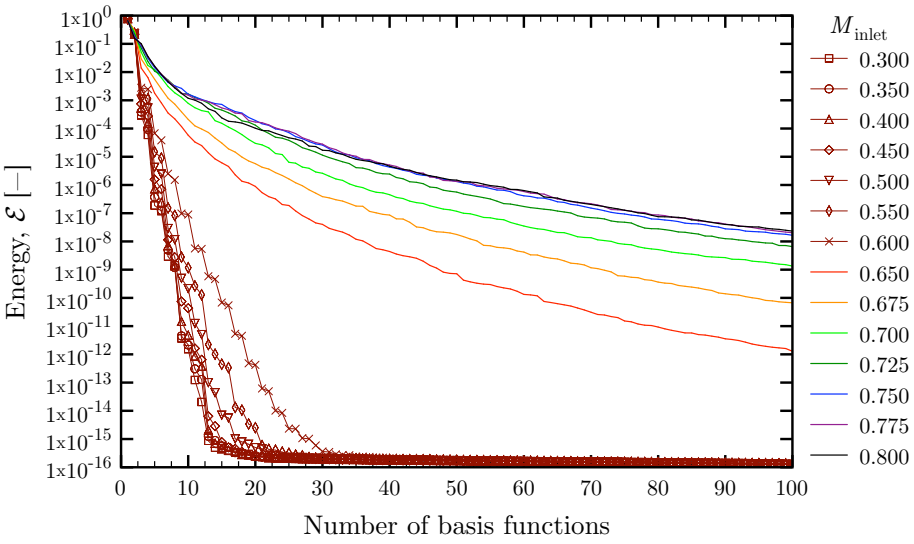

Figure 6.2: Energy spectrum, $\varepsilon = 0.05$.

accumulation between the subsonic and transonic cases, particularly the subsonic case energy plateau shown in Figure 6.2, was not as evident when only the cumulative energy was taken into account, as shown in Figure 6.5. Letting $\varepsilon = 0.05$ henceforth, these observations are further emphasized in Figures 6.6 and 6.7, which show the absolute value of the time coefficients that arose from projecting the snapshots onto the basis functions obtained directly from the snapshots for the Mach 0.40 and 0.75 cases. For the Mach 0.40 case, the amplitude of the oscillation decreased considerably for each subsequent basis function. For the Mach 0.75 case, the decrease in amplitude was much more gradual.

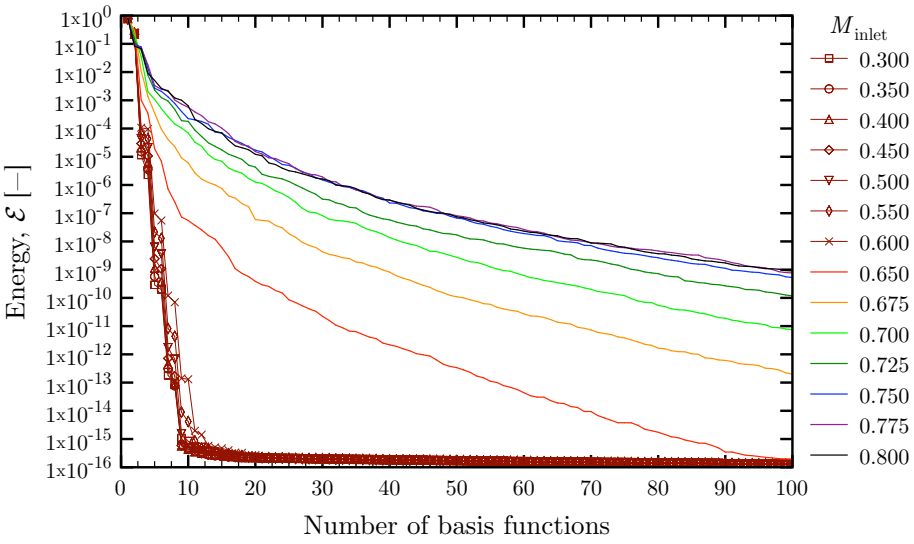

Figure 6.3: Energy spectrum, $\varepsilon = 0.01$.

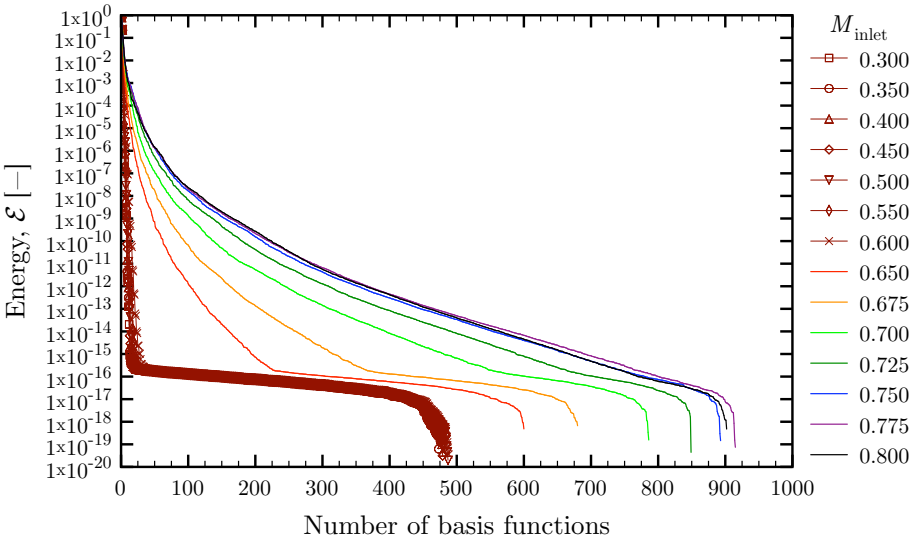

Figure 6.4: Energy spectrum, $\varepsilon = 0.05$.

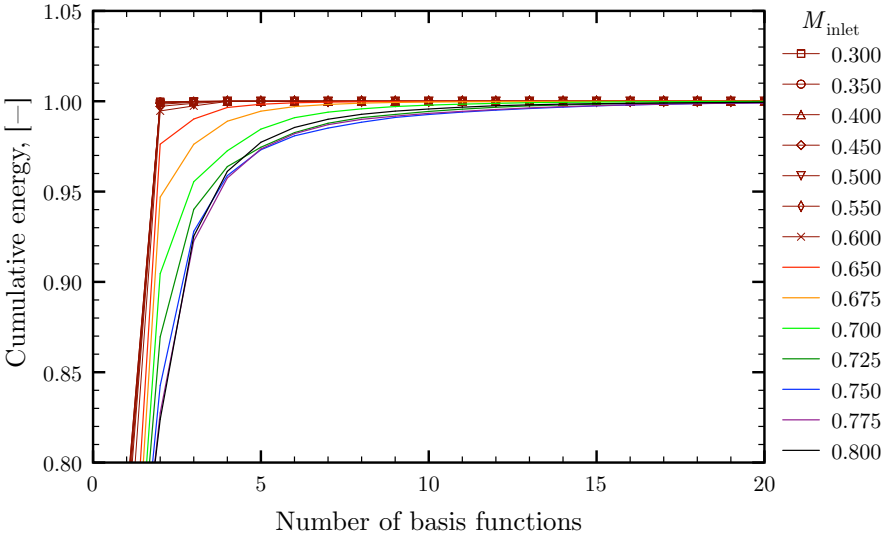

Figure 6.5: Cumulative energy, $\varepsilon = 0.05$.

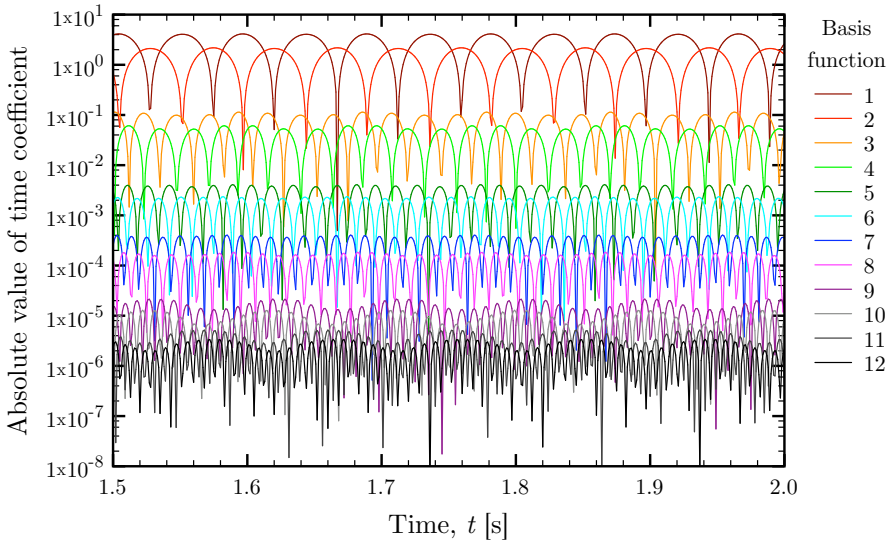

Figure 6.6: Absolute value of snapshots projected onto basis functions, $M_{\text{inlet}} = 0.40$.

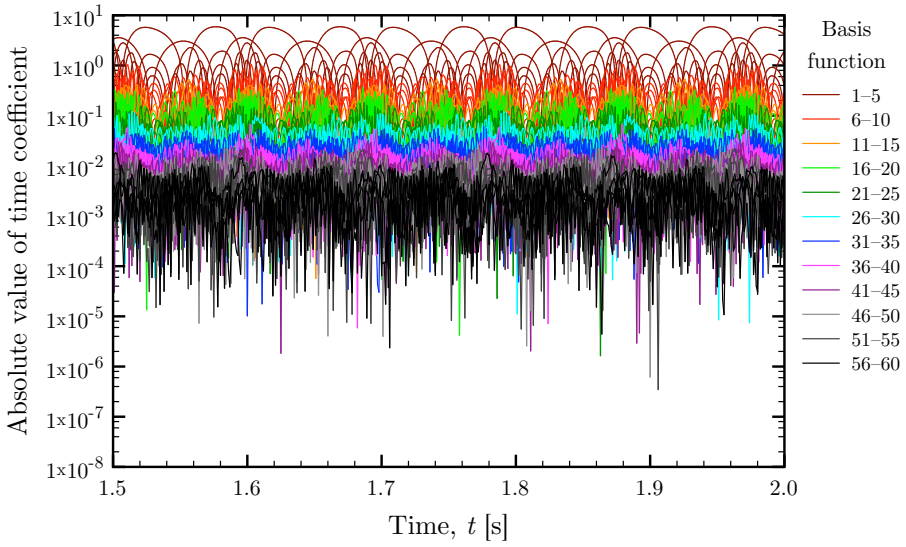

Figure 6.7: Absolute value of snapshots projected onto basis functions, $M_{\text{inlet}} = 0.75$.

The force acting on the bump between one and two seconds was used in this section as a measure for assessing the accuracy of the ROMs. Since the back pressure was prescribed by (6.1), the force acting on the bump was able to be described by an average value and an amplitude. Figures 6.8 and 6.9 plot the error in the average value of the force, as well as the error in the amplitude of the oscillation of the force in the y-direction for inlet Mach numbers of 0.40 and 0.75. These plots show the results for the ROMs using both basis functions obtained directly from the FOM snapshots and through linear interpolation on the tangent space to the Grassmann manifold.

For the subsonic simulations that used basis functions obtained directly from the FOM, Figure 6.8 indicates that exceeding the number of basis functions beyond which the energy contribution plateaued in Figure 6.2 did not necessarily improve the accuracy.

From (2.4), the eigenvalue can be computed for an arbitrary basis function:

$$\tilde{\lambda}_j = \left\langle \frac{(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_j)^2}{\left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j\right)} \right\rangle. \tag{6.2}$$

In (6.2), the basis function can be obtained through POD or interpolation.

