Coarse analysis of multiscale systems: diffuser flows, charged particle motion, and connections to averaging theory

Thesis by

Jimmy Fung

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

2005

(Defended May 10, 2005)

to Elizabeth-Sharon

Acknowledgements

I thank my advisors, Richard Murray and Jerrold Marsden, for their support and for their contributions to this thesis. It was my pleasure to observe in them not only their technical insight, but their inclination towards exploration and their patience in guiding my research.

I also thank Yannis Kevrekidis for many fruitful conversations and guidance throughout the course of my research, including two prolific visits to Princeton University. I am grateful to Tim Colonius, who served on my committee, for his help with many of the computational aspects of my work, including the diffuser code. Michael Ortiz and Mory Gharib also served on my thesis committee and offered many helpful comments. I also enjoyed many conversations, both personal and professional, with Wang-Sang Koon and Doug MacMynowski.

During my studies at Caltech I had the pleasure of working with Harish Bhat (without whom the particles work would not have been done), Matt West, Troy Smith, Abhishek Tiwari, and Raktim Bhattacharya; I thank them for their patience, persistence, and friendship. I also thank Clancy Rowley, whose work inspired me towards multidisciplinary research.

Early in my studies at Caltech, I had the pleasure of working with Paul Dimotakis, his staff including Pavel Svitek and Earl Dahl, and the GALCIT shop guys. To them I owe a great appreciation

for experimental fluid mechanics. Michel Tanguay and Jeff Krimmel were extremely effective in ensuring the reliability of the computers used during my research. I would like to thank the many staff members who spent much time working to make my life easier: Wendy McKay, Charmaine Boyd, Betty Sue Herrala, Arpy Hindoyan, Gloria Bain, and Susan Dimotakis. I have many fond memories of lunches with Phil Harter.

My master's thesis advisors at Virginia Tech, Muhammad Hajj and Ali Nayfeh, instilled in me a love for engineering science. They were instrumental in my choice to attend Caltech.

Wei-Jen Su, Tony Chao, and others who entered GALCIT with me at the same time have made my first year here a wonderfully memorable one. Jerry Shan, Genti Buzi, John Choi, Frederick Balagadde, Amos Anderson, and the folks at CCF kept me going with their prayers and support. I am grateful for my fellowship at Trinity Baptist Church. I would also like to thank my parents Maureen and Kam Chuen for their strength and attention to my education, and to my brother Kamy for always watching my back. I am very grateful to my wife, Elizabeth-Sharon, for her support and her confidence in me.

Finally, I would like to thank my Lord and Savior Jesus Christ.

> This is a faithful saying, and worthy of all acceptation, that Christ Jesus came into the world to save sinners; of whom I am chief. Howbeit for this cause I obtained mercy, that in me first Jesus Christ might shew forth all longsuffering, for a pattern to them which should hereafter believe on him to life everlasting. Now unto the King eternal, immortal, invisible, the only wise God, be honour and glory for ever and ever. Amen.
>
> 1 Timothy 1:15-17

Abstract

We describe a technique for the efficient computation of the dominant-scale dynamics of a fluid system when only a high-fidelity simulation is available. Such a technique is desirable when governing equations for the dominant scales are unavailable, when model reduction is impractical, or when the original high-fidelity computation is expensive. We adopt the coarse analysis framework proposed by I. G. Kevrekidis (Comm. Math. Sci. 2003), where a computational superstructure is designed to use short-time, high-fidelity simulations to extract the dominant features for a multiscale system. We apply this technique to compute the dominant features of the compressible flow through a planar diffuser. We apply the proper orthogonal decomposition to classify the dominant and subdominant scales of diffuser flows. We derive a suitable coarse projective Adams-Bashforth time integration routine and apply it to compute averaged diffuser flows. The results include accurate tracking of the dominant-scale dynamics for a range of parameter values for the computational superstructure. These results demonstrate that coarse analysis methods are useful for solving fluid flow problems of a multiscale nature.

In order to elucidate the behavior of coarse analysis techniques, we make comparisons to averaging theory. To this end, we derive governing equations for the average motion of charged particles in a magnetic field in a number of different settings. First, we apply a novel procedure, inspired by

WKB theory and Whitham averaging, to average the variational principle. The resulting equations are equivalent to the guiding center equations for charged particle motion; this marks an instance where averaging and variational principles commute. Secondly, we apply Lagrangian averaging techniques, previously applied in fluid mechanics, to derive averaged equations. Making comparisons to the WKB/Whitham-style derivation allows for the necessary closure of the Lagrangian averaging formulation. We also discuss the Hamiltonian setting and show that averaged Hamiltonian systems may be derivable using concepts from coarse analysis. Finally, we apply a prototypical coarse analysis procedure to the system of charged particles and generate trajectories that resemble guiding center trajectories. We make connections to perturbation theory to derive guidelines for the design of coarse analysis techniques and comment on the prototypical coarse analysis application.

Contents

List of Figures

Chapter 1

Introduction

Many fluid flows in natural and scientific environments display features common to complex, multiscale systems. First of all, descriptions of the system dynamics are readily available at the fundamental or fully detailed level; for fluids, the common descriptions are the Euler equations, Navier-Stokes equations, or kinetic theories. Computational simulations are readily constructed that integrate these governing equations over space and time. Another feature common to these systems is the development of systems behavior across a range of scales; for example, in turbulent flows the fluid organizes itself into eddies and other flow structures over a range of temporal and spatial scales. In addition, coherent structures develop in multiscale systems where a separation of scales distinguishes dominant (macroscale) structures and associated dynamics. While the system dynamics are dependent on the component dynamics at every scale, the analyst is usually interested in answering questions related to the dominant scales. For fluid flows, the largest structures in the flow affect performance in engineering applications, and in nature the weather analyst is most interested in the dynamics and trajectories of the large-scale storm fronts.

While simulations of the fundamental equations are accurate in computing the dynamics of the resolved scales, these simulations often incur expensive computational costs. This may prohibit the rapid analysis of the dominant scales of a fluid system, particularly when efficiency is desired. For example, the current state of the art in weather prediction is capable of weather forecasting on the order of days into the future. This may not be adequate for emergency evacuation and safety procedures.

Many approaches exist that attempt to increase computational efficiency while focusing on extracting or computing the dominant features of fluid flows. For turbulent flows, RANS and LES simulations avoid explicit computations of the small scales by making modeling assumptions for the small-scale (”subgrid”) interactions. Model reduction procedures, such as Galerkin projection, serve to project the governing equations onto a smaller finite-dimensional subspace. Through this projection and a truncation of the small-scale dynamics, the reduction will theoretically lead to a model that accurately accounts for the dominant dynamics. Furthermore, theoretical variational reduction procedures such as LANS-α theories represent averaging methods that implicitly construct models for the small-scale interactions.

The coarse analysis approach being developed by I. G. Kevrekidis avoids both subgrid modeling and Galerkin projection. His approach assumes that a detailed simulation is available for the multiscale system of interest and that the identification of the dominant scales of interest has already been made. Rather than develop equations (and associated simulations) of the dominant scale dynamics, the coarse analysis approach seeks to update the state of the dominant scales by making efficient use of bursts of detailed simulation. The utility of this approach is found in its extraction of dominant-scale dynamics using black-box, possibly legacy, simulation and without explicit knowledge of the governing equations (even the governing equations for the dominant scales). This approach has been validated for many systems such as chemical reaction-diffusion systems and is being applied to a growing number of other systems such as mechanical and fluid systems.

Much of the setting is in place to suggest an application to fluid flows, particularly complex internal flows. As mentioned above, high-fidelity simulations are available that are capable of resolving the broad range of scales represented in a complex fluid flow, although at high computational cost. Many fluid flows exist that display dominant-scale behavior and are amenable to averaging theory and model reduction. Furthermore, the efficient computation of dominant-scale fluid dynamics is necessary in many prediction, estimation, and control applications; the resources may not be available to perform long-time, high-fidelity simulation. Finally, although observations and extraction of the dominant scales of fluid flows are often readily made, their dynamic computation sometimes eludes attempts at model reduction. As coarse analysis techniques have been applied to other systems displaying these characteristics, we are motivated to investigate their application to fluid flows, particularly compressible diffuser flows.

Properties of coarse analysis techniques have been determined by Kevrekidis and collaborators using standard numerical analysis techniques. For instance, the absolute stability of coarse time integration schemes may be derived directly from standard numerical analysis. However, there are other properties of importance, particularly to a class of techniques that purport to extract averaged information from a multiscale system. In fact, considering coarse analysis algorithms as *multiscale* systems in their own right may yield insight into their behavior and design. We are also motivated to explore these issues, primarily by regarding them from a geometric perspective. In doing so, we seek to explore connections between coarse analysis and averaging theory.

This thesis documents contributions in the coarse analysis of diffuser flows and in averaging techniques, particularly for charged particle motion. For diffuser flows, the stable, accurate computation of the dominant scales for diffuser flows is made using coarse time integration. To that end, vector-valued proper orthogonal decomposition is successfully used as a scale classification towards the coarse analysis of compressible flows. In addition, a description of a coarse projective Adams-Bashforth time integrator is given along with its associated numerical stability analysis.

Finally, an analysis is made of the average motion of charged particles in a magnetic field using both coarse analysis and averaging techniques. Lagrangian averaging theories, inspired by WKB and LANS-α techniques, are used to derive the guiding center equations. A prototypical coarse time integration of charged particle motion is made that generates trajectories resembling guiding center trajectories, and coarse analysis design guidelines are suggested by appealing to averaging theory.

This thesis is organized as follows. We give a summary of coarse analysis techniques for multiscale systems in Chapter 2. In this chapter, we describe the framework for coarse analysis, and the major tools and ingredients used for coarse analysis. We describe diffuser flows in historical and phenomenological settings in Chapter 3. We also place diffuser flow analysis in the coarse analysis framework and apply coarse numerical integration techniques to extract the dominant-scale dynamics of a diffuser flow. In Chapter 4 we explore the connections between averaging theory and coarse analysis. We focus on the system of charged particle motion in a magnetic field and apply different averaging procedures, including coarse analysis, to extract the dominant-scale behavior.

Two appendices are provided at the end of this thesis. We describe the derivative computation in the diffuser simulation in Appendix A. We describe the parallel implementation of the diffuser simulation in Appendix B.

Chapter 2

Coarse Analysis

2.1 Literature Review

Philosophy When considering a multiscale system, most analyses make two important classifications and assumptions. The first is a classification of *scale*; that is, a distinction is made to identify the dominant and subdominant scales. The second is a *modeling assumption* of the influence between the dominant and subdominant scales. Many approaches exist that aim to reduce multiscale systems for the purposes of efficient computation and analysis. These include LES modeling for fluid turbulence, Galerkin projection methods, and averaging/perturbation methods. While these approaches are often successful, most of their shortcomings lie not in their choice of scale classification, but in their choice of modeling assumptions.

A novel numerical approach, called *coarse numerical analysis*, has been proposed by I. G. Kevrekidis that addresses the sensitivity of systems analyses to modeling assumptions (Kevrekidis et al., 2003). His approach assumes that a proper scale classification has been made for a system. Rather than make modeling assumptions, however, his approach uses bursts of full-scale simulation (which in principle are generated without modeling assumptions) to extract the interdependence of the dominant (referred to as *coarse*) and subdominant (referred to as *fine*) scales. With the scale classification and the numerically computed scale influence, systems analysis can proceed for the dominant scales. The accuracy and convergence of this approach to a representative dominant-scale system is dependent on the multiscale nature of the original system. For instance, coarse numerical analysis is often very successful for systems whose scale classification is very natural; that is, the dominant and subdominant scales are well-separated.

The following subsections describe some of the key ingredients in a coarse numerical analysis and some applications of coarse analyses.

History Shroff and Keller, while developing the recursive projection method (RPM) (Shroff and Keller, 1993), considered the notion of extracting the dominant features of multiscale systems using black-box simulations. This was formally introduced in the coarse analysis framework by I. G. Kevrekidis and collaborators (Theodoropoulos, Qian, and Kevrekidis, 2000), who have continued to explore many elements and applications of coarse analysis. These include coarse time integration (Gear et al., 2002; Siettos et al., 2003b; Gear and Kevrekidis, 2003a,b; Gear et al., 2003; Gear and Kevrekidis, 2004; Rico-Martinez et al., 2004), coarse dynamical systems analysis such as fixed point determination, stability and bifurcation analysis (Theodoropoulos et al., 2000; Gear et al., 2002; Makeev et al., 2002a; Runborg et al., 2002; Makeev et al., 2002c; Siettos et al., 2003b; Hummer and Kevrekidis, 2003; Theodoropoulos et al., 2004; Siettos et al., 2004), coarse control (Siettos et al., 2003b,a, 2004), and other applications such as extracting coarse conservation laws and constructing a coarse Hamiltonian (Li et al., 2003), and the acceleration of gPROMS simulations (Siettos et al., 2003c).

The majority of these studies used kinetic Monte Carlo simulations as the detailed simulation for chemical reaction-diffusion systems; however, at least one study has been performed to study bubble flows using a Lattice Boltzmann simulation for the detailed simulation (Theodoropoulos et al., 2004). The numerical analysis conducted in Kevrekidis's collaboration with C. W. Gear (Gear and Kevrekidis, 2003a,b, 2004) is focused on coarse analysis of systems described by ordinary differential equations. I. G. Kevrekidis has written two detailed reviews of coarse analysis (Kevrekidis et al., 2003, 2004).

2.2 The Coarse Analysis Framework

The primary analysis tool is a computational superstructure that has access to a fine-scale model or simulation. The basic superstructure defined by Kevrekidis is the coarse integrator. The primary task of the coarse numerical integrator is to compute a time history of the coarse information of a multiscale system. A schematic is shown in Figure 2.1 and includes the essential components of the integrator. The following is a list of these components.

Scale classification As follows from the coarse analysis philosophy, a notion of scale classification is required. This classification provides a well-posed, well-defined heuristic from which the dominant scales of interest may be identified.

Projection operator A projection operator extracts the dominant scales from a full-scale obser-

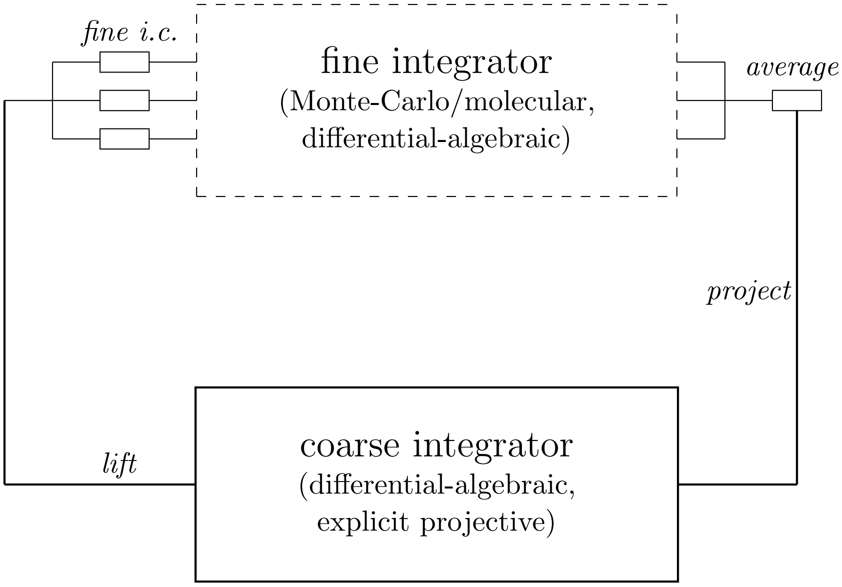


Figure 2.1: **Computational superstructure for a coarse integrator. After Kevrekidis Kevrekidis et al. (2003).**

vation of the system. This often naturally arises from the scale classification.

Average map This averages the resulting fine conditions after the fine integrator is used. The input is a set of fine conditions and the output is a single (averaged) fine condition. This is used with the projection map.

Lifting operator The lifting operator maps observations of the dominant scales to consistent observations on the full-scale system. This operator is used to initialize a detailed simulation. Observe that this map may not be unique.

Fine integrator This is assumed to be available *a priori*; this is the *workhorse* of any coarse analysis tool. This is the detailed simulation of the system that resolves the fine scales. Its operation is subject to the access of the superstructure.

Detailed simulation tools These are algorithms for extracting information using the detailed simulation. Typical quantities of interest are temporal or spatial gradients.

Dominant-scale (”coarse”) analysis For the dominant scales of interest, the dominant-scale analysis component is used for computation, for example of time histories or dynamical systems quantities.

Coarse integrator This advances the state to the next *coarse* time step. If the integrator is differential-algebraic, a coarse model or simulation must already be defined. If the integrator is of an explicit projective nature, then current and past *coarse* states/iterates are combined as an extrapolation into the future. Note that the input to the coarse integrator is a coarse initial condition. The corresponding output is a coarse final condition defined at a later time.

In computing the coarse evolution of a multiscale system, the coarse integration begins with coarse initial conditions (the initial conditions corresponding to the coarse information of the system). For a differential-algebraic superstructure, only a single set of conditions are required to initiate the coarse integrator. For an explicit projective superstructure, multiple sets of initial conditions are required, as necessary to perform the extrapolation or projection. For example, a explicit projector based on the Euler method would require two sets of initial conditions. In that case, conditions corresponding to time t_{-1} and t_{-2} would be used to extrapolate to a time t_0.

After the coarse integrator is initialized, coarse integration is performed for a *coarse* time interval. At this point, the superstructure accesses the fine integrator. The lift map is applied to the current coarse condition, which generates a corresponding set of fine initial conditions. These are input into the fine integrator and a fine evolution is computed over a short (*fine* scale) time interval.

Then the results of the fine evolution are collected, averaged using the average map, and projected forward to a corresponding coarse condition. Then the procedure repeats, beginning with another coarse integration for a coarse time interval.

The coarse integrator is just one example of a computational superstructure. Kevrekidis has used a coarse integrator in computing the evolution of system dynamics in a number of different applications (Gear and Kevrekidis, 2003a; Makeev et al., 2002b; Gear et al., 2002; Kevrekidis et al., 2003). Coarse integrators have also been used as a component of coarse stability and bifurcation analysis tools (Theodoropoulos et al., 2000; Makeev et al., 2002b; Gear et al., 2002; Runborg et al., 2002; Kevrekidis et al., 2003). Additional applications as well as analyses of the superstructure algorithms are listed in (Gear and Kevrekidis, 2003a; Kevrekidis et al., 2003; Li et al., 2003).

2.3 Coarse Timestepping

Example Consider a system where the scale classification has already been applied and where the projection and lifting operators are natural. That is, consider a system of states $z = (x, y)$, where z represents the full state, x represents the dominant scales, and y represents the subdominant scales. Here, the projection operator is represented by $x = \mathbb{P}(z)$ and the lifting operator is represented by $\tilde{z} = (x, \tilde{y}) = \mathbb{L}(x)$ with $\tilde{y}$ chosen from some distribution. Averaging is represented by the operator $\mathbb{A}(\cdot)$. Now consider a detailed simulation ϕ for the full-scale system as represented by

$$\phi_{\triangle t} : z(t) \longmapsto z(t + \triangle t) \tag{2.1}$$

with $\triangle t$ as the time interval. Using time indices, this may be rewritten as

$$\phi_{kh} : z_n \longmapsto z_{n+kh}, \tag{2.2}$$

where n refers to the current time step, h is the simulation time step and k is an integer. Suppose a timestepper Φ is desired for the dominant scales, i.e.,

$$\Phi_{\triangle t} : x(t) \longmapsto x(t + \triangle t). \tag{2.3}$$

Using time indices, this may be rewritten as

$$\Phi_{Mh} : x_n \longmapsto x_{n+Mh}, \tag{2.4}$$

where M is an integer. Such a timestepper may be constructed using tools from standard numerical analysis and from coarse numerical analysis. The detailed simulation can be used to generate vector fields using finite difference formulas; for instance, consider a forward difference formula for the first derivative:

$$\dot{z}_{n+(k+1)h} \approx \frac{1}{h}\Big(\phi_{(k+1)h}\big(z_n\big) - \phi_{kh}\big(z_n\big)\Big), \tag{2.5}$$

where the detailed simulation time is $k+1$ multiples of the detailed simulation time step h. Considering this vector field approximation as a detailed simulation tool, a corresponding coarse vector field may be computed using the projection operator:

$$\dot{x}_{n+(k+1)h} \approx \frac{1}{h}\Bigg(\mathbb{P}\Big(\phi_{(k+1)h}\big(z_n\big)\Big) - \mathbb{P}\Big(\phi_{kh}\big(z_n\big)\Big)\Bigg). \tag{2.6}$$

Now, consider a forward Euler time integration for the dominant scales. If the dominant dynamics are governed by a system $\dot{x} = f(x,t)$, the timestepper would be

$$\Phi_{Mh} : x_n \longmapsto x_n + f(x_n, t_n)Mh. \tag{2.7}$$

However, the equations of motion for the dominant scales are not explicitly given. Using the coarse vector field approximation in Eq. (2.6), the Euler timestepper may be written instead as

$$\Phi_{(M+k+1)h} : x_n \longmapsto x_n + \dot{x}_n Mh \tag{2.8}$$

or

$$\Phi_{(M+k+1)h} : x_n \longmapsto x_n + M\mathbb{P}\Bigg(\mathbb{A}\Bigg(\phi_{(k+1)h}(\mathbb{L}(x_n)) - \phi_{kh}\Big(\mathbb{L}\big(x_n\big)\Big)\Bigg)\Bigg), \tag{2.9}$$

where the timestepper is constructed for time steps of M multiples of the detailed simulation step size h. It is apparent from the coarse Euler timestepper that the detailed simulation is used primarily for extracting vector field quantities. The vector field is computed using the detailed simulation over a burst time of h; therefore, one measure of the efficiency of the coarse algorithm is the ratio $M/(k+1)$. For coarse timesteppers whose *burst interval* $(k+1)h$ is much smaller than the *coarse time interval* Mh, the efficiency is high.

2.4 Numerical Analysis

Properties of algorithms such as coarse time integrators may be identified and compared with standard algorithms through standard numerical analysis. Such comparisons are made in Siettos et al. (2003b) and Gear and Kevrekidis (2003b). One important property of time integration algorithms is the notion of *absolute stability.* Absolute stability is defined to the region on the complex $h\lambda$ plane where the numerical solution y_n decays to zero as n goes to infinity for the *linear* or *scalar* test equation,

$$\dot{y} = \lambda y \tag{2.10}$$

where h is the step size of the applied time integration algorithm.[1] For instance, the absolute stability region for the explicit Euler method is the disk of radius 1 and center $(-1, 0)$ on the complex plane. The absolute stability regions for the explicit Euler, fourth-order Runge-Kutta, and second- and third-order Adams-Bashforth schemes are shown in Figure 2.2. As shown in Figure 2.2, the fourth-order Runge-Kutta scheme is stable over a much larger region than the explicit Euler scheme, including a wide range over the imaginary axis (indicating stability for oscillatory systems). Note that one drawback of Runge-Kutta methods are their computational cost. The Adams-Bashforth schemes may have a smaller region of stability than even the explicit Euler scheme, but their advantages lie in their accuracy and stability for oscillatory systems (without incurring a heavy cost penalty than the Runge-Kutta schemes). For additional insight into time integration algorithms and numerical stability properties, the reader is referred to the texts by Gear (1971) and Shampine and Gordon (1975).

The following is a summary of the discussion found in Siettos et al. (2003b) on absolute stability of projective schemes. Coarse time integration schemes may also be evaluated by approximating their regions of absolute stability when applied to the scalar test case. In this setting, the coarse and fine representations are considered to be equivalent, and the lift and projection maps are defined

[1] Although the scalar test equation is a linear equation, this absolute stability concept is useful for determining the error management of time integration algorithms for nonlinear systems.

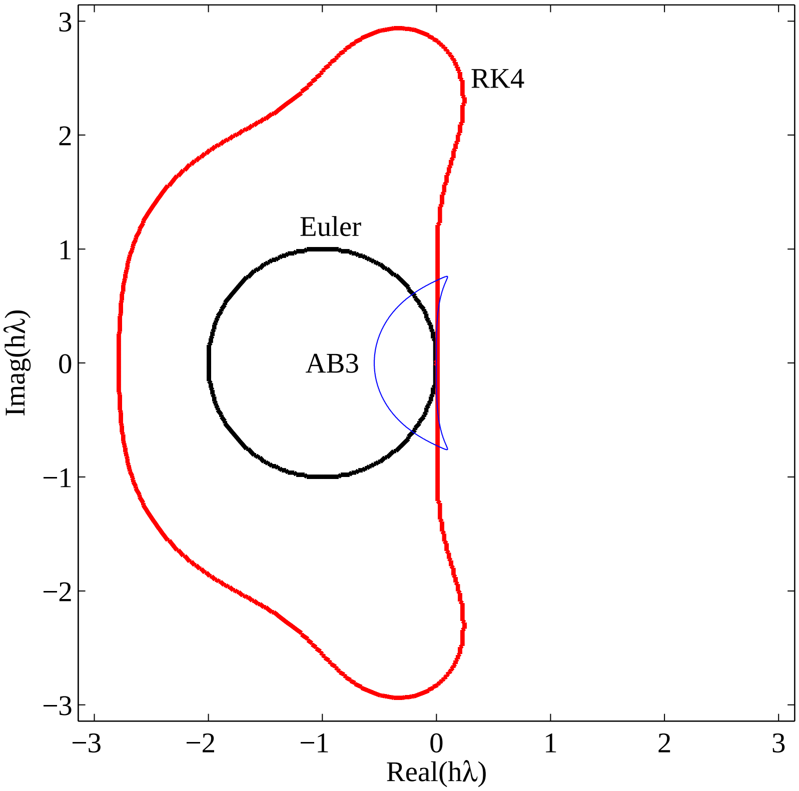


Figure 2.2: Regions of absolute stability for the Euler scheme, fourth-order Runge-Kutta, and second- and third-order Adams-Bashforth schemes.

as identity maps. Therefore, absolute stability analyses of coarse integration schemes isolate the effects of the coarse algorithm only (without the effects of the lift and projection). For instance, consider the coarse projective Euler scheme (Eq. 2.9) where the lift and projection maps are identity maps. The coarse projective Euler step from state x at time index n to the state x at time index $n+k+1+M$ is then

$$x_{n+k+1+M} = x_{n+k+1} + Mh\frac{x_{n+k+1} - x_{n+k}}{h}. \tag{2.11}$$

Observe that this coarse integration scheme may be regarded as a multistep scheme, where the next iterate $x_{n+k+1+M}$ is a function of previous states x_{n+k+1} and x_{n+k}. Assuming a geometric error amplification $\rho(h\lambda)$; that is, for the error ϵ at time index $n+k$,

$$\epsilon_{n+k} = \rho^k \epsilon_n, \tag{2.12}$$

applying the scheme in Eq. 2.11 to the scalar test equation (Eq. 2.10) yields the following error amplification:

$$\epsilon_{n+k+1+M} = \left[(M+1)\rho - M\right]\rho^k \epsilon_n. \tag{2.13}$$

Thus, the region of absolute stability is the region where $\|\sigma(h\lambda)\| < 1$; that is,

$$\|\left[(M+1)\rho - M\right]\rho^k\| < 1. \tag{2.14}$$

Regions of absolute stability are shown in Figure 2.3 for the projective Euler scheme where the detailed simulation is either itself an explicit Euler scheme or a fourth-order Runge-Kutta scheme. Note that for ratios of $M/(k+1)$ sufficiently high, the region of absolute stability for both projective

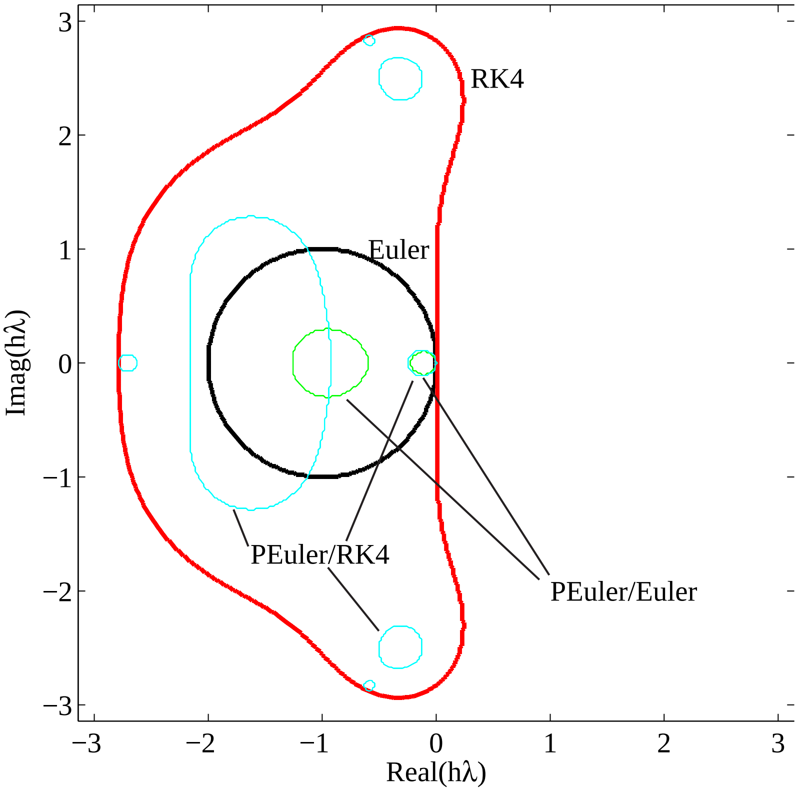


Figure 2.3: Regions of absolute stability for the projective Euler scheme where the detailed simulation is an explicit Euler scheme or a fourth-order Runge-Kutta scheme. The region of absolute stability for both the standard explicit Euler and the fourth-order Runge-Kutta schemes are included for comparison.

schemes consist of two regions that are not connected. There is a smaller region near the origin and a larger region farther away from the origin. These represent stability regions for *slow* and *fast* scales, respectively. For systems that display multiscale (temporal) behavior with decay rates falling within the regions of absolute stability, the projective schemes are stable. For instance, according to Figure 2.3, the projective Euler scheme (with an inner Euler integrator) is stable for $(h\lambda)_{\text{coarse}}$ falling within the smaller region near the origin and for $(h\lambda)_{\text{fine}}$ falling within the larger region farther out along the negative real axis. Thus, this projective scheme may be stable for a multiscale system at a *coarse time step* of Mh. Note that the detailed simulation may not be stable for such large time steps. This indicates that efficient, stable coarse integration schemes may be designed according to the dynamics of the system of interest. For additional insight into these numerical stability issues, the reader is referred to Siettos et al. (2003b). Later in Section 3.8, we return to the discussion of absolute stability with a particular emphasis on a coarse projective Adams-Bashforth scheme.

2.5 Limitations and Challenges

Coarse analysis techniques, particularly time integration routines, take advantage of the multiscale nature of a system. These techniques are well suited for systems displaying a wide separation of scales. For systems that do not display such a scale separation, numerical errors may arise during the coarse analysis. Similarly, for systems that do not admit readily to scale classification, they may not easily be placed in the coarse analysis framework. In other words, one must be able to define the right coarse variables for a given system, prior to applying coarse analysis.

Chapter 3

Coarse Analysis of Diffuser Flows

In this chapter we apply the coarse analysis methodology to extract the dominant features, and associated dynamics, of the compressible flow in a planar diffuser. We begin with an existing simulation (of the compressible Navier-Stokes equations over a diffuser geometry) and an identification of the large-scale structures. We wish to demonstrate that coarse analysis is a promising tool for the efficient extraction of the dominant features of a system, without making modeling assumptions or performing explicit model reduction. An appropriate demonstration would be an accurate reconstruction of the dominant-scale dynamics of a diffuser flow that is computationally less expensive than extracting the dominant dynamics from a full, long-time simulation. Such a demonstration would indicate that is possible to use existing detailed simulation efficiently for dominant-feature extraction, and this would also suggest possibilities for exploring the dominant-scale dynamics for diffuser flows under different environments.

This chapter is organized as follows. We give a description of the existing literature and known phenomenology for diffuser flows in Sections 3.1 and 3.2. We list the appropriate equations of motion for compressible flow and describe the numerical technique for solving those equations in Sections 3.3 and 3.4. In propose a coarse analysis framework for fluid simulation, we consider two choices for scale classification. We consider using grid refinement as scale classification in Section 3.5. In this section, we perform coarse time integration using grid interpolation techniques. We then consider a coarse analysis framework for fluids in Section 3.6 using the proper orthogonal decomposition for scale classification. This section includes a description of each of the primary components of coarse analysis as applied to this fluid flow system. We present our coarse diffuser flow analysis in Section 3.9, which includes a discussion of POD-based scale classification results for the diffuser flow and results for coarse time integration of diffuser flow. Finally, we make concluding remarks in Section 3.10.

3.1 Literature Review

Diffuser flows are found in many engineered devices, including heart-lung assistive devices, hydraulic devices, and aerospace propulsion systems. Adverse pressure gradients are intrinsic to diffuser flows, and may lead to flow unsteadiness and *flow separation.* Across all of these applications, diffuser performance is often inversely related to the level of flow distortion or blockage naturally generated from instabilities induced by the pressure gradients. For instance, the presence of flow separation may induce dead zones and blockages in heart-assist devices, increasing the likelihood of clot development and other hazardous complications (Tsukiya et al., 2003b,a). Flow distortion in propulsion systems may lead either to fatigue and failure in device components or to dangerous flow-induced instabilities such as compressor surge and stall (Hamstra et al., 2000; MacMartin et al., 2001). Interest in understanding, reducing, and controlling such phenomena in these applications has placed these flow systems under study for over fifty years. As the environment in a diffuser allows for flows with broad ranges of spatial and temporal scales, this multiscale system continues to interest engineers, scientists, and mathematicians. The majority of the research effort appears to lie in experimental characterization and in control applications; other portions include reduced-order modeling, turbulence modeling and simulation, and theory.

Flow control Diffuser flow researchers have been interested in flow control applications since the 1950s. The earliest work included experimental investigations of distributed (steady) suction as a means of boundary-layer and separation control in conical diffusers (Holzhauser and Hall, 1956; Moon, 1958). Since then, the control applications have grown to include a wide variety of actuation mechanisms and control philosophies. Passive and structural actuation was implemented in the form of vane splitters and moving walls in the 1970s (Rao, 1971; Tennant, 1973). In the latter study, moving cylinders formed part of the boundary in the region local to the separation point.

Steady blowing and suction have been thoroughly investigated. Steady inlet injection in conical diffusers was implemented by Nicoll and Ramaprian (1970). In that study, optimum actuation velocities were determined to be on the same order as the inlet flow velocity. Steady injection applied from the diffuser centerline was implemented in conical diffusers by Nishi et al. (1976). Wall injection upstream of the separation point in a rectangular diffuser was implemented by Back and Cuffel (1982). Injection in serpentine inlets, of the type found in current designs of advanced propulsion systems, was implemented by Hamstra et al. (2000). In this study, it was generally found that steady injection tended to stabilize the flow and, in many cases, reattach the separated (boundary-layer) flow.

Unsteady flow actuation has been applied in a number of settings and represents the bulk of the diffuser flow control literature. Unsteady flow control represents a boon to design engineers; unsteady actuation is ideally superior to steady actuation in terms of greater efficiency. Unsteady wall-normal inlet blowing was applied in rectangular diffusers by Smith and Kline (1974). A methodology for applying unsteady separation point and downstream injection was proposed by MacMartin et al. (2001) to control flow instabilities in a system of a serpentine diffuser coupled with a compressor. Separation point injection was applied in asymmetric rectangular diffuser flows in Brunn and Nitsche (2003) and Feakins et al. (2003). A computational study of separation point injection was made by Suzuki et al. (2004). For completeness, the work in oscillatory blowing for flow reattachment over airfoils by Seifert et al. (1996); Seifert and Pack (1999) were very influential to many of the flow control activities done after the 1990s.

Some of the unsteady flow actuation mentioned previously used audio speakers as actuators. The use of speakers to introduce acoustic waves into the diffuser has also been implemented at the diffuser exit by Salmon et al. (1983) for modulating shock-induced separation.

Rather than actuation through straightforward mass injection, more sophisticated means of actuation have been developed. Synthetic jet actuation has been developed to deliver momentum (and vorticity) with a zero-net mass flux. The work mentioned previously of MacMartin et al. (2001) also involved synthetic jet actuators. Synthetic jet actuation was also implemented in serpentine diffusers by Amitay et al. (2002) and in planar diffusers by Ben Chiekh et al. (2003). For completeness, the reader is also referred to a general review of the synthetic jet literature in (Glezer and Amitay, 2002).

In addition, vortex generator jets (VGJs) are actuators designed to inject mass at an angle away from wall-normal; the nature of the (horseshoe) vortex released from actuation is different than that of wall-normal injection. Vortex generator jets were applied well upstream of flow separation in a conical diffuser by Nishi et al. (1998). In that study, the ratio of jet speed to freestream velocity was found to be more important than the ratio of jet flow rate to total flow rate. Computational modeling of planar diffuser flows controlled using VGJs was performed by Coller et al. (2000); Coller (2000). VGJs were also implemented in the serpentine inlet studies by Hamstra et al. (2000) and Sullerey and Pradeep (2004). It is also important to note that the unsteady separation point injection studied by Suzuki et al. (2004) models the flow generated by vortex generator jets.

Theory and Modeling Many efforts have been undertaken to understand diffuser flows, associated flow regimes of low performance, and the application of control. Measurements of diffuser performance date back at least to the works by Gibson (1911–1912), Peters (1934), and Holzhauser and Hall (1956). Water experiments in planar diffusers performed by Fox and Kline (1962)[1] and in conical diffusers by McDonald and Fox (1966)[2] represent some of the earliest studies into diffuser flow characterization. In these two studies, specific flow regimes were listed and observed, including steady laminar flow, transitory stall, fully developed stall, and fixed stall. Experiments were performed in air through rectangular diffusers by Reneau et al. (1967); in these experiments, they also observed four different flow regimes: steady flow, unsteady flow, unsteady flow with

[1] This research was performed as part of Fox's doctoral studies.

[2] Fox and MacDonald have since written a commonly used fluid mechanics textbook with P. J. Pritchard. The reader is referred to Fox et al. (2004).

steady separation, jet flow. In that study, Reneau and collaborators also measured and discussed diffuser performance. Experiments were made in conical and rectangular diffusers to characterize noise and duct acoustics (Kwong and Dowling, 1994). They defined lag laws for unsteady stall motion and characterized the dependence of noise and acoustic behavior on the diffuser geometry and boundary. Finally, Suzuki and collaborators developed simulations and theory for quantifying diffuser performance in terms of the vortex structures found in unsteady diffuser flows (Suzuki and Colonius, 2003; Suzuki et al., 2004). They also developed identification and control strategies to address vortex structure location and modulation.

Simulation and Model Reduction Computer simulations developed since the 1990s have been capable of capturing many of the features of complex internal flows such as diffuser flows. Furthermore, model reduction has been employed on several occasions to aid in simulation, prediction, and control. Commercial computational fluid dynamics (CFD) codes have been used in incompressible diffuser simulations (Iaccarino, 2001). T. Suzuki and collaborators have developed high-fidelity simulations for compressible diffuser flows (Suzuki and Colonius, 2003; Suzuki et al., 2004). Large eddy and turbulence simulations using conventional and advanced subgrid physics models have been applied to computing incompressible flows in asymmetric planar diffusers (Kaltenbach et al., 1999; Apsley and Leschziner, 1999; Iaccarino, 2001; Lim and Choi, 2004; Schlüter et al., 2005). In most of these studies, the choice of subgrid stress models was found to affect the mean momentum and turbulent kinetic energy balances. It is worth mentioning that the study by Lim and Choi was motivated by optimal shape design of the diffuser geometry. In that study, fluid simulations were used in the setting of a larger optimization program. B. Coller and collaborators have developed vortex simulations of diffuser flows as a form of model reduction (Coller et al., 2000; Coller, 2000). Finally, the proper orthogonal decomposition has been applied to construct empirical basis functions from compressible diffuser flow simulation data (Narayanan et al., 1999; Khibnik et al., 2000). The results from these two groups include a proposed bifurcation map for diffuser flows as a function of flow Reynolds number.

3.2 Diffuser Flow Phenomena

Diffusers are designed to slow down a flow and increase its static pressure. This is done by converting kinetic energy (dynamic pressure) to potential energy (static pressure) generally through passive means, such as taking the (subsonic) flow through a channel with diverging boundaries.[3] The adverse pressure gradients induced on the flow by the diffuser geometry and flow conditions allow for flow complications such as boundary layer separation, blockage, stall, flow reversal, and vortex dynamics.