Using (6.2), Figure 6.10 shows the energy of each basis function. Figure 6.10 is consistent with Figure 6.2 for the transonic simulations, as shown in Figure 6.11 for $M_{\text{inlet}} = 0.75$. However, for the subsonic simulations, the basis functions that were in the plateaued region in Figure 6.2 had noticeably more energy in Figure 6.10. Furthermore, from Figure 6.8, the introduction of these superfluous basis functions in the plateaued region appears to have contaminated the basis functions arising from interpolation.

For the transonic simulations, in the absence of a plateau of energy variation,

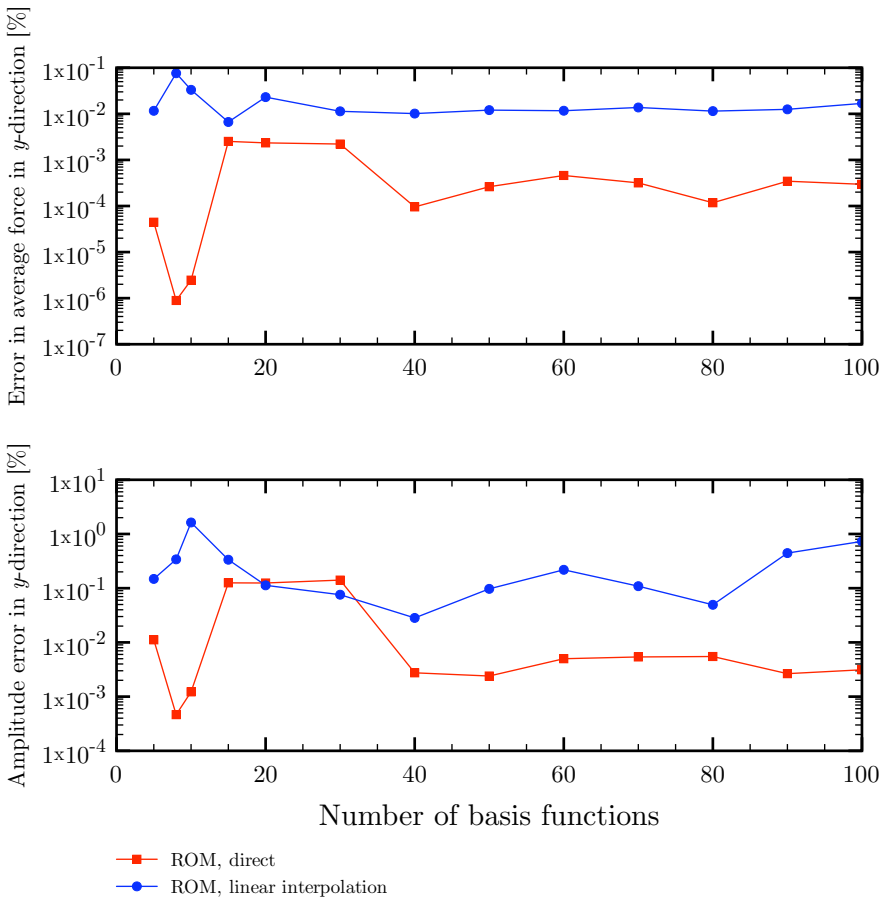

Figure 6.8: Error in average force and in amplitude of oscillation for y-component of force acting on bump, $M_{\text{inlet}} = 0.40$.

increasing the amount of basis functions generally increased the accuracy, as shown in Figure 6.9. However, the higher-numbered basis functions required a finer time discretization when used in the ROM. The basis functions obtained from interpolation did not accumulate energy as rapidly as those obtained directly from the FOM as shown in Figure 6.11. Therefore, the ROM required more than 50 basis functions for linear interpolation on the tangent space to the Grassmann manifold.

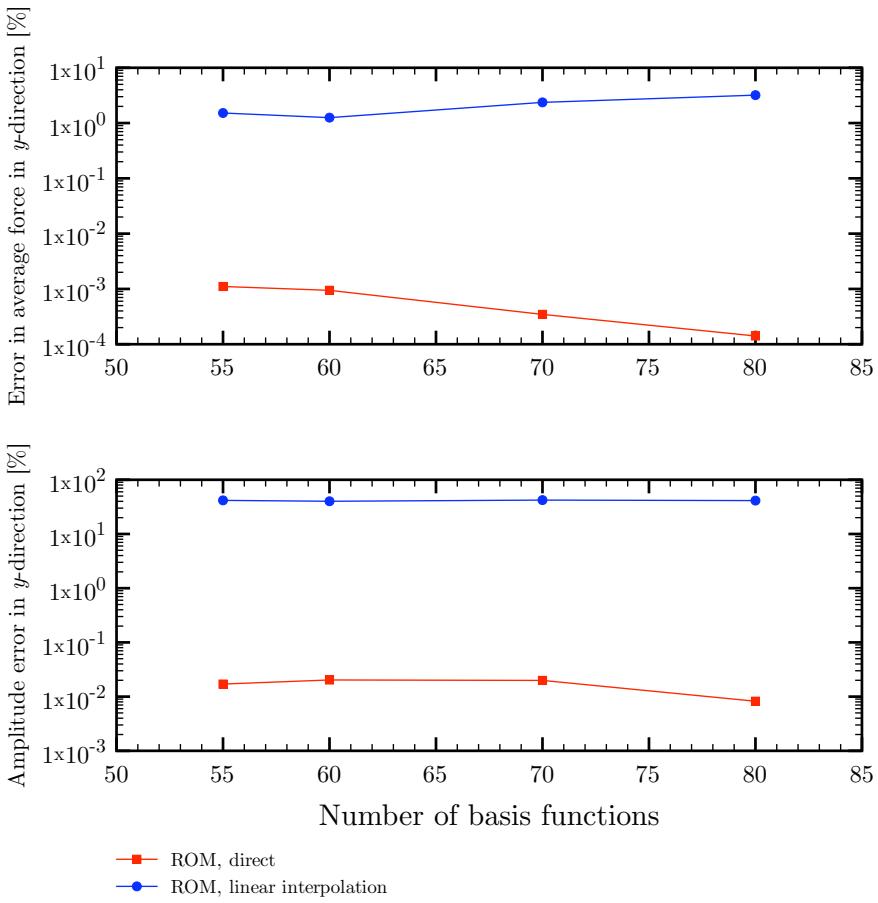

Figure 6.9: Error in average force and in amplitude of oscillation for y-component of force acting on bump, $M_{\text{inlet}} = 0.75$.

Due to the differences in energy accumulation between the subsonic and transonic simulations, for the ROM results that follow, eight basis functions were used for the subsonic simulations and sixty basis functions were used for the transonic simulations.

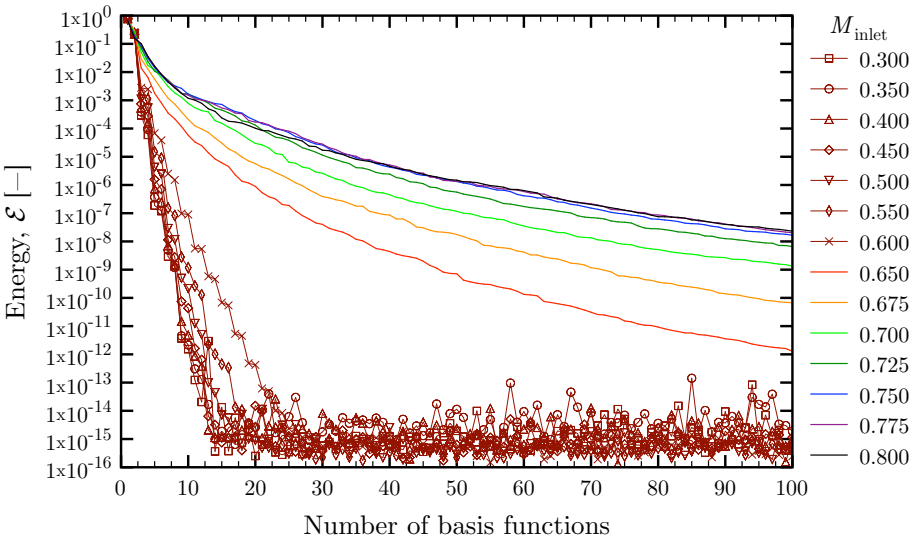

Figure 6.10: Energy spectrum using Equation (6.2), $\varepsilon = 0.05$.

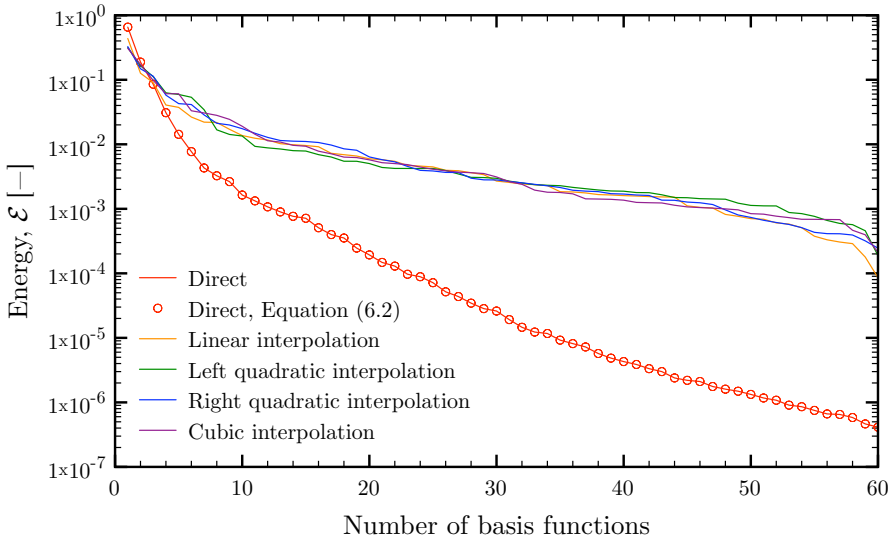

Figure 6.11: Energy spectrum for $M_{\text{inlet}} = 0.75$, $\varepsilon = 0.05$.

6.2 Interpolation Order

The ROM simulations presented in this dissertation include those that used basis functions that were derived directly from the FOM, as well as those that were obtained by interpolating between basis functions derived from snapshots corresponding to different flow simulations. Linear, two types of quadratic, and cubic interpolation were performed on the tangent space to the Grassmann manifold. Figure 6.12 shows the conditions associated with each set of basis functions used in the interpolation.

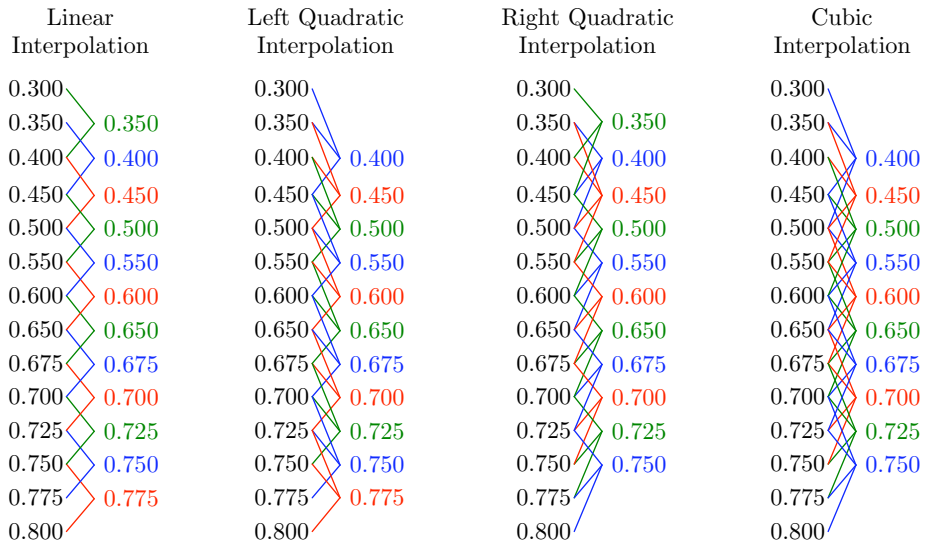

Figure 6.12: The four types of interpolation performed on the tangent space to the Grassmann manifold for the inlet Mach numbers.

The basis functions that arise from interpolation are dependent upon the number of basis functions used in the interpolation. For example, if linear interpolation is performed on the tangent space to the Grassmann manifold, two sets of basis functions corresponding to two bracketing flow conditions would be used. In one scenario, each set could contain m basis functions, and interpolating between the

two sets would yield m basis functions. In a second scenario, each set could contain M basis functions, resulting in M basis functions. In general, if M is greater than m, the m basis functions arising from interpolation in the first scenario will not be a subset of the M basis functions arising in the second scenario.

Figures 6.13–6.16 respectively show the energy of each basis function: $\tilde{\lambda}_i / \sum_{j=1}^{m} \tilde{\lambda}_j$, where $\tilde{\lambda}_j$ was computed from (6.2), for linear, left quadratic, right quadratic, and cubic interpolation. The amount of basis functions shown corresponds to the number used for the simulations. Eight basis functions were used for the subsonic simulations ($M_{\text{inlet}} \leq 0.60$), and sixty basis functions were used for the transonic simulations ($M_{\text{inlet}} \geq 0.65$).

Compared to Figure 6.2, the basis functions obtained from interpolation did not accumulate energy as rapidly as those obtained directly from the FOM. Summing the energy of each basis function obtained from interpolation approached one more slowly. For example, for the Mach 0.40 case, the sum of the energy of the first three basis functions was 0.96316 when the basis functions were obtained using linear interpolation and 0.99985 when the basis functions were obtained directly from the FOM. For the Mach 0.75 case, the sum of the energy of the first three basis functions was 0.65325 when the basis functions were obtained using linear interpolation and 0.92798 when the basis functions were obtained directly.

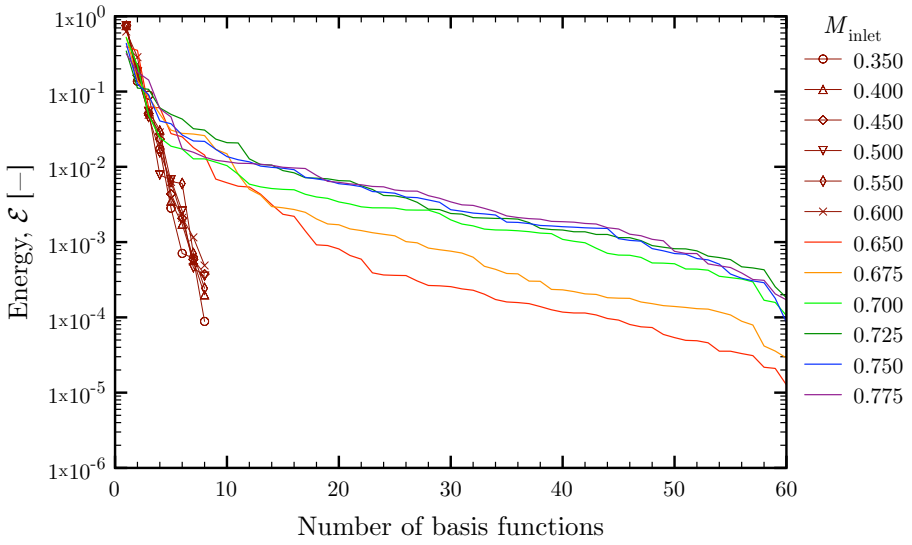

Figure 6.13: Energy spectrum, linear interpolation, $\varepsilon = 0.05$.

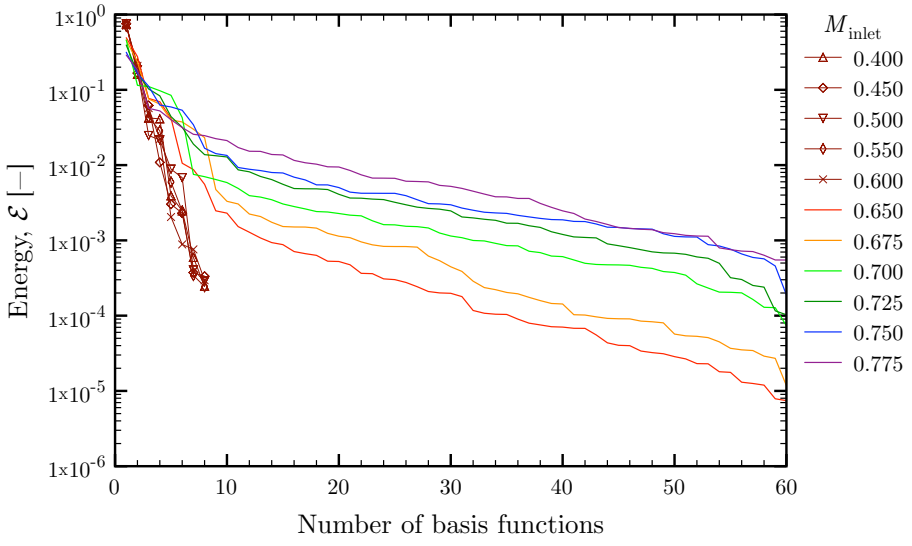

Figure 6.14: Energy spectrum, left quadratic interpolation, $\varepsilon = 0.05$.

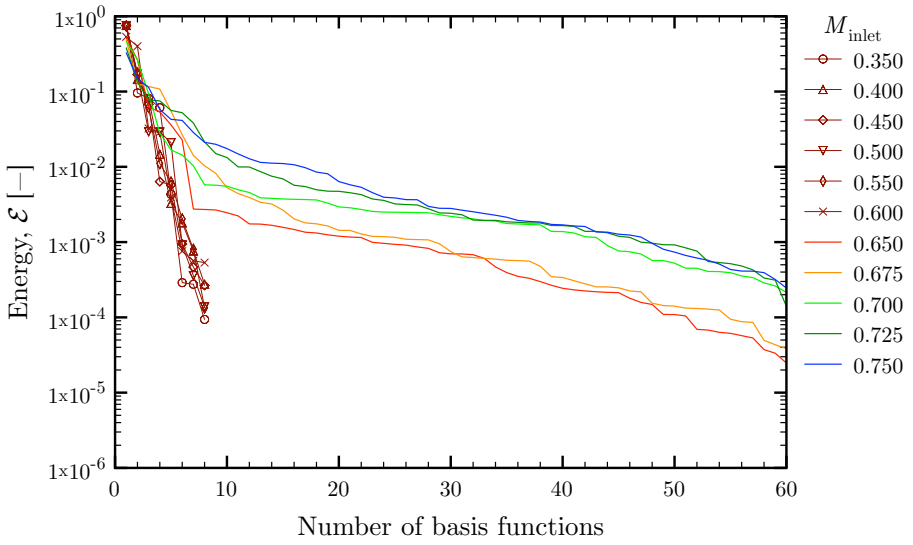

Figure 6.15: Energy spectrum, right quadratic interpolation, $\varepsilon = 0.05$.

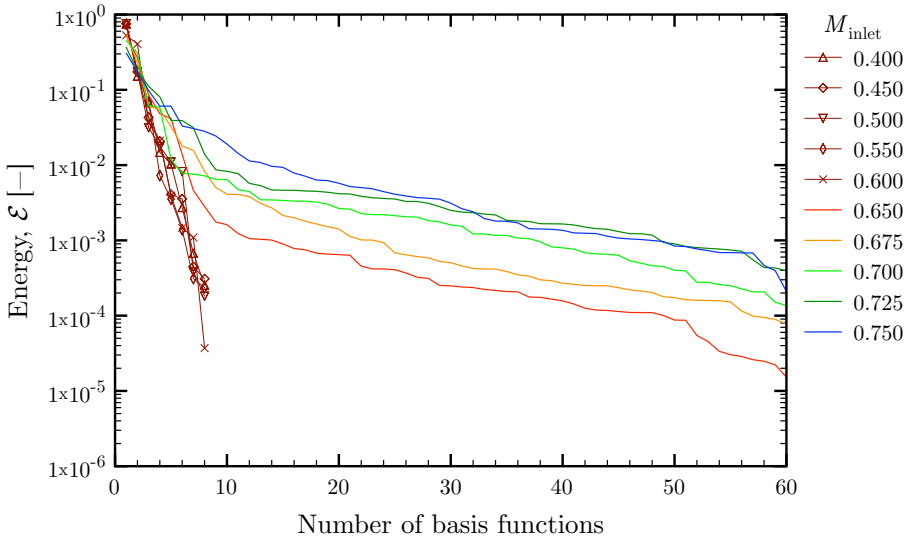

Figure 6.16: Energy spectrum, cubic interpolation, $\varepsilon = 0.05$.

Contour plots of the Mach number and the error at two seconds are shown in Figures 6.17–6.20 for inlet Mach numbers of 0.40 and 0.75. For an inlet Mach number of 0.40, the ROMs using interpolated functions accurately predicted the flow field. Though some of the flow features for an inlet Mach number of 0.75 were qualitatively modeled in Figure 6.19, the accuracy of the interpolated functions noticeably decreased, as shown in Figure 6.20.

The force acting on the bump along the y-direction between 1 and 2 seconds was compared for each of the cases simulated. The time history of the force is plotted in Figure 6.21 for an inlet Mach number of 0.40 and in Figure 6.22 for an inlet Mach number of 0.75.

For an inlet Mach number of 0.40, the force acting on the bump was accurately modeled using the interpolated functions. There was a greater discrepancy in the force for the Mach 0.75 case.

Figure 6.23 shows the average value of the force for each Mach number. The amplitude of the oscillation about the average value is plotted in Figure 6.24.

For the average value and amplitude of oscillation of the force acting on the bump, the error of the ROMs was computed relative to the FOM and is plotted in Figures 6.25 and 6.26. Relative to the FOM, Figure 6.26 shows that the ROMs did not accurately account for the amplitude of the force oscillation as the Mach number increased.