Flow regimes The studies by Fox and Kline (1962) and Reneau et al. (1967) include definitions of flow regimes observable in planar diffusers. They defined four flow regimes of interest, which are drawn in Figure 3.1 for clarity:

“Well-behaved” or “no appreciable stall” This regime is observable at low flow Reynolds numbers and is described by laminar flow throughout the diffuser, except for boundary layers that are possibly thick. Except for minor incidents, no areas of stall, flow reversal, or vortex dynamics are present.

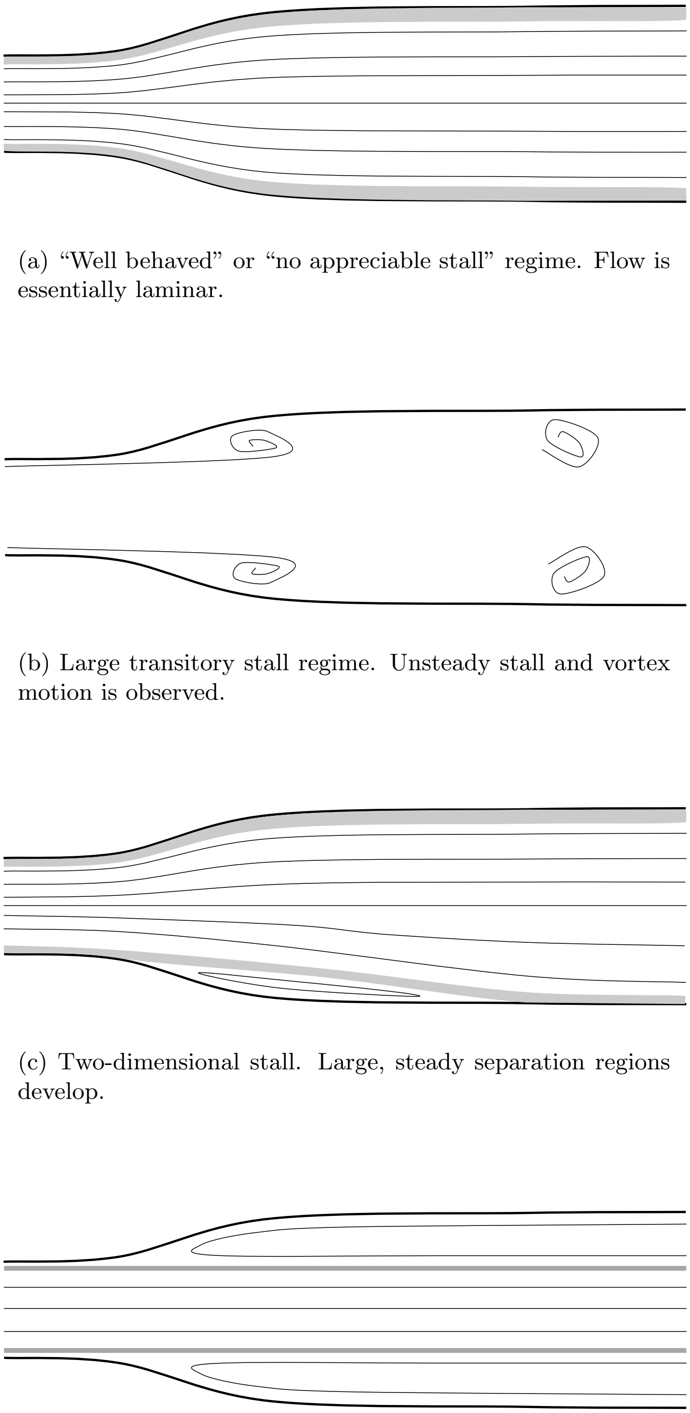
[3] The reverse is true for supersonic flow diffuser design.

Large transitory stall This regime is characterized by unsteady stall development and organized vortex dynamics. The flow may generally be regarded as oscillatory (with respect to flow quantities such as velocity, vorticity, and performance).

Two-dimensional stall This regime is characterized by well-defined stall or separation regions near the diffuser walls. The core flow and the flow within the separated regions may be well organized or laminar, with the mixing regions disorganized or turbulent. Separated regions may be present on both walls or just one of the walls. In the latter case, it is possible to affect the flow and cause the separation region to shrink and a new separation region to develop on the other wall (Reneau et al., 1967).

Jet Flow In this regime, flow separation occurs on both walls without reattachment; the core flow convects through the diffuser with straight mixing regions between the core flow and the separated flow. The flow on either side of the mixing regions are relatively steady.

Bifurcation Scenario A scenario was proposed in Khibnik et al. (2000) for the bifurcation behavior of planar diffusers with increasing diffuser angle. This bifurcation scenario is reproduced in Figure 3.2. As diffuser angle is increased, the diffuser action and the pressure gradients become increasingly severe, leading to flow regime changes roughly in the order of the list given above. The noappreciable stall regime is found to the left in the diagram, and the jet flow regime is found to the right in the diagram. The first bifurcation, approximately at $2\theta = 2.5^o$ in the figure, is a symmetry-breaking bifurcation that would be expected of the transition from the symmetric no-appreciable stall regime to an asymmetric unsteady stall (separation) regime. The reader is referred to the discussion in Khibnik et al. (2000) for a complete discussion on this proposed bifurcation



(a) "Well behaved" or "no appreciable stall" regime. Flow is essentially laminar.

(b) Large transitory stall regime. Unsteady stall and vortex motion is observed.

(c) Two-dimensional stall. Large, steady separation regions develop.

(d) Jet flow. Mixing regions become parallel to diffuser centerline.

Figure 3.1: Representative flow regimes in planar diffusers.

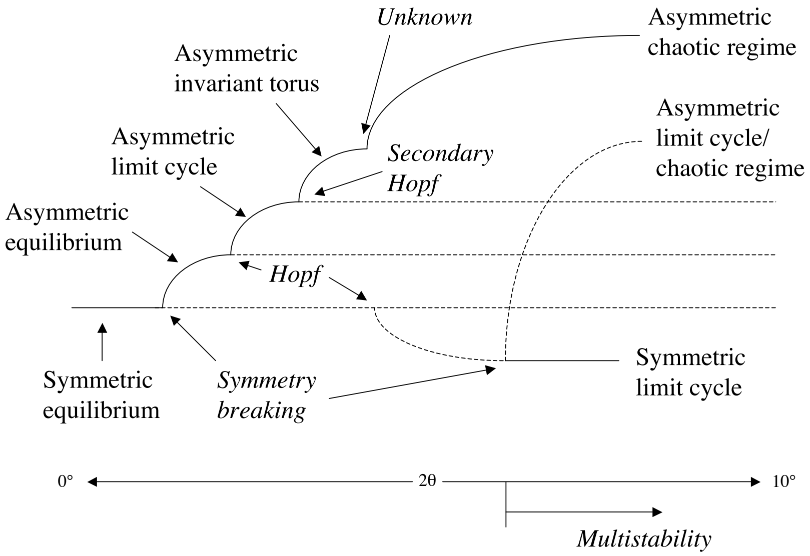


Figure 3.2: **Bifurcation scenario as proposed in Khibnik et al. (2000). The flow regimes addressed are, roughly, the no appreciable stall regime to the left, and the jet flow regime to the right.**

scenario. It is important, however, to note the complicated nature of the proposed bifurcation scenario. Not only is this indicative of a rich dynamical system, but this also suggests that any modeling of diffuser flows must be sufficiently robust and complex to capture such a bifurcation scenario.

Diffuser performance metrics Flow phenomena such as boundary layer separation, flow reversal, vortex dynamics, and turbulence may in turn affect the performance of the diffuser. The performance of a diffuser is related to the effectiveness by which the diffuser is capable of converting dynamic pressure into static pressure. To this end, performance measurements usually quantify the losses associated with this conversion. There are many different metrics available for measuring performance; these include *pressure recovery*, *head loss*, *effectiveness*, and *stagnation pressure ratio.* The first three metrics have been used throughout the history of diffuser flow studies, including the work by Fox and Kline (1962) and Reneau et al. (1967). The latter study details diffuser performance measurements as a function of diffuser geometry. The fourth metric, while not explicitly encountered in the literature, is related to the *diffuser loss factor* defined in Holzhauser and Hall (1956) which involved the ratio of average total-pressure loss to the theoretical incompressible, irrotational value.

Pressure recovery is usually defined as a normalized static pressure difference across the diffuser. That is,

$$C_{\mathrm{pr}} = \frac{\bar{p}_2 - \bar{p}_1}{\bar{q}_1}, \tag{3.1}$$

where p is the pressure, $q = \frac{1}{2}\rho U^2$ the dynamic pressure, ρ the density, and U the velocity across downstream and upstream stations 1 and 2, respectively. Overbars indicate cross-sectional averages. For irrotational flows and assuming incompressibility, Bernoulli's equation may be applied, leading to an expression for the *ideal pressure recovery* for two-dimensional (planar) diffusers:

$$C_{\mathrm{pr}_i} = 1 - \frac{1}{(\mathrm{AR})^2}, \tag{3.2}$$

where the aspect ratio AR for planar diffusers is reduced to the ratio of exit to inlet diffuser width.

Two additional measures of diffuser performance are derived from the pressure recovery; these are the diffuser *head loss* and *efficiency.* Head loss is defined as the difference between the ideal and actual pressure recovery,

$$H_L = C_{\mathrm{pr}_i} - C_{\mathrm{pr}}, \tag{3.3}$$

and diffuser effectiveness is defined as the ratio of actual to ideal pressure recovery,

$$\eta = \frac{C_{\mathrm{pr}}}{C_{\mathrm{pr}_i}}. \tag{3.4}$$

Observe that when losses are incurred in the flow, the stagnation pressure may drop in the diffuser. When stagnation pressure drops, diffuser pressure recovery and efficiency decreases, and head losses increase. Therefore, an additional diffuser performance metric is the stagnation pressure ratio across the diffuser,

$$\eta_{p_t} = \frac{\bar{p}_{t_2}}{\bar{p}_{t_1}}, \tag{3.5}$$

where p_t is the stagnation pressure. A similar performance metric, the diffuser loss factor K, is defined in Holzhauser and Hall (1956) which is the ratio of average total pressure losses to the theoretical incompressible value:

$$K = \frac{(\bar{p}_{t_1} - \bar{p}_{t_2})/(\bar{q}_1)}{(1 - A_1/A_2)^2}, \tag{3.6}$$

where A is the cross-sectional area. Note that the efficiency η, stagnation pressure ratio η_{p_t}, and the quantity $1 - K$, maxima occur at a value of one. For this reason, these two metrics are of particular value in diffuser performance measurement.

The reader is referred to the works by Fox and Kline (1962) and Reneau et al. (1967) to gain additional insight into diffuser flow characterization and performance.

Multiscale descriptions For the large transitory stall regime, in the case of planar diffusers, flow unsteadiness may be localized to occurring over just one side of the diffuser. A description of the range of flow timescales is given in Reneau et al. (1967). In terms of diffuser characteristic times $\tau = L/U_1$, with L the diffuser length, flow unsteadiness is observed at times on the order of 10τ for the fastest variations and on the order of 100τ for the large-scale variations. Furthermore, the spatial scales of the flow structures range between the smallest spatial scales, representing the shear induced by the large vortex dynamics, and the largest scales, which is on the order of the diffuser length scales, represent the large vortices themselves. In this sense, diffuser flows in the large transitory stall regime may be described as multiscale flow systems.

3.3 Equations of Motion

The equations of motion are the fully incompressible Navier-Stokes equations in two dimensions. These equations describe the conservation of mass, momentum, and energy of a compressible viscous fluid flow on a plane. In conservative form,

$$\frac{\partial u_i}{\partial t} + \frac{\partial}{\partial x_j}(\rho u_j) = 0 \tag{3.7}$$

$$\frac{\partial \rho u_i}{\partial t} + \frac{\partial}{\partial x_j}(\rho u_i u_j + p\delta_{ij}) = \frac{\partial}{\partial x_j}\tau_{ij} \tag{3.8}$$

$$\frac{\partial e}{\partial t} + \frac{\partial}{\partial x_j}\left((e+p)u_j\right) = \frac{\partial}{\partial x_j}u_i\tau_{ij} + \frac{1}{\text{Re Pr}}\frac{\partial^2 T}{\partial x_j \partial x_j}, \tag{3.9}$$

where u and v are streamwise and wall-normal velocity components, respectively, ρ the density, p the pressure, e the total energy, τ_{ij} the viscous stress tensor, and T the temperature, with the identities

$$e = \rho(E + |u|^2/2) \tag{3.10}$$

$$\tau_{ij} = \frac{1}{\mathrm{Re}}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right), \tag{3.11}$$

where E is the total energy density. In addition, Re is the Reynolds number and Pr is the Prandtl number. The Reynolds number is defined here based on the inlet velocity U_{in} and inlet height d, defined by

$$\mathrm{Re}_d = \frac{\rho_{\mathrm{in}} U_{\mathrm{in}} d}{\mu} \tag{3.12}$$

with dynamic viscosity μ. The Prandtl number is ratio of kinematic viscosity to thermal diffusivity,

$$\mathrm{Pr} = \frac{\nu}{\kappa}. \tag{3.13}$$

The fluid is also assumed to be a perfect fluid, with equation of state

$$p = \frac{\gamma - 1}{\gamma}\rho T, \tag{3.14}$$

where γ is the ratio of specific heats.

3.4 Numerical Simulation

The numerical method used for solving the governing equations was developed by Colonius and Pirozzoli (2002) and implemented by Suzuki, Colonius, and Pirozzoli for investigating vortex shedding and separation modeling and control in diffuser geometries (Suzuki et al., 2004). We have further developed the numerical simulation for execution on parallel architectures. We document the main features of the method below.

The governing equations are first recast in orthogonal curvilinear coordinates according to Vinokur (1974). The original numerical scheme of the diffuser code solves these equations using a compact sixth-order Padé finite difference scheme for spatial differencing and a fourth-order Runge-Kutta scheme for time advancement (Lele, 1992). Fourth- and fifth-order differencing is used along the inflow, exit, and wall boundaries.

The grid corresponding to the physical diffuser geometry is generated through a conformal mapping from a Cartesian grid. The conformal mapping is defined as

$$z = \zeta + \frac{\beta}{2} \log \cosh \alpha z, \qquad \zeta = \xi + i\eta, \qquad z = x + iy,$$

with α and β as constants, from a rectangular domain with coordinates $\zeta = (\xi, \eta)$ to the physical domain with coordinates $z = (x, y)$. While a uniform rectangular grid is used for the spatial derivative computation, an intermediate nonuniform Cartesian grid so that the user can cluster points (such as toward walls and separation regions). For details on the grid generation, the reader is referred to Suzuki et al. (2004).

The implementation follows the approach of Poinsot and Lele (1992) and Visbal and Gaitonde (1999) for inflow, outflow, and wall-boundary conditions. Wall-boundary conditions are no-slip and isothermal, corresponding to the stagnation temperature of the inflow. Inflow and outflow boundary conditions are nonreflecting, and buffer zones are implemented according to Freund (1997). In the buffer zone approach, the flow $\mathbf{q}$ is relaxed to a predefined flow $\mathbf{q}_{\text{target}}$ (here, the initial condition) through a forcing term

$$\sigma(\mathbf{q} - \mathbf{q}_{\text{target}}) \tag{3.15}$$

where the buffer coefficient σ is chosen here to take on a hyperbolic tangent profile.

For the purposes of this work, the diffuser code was parallelized with a domain decomposition method using MPI (see Appendix B). To this end, the implicit Padé spatial differencing schemes were replaced by explicit differencing. Here, the wavenumber–optimal differencing schemes proposed by Lele (1992) and Tam and Webb (1993) are implemented with interior spatial derivatives computed to fourth-order and fourth/fifth-order differencing along the boundaries. The resulting differencing schemes were validated against the original implementation (see Appendix A).

The initial condition is defined as the potential flow solution, with thermodynamic quantities defined using homoentropic assumptions. Initial boundary-layer profiles are defined assuming an incompressible Blasius profile. The reader is referred to Suzuki et al. (2004) for additional discussion on the numerical procedure.

Unless otherwise stated, the detailed fluid system dynamics reported here represent computations for flows at an inlet Reynolds number of 2,550 and a Mach number of 0.55 through a diffuser of an exit-to-inlet ratio of 1.4. The orthonormal grid size is 321 x 81. For clarity, a reduced grid (taken every five points in each direction on the original grid) is shown in Figure 3.3. Flows under these conditions exhibit separation and vortex dynamics in the diffuser. The vorticity of a typical flow is shown in Figure 3.3.

As shown in Figure 3.3, the multiscale nature of the fluid system arises from the vortex dynamics of the flow. The largest spatial scales of the system are represented by the large vortical structures (downstream in Figure 3.3) whose length scale is on the order of the inlet height (a length scale for the diffuser). The smallest spatial scales of the system are represented by the shearing observed at the shear layers near the separation point and around the large vortical structures.

The dominant dynamics for the system are related to the vortex dynamics of the large structures found in the flow. As shown in Figure 3.4, the diffuser flow exhibits limit cycling behavior, as measured by stagnation pressure ratio. Computing the vortex trajectories, or other system quantities related to the vortex motion, is equivalent to computing the dominant dynamics of the flow.

Using the DNS simulation for computation, although highly accurate for the resolved scales, is inefficient for extracting the dominant system dynamics. In considering the dimensionality of the system, the simulation computes the flow (two momentum quantities, density, and temperature) over the entire computational grid (over 24,000 grid points). The system dimension is then on the order of 10^5.

2-D diffuser flows It is important to note that two-dimensional diffuser flows may be different than three-dimensional diffuser flows. Because of the two-dimensional constraint, vortex stretching does not occur. Also, the boundary layer separation dynamics may be different for two-dimensional flows. Furthermore, the transition to turbulence may be delayed for two-dimensional flows. De-

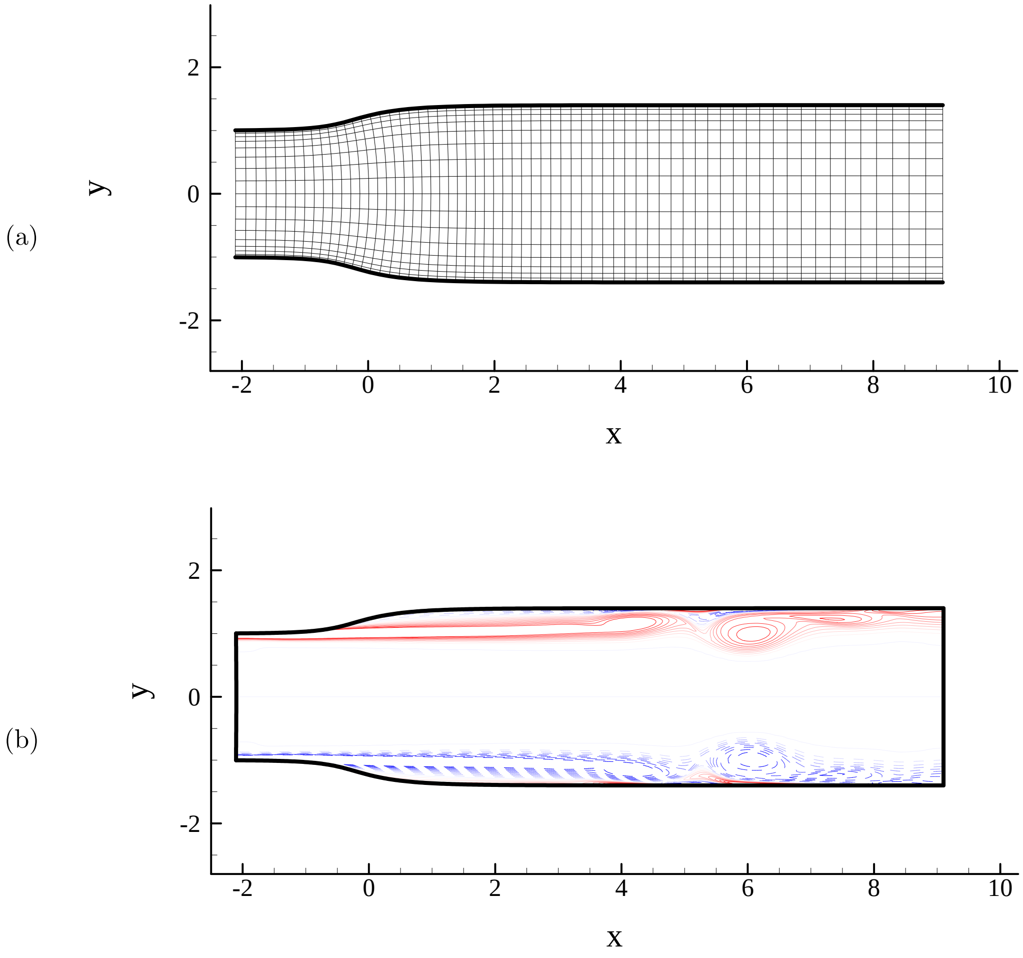


Figure 3.3: Diffuser flow (a) grid, plotted every five points in each direction for clarity; (b) representative snapshot of vorticity contours.

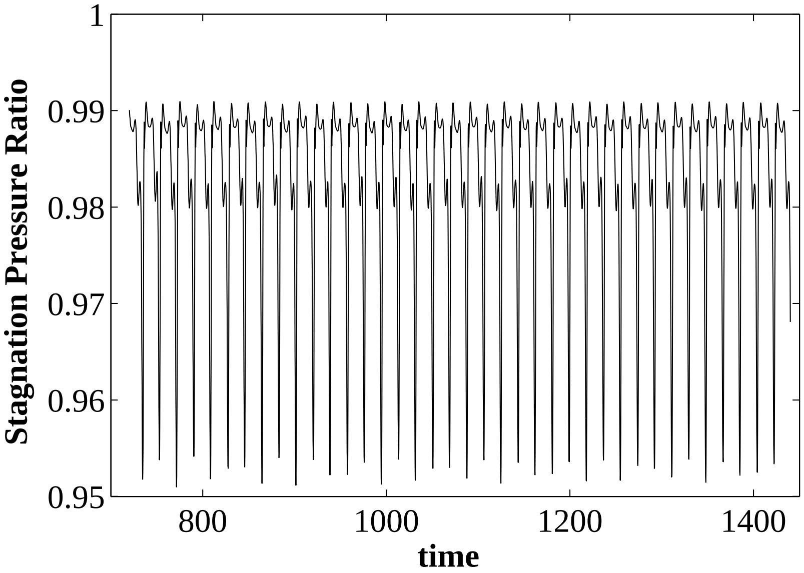


Figure 3.4: Diffuser flow stagnation pressure ratio.

spite these differences, the coarse numerical techniques developed here treat the flow as a general dynamical system. In other words, the success (or failure) of coarse analysis for fluids depends on the ability of the scale classification to represent (in other words, make observations) of a fluid flow and on the accuracy of the detailed simulation. As long as a detailed simulation is faithful in resolving a two-dimensional fluid flow, a corresponding coarse analysis should be successful for a given multiscale fluid flow that admits to a coarse representation. Alternatively, an accurate three-dimensional simulation would be required for the coarse analysis of a fully three-dimensional fluid flow.

3.5 Fluids: Coarse Analysis Using Grid Representation for Scale Classification

In this section, the components of coarse analysis are designed using a scale classification based on grid refinement. First, the grid refinement method, based on interpolation, is described. Then the components of coarse analysis, including the lift and projection algorithms, are described in this context for time integration. Finally, results from a coarse time integration using grid-refinement-based scale classification are discussed.

Grid refinement has been used extensively in the numerical computation of fluid mechanics; in particular, the adaptive mesh refinement (AMR) (Berger and Oliger, 1984) and multigrid (Brandt, 1977) techniques have been developed to reduce computational cost in fluid simulations. Adaptive mesh refinement methods seek to make spatial and temporal grid refinements to more accurately resolve small-scale or fast dynamics in the domain. Multigrid methods involve a combination of interpolation-based lifting (*prolongation*) and projection (*restriction*) with iterative methods (used to solve problems of the form $Au = f$ for matrices A and vectors u and f) to accelerate a solution for u. Although these methods are effectively used for the computation of fluids, they would either 1) require significant modifications to non-AMR and non-multigrid codes or 2) require explicit access to their subroutines for the dynamics to compute quantities such as residuals. Therefore, traditional AMR and multigridTherefore, although the traditional AMR and multigrid approaches are not implemented here, the notion of *grid refinement* is now considered in the context of coarse analysis.

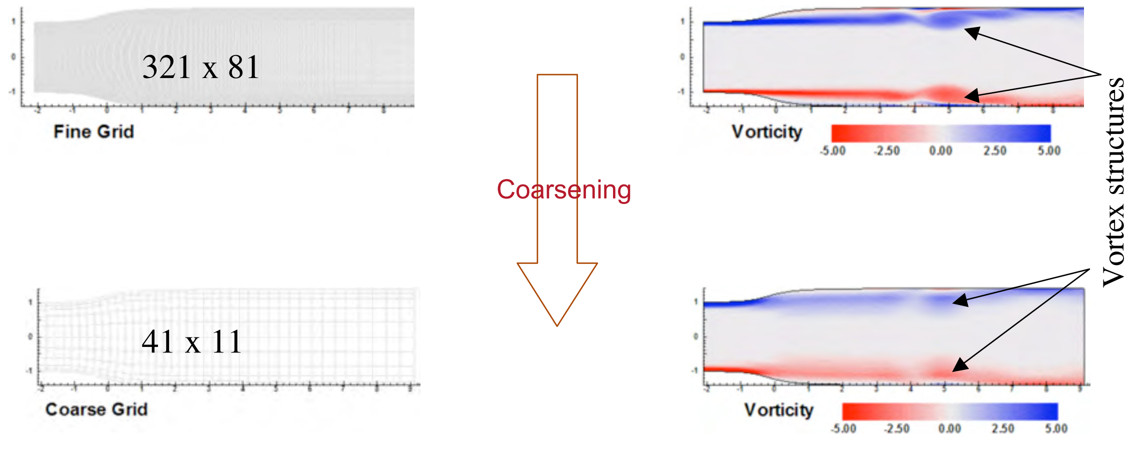


Figure 3.5: Grid refinement for scale classification.

As used in the AMR and multigrid techniques, scale classification by grid refinement is simply by grid representation; in other words, coarse and fine representations take the form of coarsely and finely resolved grids. Using the conformal mapping algorithm for the grid generation in the diffuser flow simulation, cubic spline interpolation (of each of the four flow variables) is used to coarsen a flow.[4] This is shown in Figure 3.5.

As grid refinement is performed using interpolation, lifting and projection maps follow from the interpolation. Thus, coarse time integration is straightforward using this type of scale classification; in fact, the original DNS simulation may be used on the coarse grid to produce a coarse time integrator. In other words, coarse projective integration is not necessary. A cycle of coarse time integration thus proceeds as follows: beginning with a coarse grid representation, interpolation is used to lift to a fine grid representation. A short burst of DNS simulation is performed on the fine grid using the lifted fine representation as an initial condition. This is followed by projection onto the coarse grid (using interpolation). Then a DNS simulation is computed over a longer time

[4] Linear interpolation is commonly used in mesh refinement techniques, but cubic spline interpolation is used here to ensure a smoother interpolation, which would reduce noise in the lifting algorithm.

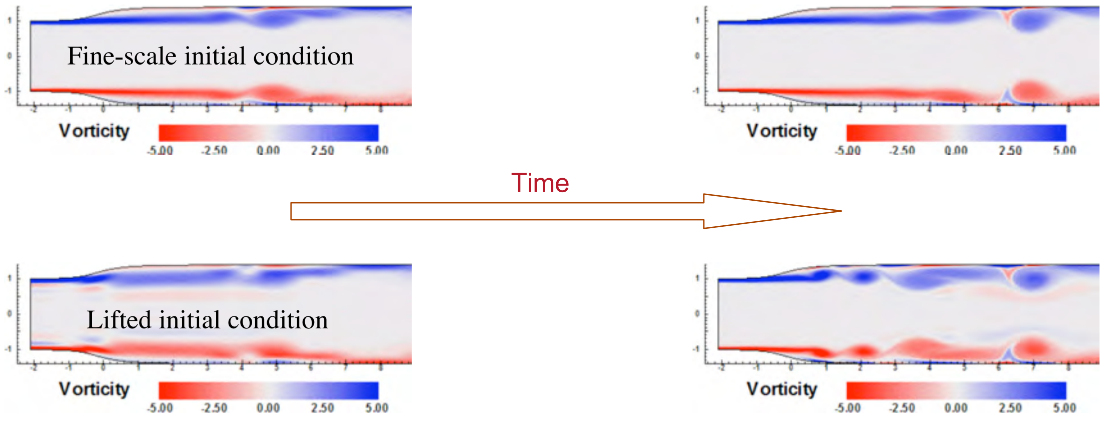


Figure 3.6: Results of coarse time integration using grid refinement.

interval on the coarse grid, using the coarsened representation as an initial condition.

A cycle of coarse time integration is shown in Figure 3.6. Beginning with a fine-scale initial condition, interpolation is used to generate a corresponding coarse initial condition, from which the cycle of coarse time integration begins. The interpolation is performed on the four flow variables: streamwise and spanwise momentum, density, and energy; the noise induced by the interpolation appears as artifacts in the vorticity of the fine-grid representation. This may be seen in the figure as light bands near the shear layers downstream from the boundary layer separation. The figure also contains a vorticity snapshot taken after a burst of fine-scale DNS simulation using the lifted flow as an initial condition. As shown in the lower left in the figure, the shear layers undergo Kelvin-Helmholtz shear instabilities due to the noise induced by the lifting algorithm. Although new vortex structures arise as a result of the induced fluid instabilities, the vortex pair observed in the original flow survives the lifting/fine-scale DNS burst.

A proper comparison would actually be between the cycle of coarse time integration and a projection of the original flow, computed using DNS simulation on the fine grid, onto a coarse grid. This is shown in Figure 3.7. As shown in this figure, the projection (interpolation) smears out the new vortex structures induced by the fluid instabilities. Although the coarsened flow still contains artifacts that are undesirable, the original vortex pair of interest is resolved accurately by the cycle of coarse time integration.

A straightforward implementation of coarse time integration using grid refinement for scale classification yields coarsened flows that do not compare well with a projection of the original DNS flow. While the results are unsatisfactory, it may be possible to increase the accuracy and reduce the errors arising from the grid interpolation. Possible techniques for noise (variance) reduction would involve lifting to multiple copies of fine-scale representations for multiple DNS bursts (and subsequent ensemble averaging). Each lifted representation would be generated from a distribution applied over the usual grid interpolation; then the induced flow instabilities might be reduced by ensemble averaging. In addition, the interpolation technique might be revised to account for desired smoothness of the derivatives of the flow variables (from which the fluid vorticity is derived). These revisions might be made using techniques from AMR and multigrid methods, and are suggested for future work. In the next section, we consider the coarse analysis of fluids with a scale analysis using the proper orthogonal decomposition.

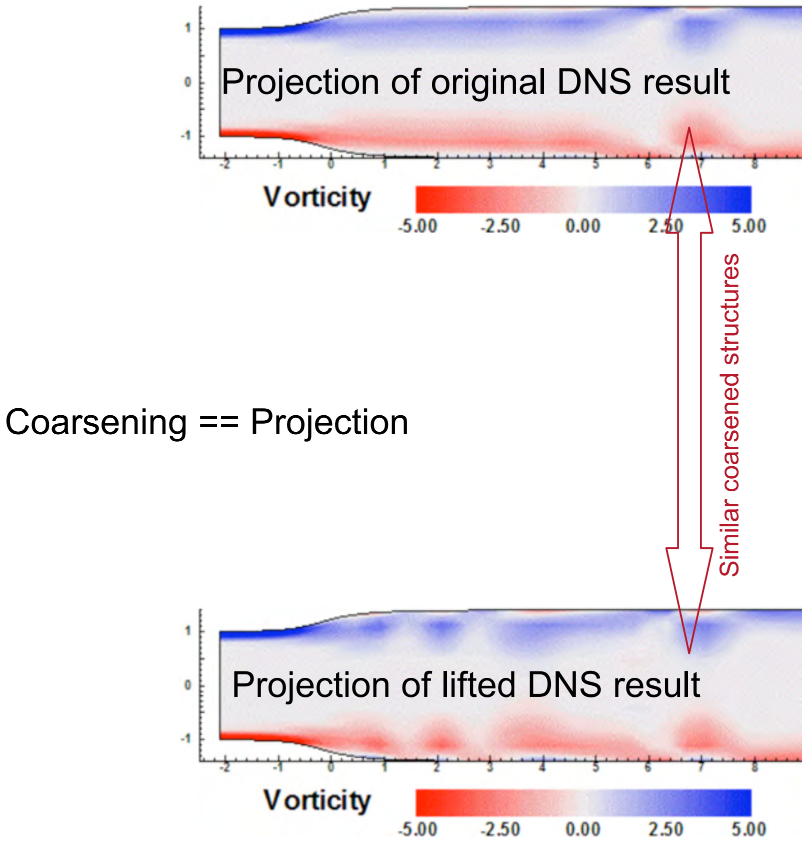


Figure 3.7: Comparison of vorticity computed by DNS and by grid-refinement-based coarse time integration.

3.6 Fluids: Coarse Analysis Using the POD for Scale Classification

In this section, the components of coarse analysis will be developed for fluid systems using a different type of scale classification. Here, we propose the proper orthogonal decomposition as a well-defined scale classification for compressible fluid flows. We then pose the necessary conditions for the projection and lift operators and tools constructed from bursts of detailed simulation. Finally, we describe the coarse timestepper.

Scale classification using the proper orthogonal decomposition Although the observation of the large-scale vortex dynamics as the dominant dynamics is valid, a more rigorous scale classification is necessary for the coarse numerical analysis. For fluid systems, natural choices for scale classification follow from a spectral decomposition of the flow. For instance, in turbulent flows a traditional scale classification is made from kinetic energy spectra. A natural scale classification that allows for both flow description and for reduced order modeling is the proper orthogonal decomposition (Holmes et al., 1996). The POD is shown to be a valid scale classification for coarse analysis of chemical reaction-diffuser systems in Gear et al. (2002). The proper orthogonal decomposition is the scale classification used for the analysis documented here.

The proper orthogonal decomposition seeks to construct an optimal basis $\{\phi_j(x)\}, j \in \mathbb{N}$ for $\mathbb{Q}$, where $\mathbb{Q}$ is a linear, infinite-dimensional Hilbert space. For this discussion, $\mathbb{Q}$ is the space of fluids with state $q = (\rho u, \rho v, \rho, \rho E)$ with the usual inner product. For finite representations of the fluid in the form of

$$q_N(x;t) = \sum_{j=1}^{N} a_j(t)\phi_j(x), \tag{3.16}$$

the basis $\{\phi_j(x)\}$ is considered optimal in the sense that the averaged projection of a fluid state q onto ϕ is maximal:

$$\max_{\phi \in \mathbb{Q}} \frac{\langle |(q,\phi)|^2 \rangle}{\|\phi\|^2} \tag{3.17}$$

subject to $\|\phi\|^2 = 1$. Here, the operator $\langle \cdot \rangle$ refers to ensemble average. This constrained variational problem is equivalent to the following eigenfunction problem:

$$\int_V \langle q(x)q^*(x')\rangle \phi(x')dx' = \lambda\phi(x) \tag{3.18}$$

with $\langle q(x)q^*(x')\rangle = R(x,x')$ the averaged autocorrelation function and (λ, ϕ) the corresponding eigenvalue/eigenfunction pair. The eigenfunctions are referred to as *empirical eigenfunctions* or *POD modes* and the eigenvalues are a measure of the energy content of the corresponding POD mode.

An efficient method for computing the basis functions ϕ_j is the *method of snapshots* (Holmes et al., 1996). Consider a system where one obtains an ensemble of M snapshots (instances) q^k of the system. If ϕ is an eigenvector for the problem in Eq. (3.18), then it can be written as

$$\phi = \sum_{k=1}^{M} b_k q^k \tag{3.19}$$

with unknown coefficients b_k. The eigenfunction problem may be written as

$$\left(\frac{1}{M} \sum_{i=1}^{M} q^i \otimes q^i , \sum_{k=1}^{M} b_k q^k \right) = \lambda \sum_{i=1}^{M} b_k q^k . \tag{3.20}$$

This may be rewritten as

$$\sum_{k=1}^{M} \frac{1}{M} (q^i , q^k) b_k = \lambda b_i , \tag{3.21}$$

which is an $M \times M$ eigenfunction problem for the coefficients b_i in the expansion for ϕ. The method of snapshots may also be applied to affine systems.

The proper orthogonal decomposition and the method of snapshots have been applied to incompressible fluids (Holmes et al., 1996) and compressible fluids (Rowley et al., 2004). Note that the issue of compressibility is addressed in the details of the scale classification (and associated projection and lifting operators) rather than in the fundamental nature of the classification. In other words, the POD is a valid scale classification for both incompressible and compressible flows. Specifically, the issue of compressibility influences the choice of the elements of the flow vector q and on the choice of the inner product used for the POD. For the purposes of this study, the flow vector q is chosen to correspond to that for the detailed simulation,

$$q = (\rho u, \rho v, \rho, \rho E), \tag{3.22}$$

and the usual inner product is chosen for the POD-related computations. As discussed in the following, choosing usual inner product leads to well-posed lifting operators.

Projection operator Because the POD-scale classification leads to orthogonal descriptions of the scales, projection follows naturally. A choice is made to identify the most energetic empirical eigenfunctions as the dominant scales of the flow, and projection is equivalent to projection of the flow onto the POD eigenfunctions. As a given flow state is identified with a linear combination of the eigenfunctions (the POD coefficients), projection is a truncation to the corresponding dominant-scale POD coefficient vector.

For an expansion of the flow onto M POD modes,

$$q_M(x;t) = \sum_{j=1}^{M} a_j(t)\phi_j(x), \tag{3.23}$$

a projection onto $N < M$ modes yields a truncated expansion,

$$q_N(x;t) = \sum_{j=1}^{N} a_j(t)\phi_j(x). \tag{3.24}$$

In other words, the projection of the flow onto N POD modes is a mapping from $q \in \mathbb{Q}$ to $a \in \mathbb{R}^N$:

$$\mathbb{P}_{\mathbb{N}}(q) = \{a_j\},\ j = 1, \cdots, N. \tag{3.25}$$

For $N < M$ with a number of snapshots M much smaller than the dimension of $\mathbb{Q}$, this is a considerable reduction. For instance, given the earlier estimate of dim $(\mathbb{Q}) = 100,000$, choosing $M = 125$ and $N = 20$ represents an overall reduction of four orders of magnitude.

Lifting operator The lifting operator is related to the reconstruction procedure in the POD framework. That is, the POD coefficient vector yields a linear combination of the POD eigenfunctions that approximates the corresponding flow state. For the analysis presented in this document, the lifting operator is POD reconstruction with the higher-order (subdominant) scales set to zero. In other words, the lifting operator is the corresponding linear combination of the POD modes where only the coefficients corresponding to the dominant scales are nonzero.