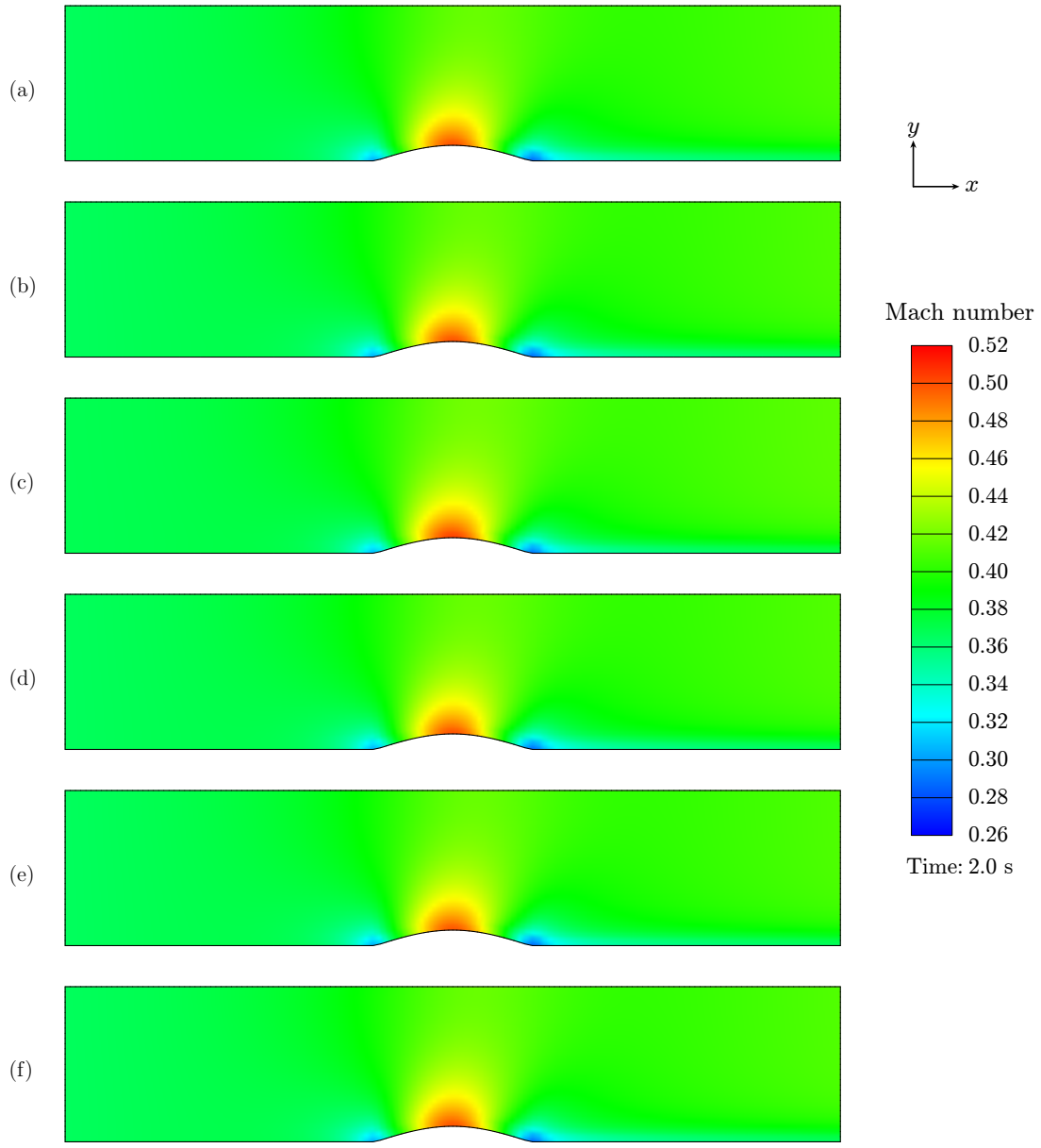

Figure 6.17: Mach number contour plots, $M_{\text{inlet}} = 0.40$. (a) FOM; (b) ROM, direct; (c) ROM, linear interpolation; (d) ROM, left quadratic interpolation; (e) ROM, right quadratic interpolation; and (f) ROM, cubic interpolation.

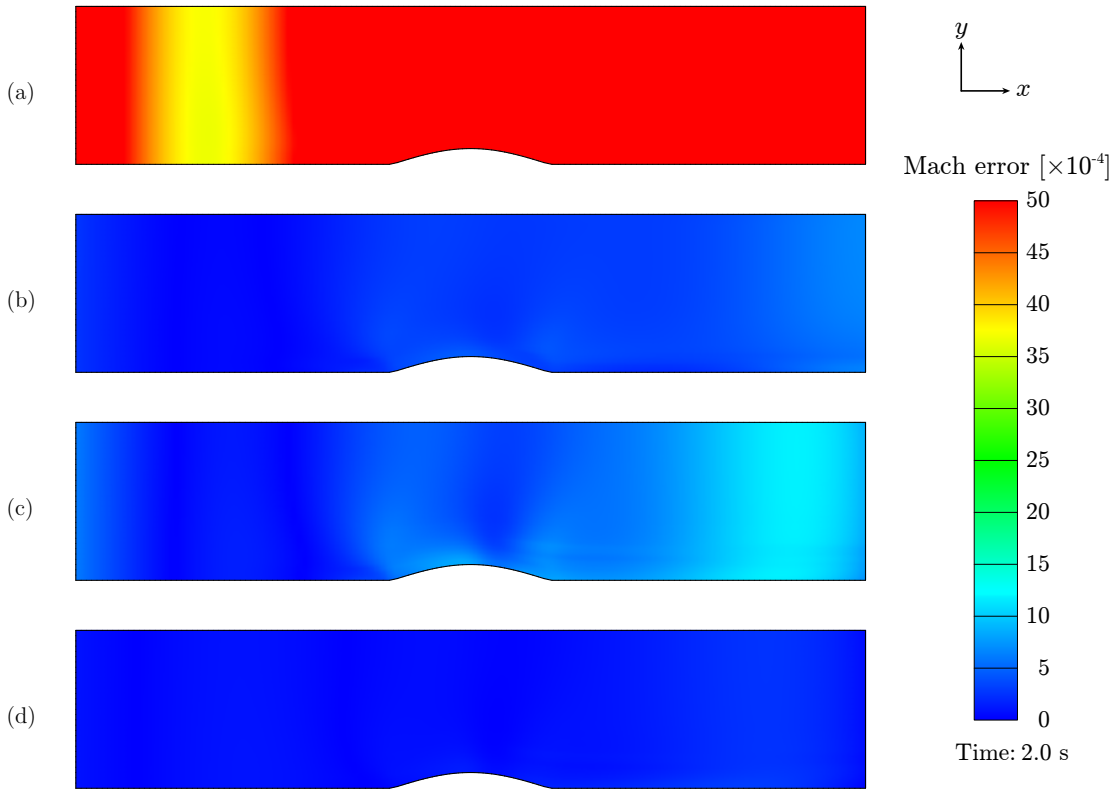

Figure 6.18: Mach number error contour plots, $M_{\text{inlet}} = 0.40$. (a) ROM, linear interpolation; (b) ROM, left quadratic interpolation; (c) ROM, right quadratic interpolation; and (d) ROM, cubic interpolation.

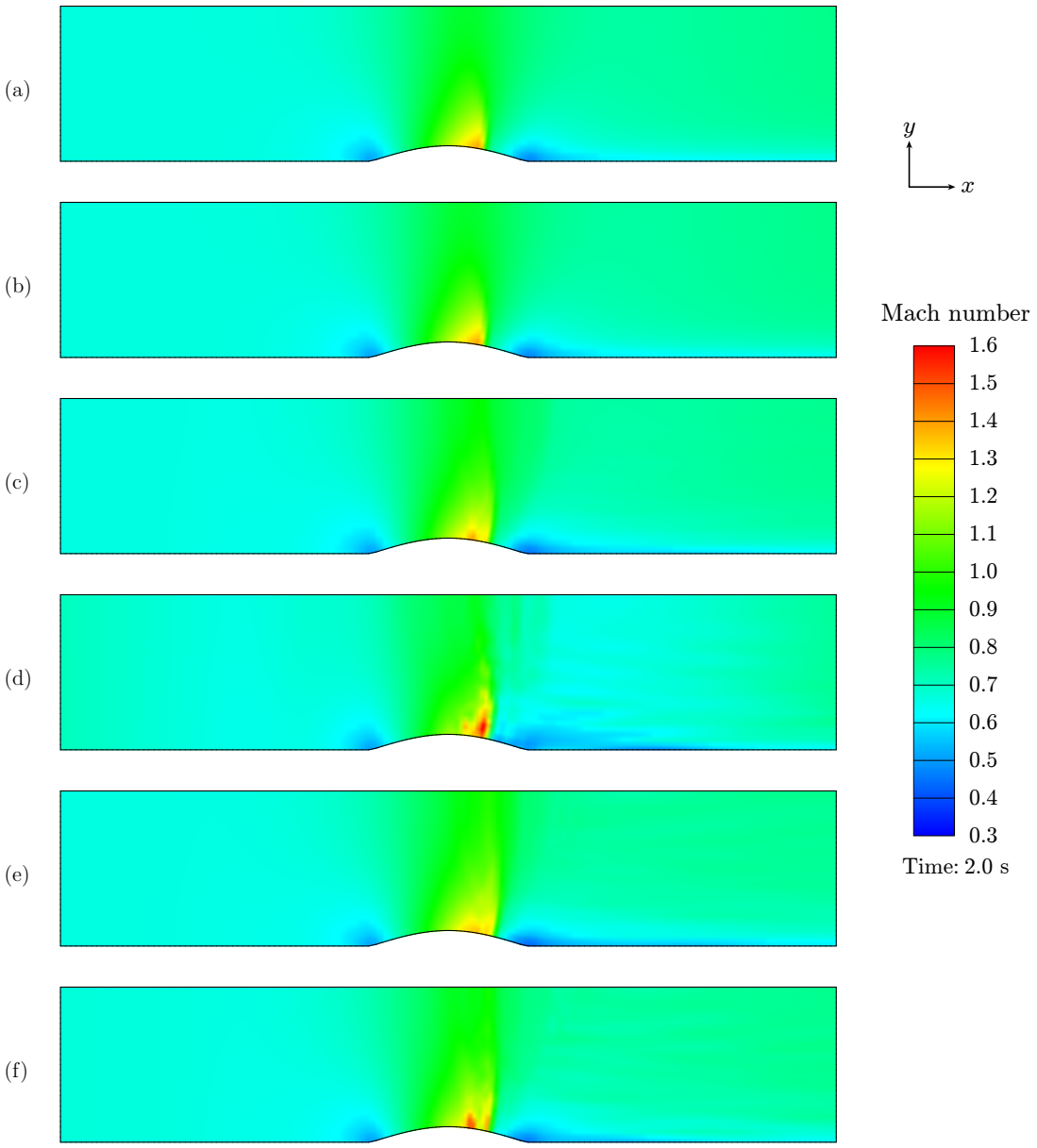

Figure 6.19: Mach number contour plots, $M_{\text{inlet}} = 0.75$. (a) FOM; (b) ROM, direct; (c) ROM, linear interpolation; (d) ROM, left quadratic interpolation; (e) ROM, right quadratic interpolation; and (f) ROM, cubic interpolation.

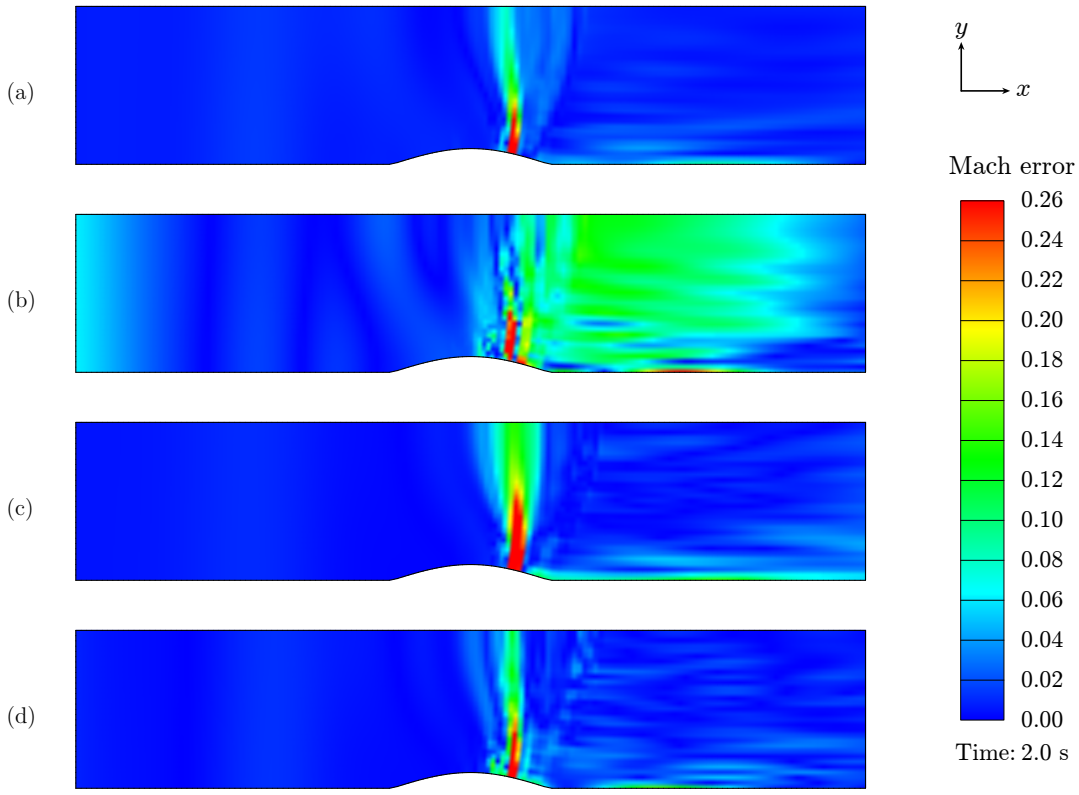

Figure 6.20: Mach number error contour plots, $M_{\text{inlet}} = 0.75$. (a) ROM, linear interpolation; (b) ROM, left quadratic interpolation; (c) ROM, right quadratic interpolation; and (d) ROM, cubic interpolation.