For the analysis documented here, the lifting operator $\mathbb{L} : \mathbb{R}^N \longmapsto \mathbb{Q}$ is

$$\mathbb{L}(\{a\}) = q(x;t) = \sum_{j=1}^{N} a_j(t)\phi_j(x). \tag{3.26}$$

Other lifting operators may be defined where the higher-order modes are initialized from a distribution. In addition, observe that because of the dependence on POD reconstruction, the decomposition, including the choice of q and the choice of inner product, must produce a well-posed lift. To this end, the choice of q in the scale classification is consistent with the lifting required for the DNS, and the choice of the usual inner product allows for a well-posed reconstruction of q.[5]

3.7 Coarse Time Stepping

Detailed simulation tools Vector fields for the fluid flow may be computed using bursts of DNS. Consider the DNS simulation as a mapping $\phi_{\mathrm{DNS},\tau}$ of a given initial condition $q_n(i,j)$ for

[5] One might choose a different inner product, for instance the kinetic energy, or enthalpy, or enstrophy. Choosing these inner products for computing POD modes would yield modes whose dimension are not the same as that for the original flow. In this case, reconstruction through the POD would be ill-posed.

the flow at time index n to the flow state $q_{n+\tau}(i,j)$ at a later time index $n+\tau$:

$$\phi_{\text{DNS},\tau}: \qquad q_n(i,j) \longmapsto q_{n+\tau}(i,j). \tag{3.27}$$

Now, using the DNS simulation, an approximation to the vector field q at a time index $n+(k+1)h$ may be computed using forward finite differencing. Using the vector field $\dot{q}_n$ as an initial condition for a DNS simulation running for $k+1$ time steps (at a DNS time step h), the vector fields corresponding to the final two time steps kh and $(k+1)h$ may be used to construct a finite difference approximation to $\dot{q}_{n+(k+1)h}$:

$$\dot{q}_{n+(k+1)h}(i,j) \approx \frac{1}{h}\left(\phi_{\text{DNS},(k+1)h}\big(q_n(i,j)\big) - \phi_{\text{DNS},kh}\big(q_n(i,j)\big)\right). \tag{3.28}$$

Using the projection operator, the vector field corresponding to the dominant scales may be computed as follows:

$$\dot{a}_{n+(k+1)h} \approx \frac{1}{h}\bigg(\mathbb{P}\Big(\phi_{\text{DNS},(k+1)h}\big(q_n(i,j)\big)\Big) - \mathbb{P}\Big(\phi_{\text{DNS},kh}\big(q_n(i,j)\big)\Big)\bigg). \tag{3.29}$$

We apply the lift operator to generate a single approximation for q, yielding

$$\dot{a}_{n+(k+1)h} \approx \frac{1}{h}\bigg(\mathbb{P}\Big(\phi_{\text{DNS},(k+1)h}\big(\mathbb{L}(\{a_n\})\big)\Big) - \mathbb{P}\Big(\phi_{\text{DNS},kh}\big(\mathbb{L}(\{a_n\})\big)\Big)\bigg). \tag{3.30}$$

Note that the averaging operator is not present, because only one lifted initial condition is generated. In the work documented in this thesis, only one initial condition q is generated for each application of the lift operator, and the averaging operator is not used.

In this way, a vector field to the coarse representation $a_{n+(k+1)h}$ may be computed from a_n, $a_{n+(k+1)h}$, and a_{n+kh} through the use of the projection and lift operators and detailed simulation tool.

Coarse Time Integration Following from POD-based scale classification, the dominant scale quantities of interest are the low-order energetic POD modes. A basic tool of coarse analysis is the coarse timestepper; this tool is used for computing the time histories of the dominant scales. Using the components listed above, conventional tools in numerical analysis may be adapted for coarse analysis. For instance, consider the system $\dot{q} = f(q)$; for fluids these are the Navier Stokes equations. It is desired to compute time histories for the dominant scales ϕ; in other words, it is desired to compute trajectories $a(t)$. If a reduced system $\dot{a} = g(a)$ were available, for instance, a Euler time integrator may be constructed to simulate the system:

$$a_{n+Mh} = a_n + g(a)Mh. \tag{3.31}$$

Replacing the vector field $g(a)$ with the approximation provided in the previous subsection yields the following time integrator:

$$a_{n+(M+k+1)h} = a_n + M\mathbb{P}\Big(\phi_{\mathrm{DNS},(k+1)h}\big(\mathbb{L}(\{a_n\})\big) - \phi_{\mathrm{DNS},kh}\big(\mathbb{L}(\{a_n\})\big)\Big). \tag{3.32}$$

Similarly, coarse timestepping algorithms may be constructed using more sophisticated techniques (Gear and Kevrekidis, 2003a).

Derivation of a modified Adams-Bashforth Integration Routine We derive third-order multistep Adams-Bashforth integration routines using Lagrange interpolating polynomials according to Shampine and Gordon (1975). In other words, the approximation of state y_{n+1} at time index $n+1$ given previous and existing states (y_n, y_{n-1}, y_{n-2}) at time indices n, $n-1$, and $n-2$ is

$$y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} P_{3,n}(t) dt, \tag{3.33}$$

where $P_{3,n}(t)$ is composed of three Lagrange interpolating polynomials:

$$P_{3,n}(t) = \sum_{i=1}^{3} l_i(t) f_{n+1-i} \tag{3.34}$$

$$l_i(t) = \prod_{j=1}^{3} \frac{t - t_{n+1-j}}{t_{n+1-i} - t_{n+1-j}}. \tag{3.35}$$

Substituting these Lagrange polynomials into Equation 3.33 and carrying out the integration leads to, after some algebra,

$$y_{n+1} = y_n + f_n \frac{c}{6ab}(6ab + 3ac + 3bc + 2c^2) - f_{n-1} \frac{1}{b(a-b)} \frac{1}{6} c^2 (3a + 2c) - f_{n-2} \frac{1}{a(b-a)} \frac{1}{6} c^2 (3b + 2c), \tag{3.36}$$

where $f = \dot{y}$ are the vector fields for y and where the point spacing is given as follows:

$$t_{n-2} = t_n - a \tag{3.37}$$

$$t_{n-1} = t_n - b \tag{3.38}$$

$$t_{n+1} = t_n + c. \tag{3.39}$$

Therefore, the **standard third-order Adams-Bashforth scheme** is given by setting $a = 2h$, $b = h$, $c = h$ with step size h:

$$y_{n+1} = y_n + \frac{23}{12} h f_n - \frac{4}{3} h f_{n-1} + \frac{5}{12} h f_{n-2}. \tag{3.40}$$

Application of this classical algorithm directly to the coarse representation a_n yields the following time integrator:

$$a_{n+Mh} = a_n + \frac{27Mh}{12}\dot{a}_n - \frac{4Mh}{3}\dot{a}_{n-Mh} + \frac{5Mh}{12}\dot{a}_{n-2Mh}. \tag{3.41}$$

Coarsening this integration routine in the same manner as above (approximating time derivatives with bursts of DNS simulation and first-order finite differences) yields

$$\begin{aligned} a_{n+Mh} = a_n + \frac{27M}{12}\Big(\mathbb{P}\Big(\phi_{\text{DNS},h}(\mathbb{L}(\{a_n\}))\Big) - \mathbb{P}\Big(\mathbb{L}(\{a_n\})\Big)\Big) \\ -\frac{4M}{3}\Big(\mathbb{P}\Big(\phi_{\text{DNS},h}(\mathbb{L}(\{a_{n-Mh}\}))\Big) - \mathbb{P}\Big(\mathbb{L}(\{a_{n-Mh}\})\Big)\Big) \\ +\frac{5M}{12}\Big(\mathbb{P}\Big(\phi_{\text{DNS},h}(\mathbb{L}(\{a_{n-2Mh}\}))\Big) - \mathbb{P}\Big(\mathbb{L}(\{a_{n-2Mh}\})\Big)\Big). \end{aligned} \tag{3.42}$$

This scheme applies bursts of simulation beginning at the coarse times t, $t - Mh$, $t - 2Mh$, and approximates the derivatives of POD coefficients $\dot{a}$ accordingly. Errors may naturally arise, however, if the POD coefficients and approximate derivatives used are not aligned with those times. For instance, it may be advantageous to compute DNS bursts for times longer than a single time step h to allow for interaction among scales. If low-order finite differencing is imposed, either the information at the end of the DNS bursts are not used, or the corresponding finite difference approximations closer to $\dot{a}_{t+jMh+kh}$ are used instead. The latter approximation may lead to errors.

Instead, to apply low-order finite differencing and allow for longer DNS bursts, the DNS burst size kh should be accounted for in the coarse integration scheme. To this end, the Adams-Bashforth routine is now rederived to account for a DNS burst over a time interval kh.

A new scheme may be derived by setting different values of a, b, and c for Eq. (3.36). For instance, setting the following values

$$a = 2Mh \tag{3.43}$$

$$b = Mh \tag{3.44}$$

$$c = (M-k)h \tag{3.45}$$

with M and k integers and h the stepsize, yields the following scheme:

$$y_{n+1} = y_n + \frac{M-k}{12M^2}(2k^2 - 13kM + 23M^2)hf_n + \frac{k-4M}{3M^2}(M-k)^2 hf_{n-1} + \frac{(M-k)^2}{12M^2}(-2k+5M)hf_{n-2}. \tag{3.46}$$

Therefore, before coarsening, the modified third-order Adams-Bashforth scheme for the dominant scales a is

$$\begin{aligned} a_{n+(M-k)h} = a_n + \frac{(M-k)h}{12M^2}(2k^2 - 13kM + 23M^2)\dot{a}_n \\ + \frac{(k-4M)h}{3M^2}(M-k)^2 \dot{a}_{n-Mh} \\ + \frac{(M-k)^2 h}{12M^2}(-2k+5M)\dot{a}_{n-2Mh}. \end{aligned} \tag{3.47}$$

Note that for $k = 0$, this scheme reduces to that of Eq. (3.41). Coarsening this scheme leads to the following *modified* coarse third-order Adams-Bashforth scheme:

$$\begin{aligned}a_{n+(M-k)h} = a_n + \frac{M-k}{12kM^2}(2k^2 - 13kM + 23M^2)(a_n - a_{n-h}) \\ + \frac{k-4M}{3kM^2}(M-k)^2(a_{n-Mh} - a_{n-(M-1)h}) \\ + \frac{(M-k)^2}{12kM^2}(-2k+5M)(a_{n-2Mh} - a_{n-(2M-1)h}).\end{aligned} \tag{3.48}$$

At the completion of this step, a burst of DNS simulation is made at a burst time of kh. Projections are made at the final two (detailed simulation) time steps of this burst. Note that at the coarse Adams-Bashforth routine may now be repeated. Observe that this modified scheme does not suffer from the penalty imposed by making low-order, DNS-based, finite-difference approximations that are away from the intended time instance. In this scheme, the DNS burst is accounted for, and both POD mode coefficients and their approximate time derivatives are represented correctly. Finally, note that the modified coarse Adams-Bashforth scheme in Eq. (3.48) does not simplify to the coarsened classical scheme in Eq. (3.42), because the nature of the coarsening is different for the schemes.

3.8 Numerical Analysis of a Coarse Adams-Bashforth Routine

Similar to the treatment discussed in Section 2.4, the coarsened Adams-Bashforth routine, Eq. (3.48), may be analyzed for its global stability properties as a multistep routine. The error growth may be written directly as

$$\begin{aligned}\epsilon_{n+(M-k)h} = \epsilon_n \rho^{k+1} + \frac{M-k}{12kM^2}\Big(\epsilon_n\rho - \epsilon_{n-h}\Big)\rho^k \\ + \frac{k-4M}{3kM^2}\Big(\epsilon_{n-Mh} - \epsilon_{n-(M-1)h}\Big)\rho^k \\ + \frac{(M-k)^2}{12kM^2}\Big(\epsilon_{n-2Mh}) - \epsilon_{n-(2M-1)h}\Big)\rho^k.\end{aligned} \tag{3.49}$$

Note that the right-hand side is dependent on quantities on time intervals that are not connected through 'detailed simulation'; that is, the right hand side is composed of three disparate portions near time indices n, $n - Mh$, and $n - 2Mh$. In order to proceed, an assumption of *consistency* in the error growth is assumed; that is,

$$\epsilon_{n-jMh} - \epsilon_{n-(jM-1)h} = (\rho - 1)\epsilon_n \rho^k \tag{3.50}$$

for any integers j and M. Substituting this into our error expression in Eq. (3.49), we have

$$\begin{aligned}\epsilon_{n+(M-k)h} = \epsilon_n \rho^{k+1} + \frac{M-k}{12kM^2}(\rho-1)\epsilon_n\rho^k \\ + \frac{k-4M}{3kM^2}(\rho-1)\epsilon_n\rho^k + \frac{(M-k)^2}{12kM^2}(\rho-1)\epsilon_n\rho^k.\end{aligned} \tag{3.51}$$

Now our error growth is

$$\epsilon_{n+(M-k)h} = \sigma(h\lambda)\epsilon_n, \tag{3.52}$$

where

$$\sigma(h\lambda) = \rho^{k+1} + \frac{M-k}{12kM^2}(\rho-1)\rho^k + \frac{k-4M}{3kM^2}(\rho-1)\rho^k + \frac{(M-k)^2}{12kM^2}(\rho-1)\rho^k. \tag{3.53}$$

Now the absolute stability criterion is defined when $\|\sigma(h\lambda)\| < 1$. This region is plotted and compared with other time integration schemes in Figure 3.8.

Note on error analysis. The coarse projective Adams-Bashforth scheme is stable for systems whose eigenvalues exist within the stability region. Although not completed at this time, the method is expected to be accurate to third order as long as the derivative approximations (using finite differencing) are also accurate to third-order or higher. Also, the determination of consistency, convergence, and error bounds has not completed at this time. It is important to note that such analyses are being performed by Kevrekidis and collaborators and have been performed for similar methods such as the heterogeneous multiscale methods by E and Engquist (2003).

3.9 Results: Coarse Diffuser Flows

Scale classification and diffuser flow POD For this coarse analysis, we used the POD for scale classification. We then applied the method of snapshots to construct empirical basis functions for the underlying space for the diffuser flow. We consider an ensemble of 126 snapshots for computing the POD modes. These snapshots are taken from a single DNS simulation and are indicated in Figure 3.9.

The eigenvalues associated with the basis functions are shown in Figure 3.10 for cases where the ensemble average is removed from the snapshots used in the decomposition, and for cases where the ensemble average is not removed. As shown in this figure, the eigenvalues decay rapidly for both cases, and the eigenvalue associated with the ensemble average is much greater than the other

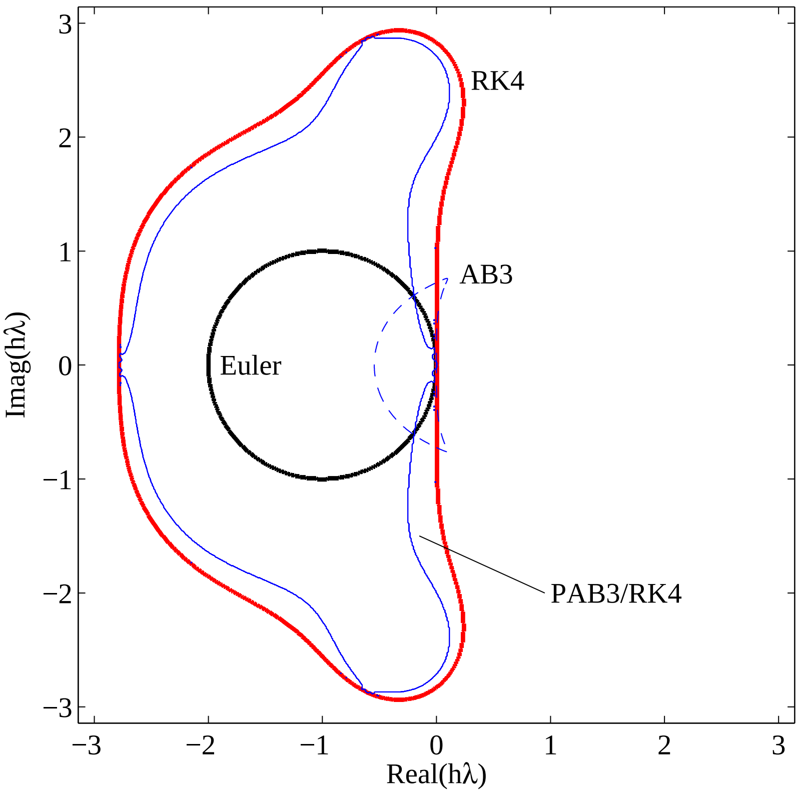


Figure 3.8: Regions of absolute stability for the Euler scheme, fourth-order Runge-Kutta, and coarse projective third-order Adams-Bashforth schemes.

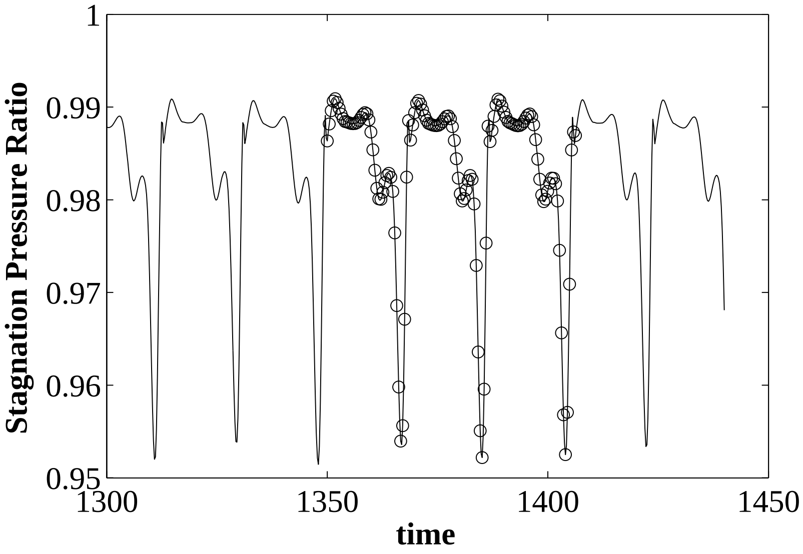


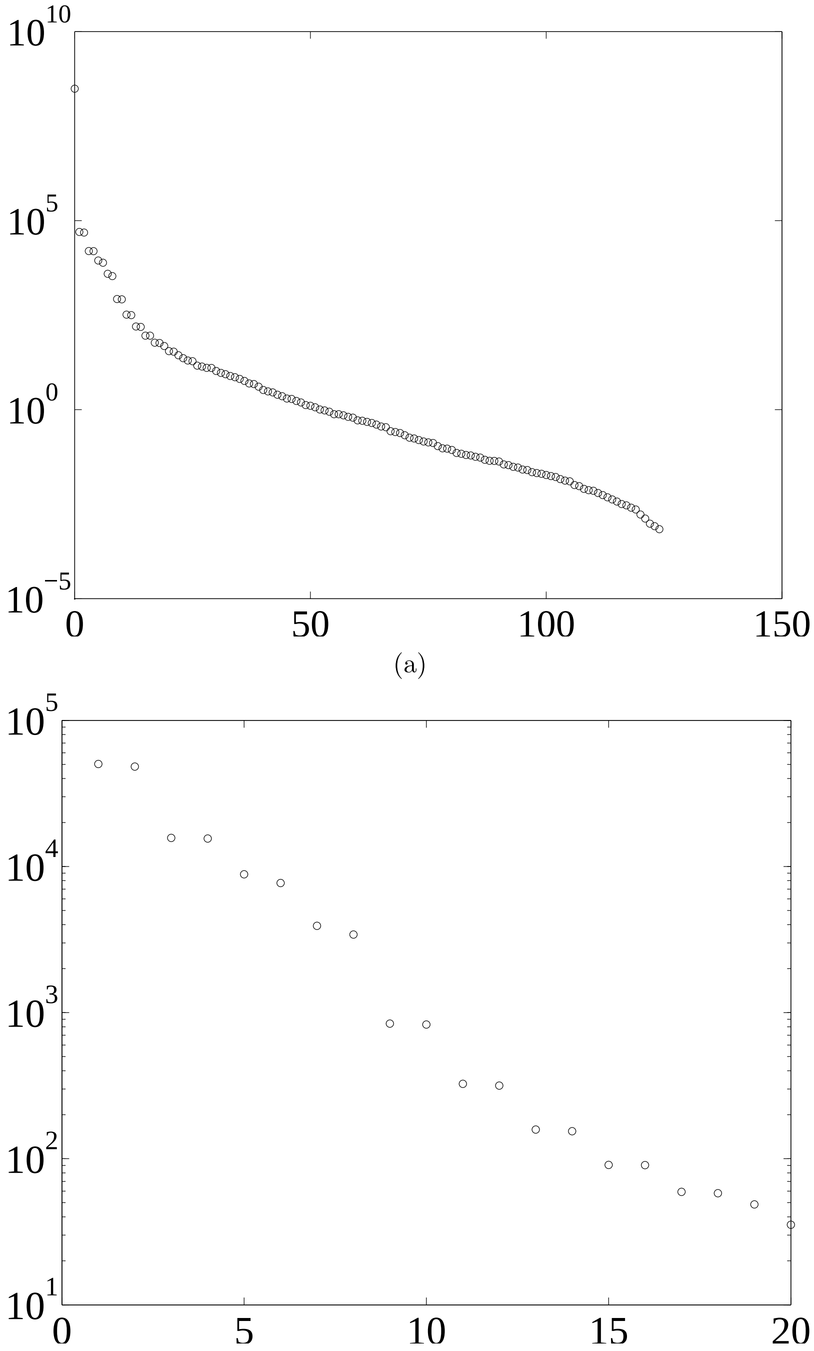
Figure 3.9: Stagnation pressure ratio. Snapshots are taken as indicated.

eigenvalues. Unless otherwise noted in this report, the ensemble average is removed from all of the snapshots taken.

Because the eigenvalues, and thus the energy, of the POD modes decay rapidly, relatively few modes are required to reconstruct a fluid flow (snapshot) to high accuracy. The scale classification arises from the ordering of the modes: low-order modes represent the dominant scales, and the high-order modes represent the subdominant scales. The rapid decay of the eigenvalues suggests that a *separation of scales* may be made for the flow. This will influence the resulting performance of the coarse analysis, as a lack of such a separation will lead to numerical instabilities.

The vorticity profiles for the ensemble average and for the first six POD modes are shown in Figures 3.11–3.12. While these modes are not directly equivalent to the vortex structures found in the diffuser flow, the lowest modes indicate structure of similar spatial scale. Furthermore, as expected, the nontrivial aspects of each mode are found near the diffuser boundary; very little variation is found towards the centerline of the diffuser. Note that the higher-order modes do resemble spatially small vortex structures, which suggests that the POD formulation would be able to model the distributed actuation of vorticity along the boundary. This may be important for flow control applications.

In Figure 3.13 we show the projections of the DNS simulation onto the first twenty POD modes. The flow is quasiperiodic, with greater difficulty in perceiving the periodicity of the trajectories of the higher-order modes. The flow is roughly organized in a series of POD mode-pairs, which displays the level of symmetry (along the diffuser centerline) of the flow. Each of the POD mode trajectories contain temporal content of more than one frequency, indicating that each mode represents different



(a)

(b)

Figure 3.10: POD mode energy (a) without subtracting the ensemble average and (b) for the first twenty modes after subtracting the ensemble average. Note that the scales are different for each figure.

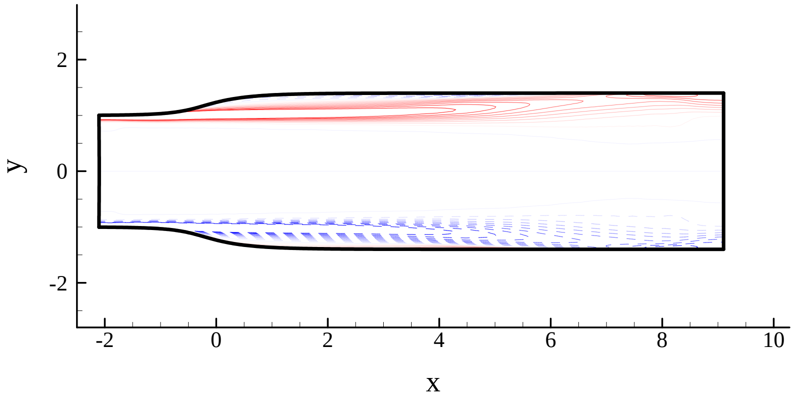


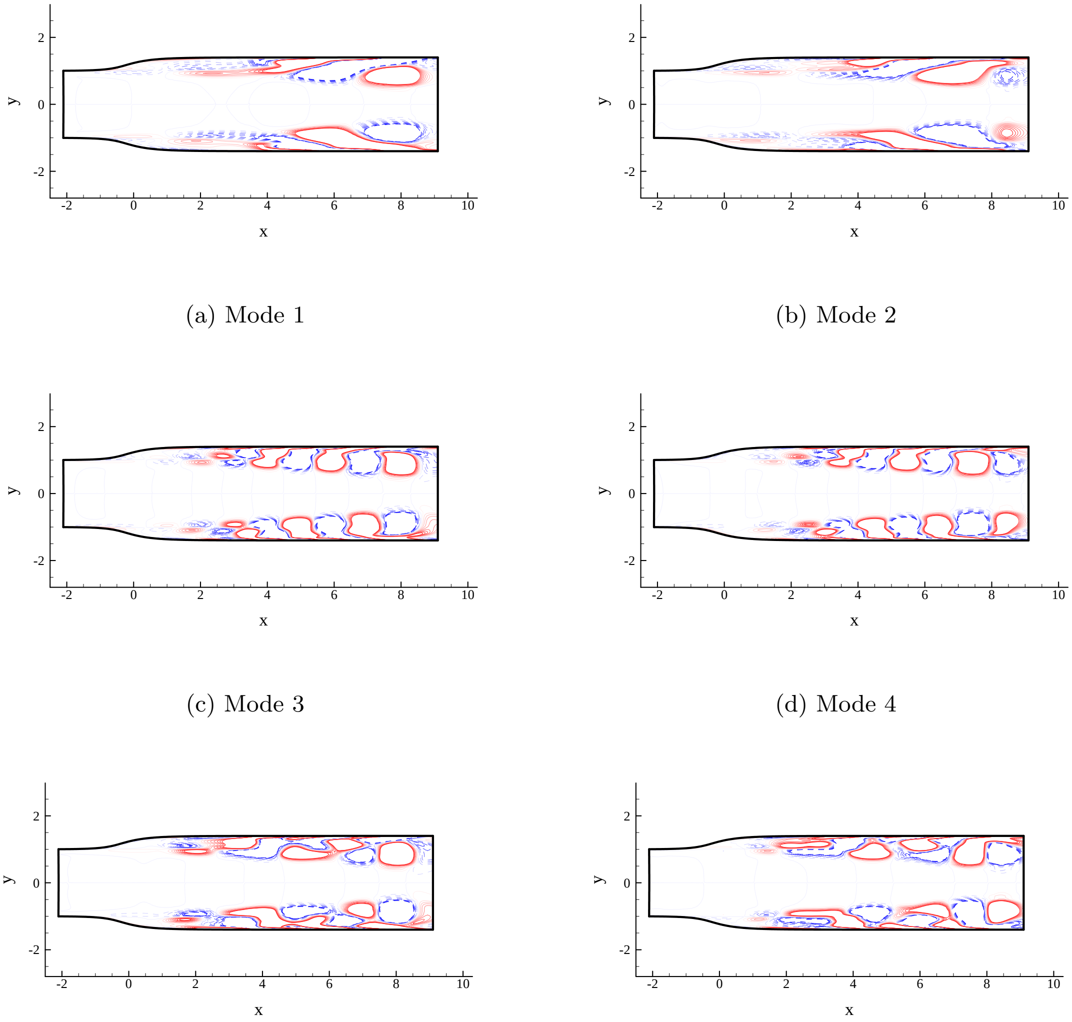
Figure 3.11: Vorticity computed for the ensemble average of the diffuser flow snapshots.

Table 3.1: Energy capture with POD reconstruction.

Number of POD modes	percent energy capture
2	62.8
4	82.7
10	99.0
20	99.8

levels of mode-coupling. Naturally, the higher-order POD modes contain higher-order frequency content; for instance, the tenth mode contains a frequency component that is approximately five times the dominant frequency in the first and second modes.

To illustrate the effectiveness of POD for scale classification, we used the POD to reconstruct the flow using low numbers of POD modes, as shown in Figure 3.14. As shown in this figure, a low number of POD modes are required for reconstruction of the flow at a high degree of accuracy. We list the energy capture (measured according to the POD theory) in Table 3.1 for various numbers of modes used in the reconstruction.



(a) Mode 1

(b) Mode 2

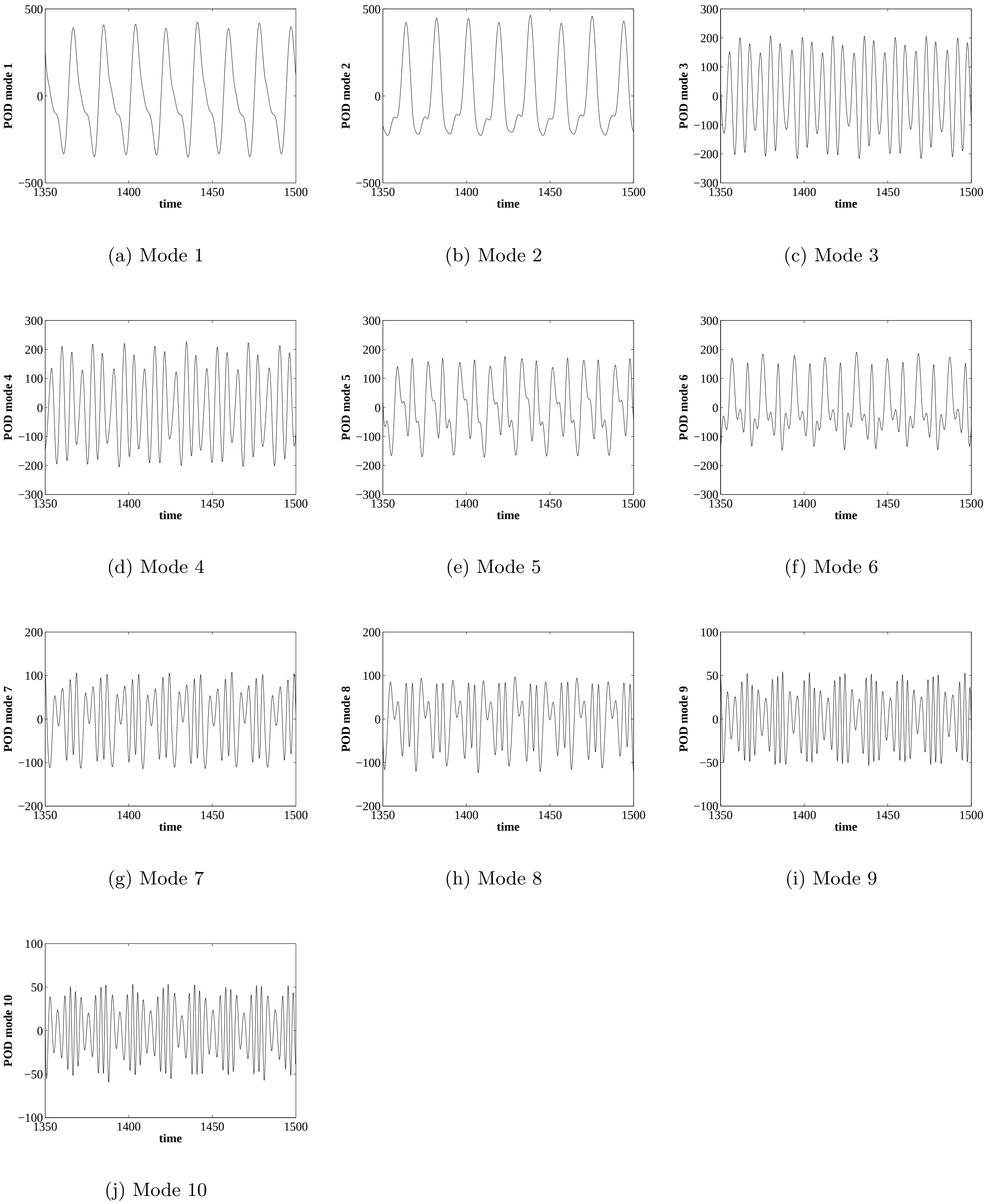
(c) Mode 3

(d) Mode 4

(e) Mode 5

(f) Mode 6

Figure 3.12: Vorticity computed for POD modes of diffuser flow.



(a) Mode 1

(b) Mode 2

(c) Mode 3

(d) Mode 4

(e) Mode 5

(f) Mode 6

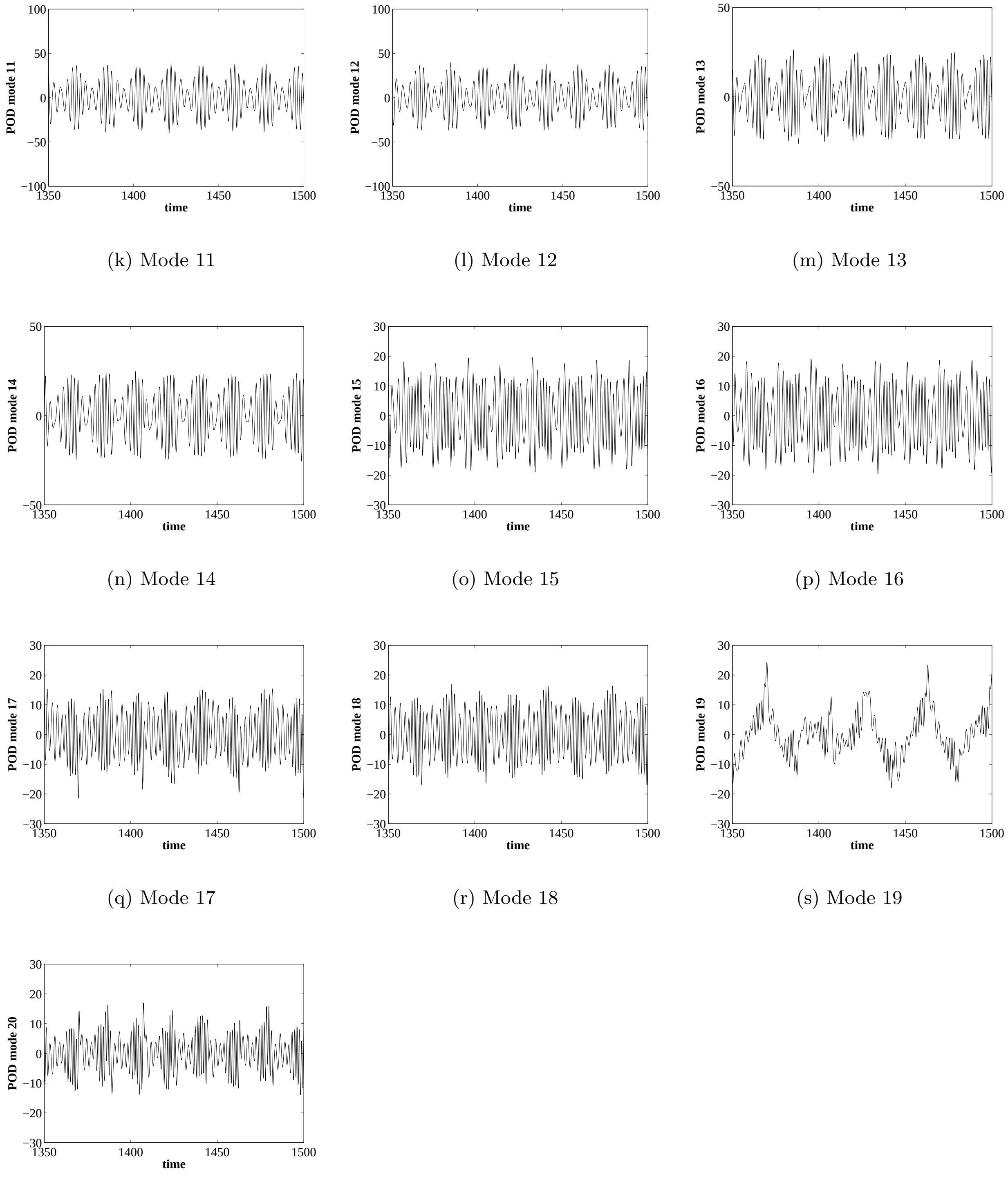
(g) Mode 7

(h) Mode 8

(i) Mode 9

(j) Mode 10

Figure 3.13: Trajectories of the DNS simulation projected onto POD modes.



(k) Mode 11

(l) Mode 12

(m) Mode 13

(n) Mode 14

(o) Mode 15

(p) Mode 16

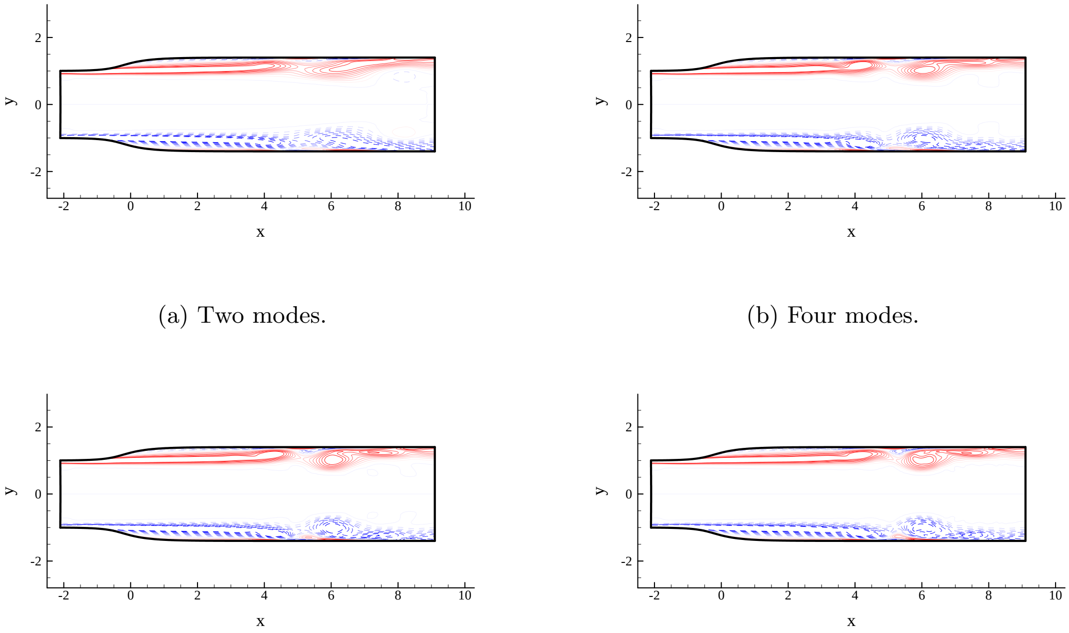
(q) Mode 17

(r) Mode 18

(s) Mode 19

(t) Mode 20

Figure 3.13: Trajectories of the DNS simulation projected onto POD modes.



(a) Two modes.

(b) Four modes.

(c) Ten modes.

(d) Twenty modes.

Figure 3.14: POD reconstruction using (a) two modes; (b) four modes; (c) ten modes; (d) twenty modes.

Behavior in phase space Here we examine a DNS simulation for a candidate limit cycle. Is there a coarse limit cycle? To this end, a DNS simulation was projected onto the first POD mode and shown in Figure 3.15. In this figure, a time history of the first POD mode amplitude a_1 is shown, as well as a phase portrait[6] of the time derivative of this mode amplitude, $\dot{a}_1$, versus the mode amplitude a_1. Indicated in red on this phase portrait is a location where a Poincaré section is taken: this section is shown in the right portion of Figure 3.15.

As shown in Figure 3.15, the projected DNS data occupies a broad region in phase space. The trajectory of this POD mode amplitude appears to be at least quasiperiodic, and perhaps chaotic. According to the Poincaré section, the trajectory does not appear to return to the same phase space position over a long simulation time.

3.9.1 Coarse time integration with the projective (standard) third-order Adams-Bashforth scheme

We compute a time integration of the compressible flow in a planar diffuser using coarse numerical analysis whose framework was described in 2 with POD as the scale classification as described in 3.6. A coarse third-order Adams-Bashforth time integrator is constructed using DNS bursts to compute the vector field as required, and a coarse third-order Runge-Kutta time integrator is used to initialize the coarse Adams-Bashforth routine. This projective Adams-Bashforth integrator is listed in Eq. (3.42). We chose this particular time integration scheme for its computational efficiency and accuracy. Simple forward Euler or Crank-Nicolson schemes were attempted with varying levels

[6] A phase portrait is usually a plot of the time derivative of the amplitude versus the amplitude for a given signal or variable. Periodic or quasi-periodic trajectories (*orbits*) are thus plotted as closed curves.