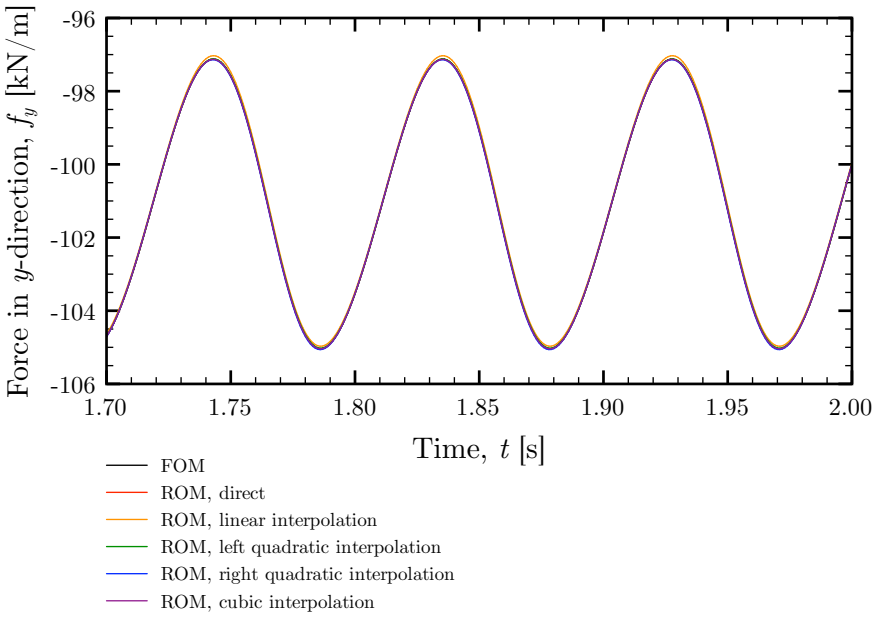

Figure 6.21: Force per unit length acting on bump along y-direction, $M_{\text{inlet}} = 0.40$.

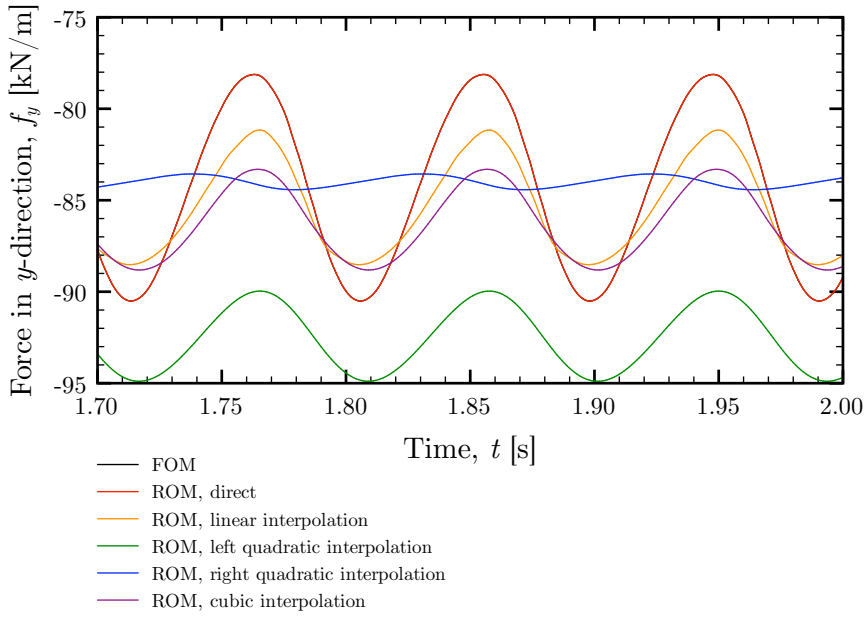

Figure 6.22: Force per unit length acting on bump along y-direction, $M_{\text{inlet}} = 0.75$.

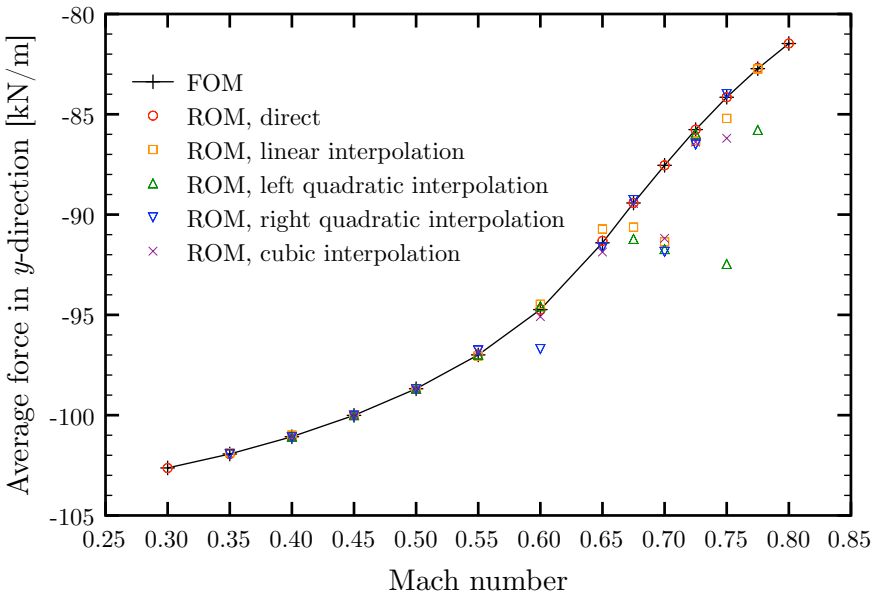

Figure 6.23: Average force acting on bump along y-direction.

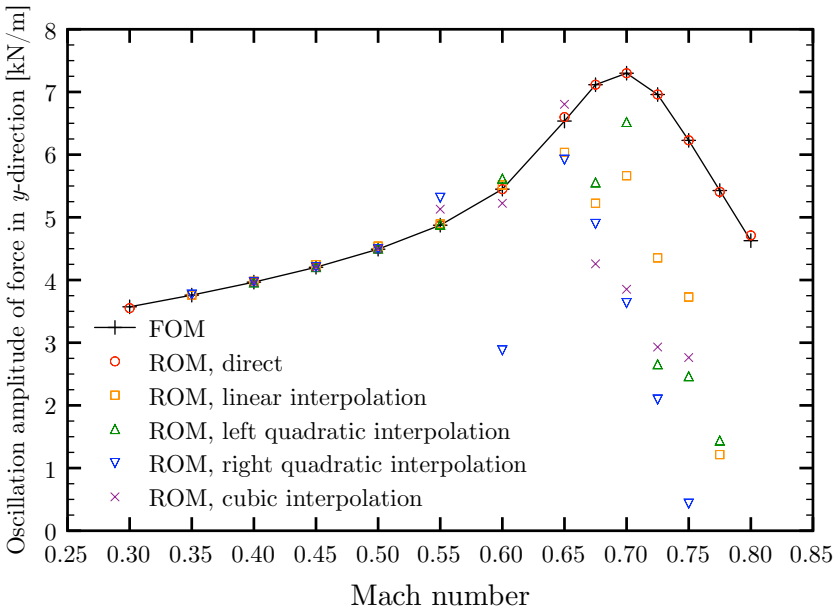

Figure 6.24: Amplitude of oscillation of force acting on bump along y-direction.

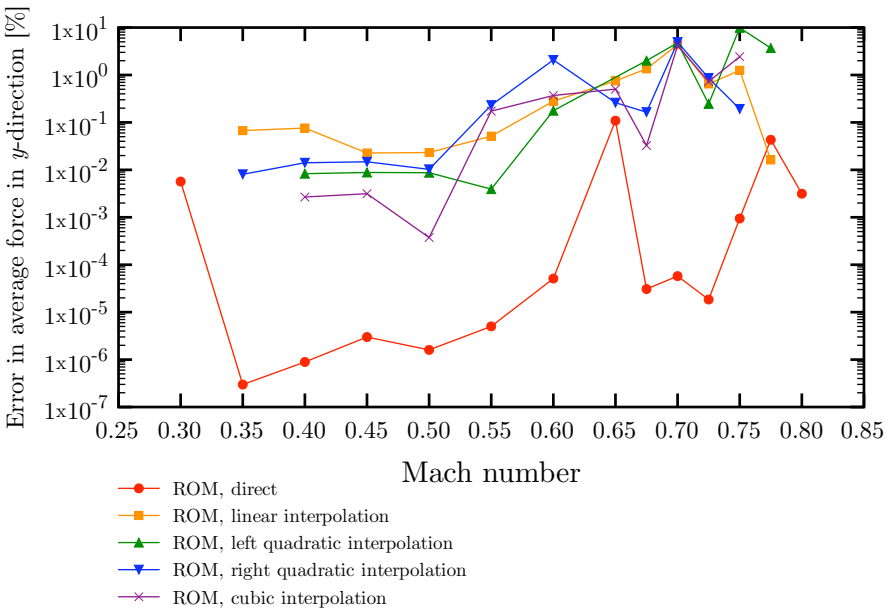

Figure 6.25: Error in average force acting on bump along y-direction.

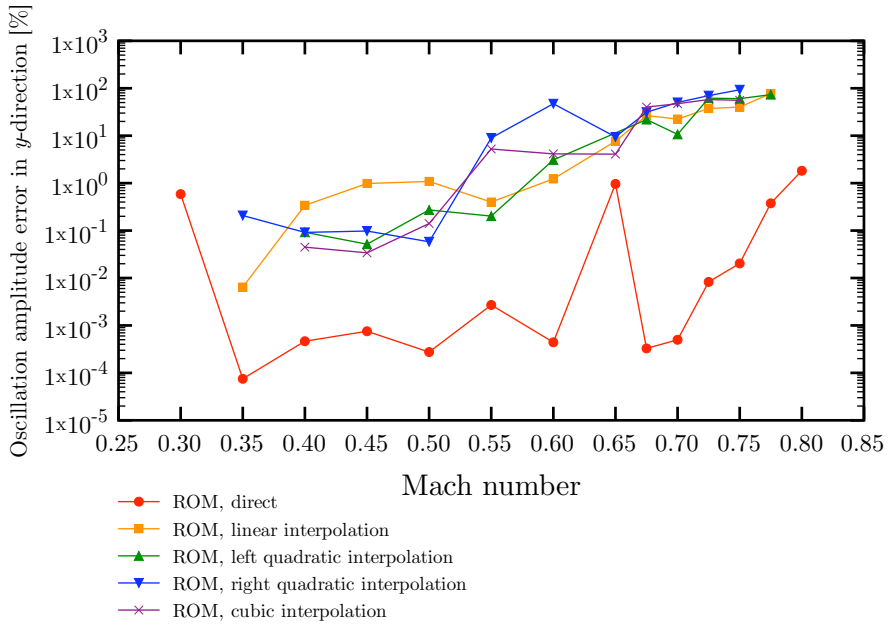

Figure 6.26: Error in oscillation amplitude of force acting on bump along y-direction.

6.3 Discussion

For low Mach numbers, the ROMs using interpolated functions accurately predicted the flow field, as shown in Figures 6.17 and 6.18. Though some of the flow features were qualitatively modeled in the transonic regime, as shown in Figures 6.19 and 6.20, the accuracy of the interpolated results noticeably decreased. The deterioration in fidelity of the ROMs can be attributed to the increasingly nonlinear flow in the transonic regime due to the strong nonlinearities associated with the shock.