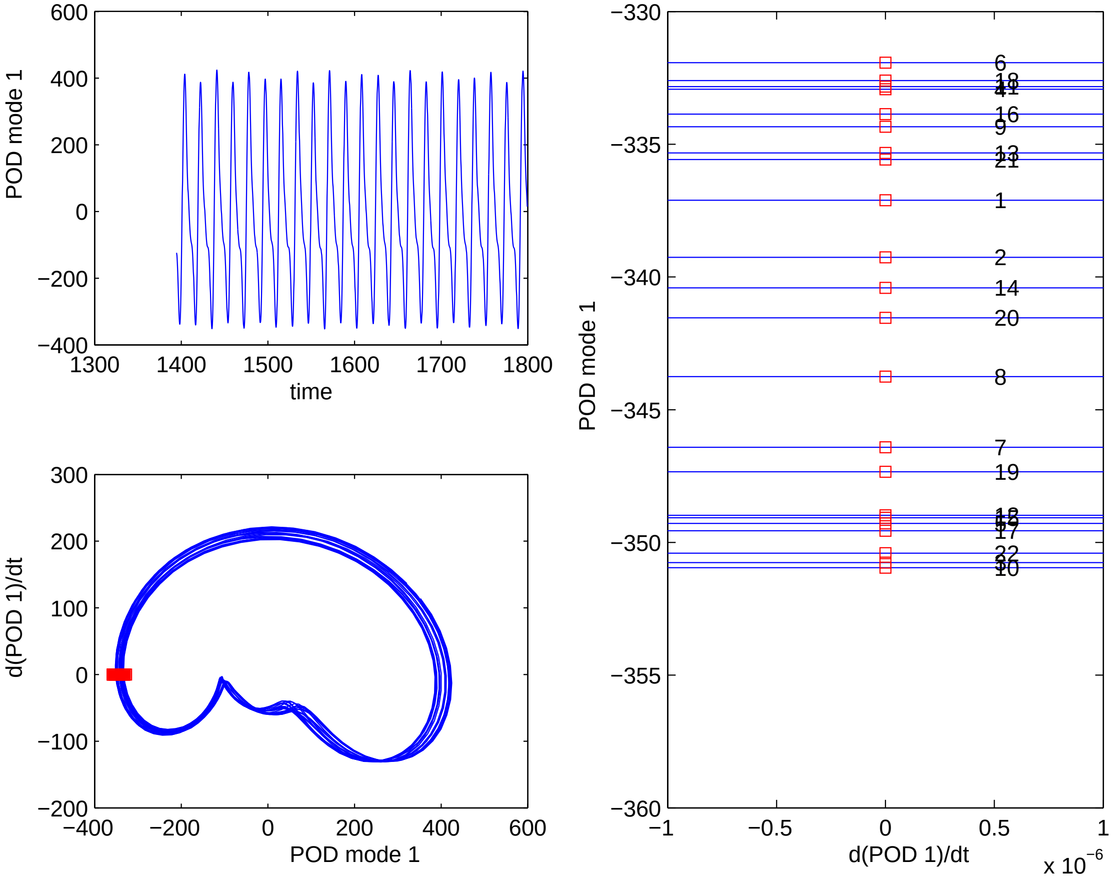


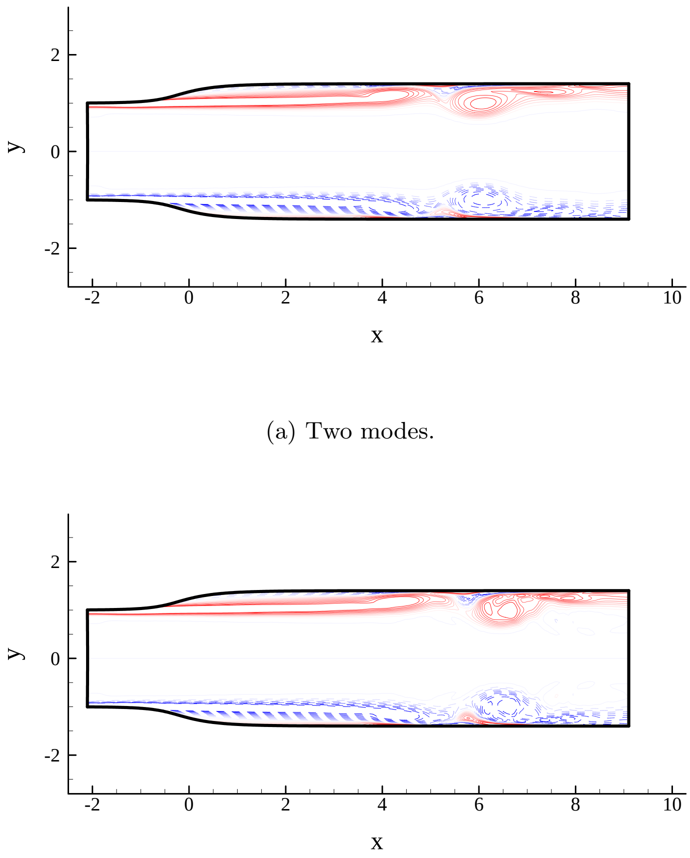
Figure 3.15: DNS data projected onto the first POD mode. A time history of the POD mode amplitude is given, as well as a phase portrait of the first POD mode (time derivative versus amplitude). Also shown is a Poincaré section near $(a_1, \dot{a}_1) = (-325, 0)$, where intersections are shown in red.

of performance and accuracy. For further details on the coarse Adams-Bashforth scheme, we refer to the numerical analysis provided in Gear and Kevrekidis (2003a) and Rico-Martinez et al. (2004).

As mentioned in Section 3.6, the projection operator is taken from the standard POD projection. The lifting operator is taken to be POD mode reconstruction with the higher-order modes initialized to their mean values. This is appropriate, as the higher-order modes have statistics centered about their mean value.

For many limit cycles, our coarse time integrator computes trajectories similar to those computed by DNS. The coarse time integration was carried out at a computational cost that was ten times smaller than that required by DNS. However, the current version of the coarse simulation code made frequent I/O calls, which reduced the overall efficiency. We note that the I/O requirements are dependent only on the code implementation, and not on the coarse analysis algorithm. A comparison of vorticity snapshots is shown in Figure 3.16. Unless otherwise stated, the coarse representation is taken to be the first twenty POD modes. As shown in this figure, the profile of the dominant vortex structures is captured successfully by the coarse integrator. However, the positions of the vortex structures are displaced slightly compared to the DNS simulation results. As is shown below, errors arise in the coarse integration that lead to a phase-shift in the coarse integration.

We give a comparison of the original diffuser flow (computed using DNS), projected onto the first twenty POD modes, and the coarse numerical simulation in Figure 3.17. A comparison of the original diffuser flow stagnation pressure ratio and the coarse numerically computed diffuser flow stagnation pressure ratio is given in Figure 3.18. As shown in these figures, the coarse numerical



(a) Two modes.

(b) Four modes.

Figure 3.16: Comparison of fluid flow snapshots computed (a) from DNS and (b) from the coarse timestepper after four limit cycles.

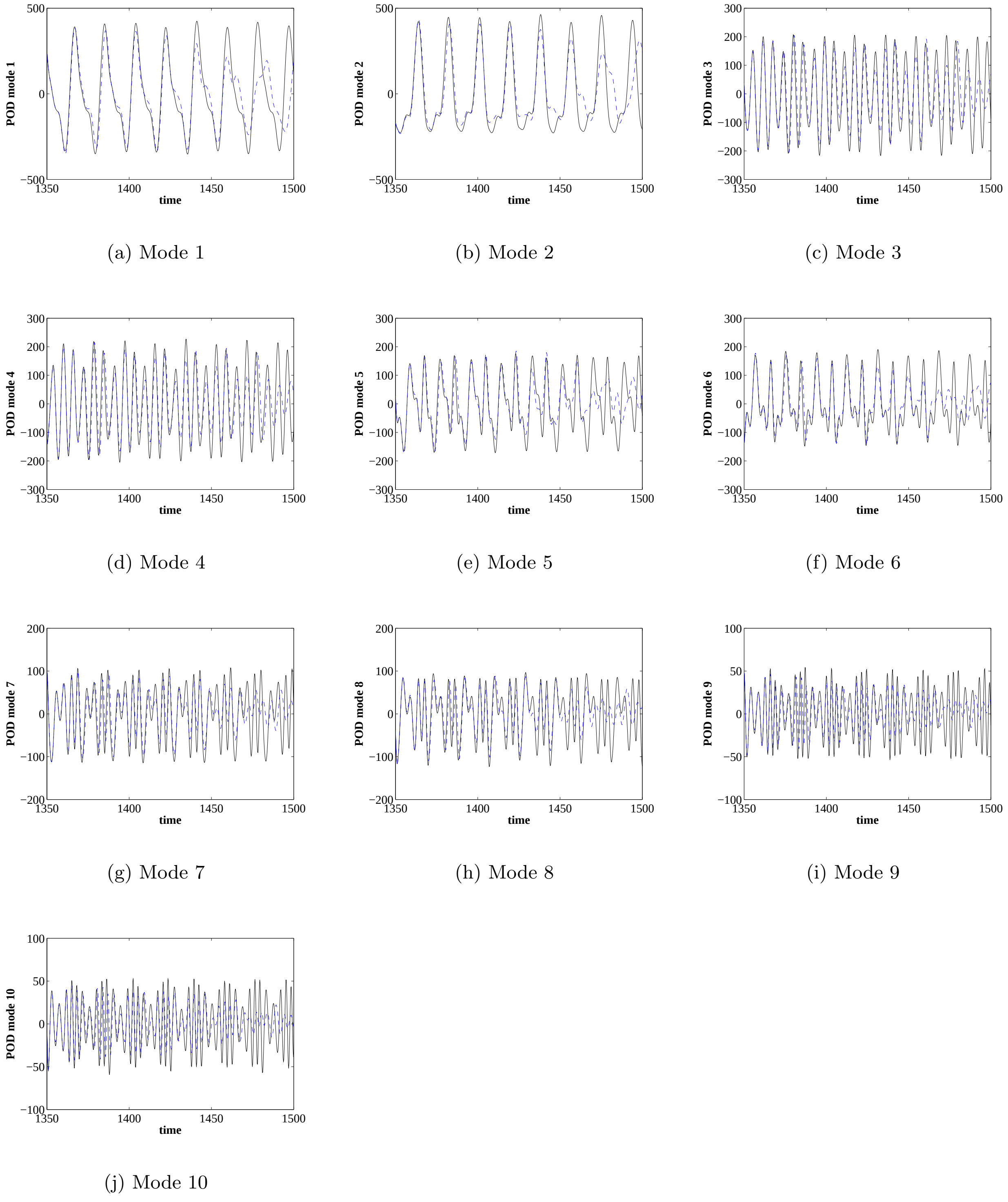
trajectories compare well with those of the DNS projected onto the POD modes.

Errors begin to arise in the higher-order modes in the coarse representation (observe the time history comparisons for the nineteenth and twentieth modes). After several periods corresponding with the trajectory of these POD modes, errors begin to grow and dominate the coarse numerical computation. The error propagation can be observed as other higher-order modes begin to deviate away from the corresponding DNS trajectories. After several periods corresponding to the trajectory of the lowest modes, the errors grow large enough to be perceived visually as in Figure 3.16. However, the phase errors observed in that figure are not apparent in the trajectories of the low order modes or in the stagnation pressure ratio time histories. This suggests that the dominant dynamics (including frequency content) are somewhat invariant to small spatial changes in the dominant scales. We are currently exploring the relationship between the diffuser flow performance, as measured by the stagnation pressure ratio, and the diffuser flow dynamics, as measured by the POD mode time histories.

3.9.2 Coarse time integration with the projective (modified) third-order Adams-Bashforth scheme

In some applications, stable, accurate, long-time coarse integration may be desired. For instance, extracting the coarse nature of the flow, such as limit cycling behavior, is done through observations of the long-time dynamics. In this case, using a projective integration scheme modeled after the standard Adams-Bashforth schemes is not appropriate.

To elucidate the coarse nature of the diffuser flow, a new coarse time integration is performed using



(a) Mode 1

(b) Mode 2

(c) Mode 3

(d) Mode 4

(e) Mode 5

(f) Mode 6

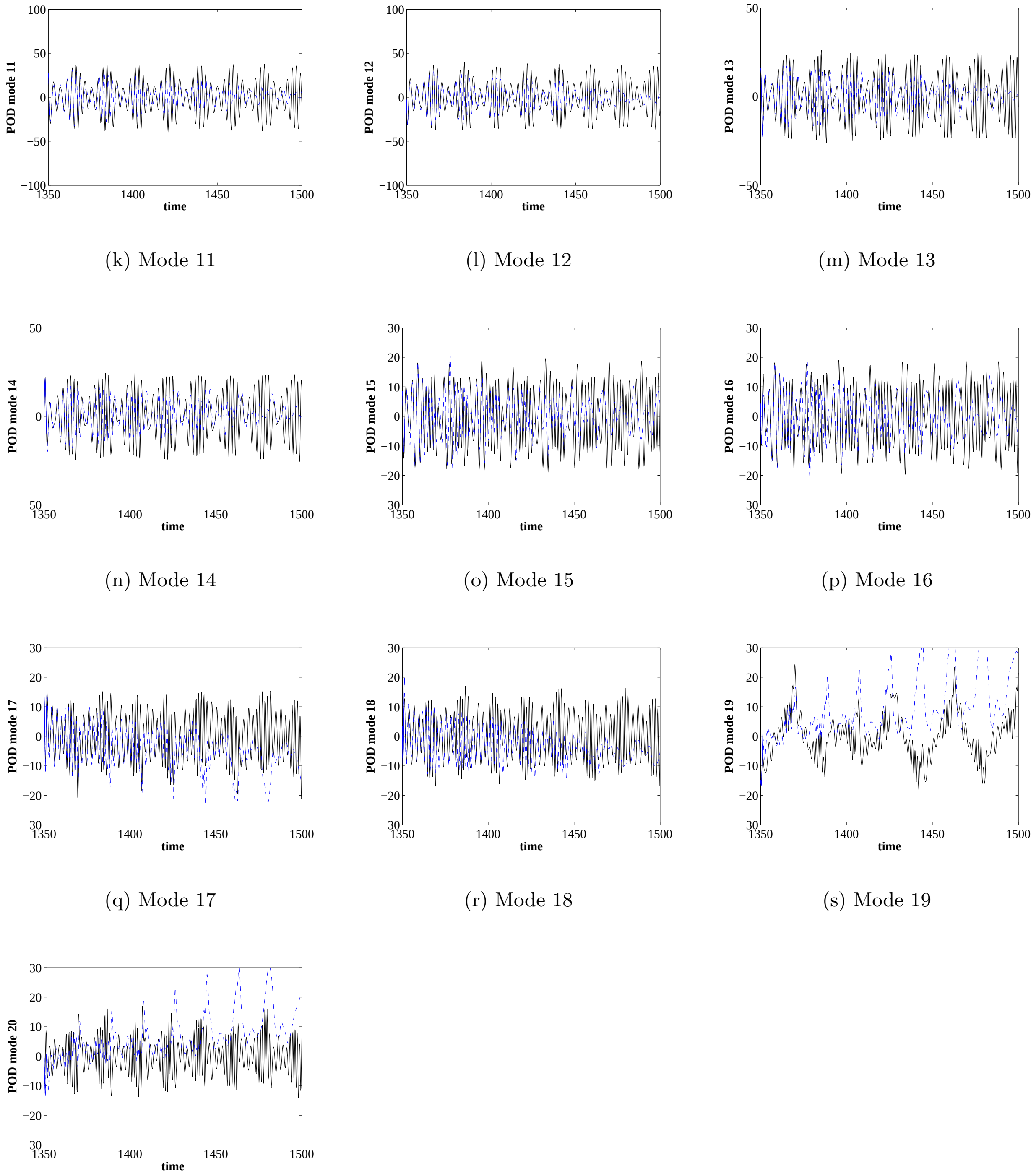
(g) Mode 7

(h) Mode 8

(i) Mode 9

(j) Mode 10

Figure 3.17: Trajectories of the coarse representation (dashed lines) compared with the DNS simulation projected onto the POD modes (solid).



(k) Mode 11

(l) Mode 12

(m) Mode 13

(n) Mode 14

(o) Mode 15

(p) Mode 16

(q) Mode 17

(r) Mode 18

(s) Mode 19

(t) Mode 20

Figure 3.17: Trajectories of the coarse representation (dashed lines) compared with the DNS simulation projected onto the POD modes (solid).

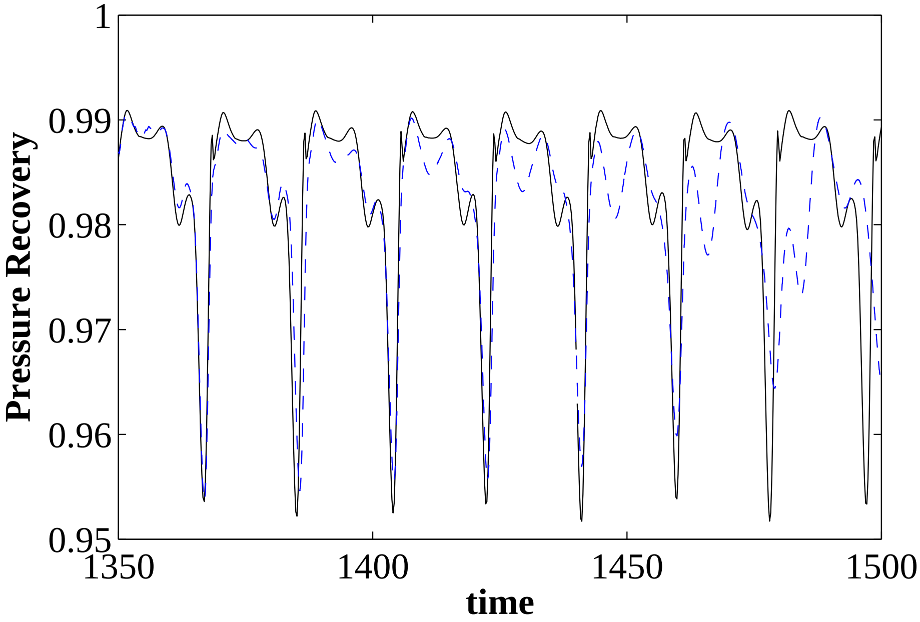


Figure 3.18: Stagnation pressure ratio.

a coarse representation consisting of twenty POD modes. The coarse integrator chosen in this case is the modified Adams-Bashforth third-order integrator listed in Eq. (3.48). For a certain DNS time step Δt, the DNS bursts used by the coarse integrator are computed over a time interval $T_{\text{burst}} = 10\Delta t$ and the coarse (Adams-Bashforth) time steps are $T_{\text{coarse}} = 50\Delta t$.

As an additional note, the coarse computations seem relatively insensitive to decreasing the efficiency ratio. In other words, we may decrease the T_{coarse} values to values comparable with DNS, without affecting the resulting coarse flow. The coarse integrator goes unstable, however, with increasing this ratio beyond 50/10, changing the number of coarse POD modes (the orbits are still stable for coarse representations of nineteen and twenty-one modes, but the amplitudes are larger), or changing the length of the DNS burst. In this work, stable orbits were achieved up to $T_{\text{burst}} = 15\Delta t$).

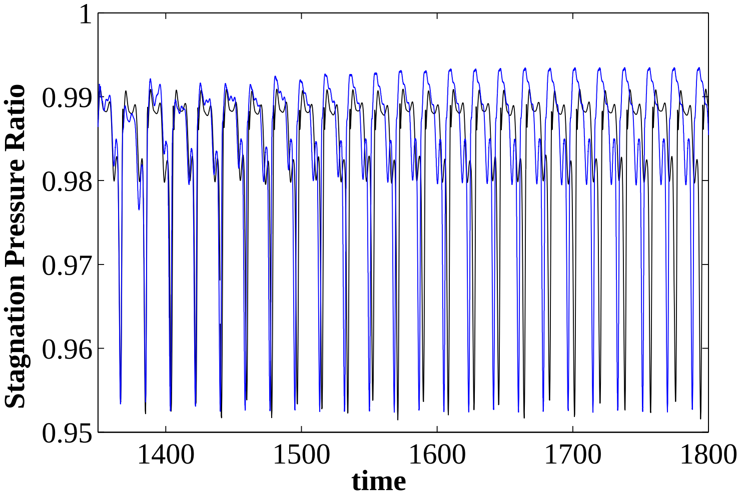


Figure 3.19: Stagnation pressure ratio for DNS (black) and coarse (blue) computations.

time histories of the stagnation pressure ratio for both the original DNS simulation and the coarse simulation are shown in Figure 3.19. Note that the coarse simulation takes several stagnation pressure ratio cycles before convergence. The coarse stagnation pressure ratio amplitude converges to a higher value than that of the DNS flow; however, the dominant frequency of the coarse trajectory remains near that of the DNS flow.

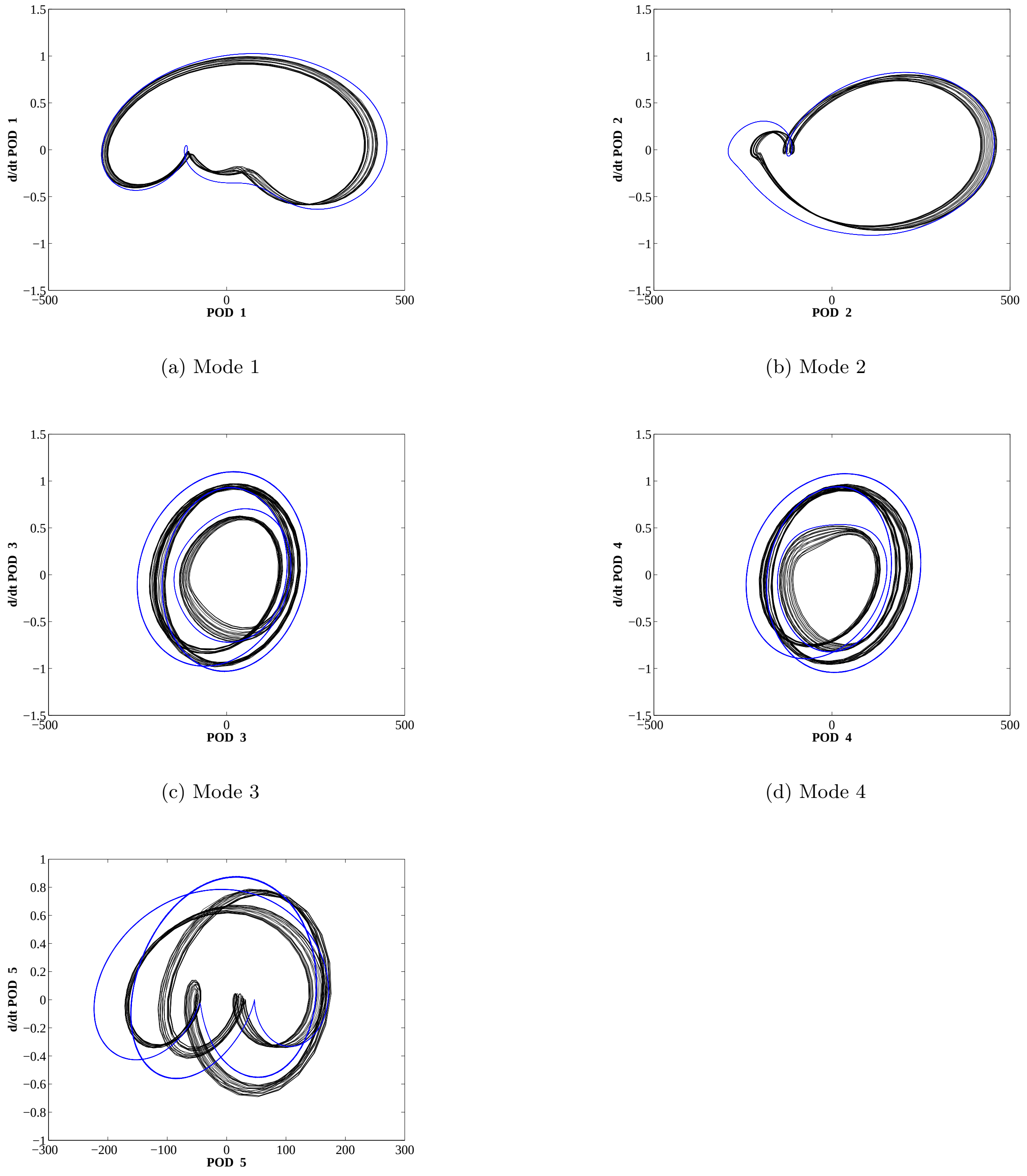
Phase portraits of the POD mode amplitude time derivatives $\dot{a}_i$ versus the POD mode amplitudes a_i would make a more clear comparison between the DNS and coarse diffuser flows. These were computed and are shown in Figure 3.20. The phase portraits of the projected DNS flow show a high quasiperiodicity that increases in complexity with increasing POD mode number. The trajectories can be organized into pairs, both by their respective POD mode energy and their phase portraits. The phase portraits of the coarse flow show a orbit defined over a very compact support-there is no apparent broad occupation of phase space. For the higher-order POD modes in the coarse representation, the trajectory was recorded at a low enough resolution that plotting may alter this perception.

time histories of the POD mode amplitudes are shown in Figures 3.21-3.22. Although the amplitudes and frequency content differ somewhat, the main structure of the time histories compare well. In fact, the phase portraits shown in Figure 3.20 indicate that the essential character of the (coarse) limit cycles are very similar between the projection of the DNS trajectories and the trajectories computed using the coarse integrator.

3.10 Conclusions

We have applied coarse analysis tools to extract the dominant-scale dynamics of the compressible flow in a planar diffuser when only a high-fidelity simulation is available. To this end, we used the proper orthogonal decomposition as our scale classification for the coarse analysis of fluid systems. This scale classification represents a reduction in the number of states by four orders of magnitude. For coarse time integration, the computational cost was observed to be ten times smaller than that for a comparable DNS simulation in some cases. This was achieved while computing flows whose coarse quantities compared well with DNS for short and medium flow times. It is remarkable that coarse analysis can provide efficient, accurate results without making modeling assumptions. While the coarse time integration results compare well for short and medium times, errors grow to dominate the coarse analysis. These errors arise from the POD truncation error and from numerical errors in the coarse time integration algorithm.

To address this issue, the coarse time integration routine was rewritten to more accurately portray



(a) Mode 1

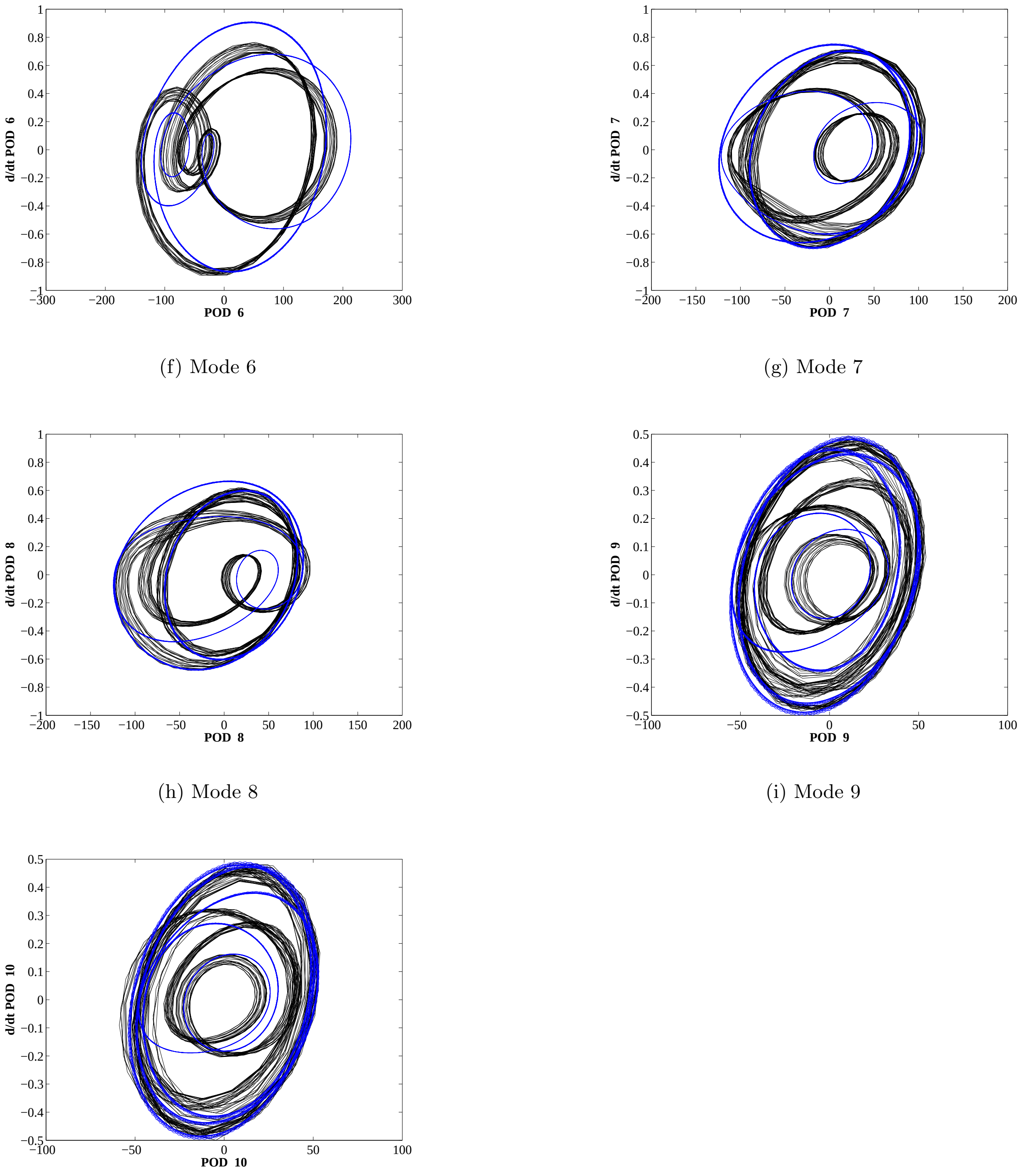
(b) Mode 2

(c) Mode 3

(d) Mode 4

(e) Mode 5

Figure 3.20: Phase portraits of the time derivative of POD mode amplitude, $\dot{a}_i$, versus POD mode amplitude a_i. Black = results from projected DNS computation, blue = coarse computation.



(f) Mode 6

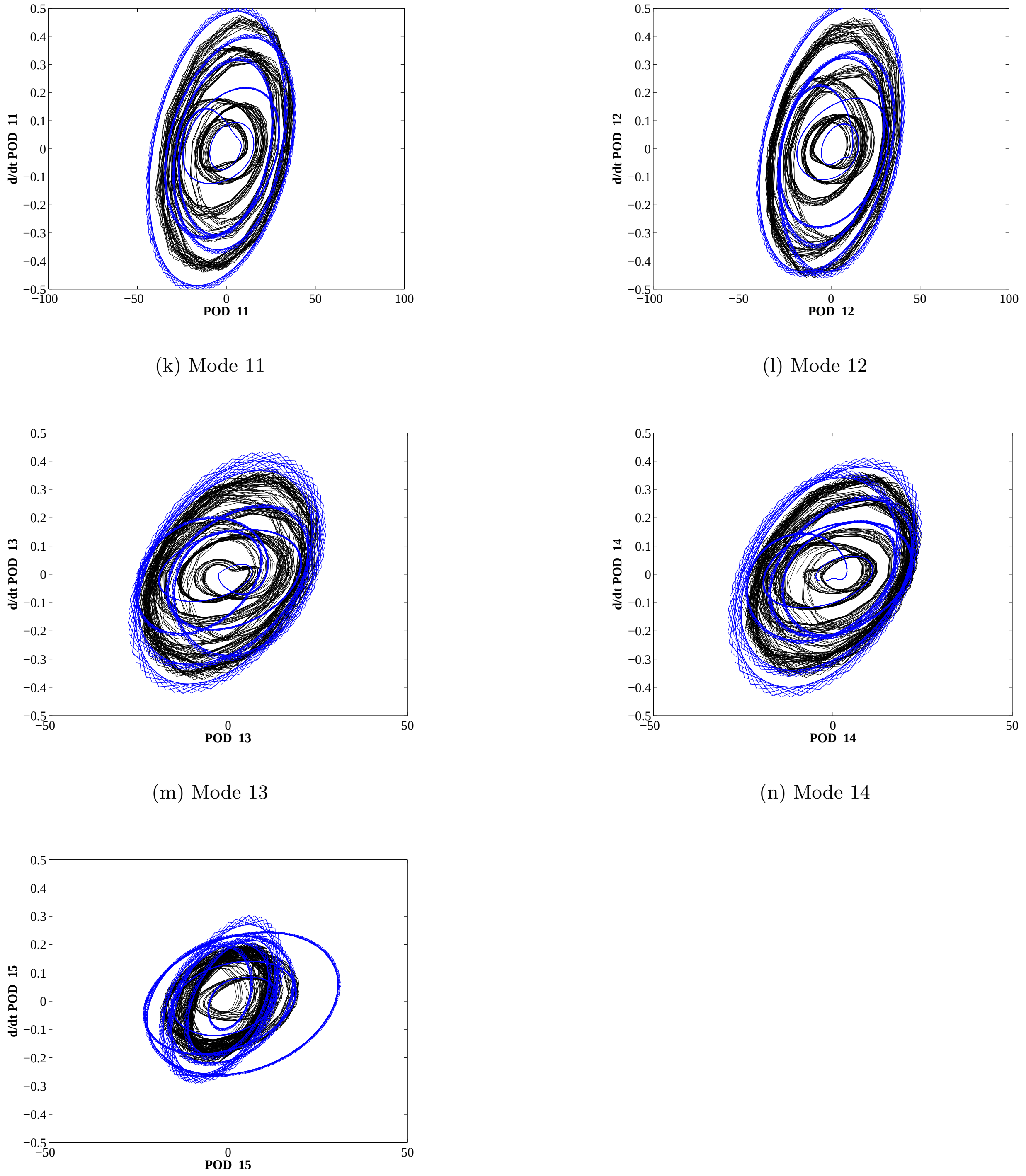
(g) Mode 7

(h) Mode 8

(i) Mode 9

(j) Mode 10

Figure 3.20: Phase portraits of the time derivative of POD mode amplitude, $\dot{a}_i$, versus POD mode amplitude a_i. Black = results from projected DNS computation, blue = coarse computation.



(k) Mode 11

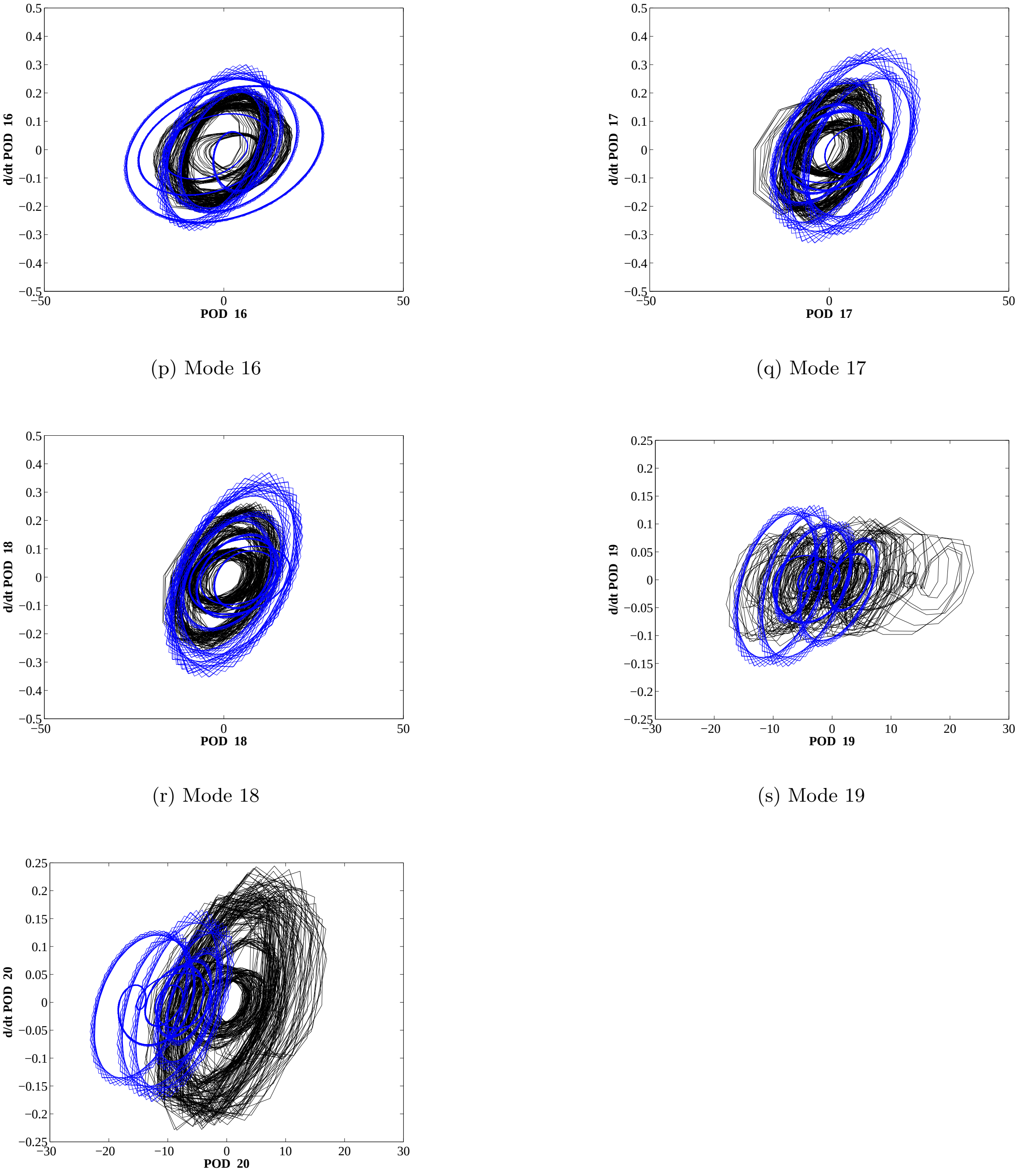
(l) Mode 12

(m) Mode 13

(n) Mode 14

(o) Mode 15

Figure 3.20: Phase portraits of the time derivative of POD mode amplitude, $\dot{a}_i$, versus POD mode amplitude a_i. Black = results from projected DNS computation, blue = coarse computation.



(p) Mode 16

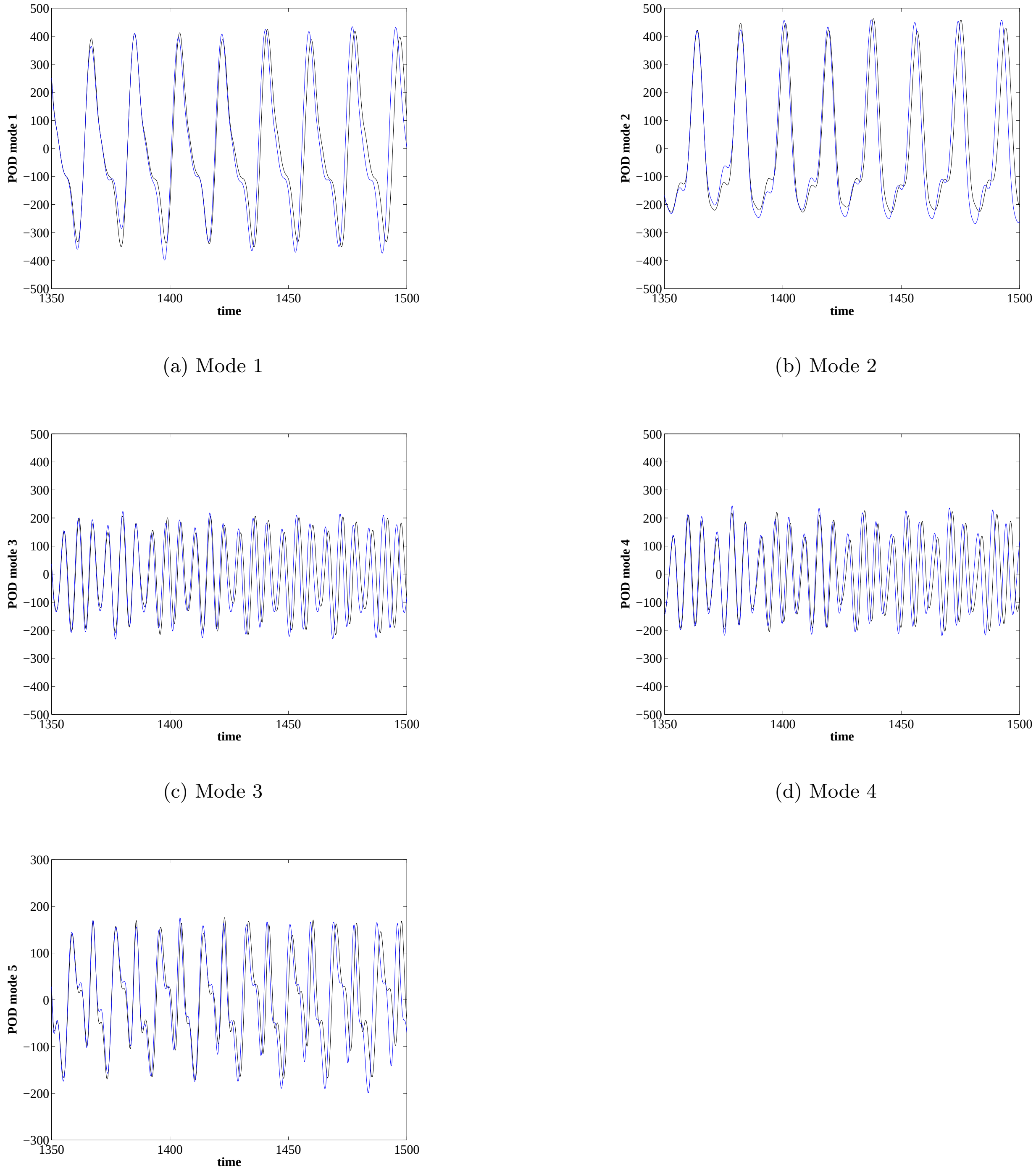
(q) Mode 17

(r) Mode 18

(s) Mode 19

(t) Mode 20

Figure 3.20: Phase portraits of the time derivative of POD mode amplitude, $\dot{a}_i$, versus POD mode amplitude a_i. Black = results from projected DNS computation, blue = coarse computation.



(a) Mode 1

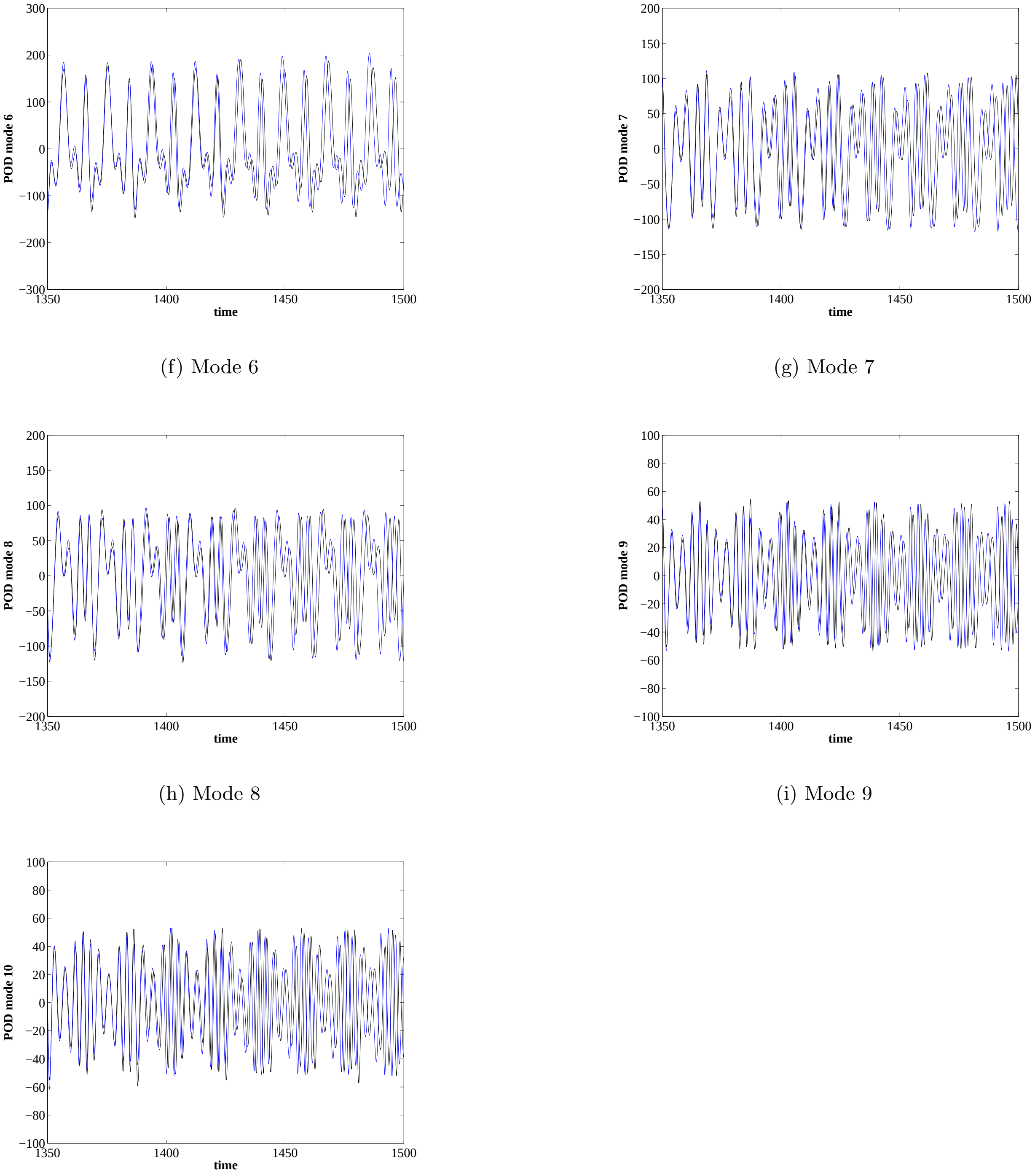
(b) Mode 2

(c) Mode 3

(d) Mode 4

(e) Mode 5

Figure 3.21: time histories of the POD mode amplitude a_i towards the beginning of coarse computation. Black = results from projected DNS computation, blue = coarse computation.



(f) Mode 6

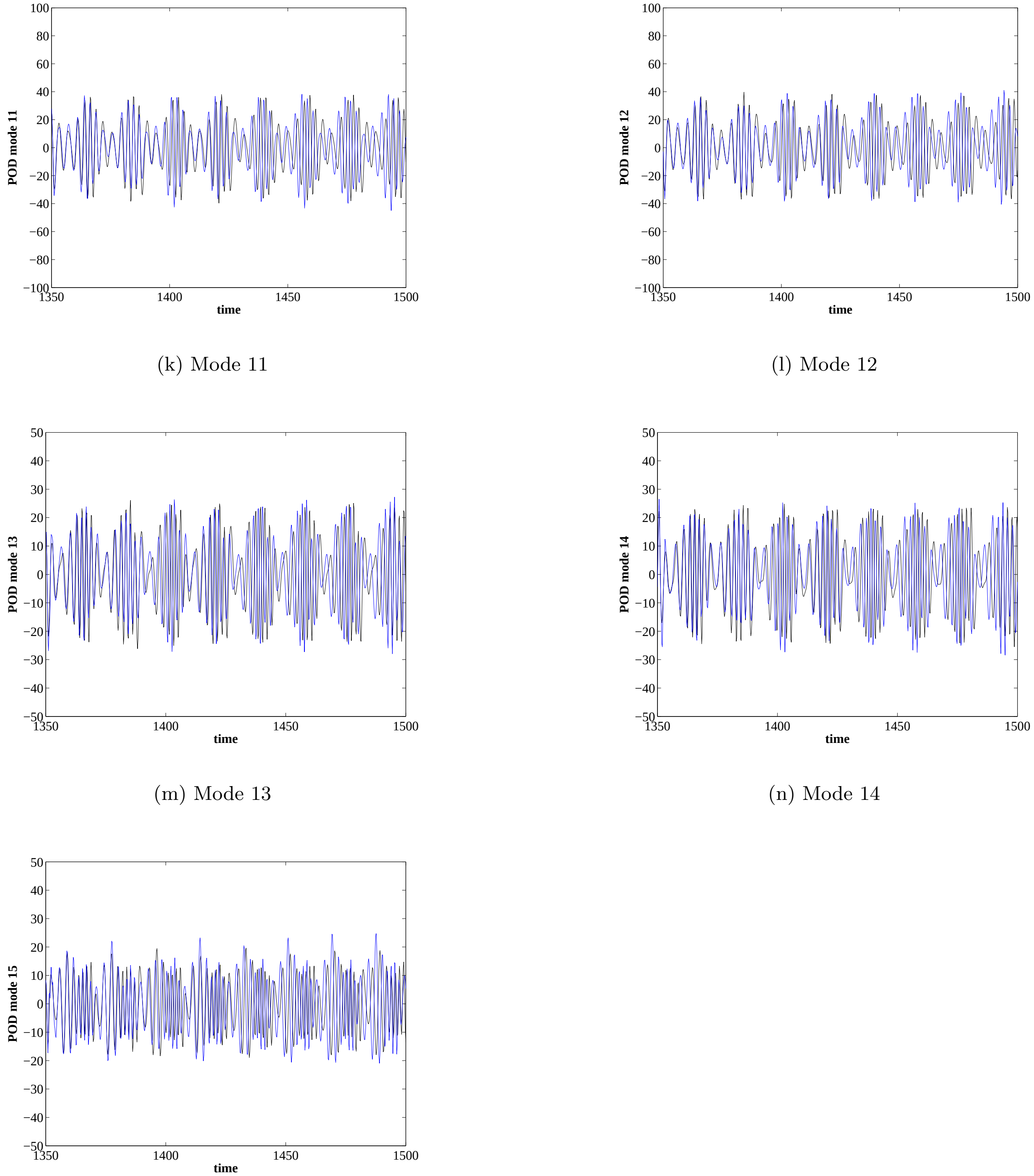
(g) Mode 7

(h) Mode 8

(i) Mode 9

(j) Mode 10

Figure 3.21: time histories of the POD mode amplitude a_i towards the beginning of coarse computation. Black = results from projected DNS computation, blue = coarse computation.



(k) Mode 11

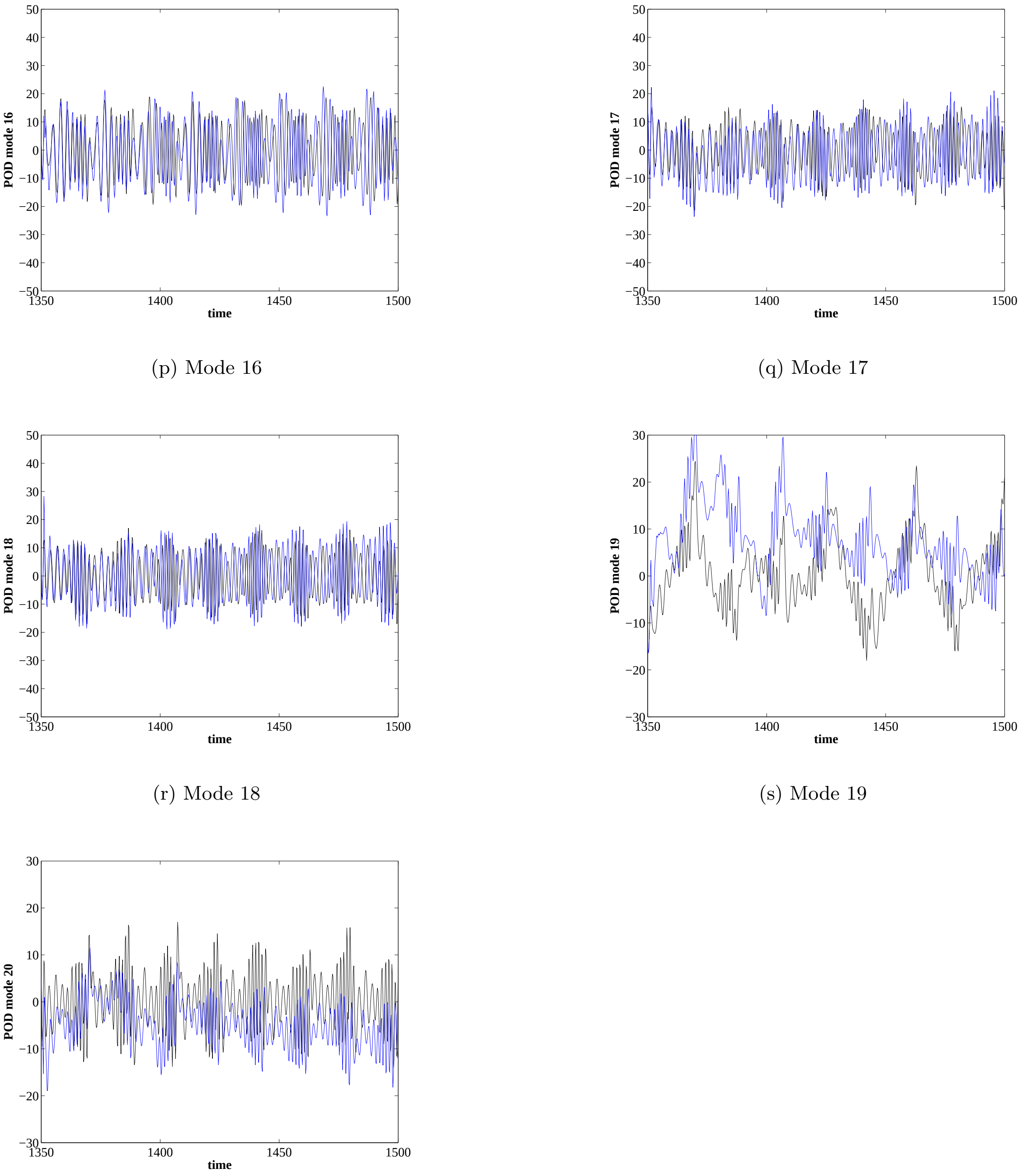
(l) Mode 12

(m) Mode 13

(n) Mode 14

(o) Mode 15

Figure 3.21: time histories of the POD mode amplitude a_i towards the beginning of coarse computation. Black = results from projected DNS computation, blue = coarse computation.



(p) Mode 16

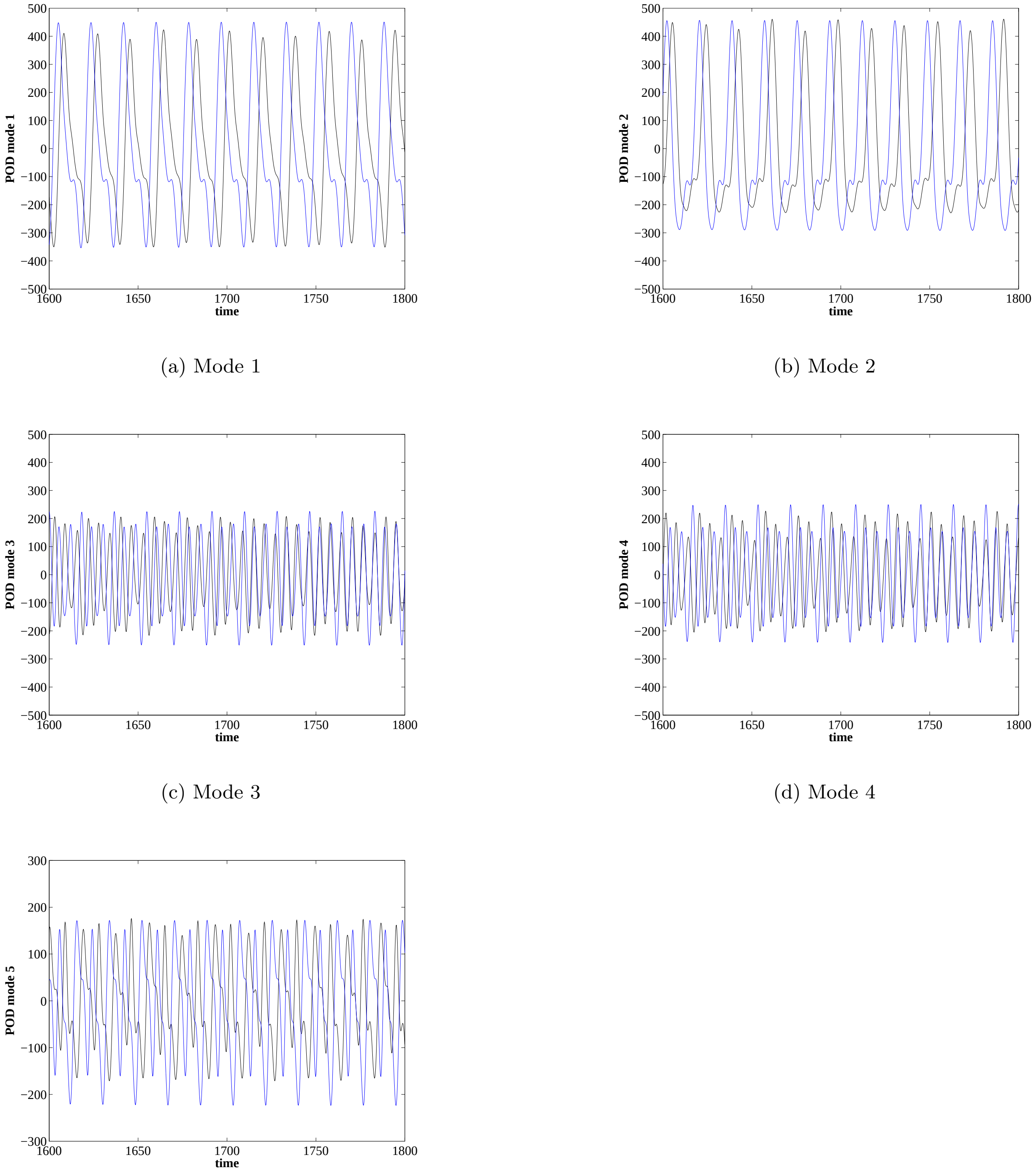
(q) Mode 17

(r) Mode 18

(s) Mode 19

(t) Mode 20

Figure 3.21: time histories of the POD mode amplitude a_i towards the beginning of coarse computation. Black = results from projected DNS computation, blue = coarse computation.



(a) Mode 1

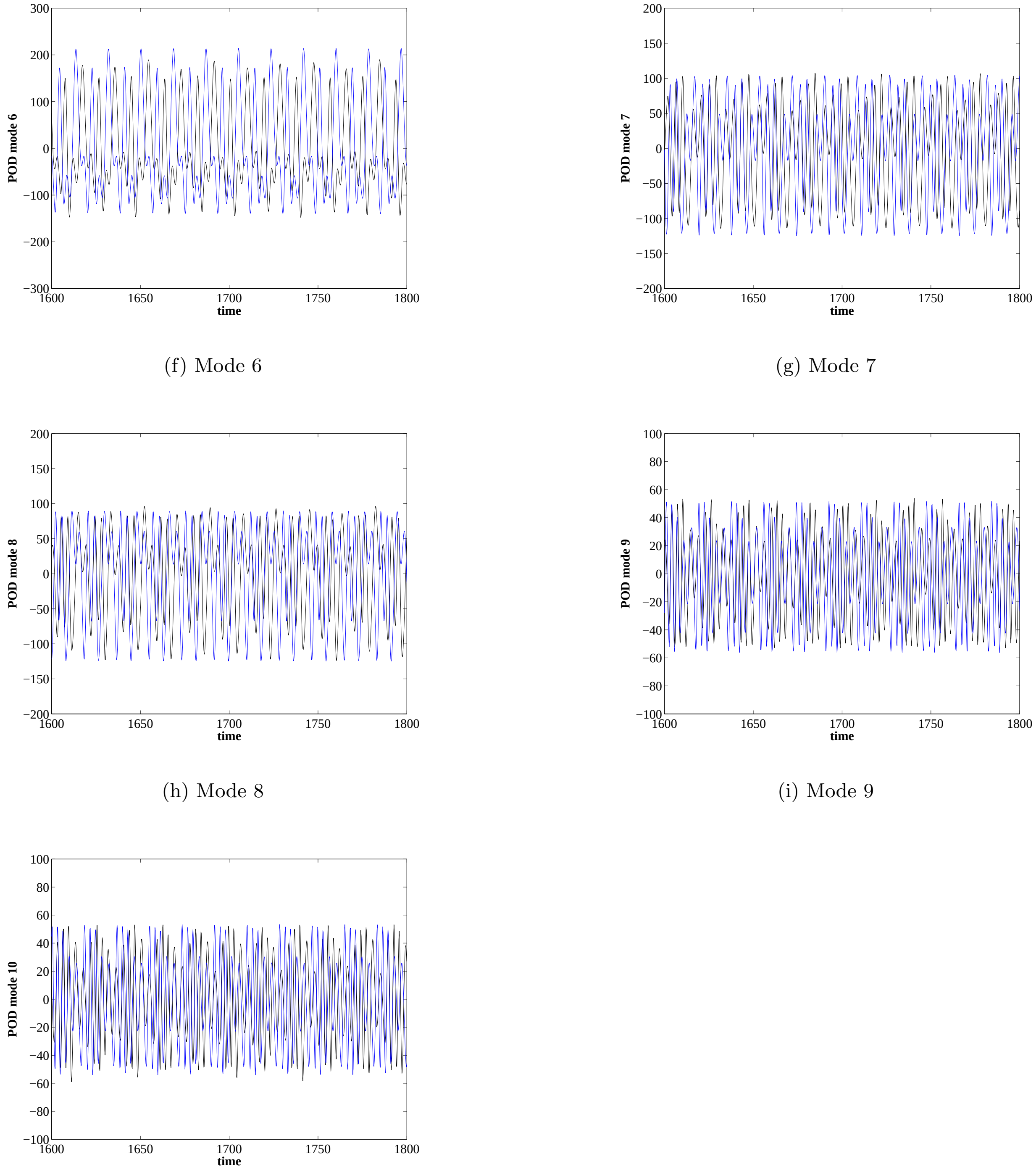
(b) Mode 2

(c) Mode 3

(d) Mode 4

(e) Mode 5

Figure 3.22: time histories of the POD mode amplitude a_i at the end of coarse computation. Black = results from projected DNS computation, blue = coarse computation.



(f) Mode 6

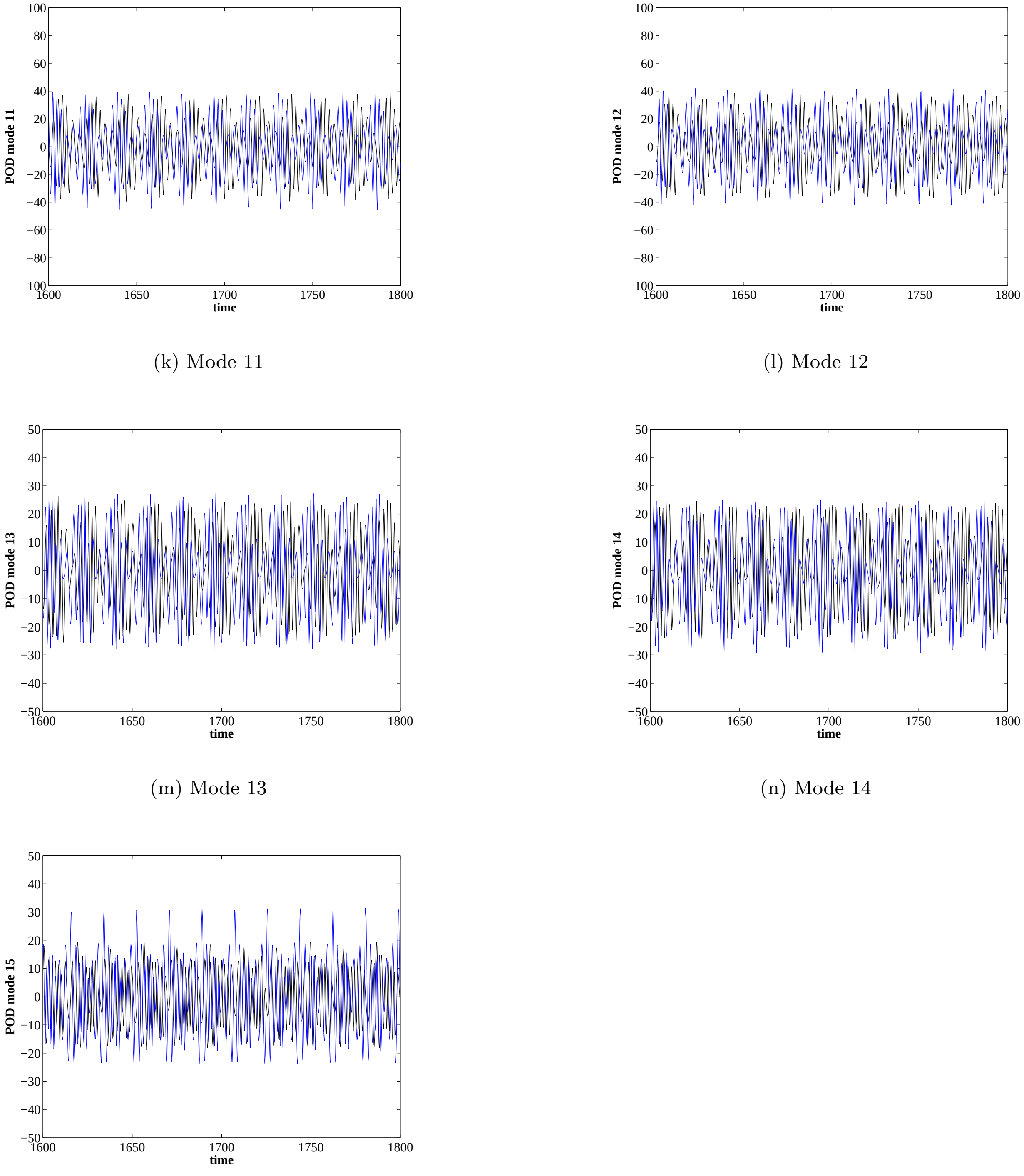
(g) Mode 7

(h) Mode 8

(i) Mode 9

(j) Mode 10

Figure 3.22: time histories of the POD mode amplitude a_i at the end of coarse computation. Black = results from projected DNS computation, blue = coarse computation.



(k) Mode 11

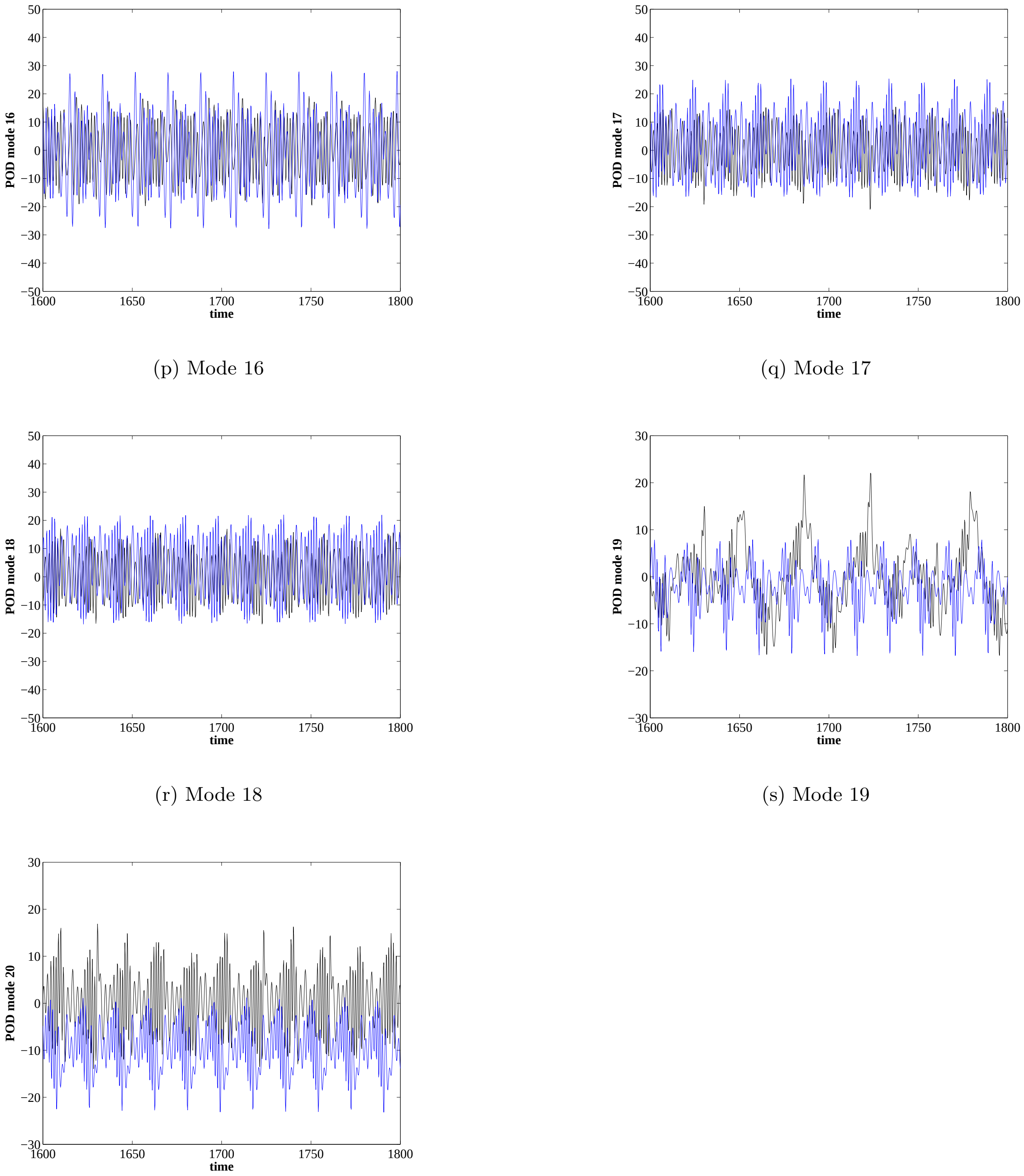
(l) Mode 12

(m) Mode 13

(n) Mode 14

(o) Mode 15

Figure 3.22: time histories of the POD mode amplitude a_i at the end of coarse computation. Black = results from projected DNS computation, blue = coarse computation.



(p) Mode 16

(q) Mode 17

(r) Mode 18

(s) Mode 19

(t) Mode 20

Figure 3.22: time histories of the POD mode amplitude a_i at the end of coarse computation. Black = results from projected DNS computation, blue = coarse computation.

a coarse third-order Adams-bashforth integrator. This led to accurate computation of the long-time system dynamics. The coarse dynamics were computed to be a compact limit cycle in phase space, which suggests that the underlying coarse dynamics are not chaotic. While the coarse computation leads to a phase shift in the temporal dynamics away from the projection of the DNS simulation, the essential features of the limit cycle are preserved. In summary, we have shown that coarse analysis represents a promising alternative for the efficient, quantitative computation of the dominant features of complex fluid flows.

It is important to note that the lifting procedure throughout the numerical simulations were used to generate only a single initial DNS initial condition at each stage in the coarse analysis. Although the lifting procedure makes an appropriate initialization of the small scales (using the mean value computed from archived simulation data), the lifting may not fully represent the statistics of the higher-order modes. In this case, lifting to several DNS initializations (and subsequent use of multiple DNS bursts) may provide a more accurate representation of the higher-order statistics and thus a more accurate coarsening.

It is also important to note that the coarse numerical analysis outlined here for fluids involves the use of the proper orthogonal decomposition for scale classification. An important comparison to make would be between coarse time integration, using POD for scale analysis, and a simulation of the Navier-Stokes equations projected onto POD modes. While the POD/Galerkin simulation represents reductions of much higher order than those offered by the coarse analysis documented here, the truncations made in the POD/Galerkin framework may yield different results than either DNS or coarse time integration. A comparison between coarse analysis and POD/Galerkin truncation might shed light on the shortcomings of the POD for scale classification and advantages/disadvantages of the two techniques. This comparison is proposed for future work, and a similar comparison is being made for the simulation of turbulence (Smith, 2005).

While this POD-based scale classification is well suited for the coarse analysis of fluids, the dependence of the POD computation on archival data raises questions on the applicability on POD–based coarse analysis to fluid flows out of the range of representation by the archival data. While this shortcoming does not affect the utility of the coarse analysis for real-time estimation and prediction of flows represented well by the archival data, improvements to the POD methodology or alternatives to the POD for scale classification would have to be made for other types of analyses.

Chapter 4

Charged Particle Motion in a Magnetic Field: Connections Among Coarse Analysis and Averaging Theories

4.1 Introduction

As we have shown in Chapter 3, coarse analysis represents a promising means of model reduction towards the extraction of system-level quantities for diffuser flows and other incompressible or compressible fluid systems. We have also shown in Chapters 2–3 that standard numerical analysis techniques may be employed to evaluate the numerical stability of coarse algorithms and to facilitate the design of coarse algorithms. However, standard numerical analysis addresses only the stability properties of coarse numerical methods. While coarse analysis techniques affect a type of averaging onto a multiscale system, it is unknown precisely how the averaging or homogenization occurs. In other words, coarse analysis tools are multiscale systems *in their own right*, and standard numerical analysis does not take this point of view. Therefore, new tools are required to determine the interaction between the coarse computational superstructure and the detailed simulation. Averaging techniques traditionally applied to physical and mechanical systems may serve to address these needs. To this end, a simpler system will be studied both from analytical and numerical points of view. The system chosen is the system of charged particles in a magnetic field. For this particular system, averaging techniques will be applied and compared with comparable techniques from coarse analysis.

This work represents a collaboration with Harish Bhat at Caltech. The numerical results presented in this thesis were performed by the author. The theoretical results presented in this thesis were derived jointly with H. Bhat, currently at Caltech. Although these results are being prepared for journal publication, their summary given in this thesis represents a revision of a preliminary draft written by the author. The original draft has subsequently undergone heavy revision by H. Bhat and the author and will be submitted soon with J. Marsden. Some of the revisions and results made after the first draft have also been incorporated here.

The organization of this chapter goes as follows. First, the detailed system, described by the Lorentz force law equations, is given in Section 4.2. Then the system is described in terms of geometric mechanics in Section 4.3. Existing averaging techniques, representing perturbation techniques applied to both the equations of motion or the underlying variational principle, are given in Section 4.4. The averaged equations are then rederived in two variational settings: one according to a WKB-style averaging procedure described in Section 4.5 and one according to a LANS-style averaging procedure described in Section 4.6. Numerical coarse integration is applied to the system and described in Section 4.7, and connections are made to perturbation theory in Section 4.8.

A connection is made to coarse analysis from a Hamiltonian setting, described in Section 4.9. Conclusions are made in Section 4.10.

4.2 Lorentz Equations

Charged particles experience forces while moving through electric and magnetic fields. In particular, for particles with charge e moving in the presence of an electric field $\mathbf{E}$ and a magnetic field $\mathbf{B}$, the particles experience a force $\mathbf{F}$ (here in MKS):

$$\begin{aligned}\mathbf{F} &= e(\mathbf{E} + \mathbf{v} \times \mathbf{B}) && (4.1)\\ &= e\left(\nabla\phi - \frac{d\mathbf{A}}{dt} + \nabla(\mathbf{A}\cdot\mathbf{v})\right), && (4.2)\end{aligned}$$

where ϕ is the electric potential, $\mathbf{A}$ the magnetic vector potential, and $\mathbf{v}$ the particle velocity. In the CGS system,

$$\mathbf{F} = e(\mathbf{E} + \frac{\mathbf{v}}{c} \times \mathbf{B}). \tag{4.3}$$

Supposing that the magnetic field is constant in time and considering a system without an electric field, the equation of motion for the particle is (in CGS),

$$\ddot{\mathbf{q}} = \frac{e}{c}\mathbf{v} \times \mathbf{B}, \tag{4.4}$$

with $\mathbf{q}$ the particle position, and $\dot{\mathbf{q}} = \mathbf{v}$. Here, $(\mathbf{q}, \mathbf{v})$ exist in $\mathbb{R}^3 \times \mathbb{R}^3$.

The action of the Lorentz force on a particle may be described as helical particle motion about the magnetic field lines. As such, the system is naturally described using two spatio-temporal scales. The small (*fine*) scale is represented by the local motion of the particle about a magnetic field line (e.g., the Larmor radius), and the large (*coarse*) scale is represented by the general magnetic field topology.

Toroidal magnetic field As an example, consider the motion of charged particles about a circular (toroidal) magnetic field. The magnetic field may be described in Cartesian coordinates with vector potential $\mathbf{A}$ and current I by

$$\begin{aligned} \mathbf{A} &= -\hat{z}\frac{I}{c}\ln(x^2+y^2) & (4.5) \\ \mathbf{B} &= \nabla\times\mathbf{A} = \frac{2I}{c(x^2+y^2)}\left(-y\hat{x}+x\hat{y}\right). & (4.6) \end{aligned}$$

Because the magnetic force on a particle is a nonlinear function of particle location, the particle motion is nonlinear. In other words, the Euler-Lagrange equations for this particular magnetic field contain $\mathbf{v}\times\mathbf{B}$ terms that are nonlinear in phase space ($\mathbf{q}$, $\dot{\mathbf{q}}$).

Particle trajectories are wrapped around the magnetic field lines, and a typical trajectory is shown in Figure 4.1. In this figure, the particle is traveling around the magnetic field line with a drift in the $-\hat{z}$ direction.

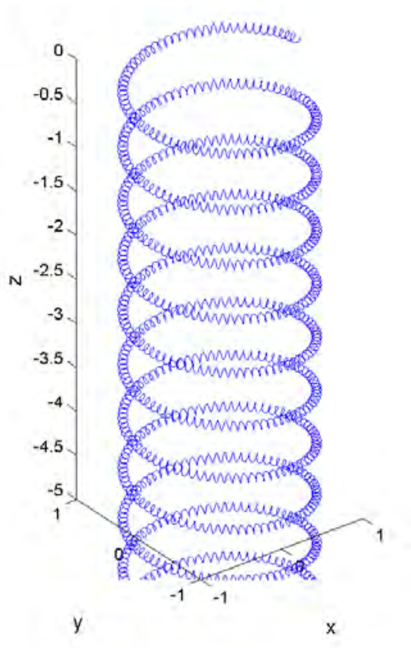


Figure 4.1: Trajectory of a single particle in a toroidal magnetic field.

Particles in constant magnetic fields move in helices Consider a system of charged particles moving under a constant magnetic field along the x-axis,

$$\mathbf{B} = \begin{pmatrix} B_x \\ 0 \\ 0 \end{pmatrix}.$$

The Lorentz force law equations are then written as

$$m\ddot{x} = 0 \tag{4.7}$$

$$m\ddot{y} = \frac{e}{c} B_x \dot{z} \tag{4.8}$$

$$m\ddot{z} = -\frac{e}{c} B_x \dot{y}. \tag{4.9}$$

Solving for $y(t)$ and $z(t)$,

$$y(t) = c_1 \cos \frac{e}{cm} B_x t + c_2 \sin \frac{e}{cm} B_x t + c_3 \tag{4.10}$$

$$y(t) = -c_1 \sin \frac{e}{cm} B_x t + c_2 \cos \frac{e}{cm} B_x t + c_4. \tag{4.11}$$

Trajectories are circles in the $y - z$ plane of radius $\sqrt{c_1^2 + c_2^2}$ with center at (x, c_3, c_4). The motion is unconstrained in the x-direction, meaning that trajectories are free to move in the x-direction. So trajectories wind around the x-axis, i.e., the motion is helical about the magnetic field lines. Thus, locally the charged particle motion is helical about the field lines.

This simple system admits to a separation of scales; upon observation of the particle trajectory in Figure 4.1, two spatial scales are apparent. As mentioned in the previous subsection, the encirclement of the particle trajectory around the magnetic field line represents the small scale particle motion. The general outline of the magnetic field line (with the drift) represents the coarse scale motion of the particle.

An objective of this research is to describe the coarse (*average*) particle motion without resorting to an exact computation of the full system or derivation of individual particle trajectories. In the following sections, coarse descriptions of the dynamics will be given in both computational and theoretical settings.

4.3 Geometric Mechanics

The geometric structure of the system of charged particles is well-known in the literature. We list many of the central results here for completion. For further insight into this system and into geometric mechanics, the reader is referred to Marsden and Ratiu (1994).