For the average force in the y-direction, Figure 6.25 shows that the ROMs using interpolated functions had an error of less than 9.9%. The maximum error occurred for an inlet Mach number of 0.75 using left quadratic interpolation, which interpolated from one subsonic and two transonic cases. On the other hand, the ROM using basis functions obtained directly from the snapshots had an error of less than 0.11%, with the maximum error occurring when the inlet Mach number was 0.65.

For the force in the y-direction, Figure 6.26 shows there was a greater discrepancy between the ROMs and the FOM in the prediction of the oscillation amplitude when compared to the average force. The ROMs using interpolated functions had an error of less than 94%. The maximum error occurred for an inlet Mach number of 0.75 using right quadratic interpolation. On the other hand, the ROM using basis functions obtained directly from the snapshots had an error of less than 1.9%, with the maximum error occurring when the inlet Mach number was 0.80.

Note that Figure 6.25 shows that increasing the interpolation order for inlet Mach numbers up to 0.50 resulted in a more accurate average force in the y-direction. Basis functions obtained from linear and left quadratic interpolation for inlet Mach numbers up to 0.55 were interpolated from basis functions arising from subsonic flow, as were basis functions obtained from right quadratic and cubic interpolation

for inlet Mach numbers up to 0.50.

In the transonic regime, there was not a clear trend between interpolation order and accuracy. Additionally, from Figure 6.26, the error in the amplitude of oscillation for interpolated functions was higher for transonic cases, as well as for subsonic cases that used basis functions obtained from using transonic basis functions in interpolation.

7. COMPUTATIONAL SAVINGS

The focus of the work presented in this dissertation has been on developing suitable basis functions for use in the reduced-order model. The primary incentive for using the ROM is to reduce the degrees of freedom and the computational time. There are two methods for reducing the computational time: (1) computing and projecting the residual onto the basis functions to increase the time step [4, 42] and (2) reformulating the equations and pre-computing the inner products to reduced the problem dimension [20, 47].

7.1 Projecting the Residual onto Basis Functions

By computing the residual and projecting it onto the basis functions, the amount of computations relative to the full-order model is increased. However, the resulting equations have a different Courant–Friedrichs–Lewy (CFL) condition, which permits a greater time step than what was possible with the FOM. Additionally, this approach requires minimal modification to the FOM to create the ROM. This was the method used to create the ROM used in this dissertation.

7.1.1 Outline of Reduced-Order Model

The ROM used in this dissertation was created by modifying the FOM. Figure 7.1 provides a flowchart of the algorithm used by the FOM. The modifications made to the FOM are shown in Figure 7.2, which contains a flowchart of the algorithm used by the ROM.

The majority of the execution time for both models is spent progressing through time. The additional costs per time step associated with the ROM arise from reconstructing the flow field and computing the inner product of the residual and the

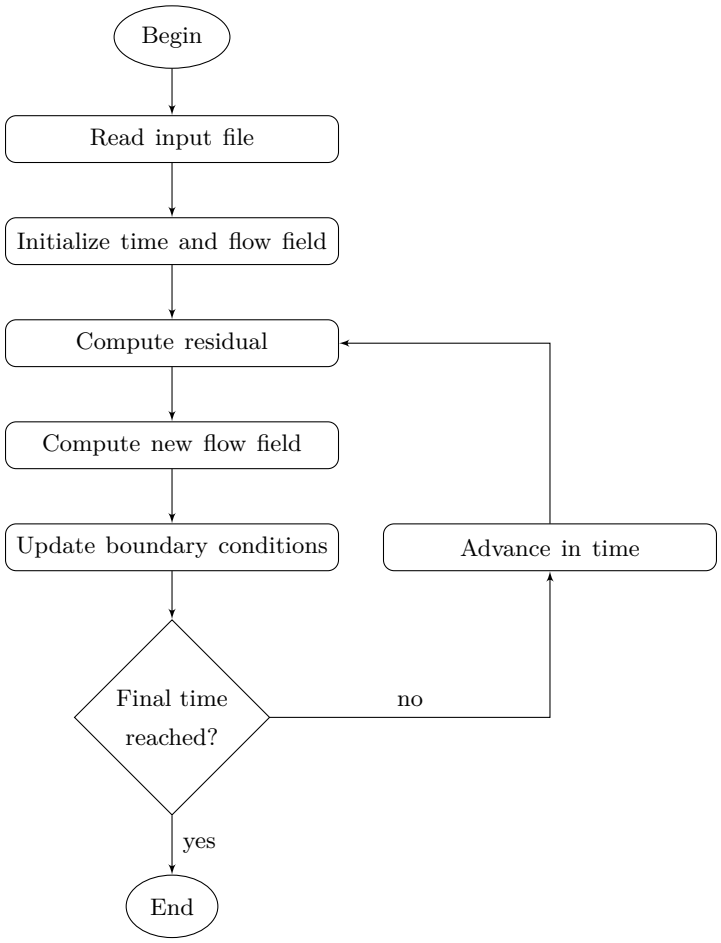

Figure 7.1: Flowchart of FOM algorithm

basis functions.

7.1.2 Computational Time Comparison

For the Tenth Standard Configuration case, fewer pseudo-time steps were needed for the ROM. Figures 7.3 and 7.4 show the relative time consumption during a real-time step for the FOM and for the ROM using three dynamic basis functions. The greatest additional cost arose from reconstructing the flow field and projecting the residual onto the basis functions.

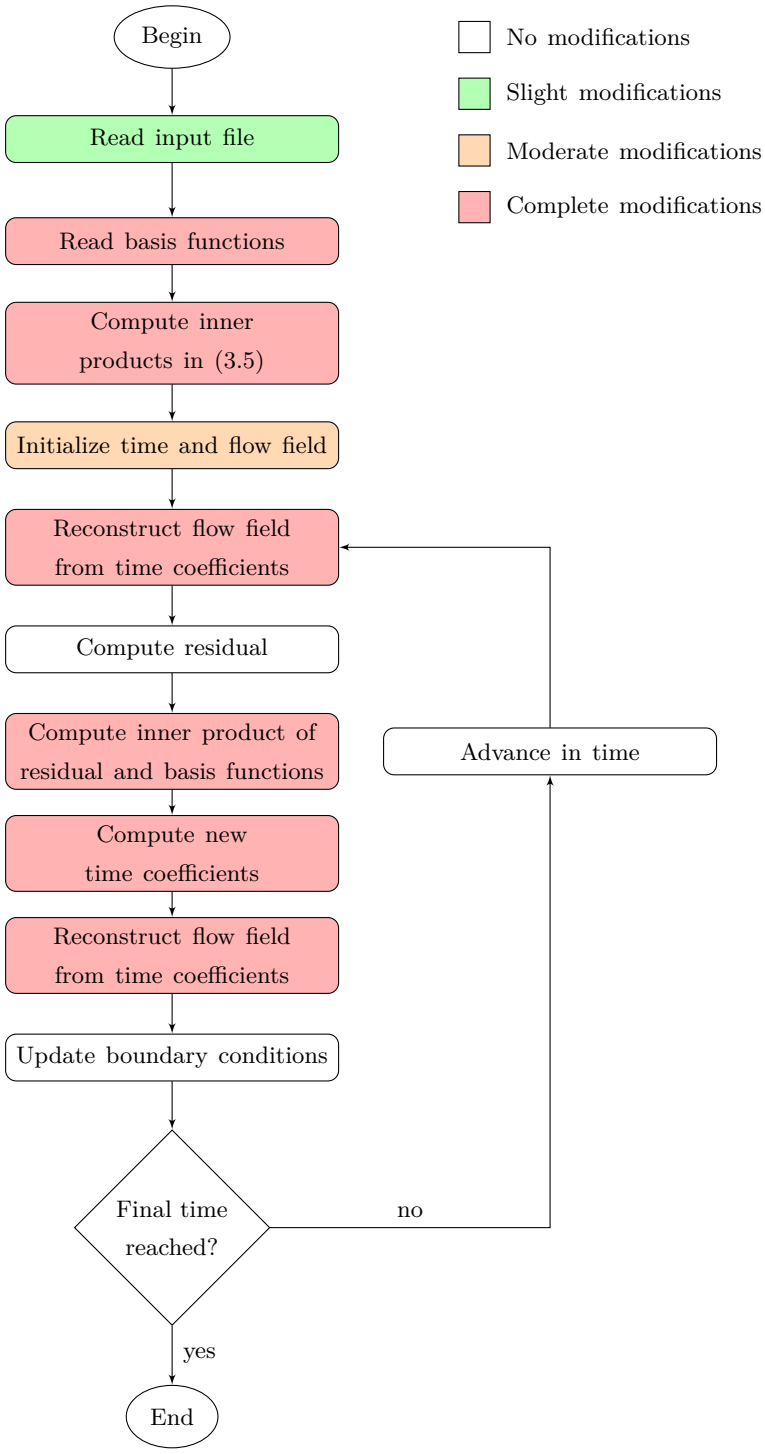

Figure 7.2: Flowchart of ROM algorithm

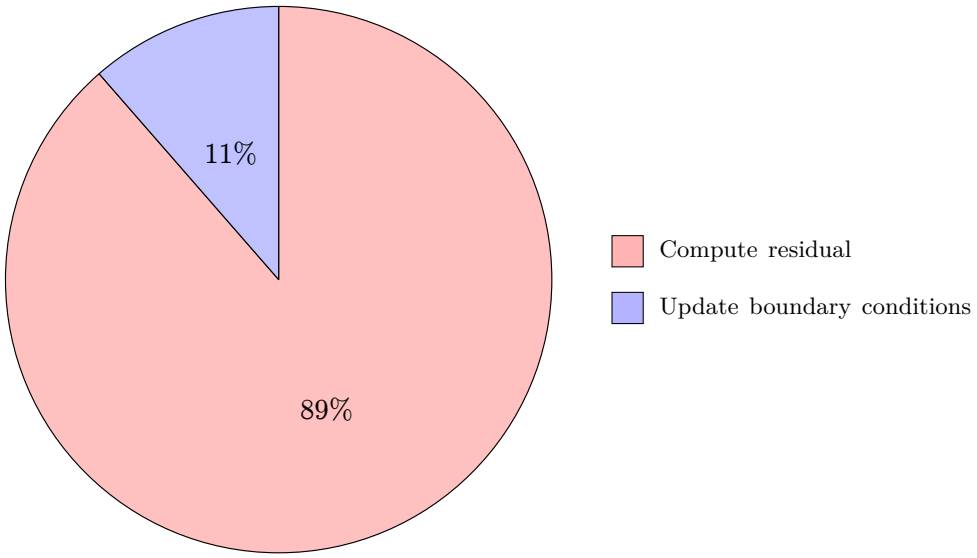

Figure 7.3: Computational time profile for FOM

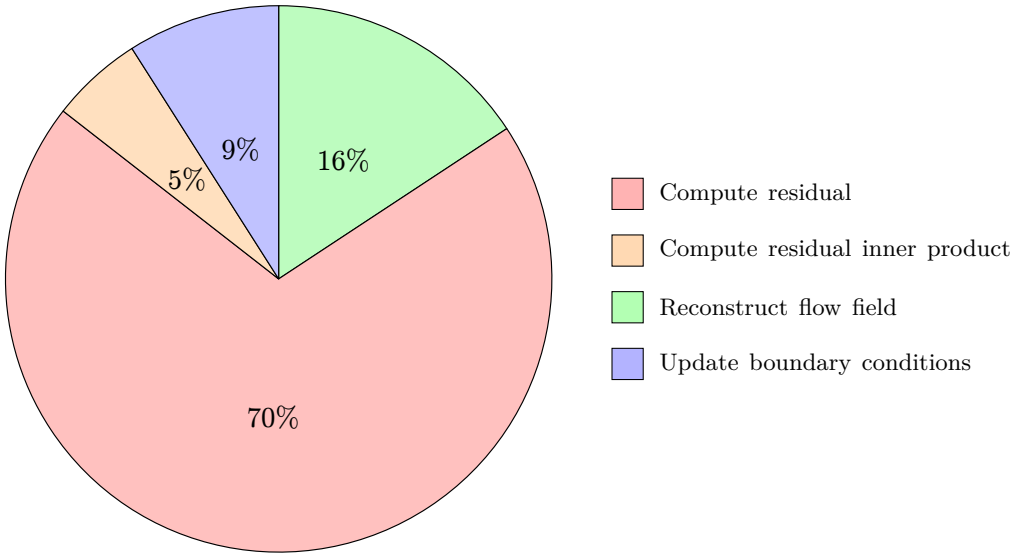

Figure 7.4: Computational time profile for ROM

Figure 7.5 plots the ratio of the execution time of the ROM to the FOM, τ. As shown in the figure, using the dynamic average with static basis functions did not significantly increase the time beyond that required by the static average with static basis functions. The static basis functions with the static and dynamic averages generally required less than half of the time required by the FOM.

The dynamic basis functions required more time than the static basis functions. Additionally, increasing the number of dynamic basis functions increased the computational time more rapidly than by increasing the number of static basis functions. Using three dynamic basis functions yielded a low error, as shown in Subsection 5.2. This resulted in a cost comparable to using the static basis functions, which was approximately 30% of the computational time of the FOM.

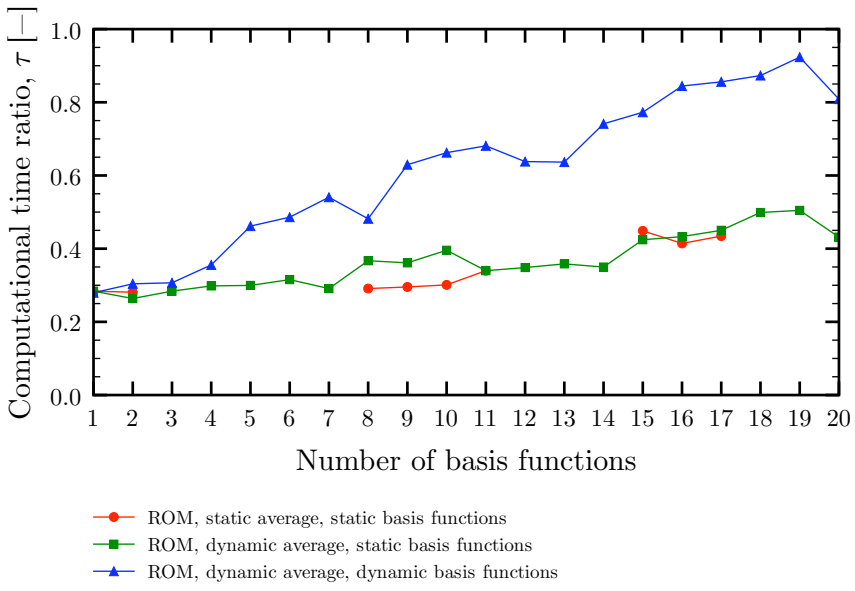

Figure 7.5: Ratio of computational time of ROM to FOM.

7.2 Reformulating the Governing Equations

The other approach to accelerate the ROM is to formulate the governing equations in terms of the primitive variables. Projecting these equations onto the basis functions yields cubic nonlinear equations, for which the inner products can be precomputed. This method requires significant modification to the FOM to create the ROM; however, the problem dimension is reduced considerably, and the flow field does not need to be reconstructed, nor do the inner products need to be computed throughout the simulation.

8. CONCLUSIONS*

This section presents the conclusions drawn from the results shown in the previous sections for the dynamic average and dynamic basis functions and for modeling off-reference flow conditions.

8.1 Dynamic Functions

In this dissertation, a novel approach was presented to model highly nonlinear flows and deforming meshes using proper orthogonal decomposition. The approach consisted of using the dynamic average with dynamic basis functions. The dynamic functions resulted in higher-fidelity results and increased stability with fewer basis functions for the highly nonlinear flows. Using the static basis functions failed or produced lower-quality results for such flows. Nonetheless, the dynamic basis functions and occasionally, the dynamic average, do not always provide an improvement for flows with weak nonlinearities. Therefore, static functions are suitable for modeling subsonic flows and flows with fixed meshes, whereas dynamic functions are better able to overcome the shortcomings of the static functions when the flow is highly nonlinear and the mesh is deforming.

8.2 Off-Reference Flow Conditions

This dissertation additionally presented a novel approach for assessing the number of basis functions used in POD. POD results were compared between subsonic and transonic flows for several cases. For off-reference flow conditions, the effect of interpolation order was investigated.

---

*Part of this section is reprinted with permission from "Using proper orthogonal decomposition to model off-reference flow conditions" by B. A. Freno, T. A. Brenner, P. G. A. Cizmas, 2013. *International Journal of Non-Linear Mechanics*, vol. 54, pp 76–84, Copyright 2013 by Elsevier.

With POD, using too few basis functions results in a low-fidelity result. Conversely, using an excessive amount of basis functions introduces the challenge of resolving higher frequencies. Additionally, it was shown that arbitrarily increasing the number of basis functions does not guarantee an improvement. Taking into account the energy contribution of each basis function provides greater insight than considering only the cumulative energy.

The results from several cases in the subsonic and transonic flow regimes were presented for inviscid flow through a channel. The FOM was compared with ROMs using basis functions generated through POD of the FOM for identical flow conditions, and through interpolation of the basis functions from flow conditions bracketing the value of interest. Interpolation yielded good results for subsonic cases; however, the fidelity of the transonic cases was noticeably lower.

Using basis functions obtained through interpolation naturally results in a decrease in fidelity compared to using basis functions extracted from snapshots of the FOM for identical flow conditions, especially for transonic flows. However, interpolation remains practical given the goal of reducing computational time while achieving acceptable results.

REFERENCES

[1] E. H. Dowell and K. C. Hall, "Modeling of fluid–structure interaction," *Annual Review of Fluid Mechanics*, vol. 33, pp. 445–490, January 2001.

[2] P. Holmes, J. L. Lumley, and G. Berkooz, *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*. Cambridge: Cambridge University Press, 1996.

[3] E. H. Dowell and D. Tang, *Dynamics of Very High Dimensional Systems*. Hackensack, NJ: World Scientific Publishing Company, 2003.

[4] D. J. Lucia, P. S. Beran, and W. A. Silva, "Reduced-order modeling: New approaches for computational physics," *Progress in Aerospace Sciences*, vol. 40, pp. 51–117, February 2004.

[5] M. F. Barone and J. L. Payne, "Methods for simulation-based analysis of fluid–structure interaction," Tech. Rep. SAND2005-6573, Sandia National Laboratories, Albuquerque, NM, October 2005.

[6] H. M. Park and M. W. Lee, "An efficient method of solving the Navier–Stokes equations for flow control," *International Journal for Numerical Methods in Engineering*, vol. 41, pp. 1133–1151, March 1998.

[7] T. Yuan, P. G. Cizmas, and T. O'Brien, "A reduced-order model for a bubbling fluidized bed based on proper orthogonal decomposition," *Computers and Chemical Engineering*, vol. 30, pp. 243–259, December 2005.

[8] R. F. Schmit and M. N. Glauser, "Improvements in low dimensional tools for flow–structure interaction problems: Using global POD," in *42nd AIAA Aerospace Sciences Meeting and Exhibit*, AIAA 2004-889, (Reno, NV), AIAA, January 2004.

[9] T. Lieu and M. Lesoinne, "Parameter adaptation of reduced order models for three-dimensional flutter analysis," in *42nd AIAA Aerospace Sciences Meeting and Exhibit*, AIAA 2004-888, (Reno, NV), AIAA, January 2004.

[10] T. Lieu, C. Farhat, and M. Lesoinne, "Reduced-order fluid/structure modeling of a complete aircraft configuration," *Computer Methods in Applied Mechanics and Engineering*, vol. 195, pp. 5730–5742, August 2006.

[11] T. Lieu and C. Farhat, "Adaptation of aeroelastic reduced-order models and application to an F-16 configuration," *AIAA Journal*, vol. 45, pp. 1244–1257, June 2007.

[12] D. Amsallem and C. Farhat, "Interpolation method for adapting reduced-order models and application to aeroelasticity," *AIAA Journal*, vol. 46, pp. 1803–1813, July 2008.

[13] D. Amsallem, *Interpolation on Manifolds of CFD-Based Fluid and Structural Reduced-Order Models for On-Line Aeroelastic Predictions*. PhD thesis, Stanford University, Stanford, CA, June 2010.

[14] B. A. Freno, T. A. Brenner, and P. G. A. Cizmas, "Using proper orthogonal decomposition to model off-reference flow conditions," *International Journal of Non-Linear Mechanics*, vol. 54, pp. 76–84, September 2013.

[15] A. Hay, J. Borggaard, and D. Pelletier, "On the use of sensitivity analysis to improve reduced-order models," in *4th Flow Control Conference*, AIAA 2008-4192, (Seattle, WA), AIAA, June 2008.

[16] A. Hay, J. Borggaard, I. Akhtar, and D. Pelletier, "Reduced-order models for parameter dependent geometries based on shape sensitivity analysis," *Journal of Computational Physics*, vol. 229, pp. 1327–1352, February 2010.

[17] C. Kasnakoğlu, A. Serrani, and M. Ö. Efe, "Control input separation by actuation mode expansion for flow control problems," *International Journal of*

Control, vol. 81, pp. 1475–1492, June 2008.

[18] R. Bourguet, M. Braza, and A. Dervieux, "Reduced-order modeling of transonic flows around an airfoil submitted to small deformations," *Journal of Computational Physics*, vol. 230, pp. 159–184, January 2011.

[19] F. Vetrano, C. L. Garrec, G. D. Mortchelewicz, and R. Ohayon, "Assessment of strategies for interpolating POD based reduced order models and application to aeroelasticity," *Journal of Aeroelasticity and Structural Dynamics*, vol. 2, no. 2, pp. 85–104, 2011.

[20] C. W. Rowley, T. Colonius, and R. M. Murray, "Model reduction for compressible flows using POD and Galerkin projection," *Physica D*, vol. 189, pp. 115–129, February 2004.

[21] M. F. Barone, I. Kalashnikova, D. J. Segalman, and H. K. Thornquist, "Stable Galerkin reduced order models for linearized compressible flow," *Journal of Computational Physics*, vol. 228, pp. 1932–1946, April 2009.

[22] T. A. Brenner, R. L. Fontenot, P. G. A. Cizmas, T. J. O'Brien, and R. W. Breault, "A reduced-order model for heat transfer in multiphase flow and practical aspects of the proper orthogonal decomposition," *Computers & Chemical Engineering*, vol. 43, pp. 68–80, August 2012.

[23] T. A. Brenner, R. L. Fontenot, P. G. A. Cizmas, T. J. O'Brien, and R. W. Breault, "Augmented proper orthogonal decomposition for problems with moving discontinuities," *Powder Technology*, vol. 203, pp. 78–85, October 2010.

[24] P. G. A. Cizmas, B. R. Richardson, T. A. Brenner, T. J. O'Brien, and R. W. Breault, "Acceleration techniques for reduced-order models based on proper orthogonal decomposition," *Journal of Computational Physics*, vol. 227, pp. 7791–7812, August 2008.

[25] K. C. Hall, J. P. Thomas, and E. H. Dowell, "Proper orthogonal decomposition

technique for transonic unsteady aerodynamic flows," *AIAA Journal*, vol. 38, pp. 1853–1862, October 2000.

[26] J. P. Thomas, E. H. Dowell, and K. C. Hall, "Three-dimensional transonic aeroelasticity using proper orthogonal decomposition-based reduced-order models," *Journal of Aircraft*, vol. 40, pp. 544–551, May–June 2003.

[27] T. Bui-Thanh, K. Willcox, and O. Ghattas, "Parametric reduced-order models for probabilistic analysis of unsteady aerodynamic applications," *AIAA Journal*, vol. 46, pp. 2520–2529, October 2008.