Hamiltonian setting Consider $\mathbf{q} = (x, y, z)$ and the magnetic field $\mathbf{B}$ as a closed two-form, consider canonical variables $(\mathbf{q}, \mathbf{p})$, with the momentum $\mathbf{p}$ defined as

$$\mathbf{p} = m\dot{\mathbf{q}} + \frac{e}{c}\mathbf{A},$$

and consider the positions $\mathbf{q}$ and momenta $\mathbf{p}$ defined as $\mathbf{q} = (x, y, z)$ and $\mathbf{p} = (p, q, r)$. Now, consider the Hamiltonian

$$H = \frac{1}{2m}\left\|\mathbf{p} - \frac{e}{c}\mathbf{A}\right\|^2$$

with the canonical bracket. The canonical bracket is associated with the canonical symplectic form

$$\Omega = (dx \wedge dp + dy \wedge dq + dz \wedge dr).$$

Taking the relation

$$\mathbf{d}H = \mathbf{i}_{X_H}\Omega,$$

where $X_H(p,q,r) = (\dot{x}, \dot{y}, \dot{z}, \dot{p}, \dot{q}, \dot{r})$, we naturally arrive at the Lorentz force law equations.[1]

Lagrangian Mechanics: The Variational Principle For the Hamiltonian with canonical variables, the Legendre transform may be applied to construct the Lagrangian for the system of charged particles. The relationship between the system Hamiltonian and Lagrangian is

$$L(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{p} \cdot \dot{\mathbf{q}} - H(\mathbf{q}, \mathbf{p}).$$

Recalling the relationship between the particle velocity and moments,

$$\mathbf{p} = m\dot{\mathbf{q}} + \frac{e}{c}\mathbf{A},$$

we have

$$L(\mathbf{q}, \dot{\mathbf{q}}) = (m\dot{\mathbf{q}} + \frac{e}{c}\mathbf{A}) \cdot \dot{\mathbf{q}} - \frac{1}{2m}(m\dot{\mathbf{q}})^2,$$

[1] In carrying out this derivation, it is important to remember that

$$\begin{aligned} \dot{\mathbf{p}} &= m\dot{\mathbf{v}} + \frac{e}{c}\dot{\mathbf{A}} \\ &= m\dot{\mathbf{v}} + \frac{e}{c}\frac{d\mathbf{A}}{dt} + (\mathbf{V} \cdot \nabla)\mathbf{A}). \end{aligned}$$

Using this, assuming that $\partial\mathbf{A}/\partial t = 0$, and applying vector identities will lead to the Lorentz force law equation. Alternatively, consider noncanonical variables $(\mathbf{q}, \dot{\mathbf{q}})$, with $\dot{\mathbf{q}} = (u, v, w)$. Considering the Hamiltonian as the kinetic energy,

$$H = \frac{m}{2}\left(\dot{x} + \dot{y} + \dot{z}\right),$$

with the alternative bracket

$$\Omega_{\mathbf{B}} = m\left(dx \wedge d\dot{x} + dy \wedge d\dot{y} + dz \wedge d\dot{z}\right) - \frac{e}{c}B,$$

applying the condition

$$\mathbf{d}H = \mathbf{i}_{X_H}\Omega_{\mathbf{B}},$$

where $X_H(u,v,w) = (u, v, w, \dot{u}, \dot{v}, \dot{w})$, also leads to the Lorentz force law equations. Note that this alternate derivation avoids the notekeeping associated with the time derivative of the magnetic vector potential.

and finally,

$$L(\mathbf{q}, \dot{\mathbf{q}}) = \frac{1}{2} m \dot{\mathbf{q}} \cdot \dot{\mathbf{q}} + \frac{e}{c} \mathbf{A} \cdot \dot{\mathbf{q}}.$$

With this Lagrangian and the action defined as

$$S = \int_{t_1}^{t_2} L(\mathbf{q}, \dot{\mathbf{q}}, t) dt, \tag{4.12}$$

Hamilton's principle of least action is described by

$$\delta S = \delta \int_{t_1}^{t_2} L(\mathbf{q}, \dot{\mathbf{q}}, t) dt = 0. \tag{4.13}$$

In other words, this principle states that particle trajectories should be minimizers of this action. Equivalently, these trajectories satisfy the Euler-Lagrange equations (here written in indicial notation),

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = 0. \tag{4.14}$$

With this action principle in mind, the equations of motion for a charged particle in a magnetic field may be derived using an appropriate action. The Lagrangian for this system of particles, in indicial notation, is

$$L(\mathbf{q}, \dot{\mathbf{q}}) = \frac{1}{2} m \dot{q}^i \dot{q}_i + e A^i(q) \dot{q}_i \tag{4.15}$$

with the particle mass m, charge e, and magnetic vector potential $A(\mathbf{q}, t)$. Applying the action principle to our system Lagrangian yields

$$\frac{\partial}{\partial t}\left(m \dot{q}^i + \frac{e}{c} A^i\right) - e \frac{\partial A^j}{\partial q^i} \dot{q}_j = 0 \tag{4.16}$$

so that the resulting Euler-Lagrange equations are

$$m\ddot{q}^i + \frac{e}{c}\frac{\partial A^i}{\partial q^j}\dot{q}_j - e\frac{\partial A^j}{\partial q^i}\dot{q}_j = 0. \tag{4.17}$$

These equations are equivalent to the Lorentz equations for a charged particle in a magnetic field $\mathbf{B} = \nabla \times \mathbf{A}$,

$$m\ddot{\mathbf{q}} = e\dot{\mathbf{q}} \times \mathbf{B} \tag{4.18}$$

after the application of the appropriate vector identities.[2]

4.4 Guiding Center Equations

Recall that particles in constant magnetic fields move in helices. It is appropriate to describe the charged particle motion as a multiscale system; in particular, with two distinct spatial or temporal scales. The dominant scales represent the motion along the magnetic field lines; the subdominant motion represents the helical motion about the dominant motion. It is also reasonable to seek an average or dominant-scale representation of the system itself; in fact, these representations have been developed as guiding center theories in plasma physics. Guiding center theory has been derived

[2] Derivation of the Lorentz force term in indicial notation:

$$\begin{aligned}
\frac{\partial A^i}{\partial q^l}\dot{q}_l - \frac{\partial A^m}{\partial q^i}\dot{q}_m &= \left(\delta^i_m\delta^j_l - \delta^i_l\delta^j_m\right)\frac{\partial A^m}{\partial q^l}\dot{q}_j \\
&= -\epsilon^{ij}_{k}\epsilon^{k}_{lm}\frac{\partial A^m}{\partial q^l}\dot{q}_j \\
&= -\epsilon^{ij}_{k}\dot{q}_j\left(\epsilon^{k}_{lm}\frac{\partial A^m}{\partial q^l}\right) \\
&= -\epsilon^{ij}_{k}\dot{q}_j B^k \\
&= -(\dot{\mathbf{q}} \times \mathbf{B})^i.
\end{aligned}$$

through the Lorentz force equations and through the variational principle.

Traditional setting Guiding center theory was primarily derived through perturbation theory; areview of traditional guiding center theory may be found in Northrop (1963). In the traditional theory, Taylor expansions of the Lorentz force equations and their subsequent averaging and truncation yield a set of equations that governs the average motion of the system. These equations, known as the guiding center equations, describe the motion of the *guiding center*, or the average particle position. In this theory, the particle position $\mathbf{r}$ is decomposed into the guiding center position $\mathbf{R}$ and fluctuation $\boldsymbol{\rho}$:

$$\mathbf{r} = \mathbf{R} + \boldsymbol{\rho} \tag{4.19}$$

The fluctuation $\boldsymbol{\rho}$ represents circular motion on the plane orthogonal to the local direction of the magnetic field; this circular motion is parameterized by a fluctuation radius ρ and phase angle θ corresponding to a phase velocity $\theta = \int \omega dt$. To first order in the mass-charge ratio m/e, the guiding center equation is

$$m\ddot{\mathbf{R}} = \frac{e}{c}\dot{\mathbf{R}} \times \mathbf{B}(\mathbf{R}) - \mu \nabla B(\mathbf{R}) \tag{4.20}$$

in the absence of gravitational and electric fields. In this equation, μ is defined as the magnetic moment, with

$$\mu = \frac{\boldsymbol{\rho}}{2B}, \tag{4.21}$$

with $v_\perp$ as the velocity perpendicular to the guiding center motion.

Variational setting Employing a form of Lagrangian averaging theory, Littlejohn (1983, 1984) derived such a set of coarse evolution equations for charged particle motion in magnetic fields. The small scales of the motion represent the fluctuations of the particle motion about a magnetic field line; averaging over these fluctuations isolates the guiding center motion. With a perturbation scale ϵ representing the separation of scales in the motion, the particle position $\mathbf{x}$ and velocity $\mathbf{v}$ may be decomposed into guiding center position and velocity ($\mathbf{X}$,$\mathbf{U}$) and fluctuating (or oscillating) position and velocity ($\mathbf{x_{osc}}$,$\mathbf{v_{osc}}$) as

$$\begin{aligned} \mathbf{x} &= \mathbf{X} + \epsilon \mathbf{x_{osc}} && (4.22) \\ \mathbf{v} &= \mathbf{U} + \epsilon \mathbf{v_{osc}}. && (4.23) \end{aligned}$$

Substitution of these perturbation expansions into the system Lagrangian, applying selected gauge transformations, and subsequent truncation to order ϵ terms yields the following (noncanonical) coarse Lagrangian:

$$L = \frac{1}{\epsilon}\mathbf{A}^* \cdot \dot{\mathbf{X}} + \epsilon\mu\dot{\Theta} - H. \tag{4.24}$$

The gyrophase Θ represents the fluctuating motion (phase) of the particle in a plane normal to the magnetic field line; the magnetic moment is then $\epsilon\mu = \partial L/\partial\dot{\theta}$. The modified vector potential is defined as $\mathbf{A}^* = \mathbf{A} + \epsilon U\hat{b}$, with the coarse(guiding center) velocity magnitude U and magnetic field direction $\hat{b}$. Finally, the coarse Hamiltonian H is

$$H = \mu B + \frac{1}{2}mU^2 \tag{4.25}$$

with the magnetic field strength B. These equations are noncanonical in the sense that the position and velocity are represented in the Lagrangian in a noncanonical manner.

Deriving the coarse equations of motion now corresponds to application of Hamilton's action principle to the coarse Lagrangian. The resulting guiding center equations are

$$m\dot{\mathbf{X}} = \frac{1}{B_1^*}\Big(mU\mathbf{B}^* + \epsilon\hat{b} \times (\mu\nabla B)\Big) \tag{4.26}$$

$$\dot{U} = -\frac{1}{B_1^*}\mathbf{B}^* \cdot (\mu\nabla B) \tag{4.27}$$

with the modified magnetic field and strength defined as

$$\mathbf{B}^* = \mathbf{B} + \epsilon U\nabla \times \hat{b} \tag{4.28}$$

$$B_1^* = B + \epsilon U(\hat{b} \cdot \nabla \times \hat{b}). \tag{4.29}$$

For parameters ϵ and μ, these guiding center equations govern the coarse motion for a charged particle in a magnetic field. The variationally derived guiding center equation (4.27) is consistent with the traditional guiding center equations (4.20) derived in Northrop (1963) by taking the traditional equations and projecting them in the local magnetic field direction. A form of this projected equation is listed as Eq. 1.20 in Northrop (1963).

For the same magnetic field whose individual particle trajectory is shown in Figure 4.1, a coarse trajectory is shown in Figure 4.2.

As expected, the coarse trajectory in Figure 4.2 follows the magnetic field without the small-scale

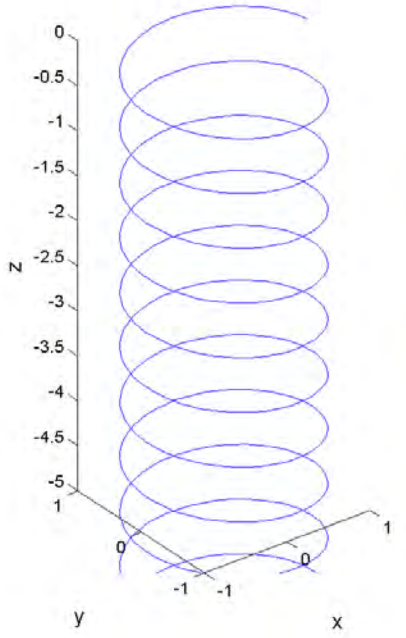


Figure 4.2: Coarse trajectory in a toroidal magnetic field, computed with the guiding center approximation.

fluctuations contained in the individual particle trajectory shown in Figure 4.1.

As documented in Littlejohn (1983, 1984), the derivation is noncanonical and relies heavily on gauge transformations. Littlejohn, in his second reference, offers a preliminary geometric interpretation of the motion and averaging process. While this treatment and derivation leads to accurate descriptions of the coarse motion, a derivation is desired that would elucidate the averaging process and the relationships among the scales in the system. In addition, making clear connections between averaging theory and coarse analysis might be complicated under a noncanonical setting. A more complete geometrical and analytical description of the averaged action principle is also desired.

4.5 Variational Averaging Inspired by WKB and Whitham Averaging

While the guiding center theory listed in the previous subsection provides an accurate representation of the averaged dynamics, little attention is paid to the exact averaging procedure and how the averaging leads to such an accurate model. This subsection seeks to rederive the averaged action principle and examine these issues.

In considering the multiscale nature of the charged particle system and the nature of the small-scale fluctuations, observe that the small-scale dynamics are essentially oscillatory motion centered around the average trajectory. Because of this, we consider two widely used averaging techniques, the WKB method and Whitham averaging, for deriving a model for the average dynamics.

WKB theory. WKB theory, named after Wentzel (1926), Kramers (1926), and Brillouin (1926), is a perturbation theory that has been successfully applied to the derivation of approximate wave solutions for differential equations such as the Schrödinger equation. In the theory, to approximate a wave function $\psi(x)$, a solution is assumed to be of the form $\psi(x) = A(x)\exp[iS(x)]$, where the amplitude $A(x)$ is slowly varying and the phase $S(x)$ is rapidly varying. This substitution is made into the differential equation to derive expressions for the amplitude and phase.

Whitham averaging. Whitham averaging techniques also seek to derive approximate solutions for problems involving wave propagation by deriving modulation equations either from the original equations of motion or from a variational principle. Whitham averaging is similar to WKB theory;

these connections are discussed in Whitham (1974).

As discussed by Whitham (1974), there are advantages of pursuing an averaging theory for the variational principle. Whitham notes in his discussion that the averaging and averaged variational principle should be unaffected in the complicating situation where the wave propagation is occurring in nonuniform media. On the other hand, these complications would have a nontrivial effect on the expansions required in applying perturbation techniques directly to the the equations of motion. Other advantages of variational averaging mentioned by Whitham include a more natural formulation of the closures required to derive averaged equations, and a natural derivation of adiabatic invariants and other quantities arising from symmetries in the system.

These techniques, although designed for approximation of solutions for wave propagation, appeal to perturbation theory and the variational principle in a very general manner. The system of charged particle motion is not a system involving wave propagation, but it is a multiscale system that appears to admit to approximation through phase averaging techniques. To this end, we generalize the two approximation theories and apply them to this system of charged particle motion.

Derivation of the guiding center equations. Consider again the original Lagrangian

$$L(\mathbf{q}, \dot{\mathbf{q}}) = \frac{1}{2} m \|\dot{\mathbf{q}}\|^2 + \frac{e}{c} \mathbf{A}(\mathbf{q}) \cdot \dot{\mathbf{q}}, \tag{4.30}$$

or in indicial notation,

$$L(\mathbf{q}, \dot{\mathbf{q}}) = \frac{1}{2} m \dot{q}^i \dot{q}_i + \frac{e}{c} A^i(\mathbf{q}) \dot{q}_i \tag{4.31}$$

where $\mathbf{q}$ is the position and $\dot{\mathbf{q}}$ is the velocity, and A is the magnetic vector potential. This potential may be thought of as a map from the underlying space to itself:

$$A : \mathbb{R}^3 \longrightarrow \mathbb{R}^3.$$

This time, consider the position $\mathbf{q}$ as having a WKB-type decomposition into a local averaged position $\mathbf{q}_{\mathrm{a}}(t)$ and a fast-varying fluctuation $\mathbf{q}_{\mathrm{f}}(t)$:

$$\mathbf{q}(t) = \mathbf{q}_{\mathrm{a}}(t) + \mathrm{Re}[\mathbf{q}_{\mathrm{f}}(t)e^{i\omega t}], \tag{4.32}$$

where $\mathbf{q}_{\mathrm{f}}(t)$ is allowed to be complex-valued. Using this decomposition, the vector potential may be written using a Taylor series expansion about the averaged position. In indicial notation, we have

$$A^i(\mathbf{q}) = A^i(\mathbf{q}_{\mathrm{a}}) + A^i_{,j}(\mathbf{q}_{\mathrm{a}})\mathrm{Re}[q^j_{\mathrm{f}}(t)e^{i\omega t}] + \frac{1}{2}A^i_{,jk}(\mathbf{q}_{\mathrm{a}})\mathrm{Re}[q^j_{\mathrm{f}}(t)e^{i\omega t}]\mathrm{Re}[q^k_{\mathrm{f}}(t)e^{i\omega t}] + \mathcal{O}(\mathrm{Re}[\mathbf{q}_{\mathrm{f}}(t)e^{i\omega t}])^3. \tag{4.33}$$

The charged particle velocity, kinetic energy, and Lorentz term may also be written using the decomposition from (4.32):

$$\dot{\mathbf{q}} = \dot{\mathbf{q}}_{\mathrm{a}}(t) + \mathrm{Re}[\dot{\mathbf{q}}_{\mathrm{f}}(t)e^{i\omega t} + i\omega\mathbf{q}_{\mathrm{f}}(t)e^{i\omega t}], \tag{4.34}$$

$$\dot{\mathbf{q}} \cdot \dot{\mathbf{q}} = \dot{\mathbf{q}}_{\mathrm{a}} \cdot \dot{\mathbf{q}}_{\mathrm{a}} + \frac{1}{2}|\dot{\mathbf{q}}_{\mathrm{f}}|^2 + \omega\mathrm{Im}[\mathbf{q}^*_{\mathrm{f}} \cdot \dot{\mathbf{q}}_{\mathrm{f}}] + \frac{1}{2}\omega^2\mathbf{q}_{\mathrm{f}} \cdot \mathbf{q}^*_{\mathrm{f}} + \mathrm{a.z}, \tag{4.35}$$

$$\mathbf{A} \cdot \dot{\mathbf{q}} = \mathbf{A} \cdot \dot{\mathbf{q}}_{\mathrm{a}} + \frac{1}{4}\dot{\mathbf{q}}_{\mathrm{a}} \cdot \left(\nabla\nabla\mathbf{A} : \mathrm{Re}\left[\mathbf{q}_{\mathrm{f}} \otimes \mathbf{q}^*_{\mathrm{f}}\right]\right) + \frac{1}{2}\mathrm{Re}\left[(\dot{\mathbf{q}}_{\mathrm{f}} + i\omega\mathbf{q}_{\mathrm{f}}) \cdot \nabla A(\mathbf{q}_{\mathrm{a}}) \cdot \mathbf{q}^*_{\mathrm{f}}\right] + \mathrm{a.z.} \tag{4.36}$$

where the term 'a.z.' refers to terms whose average over $e^{i\omega t}$ is zero. Substituting the quantities (4.32-4.34) into the Lagrangian in (4.30), averaging over $e^{i\omega t}$, and simplifying, we arrive at the *expanded Lagrangian* $l(\mathbf{q}_{\mathrm{a}}, \dot{\mathbf{q}}_{\mathrm{a}}, \mathbf{q}_{\mathrm{f}}, \dot{\mathbf{q}}_{\mathrm{f}})$:

$$\begin{aligned} l(\mathbf{q}_{\mathrm{a}}, \dot{\mathbf{q}}_{\mathrm{a}}, \mathbf{q}_{\mathrm{f}}, \dot{\mathbf{q}}_{\mathrm{f}}) \quad &= \quad \frac{e}{c}\left\{\dot{\mathbf{q}}_{\mathrm{a}} \cdot \mathbf{A}(\mathbf{q}_{\mathrm{a}}) + \frac{1}{4}\dot{\mathbf{q}}_{\mathrm{a}} \cdot \left(\nabla\nabla\mathbf{A} : \mathrm{Re}\left[\mathbf{q}_{\mathrm{f}} \otimes \mathbf{q}_{\mathrm{f}}^*\right]\right) + \frac{1}{2}\mathrm{Re}\left[(\dot{\mathbf{q}}_{\mathrm{f}} + i\omega\mathbf{q}_{\mathrm{f}}) \cdot \nabla\mathbf{A}(\mathbf{q}_{\mathrm{a}}) \cdot \mathbf{q}_{\mathrm{f}}^*\right]\right\} \\ &\quad + \frac{1}{2}m\dot{\mathbf{q}}_{\mathrm{a}}^2 + \frac{1}{4}m\dot{\mathbf{q}}_{\mathrm{f}} \cdot \dot{\mathbf{q}}_{\mathrm{f}}^* + \frac{1}{2}m\omega\mathrm{Im}[\mathbf{q}_{\mathrm{f}}^* \cdot \dot{\mathbf{q}}_{\mathrm{f}}^*] + \frac{1}{4}m\omega^2\mathbf{q}_{\mathrm{f}} \cdot \mathbf{q}_{\mathrm{f}}^*. \end{aligned} \tag{4.37}$$

We now consider two scaling assumptions:

$$|\mathbf{q}_{\mathrm{f}}| \quad \ll \quad |\mathbf{q}_{\mathrm{a}}|, \text{ and} \tag{4.38}$$

$$|\dot{\mathbf{q}}_{\mathrm{f}}| \quad \ll \quad \omega|\mathbf{q}_{\mathrm{f}}|, \tag{4.39}$$

which indicate that the average position is spatially larger than the fluctuations, and that the time variation of the fluctuations is represented in $e^{i\omega t}$ rather than in the fluctuation amplitude. Using these assumptions, we simplify the expanded Lagrangian (shown here using indicial notation):

$$\begin{aligned} l(\mathbf{q}_{\mathrm{a}}, \dot{\mathbf{q}}_{\mathrm{a}}, \mathbf{q}_{\mathrm{f}}, \dot{\mathbf{q}}_{\mathrm{f}}) \quad &= \quad \frac{e}{c}\dot{\mathbf{q}}_{\mathrm{a}} \cdot \mathbf{A}(\mathbf{q}_{\mathrm{a}}) + \frac{1}{2}\frac{e}{c}\mathrm{Re}\left[i\omega\mathbf{q}_{\mathrm{f}} \cdot \nabla\mathbf{A}(\mathbf{q}_{\mathrm{a}}) \cdot \mathbf{q}_{\mathrm{f}}^*\right] \\ &\quad + \frac{1}{2}m\dot{\mathbf{q}}_{\mathrm{a}}^2 + \frac{1}{2}m\omega\mathrm{Im}[\mathbf{q}_{\mathrm{f}}^* \cdot \dot{\mathbf{q}}_{\mathrm{f}}^*] + \frac{1}{4}m\omega^2\mathbf{q}_{\mathrm{f}} \cdot \mathbf{q}_{\mathrm{f}}^*. \end{aligned} \tag{4.40}$$

With this Lagrangian, we may proceed with computing the equations of motion according to the action principle.

Considering variations in the average position $\mathbf{q}_{\mathrm{a}}$, the corresponding Euler-Lagrange equation is

$$m\ddot{\mathbf{q}}_{\mathrm{a}} = \frac{e}{c}\dot{\mathbf{q}}_{\mathrm{a}} \times (\nabla \times \mathbf{A}) + \frac{1}{2}\omega\frac{e}{c}\mathrm{Re}[i\mathbf{q}_{\mathrm{f}} \cdot \nabla\nabla\mathbf{A} \cdot \mathbf{q}_{\mathrm{f}}^*].$$

Recognizing the magnetic field term, and applying vector algebra,[3] this equation becomes

$$m\ddot{\mathbf{q}}_{\mathrm{a}} = \frac{e}{c}\dot{\mathbf{q}}_{\mathrm{a}} \times \mathbf{B} - \frac{1}{4}\omega\frac{e}{c}\nabla\mathbf{B} \cdot \mathrm{Im}[\mathbf{q}_{\mathrm{f}}^* \times \mathbf{q}_{\mathrm{f}}]. \tag{4.41}$$

The differential equation in (4.41) represents the equation of motion for the average particle position. However, the equation is not closed in the sense that a dependence on the fluctuations remains. To examine this dependence, we derive the corresponding Euler-Lagrange equations by considering variations in the fluctuations $\mathbf{q}_{\mathrm{f}}$:

$$0 = \frac{1}{2}m\omega\mathbf{q}_{\mathrm{f}} \cdot \mathbf{q}_{\mathrm{f}}^* - \frac{1}{2}\frac{e}{c}\mathbf{B} \cdot \mathrm{Im}[\mathbf{q}_{\mathrm{f}}^* \times \mathbf{q}_{\mathrm{f}}]. \tag{4.42}$$

The *magnetic moment* of the charged particle may be recognized as

$$\mu = \frac{1}{4}m\omega^2\|\mathbf{B}\|^{-1}\mathbf{q}_{\mathrm{f}} \cdot \mathbf{q}_{\mathrm{f}}^*. \tag{4.43}$$

[3] For example, using indicial notation, and assuming that the complex-valued fluctuation has a representation $\mathbf{q}_{\mathrm{f}} = \mathbf{x}_{\mathrm{f}} + i\mathbf{y}_{\mathrm{f}}$, we may rewrite such quantities $\mathrm{Re}[i\mathbf{q}_{\mathrm{f}} \cdot \nabla\nabla A \cdot \mathbf{q}_{\mathrm{f}}^*]$ as

$$\begin{aligned}
\mathrm{Re}[i\mathbf{q}_{\mathrm{f}} \cdot \nabla\nabla\mathbf{A} \cdot \mathbf{q}_{\mathrm{f}}^*] &= -\mathrm{Im}[\mathbf{q}_{\mathrm{f}} \cdot \nabla\nabla\mathbf{A} \cdot \mathbf{q}_{\mathrm{f}}^*] \\
&= -\mathrm{Im}[q_{\mathrm{f}}^l A^l_{,jk} q_{\mathrm{f}}^{*k}] \\
&= -A^l_{,jk}(\epsilon^{kjn}\epsilon^{nrs}x_{\mathrm{f}}^r y_{\mathrm{f}}^s) \\
&= -\nabla\mathbf{B} \cdot \frac{1}{2}\mathrm{Im}[\mathbf{q}_{\mathrm{f}} \times \mathbf{q}_{\mathrm{f}} + \mathbf{q}_{\mathrm{f}}^* \times \mathbf{q}_{\mathrm{f}}].
\end{aligned}$$

With $\mathbf{b} = \mathbf{B}/\|\mathbf{B}\|$ the unit vector in the direction of the magnetic field, we may rewrite Eq. (4.42) as

$$\mu = \frac{\omega e}{4c}\mathbf{b} \cdot \mathrm{Im}\big[\mathbf{q}_\mathrm{f}^* \times \mathbf{q}_\mathrm{f}\big]. \tag{4.44}$$

One can show geometrically that μ is constant in time along any trajectory, and is therefore an integral of motion. Using Eq. (4.44), Eq. (4.41) may be closed, yielding a single equation for the average position:[4]

$$m\ddot{\mathbf{q}}_\mathrm{a} = \frac{e}{c}\dot{\mathbf{q}}_\mathrm{a} \times \mathbf{B} - \mu\nabla\|\mathbf{B}\|. \tag{4.47}$$

This is equivalent to the traditional guiding center equation (4.20). In fact, this represents a new, canonical formulation that leads from the Lagrangian for a charged particle in a magnetic field to the guiding center equation, following a WKB/Whitham-style average at the level of the

[4] To see this, substitute $\mathbf{B} = \|\mathbf{B}\|\mathbf{b}$ in the last term of (4.41) and carry out the derivative:

$$\mathrm{Im}\big[\mathbf{q}_\mathrm{f}^* \times \mathbf{q}_\mathrm{f}\big] \cdot \nabla\mathbf{B} = \nabla\|\mathbf{B}\|\Big(\mathbf{b} \cdot \mathrm{Im}\big[\mathbf{q}_\mathrm{f}^* \times \mathbf{q}_\mathrm{f}\big]\Big) + \|\mathbf{B}\|\mathrm{Im}\big[\mathbf{q}_\mathrm{f}^* \times \mathbf{q}_\mathrm{f}\big] \cdot \nabla\mathbf{b} \tag{4.45}$$

We now show that

$$\mathrm{Im}\big[\mathbf{q}_\mathrm{f}^* \times \mathbf{q}_\mathrm{f}\big] \cdot \nabla\mathbf{b} = 0. \tag{4.46}$$

Writing $\mathbf{q}_\mathrm{f} = \mathbf{x}_\mathrm{f} + i\mathbf{y}_\mathrm{f}$, we find that

$$\frac{1}{2}\mathrm{Im}\big[\mathbf{q}_\mathrm{f}^* \times \mathbf{q}_\mathrm{f}\big] = \mathbf{x}_\mathrm{f} \times \mathbf{y}_\mathrm{f}.$$

From $\mathrm{Re}\big[\mathbf{q}_\mathrm{f}(t)e^{i\omega t}\big] = \mathbf{x}_\mathrm{f}\cos\omega t - \mathbf{y}_\mathrm{f}\sin\omega t$, it is clear that $\mathbf{x}_\mathrm{f}$ and $\mathbf{y}_\mathrm{f}$ are, respectively, the cos and sin components of the fluctuation. Going back to the decomposition (4.32), we know *a priori* that the total fluctuation $\mathbf{q} - \mathbf{q}_\mathrm{a}$ is in the plane orthogonal to $\mathbf{b}$. This must hold for the components $\mathbf{x}_\mathrm{f}$ and $\mathbf{y}_\mathrm{f}$ as well, implying $\mathbf{x}_\mathrm{f} \times \mathbf{y}_\mathrm{f} \parallel \mathbf{b}$. Finally, $\|\mathbf{b}\| = 1$ implies

$$\mathbf{b} \cdot \nabla\mathbf{b} = \frac{1}{2}\nabla\|\mathbf{b}\|^2 = 0,$$

and (4.46) follows immediately. Using this, (4.45) becomes

$$\mathrm{Im}\big[\mathbf{q}_\mathrm{f}^* \times \mathbf{q}_\mathrm{f}\big] \cdot \nabla\mathbf{B} = \nabla\|\mathbf{B}\|\Big(\mathbf{b} \cdot \mathrm{Im}\big[\mathbf{q}_\mathrm{f}^* \times \mathbf{q}_\mathrm{f}\big]\Big).$$

Substituting (4.44), we are left with

$$\frac{\omega e}{2c}\mathrm{Im}\big[\mathbf{q}_\mathrm{f}^* \times \mathbf{q}_\mathrm{f}\big] \cdot \nabla\mathbf{B} = \mu\nabla\|\mathbf{B}\|,$$

proving that (4.41) implies (4.47).

Lagrangian. Note that the averaged Lagrangian is dependent on both large and small scales $\mathbf{q}_{\mathrm{a}}$ and $\mathbf{q}_{\mathrm{f}}$, respectively. In any multiscale theory, there needs to be some closure that relates these scales to each other. In this procedure, as in Whitham averaging, the closure rule is provided by the Euler-Lagrange equation for the fluctuations. This feature may hold for other multiscale systems.

Note. In order to compare this method with that of Littlejohn, suppose we examine Eq. (4.40) in light of the Euler-Lagrange equation Eq. (4.42). We find that all terms involving the fluctuations disappear, because

$$\frac{\omega e}{2c}\mathrm{Re}\left[i\mathbf{q}_{\mathrm{f}}\cdot\nabla\mathbf{A}(\mathbf{q}_{\mathrm{a}})\cdot\mathbf{q}_{\mathrm{f}}^{*}\right]=-\frac{\omega e}{4c}\mathbf{B}\cdot\mathrm{Im}\big[\mathbf{q}_{\mathrm{f}}^{*}\times\mathbf{q}_{\mathrm{f}}\big]=-\frac{1}{4}m\omega^{2}\mathbf{q}_{\mathrm{f}}\cdot\mathbf{q}_{\mathrm{f}}^{*}.$$

Then, it would appear that the Lagrangian (4.40) boils down to the original Lagrangian

$$l(\mathbf{q}_{\mathrm{a}},\dot{\mathbf{q}}_{\mathrm{a}})=\frac{1}{2}m\dot{\mathbf{q}}_{\mathrm{a}}\cdot\dot{\mathbf{q}}_{\mathrm{a}}+\frac{e}{c}\dot{\mathbf{q}}_{\mathrm{a}}\cdot\mathbf{A}(\mathbf{q}_{\mathrm{a}}). \tag{4.48}$$

This is different from Littlejohn's guiding center Lagrangian. However, the substitution of either conserved quantities or equations of motion into the Lagrangian is inappropriate.

Additional note. Applying our scaling assumptions to Eq. (4.37), we removed the term that involves $\mathrm{Im}\big[\mathbf{q}_{\mathrm{f}}^{*}\cdot\dot{\mathbf{q}}_{\mathrm{f}}^{*}\big]$. Had we retained this term, it would not have affected the Euler-Lagrange equation anyway, since

$$\frac{1}{2}m\omega\mathrm{Im}\left[\mathbf{q}_{\mathrm{f}}^{*}\cdot\dot{\mathbf{q}}_{\mathrm{f}}^{*}\right]=\frac{1}{4}m\omega\frac{d}{dt}\mathrm{Im}\left[\mathbf{q}_{\mathrm{f}}^{*}\cdot\mathbf{q}_{\mathrm{f}}^{*}\right]. \tag{4.49}$$

Hence, when considering the action

$$S = \int L \, dt,$$

this term will not contribute.

4.6 Variational Averaging Inspired by LANS Averaging

Another method for averaging the action principle is found in the LAE/LANS-α theory for fluid mechanics. The theory, also in the spirit of other Lagrangian-averaging procedures such as the guiding center theory, is very systematic in its treatment of the averaging procedure and in its geometric interpretation. The reader is encouraged to refer to Holm et al. (1998) and Marsden and Shkoller (2001) for a particular flavor of the theory and to Bhat et al. (2003) for its systematic treatment and for extensions to compressible fluids.

Consider the unaveraged Lagrangian,

$$L(\mathbf{q}, \dot{\mathbf{q}}) = \frac{1}{2} m \|\dot{\mathbf{q}}\|^2 + \frac{e}{c} \mathbf{A}(\mathbf{q}) \cdot \dot{\mathbf{q}} \tag{4.50}$$

or in indicial notation,

$$L(\mathbf{q}, \dot{\mathbf{q}}) = \frac{1}{2} m \dot{q}^i \dot{q}_i + \frac{e}{c} A^i(\mathbf{q}) \dot{q}_i, \tag{4.51}$$

where $\mathbf{q}$ is the position and $\dot{\mathbf{q}}$ is the velocity, and A is the magnetic vector potential.

Now consider the position and velocity as fluctuations from an original position and velocity; the original position and velocity are assumed to satisfy the system Euler-Lagrange equations. In that case, consider the unaveraged Lagrangian

$$L(\mathbf{q}^\epsilon, \dot{\mathbf{q}}^\epsilon) = \frac{1}{2} m \|\dot{\mathbf{q}}^\epsilon\|^2 + \frac{e}{c} \mathbf{A}(\mathbf{q}^\epsilon) \cdot \dot{\mathbf{q}}^\epsilon \tag{4.52}$$

or in indicial notation,

$$L(\mathbf{q}^\epsilon, \dot{\mathbf{q}}^\epsilon) = \frac{1}{2} (\dot{q}^\epsilon)^i (\dot{q}^\epsilon)_i + A^i(\mathbf{q}^\epsilon)(\dot{q}^\epsilon)_i, \tag{4.53}$$

where the new position $\mathbf{q}^\epsilon$ is composed of the original position $\mathbf{q}$ and a fluctuation map ξ^ϵ,

$$\mathbf{q}^\epsilon = \xi^\epsilon(\mathbf{q}). \tag{4.54}$$

Now consider a perturbation expansion of both the new position and velocity; that is,

$$\mathbf{q}^\epsilon = \mathbf{q} + \epsilon \mathbf{q}' + \epsilon^2 \mathbf{q}'' + \ldots \tag{4.55}$$

where the perturbation quantities are defined as

$$\mathbf{q}' = \left. \frac{\partial}{\partial \epsilon} \right|_{\epsilon=0} \mathbf{q}^\epsilon \tag{4.56}$$

and so on. The corresponding expansion for the time derivative is

$$\dot{\mathbf{q}}^\epsilon = \frac{\partial}{\partial t} \mathbf{q}^\epsilon, \tag{4.57}$$

which leads to the following perturbation series for $\dot{\mathbf{q}}^\epsilon$,

$$\dot{\mathbf{q}}^\epsilon = \dot{\mathbf{q}} + \epsilon\dot{\mathbf{q}}' + \epsilon^2\ddot{\mathbf{q}}'' + \dots \tag{4.58}$$

where the perturbation quantities are defined as

$$\dot{\mathbf{q}}' = \frac{\partial}{\partial t}\bigg|_{\epsilon=0}\frac{\partial}{\partial\epsilon}\dot{\mathbf{q}}^\epsilon \tag{4.59}$$

and so on.

We now seek to recast the unaveraged Lagrangian in terms of these perturbation quantities. This is done through Taylor expansion of the two terms in the unaveraged Lagrangian. First of all, we have for the kinetic energy

$$\begin{aligned}\frac{1}{2}(\dot{q}^\epsilon)^i(\dot{q}^\epsilon)_i &= \frac{1}{2}\left(\dot{q}^i + \epsilon\dot{q}'^i + \frac{1}{2}\epsilon^2\dot{q}''^i + \dots\right)\left(\dot{q}_i + \epsilon\dot{q}'_i + \frac{1}{2}\epsilon^2\dot{q}''_i + \dots\right)\\ &= \frac{1}{2}\dot{q}^i\dot{q}_i + \epsilon\dot{q}^i\dot{q}'_i + \epsilon^2\left(\dot{q}'^i\dot{q}'_i + \dot{q}''^i\dot{q}_i\right) + \dots \end{aligned}\tag{4.60}$$

and before we examine the second term, we first recast the magnetic vector potential $\mathbf{A}(\mathbf{q}^\epsilon)$:

$$\begin{aligned}A^i(\mathbf{q}^\epsilon) &= A^i(\mathbf{q}) + \epsilon\frac{\partial}{\partial\epsilon}\bigg|_{\epsilon=0}A^i(\mathbf{q}^\epsilon) + \frac{1}{2}\epsilon^2\frac{\partial^2}{\partial\epsilon^2}\bigg|_{\epsilon=0}A^i(\mathbf{q}^\epsilon) + \dots\\ &= A^i(\mathbf{q}) + \epsilon\frac{\partial A^i}{\partial q^j}\frac{\partial}{\partial\epsilon}\bigg|_{\epsilon=0}(q^\epsilon)_j\\ &\quad + \frac{1}{2}\epsilon^2\left(\frac{\partial^2 A^i}{\partial(q^\epsilon)^j\partial(q^\epsilon)^k}(\mathbf{q}^\epsilon)\frac{\partial(q^\epsilon)^j}{\partial\epsilon}\frac{\partial(q^\epsilon)^k}{\partial\epsilon} + \frac{\partial A^i}{\partial(q^\epsilon)^j}(\mathbf{q}^\epsilon)\frac{\partial^2(q^\epsilon)^j}{\partial\epsilon^2}\right)\bigg|_{\epsilon=0} + \dots\\ &= A^i(\mathbf{q}) + \epsilon\frac{\partial A^i}{\partial q^j}q'^j + \frac{1}{2}\epsilon^2\left(\frac{\partial^2 A^i}{\partial q^j\partial q^k}(\mathbf{q})q'^j q'^k + \frac{\partial A^i}{\partial q^j}(\mathbf{q})q''^j\right) + \dots. \end{aligned}\tag{4.61}$$

Then, the second term in the unaveraged Lagrangian becomes

$$\begin{aligned} A^i(\mathbf{q}^\epsilon)(\dot{q}^\epsilon)_i &= A^i(\mathbf{q})\dot{q}_i + \epsilon\left(A^i(\mathbf{q})\dot{q}'_i + \frac{\partial A^i}{\partial q^j}(\mathbf{q})q'^j\dot{q}_i\right) + \epsilon^2\left(\frac{\partial A^i}{\partial q^j}(\mathbf{q})q'^j\dot{q}'_i + \frac{1}{2}A^i\dot{q}''_i\right. \\ &\left. +\frac{1}{2}\frac{\partial^2 A^i}{\partial q^j \partial q^k}(\mathbf{q})q'^j q'^k\dot{q}_i + \frac{1}{2}\frac{\partial A^i}{\partial q^j}(\mathbf{q})q''^j\dot{q}_i\right) + \ldots \end{aligned} \tag{4.62}$$

and so the unaveraged Lagrangian becomes

$$\begin{aligned} L(\mathbf{q}^\epsilon, \dot{\mathbf{q}}^\epsilon) &= \frac{1}{2}m\dot{q}^i\dot{q}_i + \epsilon m\dot{q}^i\dot{q}'_i + \epsilon^2 m\left(\dot{q}'^i\dot{q}'_i + \dot{q}''^i\dot{q}_i\right) + \ldots \\ &+\frac{e}{c}A^i(\mathbf{q})\dot{q}_i + \epsilon\frac{e}{c}\left(A^i(\mathbf{q})\dot{q}'_i + \frac{\partial A^i}{\partial q^j}(\mathbf{q})q'^j\dot{q}_i\right) + \epsilon^2\frac{e}{c}\left(\frac{\partial A^i}{\partial q^j}(\mathbf{q})q'^j\dot{q}'_i\right. \\ &\left. +\frac{1}{2}A^i\dot{q}''_i + \frac{1}{2}\frac{\partial^2 A^i}{\partial q^j \partial q^k}(\mathbf{q})q'^j q'^k\dot{q}_i + \frac{1}{2}\frac{\partial A^i}{\partial q^j}(\mathbf{q})q''^j\dot{q}_i\right) + \ldots. \end{aligned} \tag{4.63}$$

The next step in the derivation is to compute the averaged Lagrangian. The averaged Lagrangian is precisely the ϵ-average,

$$l(\mathbf{q}, \dot{\mathbf{q}}; \alpha) = \langle L(\mathbf{q}^\epsilon, \dot{\mathbf{q}}^\epsilon)\rangle, \tag{4.64}$$

where the new parameter α arises from the averaging operator; see Bhat et al. (2003) for a description of such an averaging. It is in the averaging that assumptions are made concerning the relationship between the coarse and fine scales of the system. For example, valid assumptions for the fluctuation (fine-scale) statistics are zero-mean assumptions:

$$\langle \mathbf{q}' \rangle = 0 \tag{4.65}$$

$$\langle \mathbf{q}'' \rangle = 0 \tag{4.66}$$

$$\langle \dot{\mathbf{q}}' \rangle = 0 \tag{4.67}$$

$$\langle \dot{\mathbf{q}}'' \rangle = 0. \tag{4.68}$$

With the unaveraged Lagrangian in (4.63), terms such as $\langle \dot{q}'^i \dot{q}'_i \rangle$ and $\langle \frac{\partial^2 A^i}{\partial q^j \partial q^k}(\mathbf{q}) q'^j q'^k \dot{q}_i \rangle$ will require treatment. Symmetry and physical (phenomenological) arguments should play a role in modeling assumptions. Note that many of these questions have their analogues in the coarse integration context, and some of the answers may be provided by a concurrent investigation of these issues.

With the assumptions for the fluctuation statistics mentioned above in Eqs. (4.65-4.68), several of the cross-correlation quantities are also zero. Furthermore, an assumption is made that the following quadratic averages are constant over the configuration space:

$$\langle \dot{q}'^i \dot{q}'_i \rangle = G \tag{4.69}$$

$$\langle \frac{\partial^2 A^i}{\partial q^j \partial q^k}(\mathbf{q}) q'^j q'^k \dot{q}_i \rangle = \frac{\partial^2 A^i}{\partial q^j \partial q^k}(\mathbf{q}) F^{jk} \dot{q}_i \tag{4.70}$$

$$\langle \frac{\partial A^i}{\partial q^j}(\mathbf{q}) q'^j \dot{q}'_i \rangle = \frac{\partial A^i}{\partial q^j}(\mathbf{q}) H^j_i. \tag{4.71}$$

The resulting averaged Lagrangian to second order in ϵ is

$$l(\mathbf{q}, \dot{\mathbf{q}}; \epsilon) = \frac{1}{2} m \dot{q}^i \dot{q}_i + \frac{e}{c} A^i(\mathbf{q}) \dot{q}_i + \epsilon^2 \frac{e}{c} \left(G + \frac{1}{2} \frac{\partial^2 A^i}{\partial q^j \partial q^k}(\mathbf{q}) F^{jk} \dot{q}_i + \frac{\partial A^i}{\partial q^j}(\mathbf{q}) H^j_i \right). \tag{4.72}$$

The coarse action is now defined as

$$\hat{S} = \int_{t_1}^{t_2} l(\mathbf{q}, \dot{\mathbf{q}}; \epsilon) dt. \tag{4.73}$$

Computing Hamilton's principle should yield the average equations for the particle motion.

In computing this action principle, we observe that the constant G will not influence the variations (and therefore will not manifest itself in the equations of motion). Rewriting the averaged Lagrangian as

$$l(\mathbf{q}, \dot{\mathbf{q}}; \epsilon) \;=\; \frac{1}{2} m \dot{q}^i \dot{q}_i + \frac{e}{c}\left(A^i(\mathbf{q}) + \frac{1}{2}\epsilon^2 \frac{\partial^2 A^i}{\partial q^j \partial q^k}(\mathbf{q}) F^{jk} \right) \dot{q}_i + \epsilon^2 \frac{e}{c} \frac{\partial A^i}{\partial q^j}(\mathbf{q}) H_i^j, \tag{4.74}$$

it becomes apparent that this averaged Lagrangian differs from the original unaveraged Lagrangian by two terms (to second order in ϵ). These terms correspond to terms that appear in the WKB/Whitham-style averaged Lagrangian listed in Eq. (4.37).

The tensor F corresponds to the tensor $\mathrm{Re}\left[\mathbf{q}_{\mathrm{f}} \otimes \mathbf{q}_{\mathrm{f}}^*\right]$ in Eq. (4.37). Both yield corrections to the magnetic potential $\mathbf{A}$. Note that in the WKB/Whitham-style derivation, this term is ignored once the scaling assumptions in Eqs. (4.38-4.39) are applied.

The tensor H corresponds to the tensor that is implicitly multiplying $\nabla \mathbf{A}(\mathbf{q}_{\mathrm{a}})$ in Eq. (4.40). In the WKB/Whitham-style derivation, this term survives the scaling assumptions.

Defining the modified magnetic potential $\tilde{\mathbf{A}}$ by

$$\tilde{A}^i = A^i + \frac{1}{2}\epsilon^2 \frac{\partial^2 A^i}{\partial q^j \partial q^k} F^{jk} \tag{4.75}$$

and comparing the LANS-style Lagrangian in Eq. (4.74) with that of Littlejohn, it is clear that if

$$\epsilon^2 \frac{e}{c} \frac{\partial A^i}{\partial q^j} H_i^j = -\mu \|\mathbf{B}\|, \tag{4.76}$$

where the magnetic moment is defined by

$$\mu = \frac{1}{2} m \alpha^{-2} \|\mathbf{B}\|^{-1} \|\mathbf{q}'\|^2, \tag{4.77}$$

then the averaged Lagrangian listed in Eq. (4.74) will become

$$l(\mathbf{q}, \dot{\mathbf{q}}; \alpha) = \frac{1}{2} m \dot{\mathbf{q}} \cdot \dot{\mathbf{q}} + \frac{e}{c} \mathbf{A}(\mathbf{q}) \cdot \dot{\mathbf{q}} - \mu \|\mathbf{B}\|.$$

Determination of H. A final task is to define a tensor H that satisfies Eq. (4.76). To this end, we examine the fluctuation equations from the WKB/Whitham-style derivation, keeping in mind that in the LANS-style derivation, the fluctuations $\mathbf{q}'$ are real-valued. Examining the tensor that implicitly multiplies $\nabla \mathbf{A}(\mathbf{q}_{\mathrm{a}})$ in Eq. (4.40), observe that in the WKB/Whitham context,

$$\mathrm{Re}\big[i \mathbf{q}_{\mathrm{f}} \cdot \nabla \mathbf{A} \cdot \mathbf{q}_{\mathrm{f}}^* \big] = -A^l_{,k} \epsilon^{kln} \epsilon^{nrs} x_{\mathrm{f}}^r y_{\mathrm{f}}^s, \tag{4.78}$$

where $\mathbf{q}_{\mathrm{f}} = \mathbf{x}_{\mathrm{f}} + i\mathbf{y}_{\mathrm{f}}$. Comparing this with Eq. (4.76), we write

$$H_l^k = \beta \epsilon^{kln} \epsilon^{nrs} (q')^r (\dot{q}')^s, \tag{4.79}$$

where β is a quantity that will be determined shortly. The idea behind this choice is that in the WKB/Whitham context, $\mathbf{x}_{\mathrm{f}}$ and $\mathbf{y}_{\mathrm{f}}$ are, respectively, the cos and sin components of the fluctuation. The analogous terms in the LANS contexts are, respectively, $\mathbf{q}'$ and $\dot{\mathbf{q}}'$. Note that this is compatible with the definition of H in Eq. (4.76). Substituting Eq. (4.79) into Eq. (4.76), we derive a condition on μ:

$$\epsilon^2 \frac{e}{c} \beta \mathbf{b} \cdot \left(\mathbf{q}' \times \dot{\mathbf{q}}' \right) = -\mu.$$

For the particle in a magnetic field, we know that $\mathbf{q}' \times \dot{\mathbf{q}}'$ is parallel to $\mathbf{b}$. We further assume that $\|\dot{\mathbf{q}}'\| \approx \epsilon^{-1} \|\mathbf{q}'\|$. Finally, it can be shown that

$$\beta = \frac{mc}{2e\epsilon} \|\mathbf{B}\|^{-1}. \tag{4.80}$$

Using this formula for β, we may simplify Eq.(4.79) and rewrite it as

$$H = 2\beta \, \mathrm{Skew} \left(\mathbf{q}' \otimes \dot{\mathbf{q}}' \right). \tag{4.81}$$

With these definitions of H and β, Eq. (4.74) becomes another canonical version of the guiding center Lagrangian.

The guiding center equation. Computing the Euler-Lagrange equation for this Lagrangian (4.74) with H and β yields the following guiding center equation for the modified magnetic field:

$$m\ddot{\mathbf{q}} = e\dot{\mathbf{q}} \times \tilde{\mathbf{B}} - \mu\nabla\|\mathbf{B}\|, \tag{4.82}$$

where $\tilde{\mathbf{B}} = \nabla \times \tilde{\mathbf{A}}$. In case we choose to ignore the order α^2 correction to the magnetic field and use $\mathbf{B}$ instead of $\tilde{\mathbf{B}}$, then Eq. (4.82) is equivalent to the traditional guiding center equation (4.20). It is important to note that this LANS-inspired derivation, although canonical and appropriate for averaging of multiscale mechanical systems, is very general in implementation. To close the resulting equations, modeling assumptions must be made that represent the phenomenology of the system (in this case, using the notion of the magnetic moment).

4.7 Coarse Timestepping

A prototypical coarse integration was performed for our system of charged particles in a magnetic field. These results are shown in Figure 4.3. For this coarse integration, the computational superstructure is the trivial one: no extrapolative/projective method was used to advance the integration for coarse times. Instead, the algorithm repeated the process of lifting, flow evolution, and averaging. Projection is not required in this case because the system dimension for both the coarse and detailed variables are the same ($\mathbb{R}^3$). The lifting in this case is to multiple particles over a Gaussian distribution of positions and velocities about the average; averaging is ensemble averaging.

As expected, the coarse integrator was able to generate a coarse trajectory without demanding

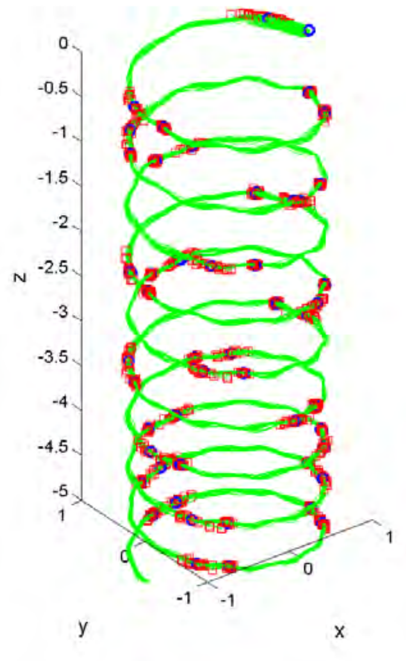


Figure 4.3: Coarse trajectory in a toroidal magnetic field, generated using coarse integration techniques. Shown in red squares and green lines are the particle initial conditions and trajectories, respectively. Shown in blue squares are averaged coarse positions.

explicit knowledge of the underlying dynamics from the user. Because this is a small system of ordinary differential equations, and because no projective time integration scheme was used, there are no computational savings in using the coarse integrator. However, this computational superstructure is scalable for integrating large systems at a lower computational cost. The coarse integration research in this area is continuing to be pursued at Caltech.

Observe that this coarse integrator was able to compute the coarse dynamics for a Hamiltonian system. This is an interesting result that shows that the computational superstructure may apply to systems other than those whose scale separation is caused by dissipation of some kind.

The framework listed in this section raises many questions concerning convergence and accuracy, algorithm philosophy, and component design. Convergence to coarse trajectories or dynamics is an underlying question not only for these algorithms but also for the averaging philosophy. In the computational setting, convergence parameters are the various algorithm parameters (the lift/project/averaging parameters, as well as the coarse and fine integrator parameters).

The algorithm presumes upon the averaging of trajectories in constructing some sort of a coarse trajectory. This may or may not be valid depending on the types of trajectories and averaging and on the underlying structure of the dynamical system. A differential-geometric perspective, with symmetry considerations, may provide the necessary interpretation for a proper construction of a coarsening algorithm and its corresponding components.

4.8 Lifting and Averaging

Averaging theory may be employed at the level of the equations of motion to derive the averaged equations. Furthermore, as shown below, connections may be made between the choice of averaging methods and coarse analysis techniques.

Preliminaries We make the following definitions, valid for phase spaces that are subsets of $\mathbb{R}^n$.

A *lift map to N particles* is a map $L : \mathbb{R}^n \to \mathbb{R}^n \times \cdots \times \mathbb{R}^n$ (N times).

An *averaging map for N particles* is a map $A : \mathbb{R}^n \times \cdots \times \mathbb{R}^n \to \mathbb{R}^n$ (N times).

Suppose we have a vector field $X(x, t)$ with flow map ϕ, i.e.,

$$X(\phi(t, x_0), t) = \frac{d}{dt}\phi(t, x_0).$$

We can extend ϕ to a flow $\tilde{\phi}$ defined on an enlarged space of N particles. The idea is that we apply ϕ to each particle separately:

$$\tilde{\phi}(t, \mathbf{q}_1, \ldots, \mathbf{q}_N) = (\phi(t, \mathbf{q}_1), \ldots, \phi(t, \mathbf{q}_N)),$$

where each $\mathbf{q}_i \in \mathbb{R}^n$. Then we see that $\tilde{\phi}$ describes integral curves of an extended vector field $\tilde{X}$ in the natural way.

Then we define the *coarse flow map* ψ (associated with X) by

$$\psi(t, x_0) = A \circ \tilde{\phi}(t, L(x_0)),$$

or for short, $\psi_t = A \circ \tilde{\phi}_t \circ L$.

We begin with the equations of motion for a charged particle in a magnetic field, written as a system of first-order equations:

$$\dot{\mathbf{q}} = \mathbf{v} \tag{4.83a}$$

$$\dot{\mathbf{v}} = \frac{1}{m}\left(\frac{e}{c}\mathbf{v} \times \mathbf{B}\right). \tag{4.83b}$$

We seek to compute the coarse flow map ψ associated with X, where X is the autonomous vector field defined by the right-hand side of Eq. (4.83). We would like to compare ψ to the flow map ψ^{GC} for the guiding center vector field. If so, this would explain why coarse analysis techniques yield the correct averaged dynamics.

First observe that the coarse flow ψ may only approximate the guiding center flow asymptotically:

$$\psi(t, x_0) = \psi^{\mathrm{GC}}(t, x_0) + O(t^{1/2}).$$

In carrying out the comparison between ψ and ψ^{GC}, we will derive conditions on lifting and averaging maps so that the coarse integration approximates guiding center dynamics. Before continuing, recall that the guiding center vector field X^{GC} is defined by the right-hand side of the equations

$$\dot{\mathbf{q}} = \mathbf{v} \tag{4.84a}$$

$$\dot{\mathbf{v}} = \frac{1}{m}\left(\frac{e}{c}\mathbf{v} \times \mathbf{B} - \mu \nabla \|\mathbf{B}\|\right). \tag{4.84b}$$

Consider the phase space of positions and velocities $\mathbb{R}^6$. Also consider a general lift map of the form

$$L\begin{pmatrix}\mathbf{q}\\ \mathbf{v}\end{pmatrix} = \left(\begin{pmatrix}\mathbf{q}+\varepsilon\mathbf{f}^1\\ \mathbf{v}+\varepsilon\mathbf{g}^1\end{pmatrix}, \dots, \begin{pmatrix}\mathbf{q}+\varepsilon\mathbf{f}^N\\ \mathbf{v}+\varepsilon\mathbf{g}^N\end{pmatrix}\right),$$

where $\mathbf{q}$, $\mathbf{v}$, $\mathbf{f}^i$, and $\mathbf{g}^i$ are all in $\mathbb{R}^3$. Assume that $\mathbf{f}^i$ and $\mathbf{g}^i$ are constant in time, and that $\|\mathbf{f}^i\| = \|\mathbf{g}^i\| = 1$ for all i. Now consider an averaging map that is the trivial ensemble average over N particles:

$$A(\mathbf{r}^1, \dots, \mathbf{r}^N) = \frac{1}{N}\sum_{i=1}^{N}\mathbf{r}^i,$$

with $\mathbf{r}^i(t)$ in $\mathbb{R}^6$. Now we postulate the existence of $\mathbf{f}^i$ and $\mathbf{g}^i$ such that

$$\frac{d}{dt}\psi_t\begin{pmatrix}\mathbf{q}\\ \mathbf{v}\end{pmatrix} = \frac{d}{dt}\psi_t^{\mathrm{GC}}\begin{pmatrix}\mathbf{q}\\ \mathbf{v}\end{pmatrix}. \tag{4.85}$$

We drop the point of evaluation $(\mathbf{q}, \mathbf{v})$ and treat it as implied. Then, using the definition of ψ, we obtain

$$\begin{aligned}\frac{d}{dt}\psi_t &= \frac{d}{dt}\left(A \circ \tilde{\phi}_t \circ L\right)\\ &= DA\left(\tilde{\phi}_t \circ L\right) \cdot \frac{d}{dt}\tilde{\phi}_t \circ L\\ &= A \circ \frac{d}{dt}\tilde{\phi}_t \circ L,\end{aligned}$$

where we have used the relationship between vector fields and their flows, along with the linearity of the averaging map A. At this stage, to compute $\tilde{\phi}$ exactly, we would require knowledge of $\mathbf{B}$. Hence we introduce the following approximation, which is actually the first-order Euler method for solving an ODE:

$$\phi_t \begin{pmatrix} \mathbf{q} + \varepsilon \mathbf{f}^i \\ \\ \mathbf{v} + \varepsilon \mathbf{g}^i \end{pmatrix} = \begin{pmatrix} \mathbf{q} + \varepsilon \mathbf{f}^i \\ \\ \mathbf{v} + \varepsilon \mathbf{g}^i \end{pmatrix} + tX \begin{pmatrix} \mathbf{q} + \varepsilon \mathbf{f}^i \\ \\ \mathbf{v} + \varepsilon \mathbf{g}^i \end{pmatrix} + O(t^2).$$

Differentiating in time, we obtain

$$\frac{d}{dt}\phi_t \begin{pmatrix} \mathbf{q} + \varepsilon \mathbf{f}^i \\ \\ \mathbf{v} + \varepsilon \mathbf{g}^i \end{pmatrix} = X \begin{pmatrix} \mathbf{q} + \varepsilon \mathbf{f}^i \\ \\ \mathbf{v} + \varepsilon \mathbf{g}^i \end{pmatrix} + O(t).$$

By definition,

$$X \begin{pmatrix} \mathbf{q} + \varepsilon \mathbf{f}^i \\ \\ \mathbf{v} + \varepsilon \mathbf{g}^i \end{pmatrix} = \begin{pmatrix} \mathbf{v} + \varepsilon \mathbf{g}^i \\ \\ \frac{e}{mc}\left(\mathbf{v} + \varepsilon \mathbf{g}^i\right) \times \mathbf{B}\left(\mathbf{q} + \varepsilon \mathbf{f}^i\right) \end{pmatrix}.$$

We will approximate

$$\mathbf{B}(\mathbf{q} + \varepsilon \mathbf{f}^i) = \mathbf{B}(\mathbf{q}) + \varepsilon \nabla \mathbf{B}(\mathbf{q}) \cdot \mathbf{f}^i + \frac{\varepsilon^2}{2} \nabla\nabla \mathbf{B} : (\mathbf{f}^i \otimes \mathbf{f}^i) + O\left(\varepsilon^3\right).$$

Hence the lifted, averaged vector field is

$$A \circ \frac{d}{dt}\tilde{\phi}_t \circ L = \frac{1}{N}\sum_{i=1}^{N} X \begin{pmatrix} \mathbf{q} + \varepsilon \mathbf{f}^i \\ \\ \mathbf{v} + \varepsilon \mathbf{g}^i \end{pmatrix} + O(t)$$

$$= \begin{pmatrix} \mathbf{v} + \varepsilon\langle \mathbf{g}\rangle \\ \\ \frac{e}{mc}\mathbf{v} \times \tilde{\mathbf{B}} + \frac{e}{mc}\varepsilon\langle \mathbf{g}\rangle \times \mathbf{B} + \frac{e}{mc}\varepsilon^2 \frac{1}{N}\sum_{i=1}^{N}\left[\mathbf{g}^i \times \left(\nabla \mathbf{B}\cdot \mathbf{f}^i\right)\right] \end{pmatrix} + O(t) + O(\varepsilon^3),$$

where

$$\tilde{\mathbf{B}} = \mathbf{B} + \varepsilon \nabla \mathbf{B} \cdot \langle \mathbf{f}\rangle + \frac{\varepsilon^2}{2}\nabla\nabla\mathbf{B} : \langle \mathbf{f} \otimes \mathbf{f}\rangle.$$

We have used the conventions

$$\langle \mathbf{f}\rangle = \frac{1}{N}\sum_{i=1}^{N}\mathbf{f}^i,$$

$$\langle \mathbf{g}\rangle = \frac{1}{N}\sum_{i=1}^{N}\mathbf{g}^i,$$

$$\langle \mathbf{f} \otimes \mathbf{f}\rangle = \frac{1}{N}\sum_{i=1}^{N}\mathbf{f}^i \otimes \mathbf{f}^i.$$

and assume that the lifting is not skewed in any particular direction:

$$\langle \mathbf{f}\rangle = \langle \mathbf{g}\rangle = 0.$$

Then we obtain, for the lifted, averaged vector field,

$$A \circ \frac{d}{dt}\tilde{\phi}_t \circ L = \begin{pmatrix} \mathbf{v} \\ \\ \frac{e}{mc}\mathbf{v} \times \tilde{\mathbf{B}} + \frac{e}{mc}\varepsilon^2 \frac{1}{N}\sum_{i=1}^{N}\left[\mathbf{g}^i \times \left(\nabla \mathbf{B}\cdot \mathbf{f}^i\right)\right] \end{pmatrix} + O(t) + O(\varepsilon^3), \tag{4.86}$$

where

$$\tilde{\mathbf{B}} = \mathbf{B} + \frac{\varepsilon^2}{2}\nabla\nabla\mathbf{B} : \langle \mathbf{f} \otimes \mathbf{f} \rangle.$$

Using the same first-order accurate scheme we applied to ϕ, we may write the time derivative of the guiding center flow as

$$\frac{d}{dt}\psi_t^{\mathrm{GC}} \begin{pmatrix} \mathbf{q} \\ \\ \mathbf{v} \end{pmatrix} = X^{\mathrm{GC}} \begin{pmatrix} \mathbf{q} \\ \\ \mathbf{v} \end{pmatrix} + O(t) = \left(\mathbf{v} \tfrac{1}{m} \left(\tfrac{e}{c}\mathbf{v} \times \mathbf{B} - \mu \nabla \|\mathbf{B}\| \right) \right) + O(t). \tag{4.87}$$

Comparing Eq. (4.87) with Eq. (4.86) and ignoring the difference between $\tilde{\mathbf{B}}$ and $\mathbf{B}$, we derive the condition on $\mathbf{f}$ and $\mathbf{g}$:

$$\frac{e}{mc}\varepsilon^2 \frac{1}{N} \sum_{i=1}^{N} \left[\mathbf{g}^i \times \left(\nabla \mathbf{B} \cdot \mathbf{f}^i \right) \right] = -\frac{\mu}{m} \nabla \|\mathbf{B}\|. \tag{4.88}$$

In coordinates, the left-hand side may be written as

$$\frac{e}{c}\varepsilon^2 \frac{1}{N} \sum_{i=1}^{N} \epsilon^{nkl} g_k^i B_{,j}^l f_j^i = \frac{e}{c}\varepsilon^2 \epsilon^{nkl} B_{,j}^l \langle \mathbf{f} \otimes \mathbf{g} \rangle_{jk}, \tag{4.89}$$

where, in the same spirit as our previous conventions,

$$\langle \mathbf{f} \otimes \mathbf{g} \rangle = \frac{1}{N} \sum_{i=1}^{N} \mathbf{f}^i \otimes \mathbf{g}^i.$$

Using Eq. (4.89), we rewrite Eq. (4.88) as

$$\frac{e}{c}\varepsilon^2 \epsilon^{nkl} B_{,j}^l \langle \mathbf{f} \otimes \mathbf{g} \rangle_{jk} = -\mu \|\mathbf{B}\|_{,n}. \tag{4.90}$$

It is a nontrivial task to solve for $\langle \mathbf{f} \otimes \mathbf{g} \rangle$ given a general magnetic field $\mathbf{B}$, much less observe the relationship between μ and this tensor. Hence we focus our efforts on the helical magnetic field, given by the vector potential

$$\mathbf{A} = -\frac{1}{2}(0, 0, \ln r^2),$$

where $r^2 = x^2 + y^2$, and the following expression for $\mathbf{B}$:

$$\mathbf{B} = r^{-2}(-y, x, 0).$$

In this case, simple algebraic manipulations show that the solution of (4.90) is given by

$$\frac{e}{c}\varepsilon^2 \langle \mathbf{f} \otimes \mathbf{g} \rangle_{jk} = \begin{cases} \mu x/r & (j,k) = (1,3) \\ \mu y/r & (j,k) = (2,3) \\ 0 & \text{otherwise.} \end{cases} \tag{4.91}$$

Now we recognize the decoupling $\langle \mathbf{f} \otimes \mathbf{g} \rangle = \widehat{\mathbf{f}} \otimes \widehat{\mathbf{g}}$, where

$$\widehat{\mathbf{f}} = \frac{c}{e}\varepsilon^{-2}(\mu x/r, \mu y/r, 0) \tag{4.92a}$$

$$\widehat{\mathbf{g}} = (0, 0, 1). \tag{4.92b}$$

Here we have specifically chosen $\widehat{\mathbf{g}}$ to have unit norm. Observe that these vectors satisfy the following relationships with the magnetic field:

$$\widehat{\mathbf{f}} \perp \mathbf{B},$$

$$\widehat{\mathbf{g}} \perp \mathbf{B},$$

$$\widehat{\mathbf{f}} \perp \widehat{\mathbf{g}}.$$

In other words, $\widehat{\mathbf{f}}, \widehat{\mathbf{g}}, \mathbf{B}$ forms a basis for $\mathbb{R}^3$ at each point in space. We also see that $\widehat{\mathbf{f}}$ rotates with the magnetic field. Hence, lifting to respect $\widehat{\mathbf{f}}, \widehat{\mathbf{g}}$ according to Eq. (4.92), and using ensemble averaging would yield an approximation to guiding center flow.

To place this in the context of coarse analysis, note that we may now approximate the guiding center vector field by choosing an appropriate lift and average maps and applying them to the original vector field for the Lorentz force system. Should a coarse (time integration) algorithm respect the choice of lift and average maps, then computing the flow on the lifted, averaged Lorentz field yields an approximation to the averaged flow itself. We now hypothesize that the numerical results discussed in Section 4.7 were successful approximations to the averaged flow because the lift map (generation of particles in a distribution normal to the local magnetic field line) is similar to the one suggested here.

While averaging theory on the level of the equations of motion are useful for guiding the development of coarse analysis techniques, averaging theory from other points of view might also prove useful. For instance, guiding center theory may also be derived from the variational principle. Because this represents another connection between averaging and the equations of motion, we now consider derivations of the guiding center equation using techniques that might allow for approximations through coarsening.

4.9 Hamiltonian Averaging

Here, the construction of a coarse integration algorithm is taken from the point of view of Hamiltonian manifolds and flows. For idealized coarse integrators that that utilize idealized fine integrators (where the underlying dynamics are Hamiltonian), the computed flow is shown to correspond to a coarse Hamiltonian flow.

Preliminaries Consider a configuration space M and its product space $M^N = M \times M \times \cdots \times M$ (N times). In this setting, individual trajectories exist on M and an ensemble of trajectories exist on M^N. With these manifolds, define corresponding vector fields

$$X: \qquad M \to TM \tag{4.93}$$

$$X^N: \qquad M^N \to T(M^N) = (TM)^N. \tag{4.94}$$

Using the terminology from coarse analysis, define a *lift map* L and an *average map* A as

$$L: \qquad T^*M \to (T^*M)^N \tag{4.95}$$

$$A: \qquad (T^*M)^N \to T^*M. \tag{4.96}$$

Observe that lift and average maps are defined here as operating on the cotangent manifold and on the cotangent-product manifold.

Hamiltonian systems For a Hamiltonian system, a vector field at $q \in M$ is related to the Hamiltonian by

$$X_H(q) = R \cdot \mathbf{d}H(q) \tag{4.97}$$

where the matrix of R is the symplectic matrix,

$$[R] = \begin{bmatrix} \mathbf{0} & \mathbf{1} \\ -\mathbf{1} & \mathbf{0} \end{bmatrix} := \mathbb{J}. \tag{4.98}$$

In other words,

$$X = \mathbb{J}\nabla H. \tag{4.99}$$

Here, the Hamiltonian H is a C^1 map,

$$H : T^*M \to \mathbb{R}. \tag{4.100}$$

Now consider a collection of points q_i in an ϵ-ball $\mathbf{B}_\epsilon(q)$ around $q \in T^*M$. For this collection, define their *effective* Hamiltonian $H^N : T^*M \to \mathbb{R}$ as

$$H_q^N = \left(A_H \circ H^N \circ L\right)(q) \tag{4.101}$$

or, for the naive average map,

$$H_q^N : \frac{1}{N}\sum_{i=1}^{N} H_{q_i}. \tag{4.102}$$

The average map A_H may be a simple ensemble averaging, as $H^N(L(q)) \in \mathbb{R}^N$. Observe that, for this to hold, we assume that

$$A_H \circ H^N = H \circ A, \tag{4.103}$$

that is, the action of taking averages and computing the Hamiltonian commute with each other. This should hold on the ball $\mathbf{B}_\epsilon q$. Now, using our definitions of the lift and average maps, observe that

$$\begin{aligned} \mathbb{J}\nabla H_q^N &= \mathbb{J}\nabla \left(\frac{1}{N} \sum_{i=1}^{N} H_{q_i} \right) \\ &= \mathbb{J}\nabla \left[\left(A_H \circ H^N \circ L \right)(q) \right] \\ &= \mathbb{J} \left[TA_H \circ (\nabla H)^N \circ TL \right](q) \\ &= \left[TA_H \circ (\mathbb{J}\nabla H)^N \circ TL \right](q) \\ &= \left(TA \circ X^N \circ TL \right)(q). \end{aligned} \tag{4.104}$$

Here, we have assumed that taking averages and computing the Hamiltonian commute, and there are additional assumptions associated with the action of the ∇ operator and the chain rule. With this progression, we can define the *average* vector field $\bar{X} : M \to TM$ as

$$\begin{aligned} \bar{X} &= \mathbb{J}\nabla H_q^N \\ &= TA \circ X^N \circ TL. \end{aligned} \tag{4.105}$$

Observe that this average vector field is Hamiltonian with the average Hamiltonian defined above.

Does this average vector field correspond to a flow of anything? This bears direct relevance to the coarse integration algorithm. The coarse integration algorithm constructs an average flow from the actual flows, using appropriate lifts and averages.

First, define a map $G : \mathbb{R} \times M \to M$ as

$$G_\tau = A \circ F_\tau^N \circ L \tag{4.106}$$

with the original flow F_τ,

$$\begin{aligned} \frac{d}{dt} F_t(q) &= X(t, F_t(q)) \qquad (4.107) \\ F_0(q) &= q. \qquad (4.108) \end{aligned}$$

We seek to show that an average flow map K_t can be constructed, such that

$$\begin{aligned} \frac{d}{dt} K_t(q) &= \bar{X}(t, K_t(q)) \qquad (4.109) \\ K_0(q) &= q. \qquad (4.110) \end{aligned}$$

First, define a candidate map K_t,

$$K_t = \lim_{\tau \to 0} G_\tau^{t/\tau}. \tag{4.111}$$

Observe that a flow property is satisfied:

$$\begin{aligned}
K_{t_1+t_2} &= \lim_{\tau\to 0} G_\tau^{\frac{t_1+t_2}{\tau}} \\
&= \lim_{\tau\to 0}\left(G_\tau^{t_2/\tau}\circ G_\tau^{t_1/\tau}\right) \\
&= \left(\lim_{\tau\to 0} G_\tau^{t_2/\tau}\right)\circ\left(\lim_{\tau\to 0} G_\tau^{t_1/\tau}\right) \\
&= K_{t_2}\circ K_{t_1}. \qquad (4.112)
\end{aligned}$$

Now construct the following limit:

$$\lim_{t\to 0}\frac{K_t - I}{t}. \qquad (4.113)$$

We seek to demonstrate that this expression is equivalent to the expression for the average vector field. Now,

$$\begin{aligned}
\lim_{t\to 0}\frac{K_t - I}{t} &= \lim_{t\to 0}\frac{\lim_{\tau\to 0} G_\tau^{t/\tau} - I}{t} \\
&= \lim_{t\to 0}\frac{\lim_{\tau\to 0} G_{\epsilon t}^{t/\epsilon t} - I}{t} \\
&= \lim_{t\to 0}\frac{\lim_{\tau\to 0} G_{\epsilon t}^{1/\epsilon} - I}{t} \qquad (4.114)
\end{aligned}$$

and we evaluate the outer limit using L'Hôpital's rule.

The quantity $G_{\epsilon t}^{1/\epsilon}$ is an application of the map $G_{\epsilon t}$, $1/\epsilon$ times; that is,

$$G_{\epsilon t}^{1/\epsilon} = \left(A\circ F_{\epsilon t}^{N}\circ L\right)^{1/\epsilon} \qquad (4.115)$$

so that the time derivative of its ϵ - limit is

$$\begin{aligned}
\frac{\partial}{\partial t}\lim_{\epsilon\to 0} G_{\epsilon t}^{1/\epsilon} &= \lim_{\epsilon\to 0}\sum_{m=1}^{1/\epsilon}\left(G_{\epsilon t}^{1/\epsilon}\right)^{m-1}\circ\frac{\partial}{\partial t}G_{\epsilon t}\circ\left(G_{\epsilon t}^{1/\epsilon}\right)^{1/\epsilon-m} \\
&= \lim_{\epsilon\to 0}\sum_{m=1}^{1/\epsilon}\left(G_{\epsilon t}^{1/\epsilon}\right)^{m-1}\circ\frac{\partial}{\partial t}G_{\epsilon t}\circ\left(G_{\epsilon t}^{1/\epsilon}\right)^{1/\epsilon-m} \\
&= \lim_{\epsilon\to 0}\sum_{m=1}^{1/\epsilon}\left(G_{\epsilon t}^{1/\epsilon}\right)^{m-1}\circ\left(TA\circ\epsilon X^N\circ L\right)\circ\left(G_{\epsilon t}^{1/\epsilon}\right)^{1/\epsilon-m} \\
&= TA\circ X^N\circ L \\
&= \bar{X}. \qquad (4.116)
\end{aligned}$$

So, application of L'Hopital's rule leads to

$$\lim_{t\to 0}\frac{K_t - I}{t} \quad = \quad \bar{X}. \qquad (4.117)$$

This means that this average vector field $\bar{X}$, which is Hamiltonian, corresponds to an averaged flow map of the system. In this sense, numerical simulations that preserve the Hamiltonian nature of the original charged particle system may be used in the sense of coarse analysis to construct numerical simulations of a coarsened system that is also Hamiltonian. This would be expected of a coarsened charged particle simulation, because the guiding center system is also a conservative system.

4.10 Conclusions

We have undertaken a study of a system of charged particles in a magnetic field for the purposes of advancing averaging theory and for making connections to coarse analysis. A summary of the theoretical and numerical averaging techniques applied to the system of charged particles is given in Table 4.1. For this system of charged particle motion, we have successfully derived the guiding center equations, computed guiding center flow, and derived design guidelines for coarse numerics using a variety of averaging techniques.

A summary of averaging techniques are given for the system of charged particles. Guiding center theory has existed for approximately forty years, involving averaging from both the point of view of the equations of motion and from the point of view of the variational principle. To elucidate the relationship between the dominant and subdominant scales of the system, two novel averaging techniques are applied at the level of the variational principle. First, a new WKB/Whitham-style variational averaging procedure is applied, yielding the well-known guiding center equations. The closure model is made clear through the Euler-Lagrange equation for the fluctuations, and an expression for the magnetic moment is produced from the analysis. The WKB/Whitham-style averaging framework appears to be applicable to other types of systems that respect a variational principle.

Secondly, LANS-style Lagrangian averaging is applied, yielding averaged equations and suggested closure models for the scale interactions. Connections are made between the WKB/Whitham and LANS-style averaging procedures that provide additional insight into the closure models and resulting Euler-Lagrange equations.

On the side of the Hamiltonian, it is shown that, in general, averaging of Hamiltonian systems and associated vector fields yields a coarsened Hamiltonian and an associated coarsened vector field. Because elements from coarse analysis are applied in the averaging, this result suggests that coarse analysis of Hamiltonian systems may yield coarsened systems that are also Hamiltonian. This is very important, as this suggests that coarse analysis techniques may be constructed that preserve mechanical structure.

Because of the success of averaging theory in deriving guiding center approximations to the original Lorentz force system, the application of coarse analysis to this conservative system is considered. A prototypical coarse numerical time integration yielded averaged trajectories that suggest that coarse analysis also applies to systems without dissipation. To elucidate this and suggest rigorous guidelines for coarse analysis algorithm design, perturbation and averaging theory is employed to derive appropriate lift maps.

To this end, averaging techniques influenced by coarse analysis are applied to the equations of motion for the charged particle system. The averaging yields guiding center flow, with a constraint on the types of lifting and averaging operators employed. This suggests both a confirmation of the coarse numerical analysis results for the charged particle system as well as the necessary structure of the specific numerical implementation.

Finally, it is important to note that while connections may be made between geometric mechanics, averaging theory and coarse analysis, we have made these connections in the simple setting of a conservative mechanical system. A similar development in coarse fluids analysis is highly nontrivial compared to our treatment of the Lorentz force system. However, in light of the successful

Table 4.1: Comparison of averaging techniques for the charged particle system.

Approach	**Setting**	**Comparison with classical g.c.e?**	**Notes**
Classical	e.o.m.		
Littlejohn	Variational	✓ (same)	Non-canonical, requires gauge transformations
WKB-inspired	Variational	✓ (same)	Canonical
LANS-inspired	Variational	✓ (same)	Canonical; Insight gained from WKB-derivation
Coarse Analysis	Computational	✓ (similar)	Lifting orthogonal to magnetic field
CA-inspired perturbation	e.o.m.	✓ (same)	Produces guidelines for coarse analysis

connections described in this chapter, we propose that averaging and perturbation theory, which have been successfully applied in fluid mechanics, might prove useful in designing coarse analysis methodologies for fluids.

Chapter 5

Concluding Remarks and Future Directions

Conclusions We have found the application of coarse analysis methods to compressible diffuser flows to be very encouraging and promising. First, the nontraditional use of the proper orthogonal decomposition is very well-posed and well-defined in the context of coarse analysis of compressible flows. Traditionally, the POD has been used for making observations and capturing the dominant scales of interest for a fluid flow. We make use of this, and extend the traditional POD application to that of *reconstruction*.

The use of the POD as a scale classification allows us to implement coarse analysis techniques. We derived a coarse projective third-order Adams-Bashforth routine suitable for the time integration of POD mode coefficients for our compressible diffuser flow, using short bursts of DNS simulation to compute the interaction between scales. Although considerations have to be made to ensure the numerical stability of these methods, we have achieved stable, accurate time integration of the dominant dynamics at a computation reduction of a factor of ten (for short-time reconstruction) and about five (for long-time convergence and accuracy). Although the analysis produces long-time trajectories that are out of phase and of slightly different amplitude than those extracted from the original DNS computation, their physical behavior is similar, as shown in the phase portraits. Furthermore, the coarse computation extracts a converged limit cycle, which is only suggested by the projected DNS data. This suggests the use of coarse analysis methods to extract stable or unstable limit cycles where the underlying detailed flow is chaotic or turbulent.