[28] J. S. Anttonen, *Techniques for Reduced Order Modeling of Aeroelastic Structures with Deforming Grids*. PhD thesis, Air Force Institute of Technology, Wright-Patterson AFB, OH, October 2001.

[29] G. Lewin and H. Haj-Hariri, "Reduced-order modeling of a heaving airfoil," *AIAA Journal*, vol. 43, pp. 270–283, February 2005.

[30] J. Anttonen, P. King, and P. Beran, "Applications of multi-POD to a pitching and plunging airfoil," *Mathematical and Computer Modelling*, vol. 42, pp. 245–259, August 2005.

[31] A. Placzek, D.-M. Tran, and R. Ohayon, "A nonlinear POD-Galerkin reduced-order model for compressible flows taking into account rigid body motions," *Computer Methods in Applied Mechanics and Engineering*, vol. 200, pp. 3497–3514, December 2011.

[32] B. A. Freno and P. G. A. Cizmas, "A proper orthogonal decomposition method for nonlinear flows with deforming meshes," in *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, AIAA 2013-0055, (Grapevine, TX), AIAA, 2013.

[33] J. Anttonen, P. King, and P. Beran, "POD-based reduced-order models with deforming grids," *Mathematical and Computer Modelling*, vol. 38, pp. 41–62,

July 2003.

[34] E. Liberge and A. Hamdouni, "Reduced order modelling method via proper orthogonal decomposition (POD) for flow around an oscillating cylinder," *Journal of Fluids and Structures*, vol. 26, pp. 292–311, February 2010.

[35] L. Sirovich, "Turbulence and the dynamics of coherent structures," *Quarterly of Applied Mathematics*, vol. 45, pp. 561–590, October 1987.

[36] P. Geladi and B. R. Kowalski, "Partial least-squares regression: A tutorial," *Analytica Chimica Acta*, vol. 185, pp. 1–17, 1986.

[37] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and Intelligent Laboratory Systems*, vol. 2, pp. 37–52, August 1987.

[38] R. Byrd, P. Lu, J. Nocedal, and C. Zhu, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on Scientific Computing*, vol. 16, pp. 1190–1208, September 1995.

[39] J. L. Morales and J. Nocedal, "Remark on "Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound constrained optimization"," *ACM Transactions on Mathematical Software*, vol. 38, pp. 7:1–7:4, November 2011.

[40] M. Vinokur, "An analysis of finite-difference and finite-volume formulations of conservation laws," *Journal of Computational Physics*, vol. 81, pp. 1–52, March 1989.

[41] J. Trépanier, M. Reggio, H. Zhang, and R. Camarero, "A finite-volume method for the Euler equations on arbitrary Lagrangian–Eulerian grids," *Computers & Fluids*, vol. 20, no. 4, pp. 399–409, 1991.

[42] T. A. Brenner, *Practical Aspects of the Implementation of Reduced-Order Models Based on Proper Orthogonal Decomposition*. PhD thesis, Texas A&M University, College Station, TX, May 2011.

[43] P. G. A. Cizmas, J. I. Gargoloff, T. W. Strganac, and P. S. Beran, "A parallel

multigrid algorithm for aeroelasticity simulations," *Journal of Aircraft*, vol. 47, pp. 53–63, January–February 2010.

[44] A. de Boer, M. van der Schoot, and H. Bijl, "Mesh deformation based on radial basis function interpolation," *Computers & Structures*, vol. 85, pp. 784–795, June–July 2007.

[45] J. M. Verdon, "Linearized unsteady aerodynamic theory," in *AGARD Manual on Aeroelasticity in Axial-Flow Turbomachines* (M. F. Platzer and F. O. Carta, eds.), vol. 1, March 1987.

[46] T. H. Fransson and J. M. Verdon, "Updated report on "standard configurations for unsteady flow through vibrating axial-flow turbomachine-cascades"," tech. rep., AGARD, July 1991.

[47] A. Iollo, S. Lanteri, and J.-A. Désidéri, "Stability properties of POD-Galerkin approximations for the compressible Navier–Stokes equations," *Theoretical and Computational Fluid Dynamics*, vol. 13, no. 6, pp. 377–396, 2000.

APPENDIX A

NORM SIMPLIFICATION

The square of the norm of the approximation when expanded is

$$\begin{aligned}
\left\| \sum_{j=1}^{m} \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_j\right)}{\left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j\right)} \boldsymbol{\varphi}_j \right\|^2 &= \left(\sum_{j=1}^{m} \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_j\right)}{\left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j\right)} \boldsymbol{\varphi}_j, \sum_{k=1}^{m} \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_k\right)}{(\boldsymbol{\varphi}_k, \boldsymbol{\varphi}_k)} \boldsymbol{\varphi}_k \right), \\
&= \sum_{j=1}^{m} \left(\frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_j\right)}{\left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j\right)} \boldsymbol{\varphi}_j, \sum_{k=1}^{m} \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_k\right)}{(\boldsymbol{\varphi}_k, \boldsymbol{\varphi}_k)} \boldsymbol{\varphi}_k \right), \\
&= \sum_{j=1}^{m} \sum_{k=1}^{m} \left(\frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_j\right)}{\left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j\right)} \boldsymbol{\varphi}_j, \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_k\right)}{(\boldsymbol{\varphi}_k, \boldsymbol{\varphi}_k)} \boldsymbol{\varphi}_k \right), \\
&= \sum_{j=1}^{m} \sum_{k=1}^{m} \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_j\right)}{\left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j\right)} \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_k\right)}{(\boldsymbol{\varphi}_k, \boldsymbol{\varphi}_k)} \left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_k\right).
\end{aligned}$$

If the basis functions are mutually orthogonal, $\left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_k\right) = 0$ when $j \neq k$. Therefore,

$$\sum_{j=1}^{m} \sum_{k=1}^{m} \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_j\right)}{\left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j\right)} \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_k\right)}{(\boldsymbol{\varphi}_k, \boldsymbol{\varphi}_k)} \left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_k\right) = \sum_{j=1}^{m} \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_j\right)}{\left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j\right)} \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_j\right)}{\left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j\right)} \left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j\right) = \sum_{j=1}^{m} \frac{\left(\tilde{\mathbf{U}}, \boldsymbol{\varphi}_j\right)^2}{\left(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j\right)}.$$

APPENDIX B

DERIVATIVE OF THE FUNCTIONAL

To extremize

$$J[\boldsymbol{\varphi}] \equiv \left\langle \frac{\boldsymbol{\varphi}^T \hat{\mathbf{A}}(t) \boldsymbol{\varphi}}{(\boldsymbol{\varphi}, \boldsymbol{\varphi})} \right\rangle,$$

the derivative $\nabla_{\boldsymbol{\varphi}} J$ is set to zero. A Taylor series expansion yields

$$J[\boldsymbol{\varphi} + \delta\boldsymbol{\psi}] = J[\boldsymbol{\varphi}] + (\nabla_{\boldsymbol{\varphi}} J) \cdot \delta\boldsymbol{\psi} + o(|\delta\boldsymbol{\psi}|).$$

Equivalently,

$$(\nabla_{\boldsymbol{\varphi}} J) \cdot \boldsymbol{\psi} = \lim_{\delta \to 0} \frac{J[\boldsymbol{\varphi} + \delta\boldsymbol{\psi}] - J[\boldsymbol{\varphi}]}{\delta} = \frac{\partial J}{\partial \delta}[\boldsymbol{\varphi} + \delta\boldsymbol{\psi}]\bigg|_{\delta=0}.$$

The derivative of the functional is obtained by computing $\dfrac{\partial J}{\partial \delta}[\boldsymbol{\varphi} + \delta\boldsymbol{\psi}]\bigg|_{\delta=0}$:

$$\left.\frac{\partial J}{\partial \delta}[\boldsymbol{\varphi}+\delta\boldsymbol{\psi}]\right|_{\delta=0} = \left\langle \frac{(\boldsymbol{\varphi},\boldsymbol{\varphi})\,\dfrac{\partial}{\partial\delta}\left[(\boldsymbol{\varphi}+\delta\boldsymbol{\psi})^T\,\hat{\mathbf{A}}(t)\,(\boldsymbol{\varphi}+\delta\boldsymbol{\psi})\right]\Big|_{\delta=0}}{(\boldsymbol{\varphi},\boldsymbol{\varphi})^2} \right.$$
$$\left. - \frac{(\boldsymbol{\varphi}^T\hat{\mathbf{A}}(t)\boldsymbol{\varphi})\,\dfrac{\partial}{\partial\delta}\,(\boldsymbol{\varphi}+\delta\boldsymbol{\psi},\boldsymbol{\varphi}+\delta\boldsymbol{\psi})\Big|_{\delta=0}}{(\boldsymbol{\varphi},\boldsymbol{\varphi})^2} \right\rangle,$$

$$= \left\langle \frac{(\boldsymbol{\varphi},\boldsymbol{\varphi})\left(\boldsymbol{\varphi}^T\hat{\mathbf{A}}(t)\boldsymbol{\psi}+\boldsymbol{\psi}^T\hat{\mathbf{A}}(t)\boldsymbol{\varphi}\right)}{(\boldsymbol{\varphi},\boldsymbol{\varphi})^2} - \frac{(\boldsymbol{\varphi}^T\hat{\mathbf{A}}(t)\boldsymbol{\varphi})\,[(\boldsymbol{\varphi},\boldsymbol{\psi})+(\boldsymbol{\psi},\boldsymbol{\varphi})]}{(\boldsymbol{\varphi},\boldsymbol{\varphi})^2} \right\rangle,$$

$$= \left\langle \frac{\left(\boldsymbol{\varphi}^T\hat{\mathbf{A}}(t)\boldsymbol{\psi}+\boldsymbol{\psi}^T\hat{\mathbf{A}}(t)\boldsymbol{\varphi}\right)}{(\boldsymbol{\varphi},\boldsymbol{\varphi})} - 2\frac{(\boldsymbol{\varphi}^T\hat{\mathbf{A}}(t)\boldsymbol{\varphi})\,(\boldsymbol{\psi},\boldsymbol{\varphi})}{(\boldsymbol{\varphi},\boldsymbol{\varphi})^2} \right\rangle.$$

Since $\hat{\mathbf{A}}$ is symmetric,

$$\left.\frac{\partial J}{\partial \delta}[\boldsymbol{\varphi}+\delta\boldsymbol{\psi}]\right|_{\delta=0} = 2\left\langle \frac{\boldsymbol{\psi}^T\hat{\mathbf{A}}(t)\boldsymbol{\varphi}}{(\boldsymbol{\varphi},\boldsymbol{\varphi})} - \frac{(\boldsymbol{\varphi}^T\hat{\mathbf{A}}(t)\boldsymbol{\varphi})\,(\boldsymbol{\psi},\boldsymbol{\varphi})}{(\boldsymbol{\varphi},\boldsymbol{\varphi})^2} \right\rangle.$$

The derivative is set to zero:

$$2\left\langle \frac{\boldsymbol{\psi}^T\hat{\mathbf{A}}(t)\boldsymbol{\varphi}}{(\boldsymbol{\varphi},\boldsymbol{\varphi})} - \frac{(\boldsymbol{\varphi}^T\hat{\mathbf{A}}(t)\boldsymbol{\varphi})\,(\boldsymbol{\psi},\boldsymbol{\varphi})}{(\boldsymbol{\varphi},\boldsymbol{\varphi})^2} \right\rangle = 0$$

and must hold for an arbitrary $\boldsymbol{\psi}$; therefore,

$$\left\langle \frac{\hat{\mathbf{A}}(t)\boldsymbol{\varphi}}{(\boldsymbol{\varphi},\boldsymbol{\varphi})} - \frac{(\boldsymbol{\varphi}^T\hat{\mathbf{A}}(t)\boldsymbol{\varphi})\boldsymbol{\varphi}}{(\boldsymbol{\varphi},\boldsymbol{\varphi})^2} \right\rangle = \mathbf{0}.$$