It is important to note that the POD-based scale classification restricts the coarse analysis to the computation of fluid flows well represented by the archival data used to compute the POD modes. While this yields coarse analysis techniques which are well suited to real-time estimation and prediction of "known" fluid flows, they are not well suited for the exploration of "unknown" flows. This would suggest seeking improvements or alternatives to the POD-based scale classification. To this end, a comparison with POD/Galerkin-based simulations of diffuser flow is proposed to elucidate the differences between analytical model reduction (POD/Galerkin) and numerical model reduction (coarse analysis). A similar development is in progress for computations of turbulent flow in a minimal flow unit (Smith, 2005). Shown in Figure 5.1 are phase portraits of three POD mode coefficients projected from DNS simulation and computed from POD/Galerkin simulation (Smith et al., 2005). Their POD/Galerkin simulation includes additional terms which model the interaction between small and large scales of the flow, and T. Smith is currently exploring a connection between these interaction terms and coarse analysis.

We then turned to averaging and geometric mechanics to gain insight and better discern the averaging and mechanical properties of coarse analysis methods. In doing so, we shifted our focus from diffuser flows to the simple system of charged particle motion in a magnetic field, and we explored the applications of both averaging theory and coarse analysis. We first derived a canonical formulation of the guiding center equations for averaged charged particle motion, which would allow for more future comparisons with coarse analysis than existing noncanonical methods. In doing so, we averaged the variational principle following inspiration from WKB methods. We also made a similar derivation using Lagrangian averaging theory of the type applied to fluid turbulence, and we were able to make connections to the other averaging methods. We were also able to make connections between Hamiltonian averaging theory and coarse analysis. Finally, we returned to a setting of applying perturbation analysis to the equations of motion for the charged particle system, and we were able to make strong connections to coarse analysis. In applying a prototypical coarse time integration of the charged particle system, we found its success compatible with design guidelines suggested by the averaging theory.

Future Directions The work documented in this thesis is very promising, as the avenues of future research follow according to directions into the coarse computation of fluid flows, averaging theory, and additional applications.

While the application of averaging techniques to fluids for the purposes of advancing coarse analysis theory is still open, these results suggest that the application to fluids might be very promising. New types of error analysis for the coarse analysis of fluids may be defined using averaging techniques. A fundamental understanding of coarse fluids analysis tools as multiscale fluid systems might provide connections to existing model reduction procedures, such as turbulence modeling and Galerkin projection. Finally, averaging techniques may be employed to facilitate the design of subroutines for coarse fluids analysis techniques. The coarse analysis of diffuser flows may continue in many directions. Continuing development of the POD as scale classification would yield insight not only into coarse fluids analysis but also into the nature of the POD as a numerical tool. Improvements to the POD methodology, such as ways to generate the modes without archival data but with data generated during the course of the simulation, would avoid a dependence on archival data. Coarse continuation and bifurcation analyses of diffuser flows would follow directly from this work; coarse diffuser flow control would represent an additional level of abstraction, as a coarse time integration routine (for estimation, for instance) would be considered a subsystem in a larger control system design. Furthermore, as the phenomena of diffuser flows and flow separation is still under active exploration in the community, we suggest using coarse analysis to facilitate these investigations. For instance, shown in Figure 5.3 is a vorticity snapshot of a diffuser flow controlled using vorticity flux control, described in Figure 5.2; computing this controlled flow in real-time, along with controller design assisted by simulation, would be a desirable goal of coarse analysis.

Finally, recent explorations by our collaborators into climate dynamics and turbulence suggest a wide realm of exploration of coarse analysis techniques into fluid dynamics and related applications areas. For instance, shown in Figure 5.4 is a root locus chart, which is a type of sensitivity analysis, computed by MacMynowski (2005) for a particular model of the El Niño phenomenon. The figure shows the dominant mode of the dynamics plotted as a function of the atmosphere-ocean coupling parameter. As shown in the figure, the mode goes unstable as the parameter is varied. This chart was computed, however, without the traditional linearization of the equations of motion; it was computed using bursts of simulation using techniques inspired by coarse analysis.

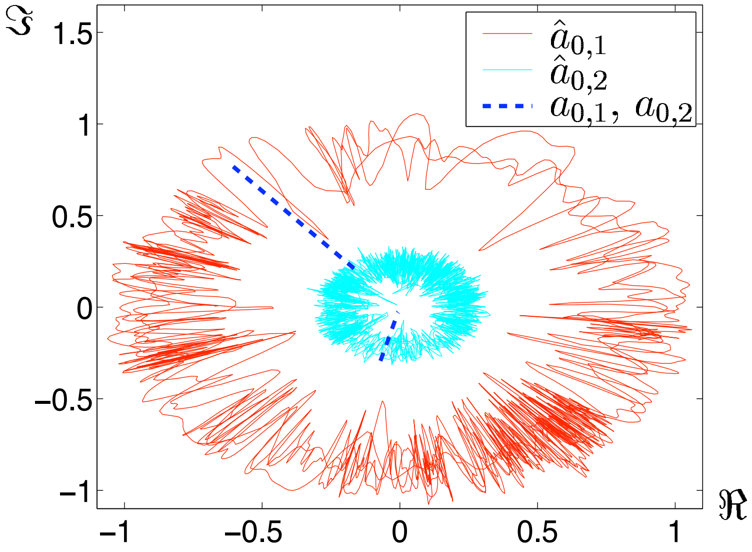


Figure 5.1: POD coefficients for turbulent plane Couette flow in a minimal flow unit, taken from Smith et al. (2005).

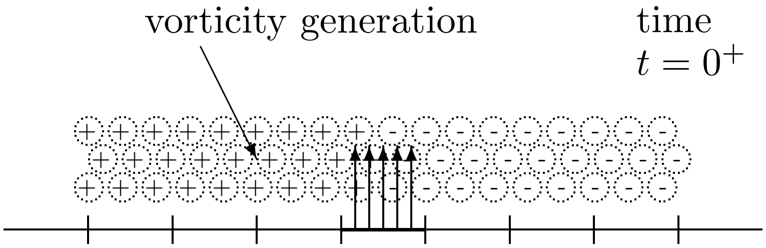


Figure 5.2: Vorticity flux control: a latter stage taken from Koumoutsakos (1997).

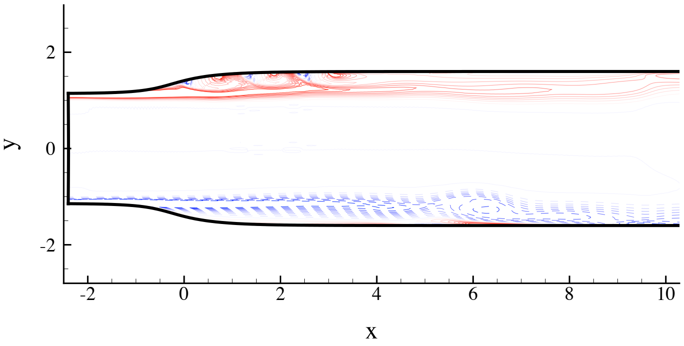


Figure 5.3: Diffuser flow under vorticity flux control.

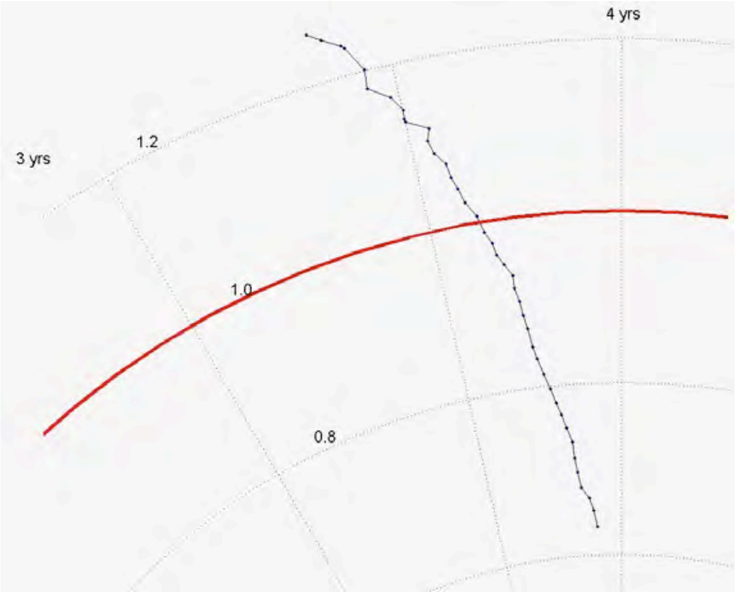


Figure 5.4: Root locus for an El Niño model, with atmosphere-ocean coupling as the parameter of interest. The stability boundary is shown in red.

Appendix A

Derivative Computation in the *diffuser* Code

A.1 Objective

The original *diffuser* code computes derivatives with implicit Padé schemes. These implicit schemes are nonlocal, and this nonlocality hinders efforts to parallelize the code. Explicit schemes, which are inherently local, allow for domain-splitting and similar straightforward parallel implementations of a numerical algorithm. Classical explicit finite-difference schemes are described, and a wavenumber–optimal explicit scheme is described. The existing and proposed differencing framework is given for computing derivatives away and near boundary points. Results of an error analysis is given to demonstrate the accuracy of the explicit schemes. Finally, computational *diffuser* results are given to show that a streamwise explicit/transverse Padé formulation compares well with a full Padé implementation.

A.2 Implicit Finite-Difference Schemes

First Derivative Here, the first and second spatial derivatives are computed using a Padé scheme. The schemes used are tridiagonal and hence implicit. The first derivative is computed as

$$\beta f'_{i-2} + \alpha f'_{i-1} + f'_i + \alpha f'_{i+1} + \beta f'_{i+2} = c\frac{f_{i+3} - f_{i-3}}{6h} + b\frac{f_{i+2} - f_{i-2}}{4h} + a\frac{f_{i+1} - f_{i-1}}{2h}. \tag{A.1}$$

Fourth-Order Observe that for a tridiagonal scheme, β is set equal to zero. For $c = 0$, we have the following family of tridiagonal fourth-order schemes:

$$\beta = 0, \quad a = \tfrac{2}{3}(\alpha + 2), \quad b = \tfrac{1}{3}(4\alpha - 1), \quad c = 0 \quad \cdot \tag{A.2}$$

For the diffuser code, α is set to 1/4, we arrive at the classical fourth-order Padé scheme. The corresponding values are

$$\alpha = \tfrac{1}{4}, \quad \beta = 0, \quad a = \tfrac{3}{2}, \quad b = 0, \quad c = 0 \quad \cdot \tag{A.3}$$

Sixth-Order Although the previous family corresponds to fourth-order schemes, setting $\alpha = 1/3$ actually yields a sixth-order accurate scheme:

$$\alpha = \tfrac{1}{3}, \quad \beta = 0, \quad a = \tfrac{14}{9}, \quad b = \tfrac{1}{9}, \quad c = 0 \quad \cdot \tag{A.4}$$

Second Derivative The second derivative is computed as

$$\beta f''_{i-2} + \alpha f''_{i-1} + f''_i + \alpha f''_{i+1} + \beta f''_{i+2} = c\frac{f_{i+3} - 2f_i + f_{i-3}}{9h^2} + b\frac{f_{i+2} - 2f_i + f_{i-2}}{4h^2} + a\frac{f_{i+1} - 2f_i + f_{i-1}}{h^2}. \tag{A.5}$$

Fourth-order Again, we seek tridiagonal schemes, so we set β equal to zero. For $c = 0$, we arrive at a family of fourth-order schemes

$$\beta = 0, \quad a = \tfrac{4}{3}(1 - \alpha), \quad b = \tfrac{1}{3}(-1 + 10\alpha), \quad c = 0 \quad \cdot \tag{A.6}$$

For the diffuser code, α is set to 1/10, we arrive at the classical fourth-order Padé scheme. The corresponding values are

$$\alpha = \tfrac{1}{10}, \quad \beta = 0, \quad a = \tfrac{6}{5}, \quad b = 0, \quad c = 0 \quad \cdot \tag{A.7}$$

Sixth-Order Although the previous family corresponds to fourth-order schemes, setting $\alpha = 2/11$ actually yields a sixth-order accurate scheme:

$$\alpha = \tfrac{2}{11}, \quad \beta = 0, \quad a = \tfrac{12}{11}, \quad b = \tfrac{3}{11}, \quad c = 0 \quad \cdot \tag{A.8}$$

These are set in the diffuser code in the `derivatives.f90` file. There are certain differences between the actual and computational algorithms. Rather than using the parameters a, b, and c, the coefficients of the respective terms are used (for instance $\pm a/2$). Observe that Eqs. (A.1) and (A.5) show what those coefficients are.

A.3 Explicit Finite-Difference Schemes

Parallelization of the existing diffuser code is nontrivial due to the implicit nature of the Padé schemes. The implementation effort would reside primarily in developing communications routines between parallel processors to compute the implicit derivatives. Explicit schemes are local by definition, and this locality can be exploited in a straightforward manner when dividing the processing load among processors. High-order explicit finite-difference schemes are described here and should be implemented prior to parallelization.

To begin, we wish to compute an approximation to the rth derivative of f, or the quantity $\partial^r f/\partial x^r$. The approximation will be computed over q nodes, providing an estimate of order $q-r$ in accuracy for odd derivatives and an estimate of order $q-r+1$ in accuracy for even derivatives. This approximation may be restated as

$$\frac{\partial^r f}{\partial x^r} \approx c_1 f_1 + c_2 f_2 + \cdots + c_q f_q. \tag{A.9}$$

Consider the approximation to be for a particular node $\tilde{x}$. Now we expand each f_i about the node $\tilde{x}$:

$$f_i = \tilde{f} + (x_i - \tilde{x})\frac{\partial \tilde{f}}{\partial x} + \frac{1}{2}(x_i - \tilde{x})^2 \frac{\partial^2 \tilde{f}}{\partial x^2} + \frac{1}{6}(x_i - \tilde{x})^3 \frac{\partial^3 \tilde{f}}{\partial x^3} + \cdots . \tag{A.10}$$

Substituting and collecting coefficients of the $\frac{\partial^i \tilde{f}}{\partial x^i}$, we arrive at

$$\frac{\partial^r f}{\partial x^r} \approx \sum_{i=1}^{q} c_i \tilde{f} + \sum_{i=1}^{q} c_i (x_i - \tilde{x})\frac{\partial \tilde{f}}{\partial x} + \sum_{i=1}^{q} c_i (x_i - \tilde{x})^2 \frac{\partial^2 \tilde{f}}{\partial x^2} + \cdots . \tag{A.11}$$

To solve for the coefficients c_i, we solve the set of equations

$$\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ (x_1-\tilde{x}) & (x_2-\tilde{x}) & (x_3-\tilde{x}) & \cdots & (x_q-\tilde{x}) \\ (x_1-\tilde{x})^2 & (x_2-\tilde{x})^2 & (x_3-\tilde{x})^2 & \cdots & (x_q-\tilde{x})^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ (x_1-\tilde{x})^q & (x_2-\tilde{x})^q & (x_3-\tilde{x})^q & \cdots & (x_q-\tilde{x})^q \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_q \end{bmatrix} = R, \tag{A.12}$$

where $R = \begin{bmatrix} 0 & \cdots & 0 & r! & 0 & \cdots & 0 \end{bmatrix}^T$. In other words, R is an array containing zeros except for an entry of value $r!$ at the rth position in the array. Upon solving for the coefficents c_i, substitution into Eq. (A.9) provides the algorithm for computing the rth derivative.

First Derivative Here, we wish to compute derivatives of order one, so we set r equal to one.

Fourth-Order Centered Difference Now, for a fourth-order approximation we must use five nodes, and we set $q = 5$ so that we have a fourth-order approximation. We now solve the following for the coefficients of the particular nodes:

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ (-2h) & (-h) & 0 & (h) & (2h) \\ (-2h)^2 & (-h)^2 & 0 & (h)^2 & (2h)^2 \\ (-2h)^3 & (-h)^3 & 0 & (h)^3 & (2h)^3 \\ (-2h)^4 & (-h)^4 & 0 & (h)^4 & (2h)^4 \end{bmatrix} \begin{bmatrix} c_{i-2} \\ c_{i-1} \\ c_i \\ c_{i+1} \\ c_{i+2} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{A.13}$$

The fourth-order scheme for the first derivative is now

$$\frac{\partial f}{\partial x} = \frac{1}{12h} f_{i-2} - \frac{2}{3h} f_{i-1} + \frac{2}{3h} f_{i+1} - \frac{1}{12h} f_{i+2}. \tag{A.14}$$

Sixth-Order Centered Difference Now, for a sixth-order approximation we must use seven nodes, and we set $q = 7$ so that we have a sixth-order approximation. We now solve the following for the coefficients of the particular nodes:

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ (-3h) & (-2h) & (-h) & 0 & (h) & (2h) & (3h) \\ (-3h)^2 & (-2h)^2 & (-h)^2 & 0 & (h)^2 & (2h)^2 & (3h)^2 \\ (-3h)^3 & (-2h)^3 & (-h)^3 & 0 & (h)^3 & (2h)^3 & (3h)^3 \\ (-3h)^4 & (-2h)^4 & (-h)^4 & 0 & (h)^4 & (2h)^4 & (3h)^4 \\ (-3h)^5 & (-2h)^5 & (-h)^5 & 0 & (h)^5 & (2h)^5 & (3h)^5 \\ (-3h)^6 & (-2h)^6 & (-h)^6 & 0 & (h)^6 & (2h)^6 & (3h)^6 \end{bmatrix} \begin{bmatrix} c_{i-3} \\ c_{i-2} \\ c_{i-1} \\ c_i \\ c_{i+1} \\ c_{i+2} \\ c_{i+3} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{A.15}$$

The sixth-order scheme for the first derivative is now

$$\frac{\partial f}{\partial x} = -\frac{1}{60h} f_{i-3} + \frac{3}{20h} f_{i-2} - \frac{3}{4h} f_{i-1} + \frac{3}{4h} f_{i+1} - \frac{3}{20h} f_{i+2} + \frac{1}{60h} f_{i+3}. \tag{A.16}$$

Third-Order Upwinding (Forward) Difference Now for a third-order approximation we must use four nodes, and we set $q = 4$ so that we have a third-order approximation. We now solve the following for the coefficients of the particular nodes:

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & h & 2h & 3h \\ 0 & h^2 & (2h)^2 & (3h)^2 \\ 0 & h^3 & (2h)^3 & (3h)^3 \\ 0 & h^4 & (2h)^4 & (3h)^4 \end{bmatrix} \begin{bmatrix} c_i \\ c_{i+1} \\ c_{i+2} \\ c_{i+3} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}. \tag{A.17}$$

The third-order scheme for the first derivative is now

$$\frac{\partial f}{\partial x} = -\frac{11}{6h} f_i + \frac{3}{h} f_{i+1} - \frac{3}{2h} f_{i+2} + \frac{1}{3h} f_{i+3}. \tag{A.18}$$

Fourth-Order Upwinding (Forward) Difference Now, for a fourth-order approximation we must use five nodes, and we set $q = 5$ so that we have a fourth-order approximation. Note that, in this case, the derivative is defined for a node i with neighbors $i-1, i+1 \ldots i+3$. We now solve the following for the coefficients of the particular nodes:

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ -h & 0 & h & 2h & 3h \\ (-h)^2 & 0 & h^2 & (2h)^2 & (3h)^2 \\ (-h)^3 & 0 & h^3 & (2h)^3 & (3h)^3 \\ (-h)^4 & 0 & h^4 & (2h)^4 & (3h)^4 \end{bmatrix} \begin{bmatrix} c_i \\ c_{i+1} \\ c_{i+2} \\ c_{i+3} \\ c_{i+4} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{A.19}$$

The fourth-order scheme for the first derivative is now

$$\frac{\partial f}{\partial x} = -\frac{1}{4h} f_i - \frac{5}{6h} f_{i+1} + \frac{3}{2h} f_{i+2} - \frac{1}{2h} f_{i+3} + \frac{1}{12h} f_{i+4}. \tag{A.20}$$

Fifth-Order Upwinding (Forward) Difference Now, for a fifth-order approximation we must use six nodes, and we set $q = 5$ so that we have a fifth-order approximation. We now solve the following for the coefficients of the particular nodes:

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & h & 2h & 3h & 4h & \\ 0 & h^2 & (2h)^2 & (3h)^2 & (4h)^2 & \\ 0 & h^3 & (2h)^3 & (3h)^3 & (4h)^3 & \\ 0 & h^4 & (2h)^4 & (3h)^4 & (4h)^4 & \\ 0 & h^5 & (2h)^5 & (3h)^5 & (4h)^5 & \end{bmatrix} \begin{bmatrix} c_i \\ c_{i+1} \\ c_{i+2} \\ c_{i+3} \\ c_{i+4} \\ c_{i+5} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{A.21}$$

The fourth-order scheme for the first derivative is now

$$\frac{\partial f}{\partial x} = -\frac{137}{60h} f_i + \frac{5}{h} f_{i+1} - \frac{5}{h} f_{i+2} + \frac{10}{3h} f_{i+3} - \frac{5}{4h} f_{i+4} + \frac{1}{5h} f_{i+5}. \tag{A.22}$$

Second Derivative Here, we wish to compute derivatives of order two, so we set r equal to two.

Fourth-Order Centered Difference We repeat the procedure for the first derivative case, except now $R = \begin{bmatrix} 0 & 0 & 2! & 0 & 0 \end{bmatrix}^T$. The fourth-order scheme for the second derivative is now

$$\frac{\partial f}{\partial x} = -\frac{1}{12h^2} f_{i-2} + \frac{4}{3h^2} f_{i-1} - \frac{5}{2h^2} f_i + \frac{4}{3h^2} f_{i+1} - \frac{1}{12h^2} f_{i+2}. \tag{A.23}$$

Sixth-Order Centered Difference We repeat the procedure for the first derivative case, except now $R = \begin{bmatrix} 0 & 0 & 2! & 0 & 0 & 0 & 0 \end{bmatrix}^T$. The sixth-order scheme for the second derivative is now

$$\frac{\partial^2 f}{\partial x^2} = \frac{1}{90h^2} f_{i-3} - \frac{3}{20h^2} f_{i-2} + \frac{3}{2h^2} f_{i-1} - \frac{49}{18h^2} f_i + \frac{3}{2h^2} f_{i+1} - \frac{3}{20h^2} f_{i+2} + \frac{1}{90h^2} f_{i+3}. \tag{A.24}$$

Third-Order Upwinding (Forward) Difference We repeat the procedure for the first derivative case, except now $R = \begin{bmatrix} 0 & 0 & 2! & 0 \end{bmatrix}^T$. The third-order scheme for the second derivative is now

$$\frac{\partial^2 f}{\partial x^2} = \frac{2}{h^2} f_i - \frac{5}{h^2} f_{i+1} + \frac{4}{h^2} f_{i+2} - \frac{1}{h^2} f_{i+3}. \tag{A.25}$$

Fourth-Order Upwinding (Forward) Difference We repeat the procedure for the first derivative case, except now $R = \begin{bmatrix} 0 & 0 & 2! & 0 & 0 \end{bmatrix}^T$. The fourth-order scheme for the second derivative is now

$$\frac{\partial^2 f}{\partial x^2} = \frac{11}{12h^2} f_i - \frac{5}{3h^2} f_{i+1} + \frac{1}{2h^2} f_{i+2} + \frac{1}{3h^2} f_{i+3} - \frac{1}{12h^2} f_{i+4}. \tag{A.26}$$

Fifth-Order Upwinding (Forward) Difference We repeat the procedure for the first derivative case, except now $R = \begin{bmatrix} 0 & 0 & 2! & 0 & 0 & 0 \end{bmatrix}^T$. The fifth-order scheme for the second derivative is now

$$\frac{\partial^2 f}{\partial x^2} = \frac{15}{4h^2} f_i - \frac{77}{6h^2} f_{i+1} + \frac{107}{6h^2} f_{i+2} - \frac{13}{h^2} f_{i+3} + \frac{61}{12h^2} f_{i+4} - \frac{5}{6h^2} f_{i+5}. \tag{A.27}$$

A.4 Optimized Spatial Discretization

High-order finite difference schemes do not necessarily lead to accurate solutions. In particular, the problems of interest involve nondispersive waves that may not be captured well by conventional (dispersive) finite difference schemes. As a remedy, one can optimize the differencing scheme in wavenumber space, as proposed by Lele (1992) and Tam and Webb (1993). These new schemes should produce wave solutions that are more similar to those of the actual governing partial-differential equations.

First Derivative Consider the following discretization of the first derivative:

$$\frac{\partial f}{\partial x}(x) \approx \frac{1}{h} \sum_{j=-M}^{M} a_j f(x + jh). \tag{A.28}$$

Observe that we only consider stencils over odd numbers of grid points. The Fourier transform and its inverse are defined as

$$F(k) = \frac{1}{2\pi} \int_{-\infty}^{\infty} f(x) e^{-ikx} dx \tag{A.29}$$

$$f(x) = \int_{-\infty}^{\infty} F(k) e^{ikx} dk \tag{A.30}$$

and computing the Fourier transform of Eq. (A.28), we arrive at

$$ikF(k) \approx \left(\frac{1}{h} \sum_{j=-M}^{M} a_j e^{ijkh} \right) F(k). \tag{A.31}$$

This defines the effective wavenumber K of the finite difference scheme, where K is defined as

$$Kh = -i \sum_{j=-M}^{M} a_j e^{ijkh}. \tag{A.32}$$

The finite differencing scheme can then be optimized in the sense that the error between E between the wavenumber and the effective wave number is a minimum. The quantity E is defined as

$$\begin{aligned} E &= \int_{-pi/2}^{pi/2} |kh - Kh|^2 d(kh) && \text{(A.33)} \\ &= \int_{-pi/2}^{pi/2} \left| kh + i \sum_{j=-M}^{M} a_j e^{ijkh} \right|^2 d(kh) && \text{(A.34)} \\ & && \text{(A.35)} \end{aligned}$$

and the scheme is optimal in the sense that

$$\frac{\partial E}{\partial a_j} = 0 \qquad j = \text{-}M \text{ to } M. \tag{A.36}$$

Instead of using all of the coefficients in the previous expression to minimize the error, a combination of error minimization and traditional Taylor series truncation is used to identify the coefficients. Here, we implement the following to produce a fourth-order accurate scheme:

$$M = 3 \qquad a_0 = 0 \qquad \lim_{k \to 0} \frac{\partial(Kh)}{\partial(kh)} = 1 \qquad \lim_{k \to 0} \frac{\partial^3(Kh)}{\partial(kh)^3} = 1 \qquad \lim_{k \to 0} \frac{\partial E}{\partial a_3} = 0 \tag{A.37}$$

and compute the following spatially optimized scheme:

$$\frac{\partial f}{\partial x} \approx \frac{496-15\pi}{-5376+1890\pi}(f_{i+1}-f_{i-1}) + \frac{5632-1725\pi}{10752-3780\pi}(f_{i+2}-f_{i-2}) + \frac{272-85\pi}{-1792+630\pi}(f_{i+3}-f_{i-3}). \tag{A.38}$$

A comparison between the optimized scheme and the standard fourth- and sixth-order finite difference schemes is shown in Figure A.1. As seen in this comparison, the optimized scheme is more accurate than two classic schemes at higher wavenumbers.

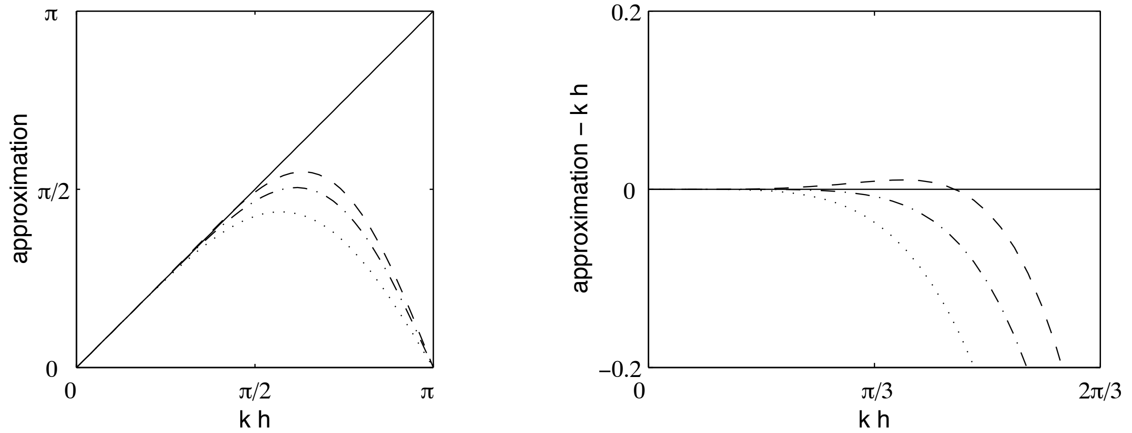


Figure A.1: Comparison in wavenumber approximation to the first derivative. (- - -) Optimized fourth-order finite difference scheme. (- · -) Sixth-order finite-difference scheme. (· · ·) Fourth-order finite-difference scheme.

Second Derivative Consider the following discretization of the second derivative:

$$\frac{\partial^2 f}{\partial x^2}(x) \approx \frac{1}{h^2}\sum_{j=-M}^{M} b_j f(x+jh). \tag{A.39}$$

Observe that we only consider stencils over odd numbers of grid points. Computing the Fourier transform of Eq. (A.39), we arrive at

$$-k^2F(k) \approx \left(\frac{1}{h^2} \sum_{j=-M}^{M} b_j e^{ijkh} \right) F(k). \tag{A.40}$$

This defines the effective wavenumber squared K^2 of the finite difference scheme, where K is defined as

$$K^2h^2 = -\sum_{j=-M}^{M} b_j e^{ijkh}. \tag{A.41}$$

The finite differencing scheme can then be optimized in the sense that the error E between the wavenumber squared and the effective wave number squared is a minimum. The quantity E is defined as

$$\begin{aligned} E &= \int_{-pi/2}^{pi/2} |k^2h^2 - K^2h^2|^2 d(kh) && \text{(A.42)} \\ &= \int_{-pi/2}^{pi/2} \left| k^2h^2 - \sum_{j=-M}^{M} b_j e^{ijkh} \right|^2 d(kh) && \text{(A.43)} \\ & && \text{(A.44)} \end{aligned}$$

and the scheme is optimal in the sense that

$$\frac{\partial E}{\partial b_j} = 0 \qquad j = \text{-}M \text{ to } M. \tag{A.45}$$

Instead of using all of the coefficients in the previous expression to minimize the error, a combination of error minimization and traditional Taylor series truncation is used to identify the coefficients.

Here, we implement the following:

$$M = 3 \tag{A.46}$$

$$\lim_{k \to 0} Kh = 0 \tag{A.47}$$

$$\lim_{k \to 0} \frac{\partial^2 (Kh)}{\partial (kh)^2} = 2 \tag{A.48}$$

$$\lim_{k \to 0} \frac{\partial^4 (Kh)}{\partial (kh)^4} = 0 \tag{A.49}$$

$$\lim_{k \to 0} \frac{\partial^6 (Kh)}{\partial (kh)^6} = 0 \tag{A.50}$$

$$\lim_{k \to 0} \frac{\partial E}{\partial b_3} = 0 \tag{A.51}$$

and compute the following spatially optimized scheme:

$$\begin{aligned} \frac{\partial^2 f}{\partial x^2} \quad \approx \quad & \frac{-50720 - 24975\pi + 19800\pi^2 - 2250\pi^3}{-194688 + 62370\pi} f_i \\ & + \frac{31888 - 3345\pi - 3300\pi^2 + 375\pi^3}{-43264 + 13860\pi} (f_{i+1} - f_{i-1}) \\ & + \frac{32224 - 7575\pi - 1320\pi^2 + 150\pi^3}{43264 - 13860\pi} (f_{i+2} - f_{i-2}) \\ & + \frac{53744 - 13095\pi - 1980\pi^2 + 225\pi^3}{-389376 + 124740\pi} (f_{i+3} - f_{i-3}). \end{aligned} \tag{A.52}$$

A comparison between the optimized scheme and the standard fourth- and sixth-order finite difference schemes is shown in Figure A.2. As for the first-derivative approximation, the optimized scheme for the second derivative is more accurate than two classic schemes at higher wavenumbers.

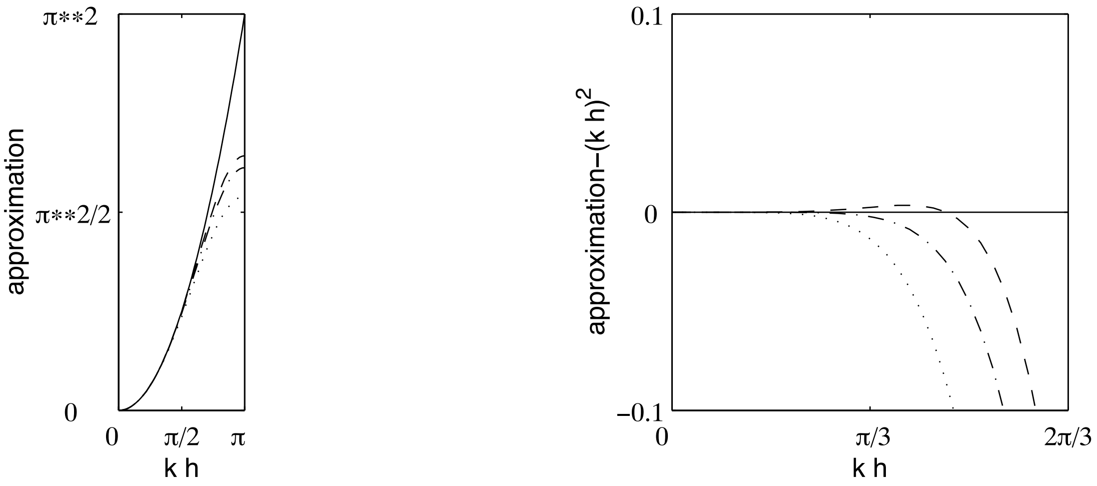


Figure A.2: Comparison in wavenumber approximation to the second derivative.// (- - -) Optimized fourth-order finite difference scheme. (- · -) Sixth-order finite-difference scheme. ($\cdots$) Fourth-order finite-difference scheme.

A.5 Implementation

Boundary stencils Derivative computation requires the use of different stencils, as the nature of the available data changes near the boundaries. In particular, the differencing scheme must switch from the primary scheme in the middle of the domain to an upwinding scheme at the domain boundary. In the *diffuser* code, the implicit formulation uses three different stencils for derivatives computed at nodes $i = \{1, n\}, \{2, n-1\}, \{3, \ldots, n-2\}$. The explicit formulation uses four different stencils for derivatives computed at nodes $i = \{1, n\}, \{2, n-1\}, \{3, n-2\}, \{4, \ldots, n-3\}$. The implicit formulation is illustrated in Figure A.3 and the explicit formulation is illustrated in Figure A.4.

As shown in Figure A.3, the implicit scheme switches from Padé sixth-order to Padé fourth-order and finally to upwinding, explicit third-order at the end node. As shown in Figure A.4, the explicit scheme switches from optimized fourth-order to centered fourth-order, upwinding fourth-order, and

Node	Scheme
1	upwind fifth
2	upwind fourth
3	centered fourth
4	optimized fourth
…	…
n-3	optimized fourth
n-2	centered fourth
n-1	upwind fourth
n	upwind fifth

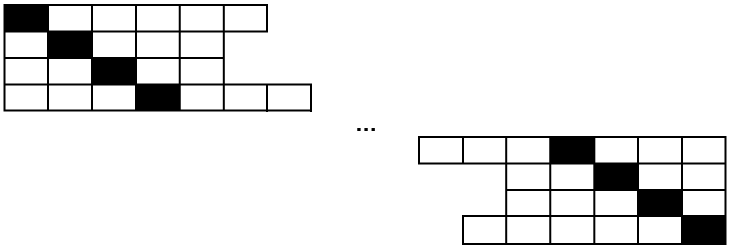


Figure A.3: Padé differencing formulation. The primary scheme is Padé sixth-order and the end-node scheme is upwinding (explicit) third-order.

finally to upwinding fifth-order at the end node. The longer stencils for the end nodes are used for higher accuracy; shorter stencils used at the end nodes, for this study, resulted in instability and the appearance of spurious modes.

Node	Scheme
1	upwind fifth
2	upwind fourth
3	centered fourth
4	optimized fourth
…	…
n-3	optimized fourth
n-2	centered fourth
n-1	upwind fourth
n	upwind fifth

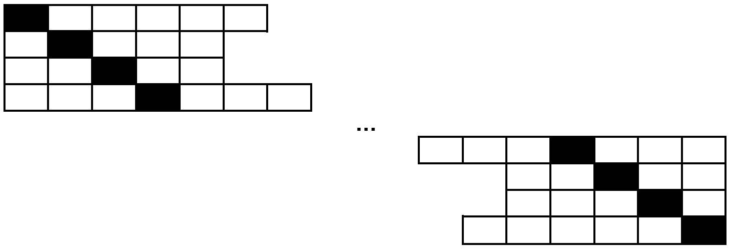


Figure A.4: Explicit differencing formulation. The primary scheme is optimized fourth-order.

Resulting Code The coefficient sets have been input into the code as follows:

```
TYPE (centered_scheme)                  :: &

   cent_pade6_1 = centered_scheme( 1.D0/3.D0 , 7.D0/9.D0  , 1.D0/36.D0,  0.D0      , 0.D0       ),    &

   cent_pade4_1 = centered_scheme( 1.D0/4.D0 , 3.D0/4.D0  , 0.D0      ,  0.D0      , 0.D0       ),    &

   cent_exp6_1  = centered_scheme( 0.D0      , 3.D0/4.D0 , -3.D0/20.D0, 1.D0/60.D0 , 0.D0       ),    &

   cent_exp4_1  = centered_scheme( 0.D0      , 2.D0/3.D0 , -1.D0/12.D0, 0.D0       , 0.D0       ),    &


   cent_pade6_2 = centered_scheme(2.D0/11.D0, -2.D0*(12.D0/11.D0+3.D0/44.D0),12.D0/11.D0,3.D0/44.D0, 0.D0),&

   cent_pade4_2 = centered_scheme( 1.D0/10.D0, -2.D0*6.D0/5.D0, 6.D0/5.D0, 0.D0    , 0.D0       ),    &
```

```
     cent_exp6_2  = centered_scheme( 0.D0      , -49.D0/18.D0, 3.D0/2.D0, -3.D0/20.D0, 1.D0/90.D0 ),    &
     cent_exp4_2  = centered_scheme( 0.D0      , -5.D0/2.D0  , 4.D0/3.D0, -1.D0/12.D0, 0.D0       ),    &

     cent_f2   = centered_scheme( 0.25D0, 0.75D0, 0.375D0, 0.D0, 0.D0 ),                   &
     cent_uf4  = centered_scheme( 1.D0, 1.D0, 0.D0, 0.D0, 0.D0 )

  TYPE (upwind_scheme)                    :: &
     upwind_exp3_1 = upwind_scheme(  (/-11.D0/6.D0 ,  3.D0, -3.D0/2.D0,  1.D0/3.D0 ,  0.D0     , 0.D0 /) ),&
     upwind_exp4_1 = upwind_scheme(  (/-25.D0/12.D0,  4.D0, -3.D0     ,  4.D0/3.D0 , -1.D0/4.D0, 0.D0 /) ),&
     upwind_exp3_2 = upwind_scheme(  (/ 2.D0       , -5.D0,  4.D0     , -1.D0      ,  0.D0     , 0.D0 /) ),&
     upwind_exp4_2 = upwind_scheme( (/ 35.D0/12.D0,-26.D0/3.D0,19.D0/2.D0, -14.D0/3.D0, -11.D0/12.D0, 0.D0 /))

!Define the optimized fourth-order explicit schemes:

  REAL(kind=2), parameter        :: opi = 3.1415926535897931D0
  REAL(kind=2), parameter        :: opta1 = (496.0D0-15.0D0*opi) / (-5376.0D0+1890.0D0*opi)
  REAL(kind=2), parameter        :: opta2 = (5632.0D0-1725.0D0*opi) / (10752.0D0-3780.0D0*opi)
  REAL(kind=2), parameter        :: opta3 = (272.0D0-85.0D0*opi) / (-1792.0D0+630.0D0*opi)

  REAL(kind=2), parameter        :: optb0 =(-50720.0D0-24975.0D0*opi+19800.0D0*opi*opi-2250.0D0*opi*opi*opi)&
                                            / (-194688.0D0+62370.0D0*pi)
  REAL(kind=2), parameter        :: optb1 =   (31888.0D0-3345.0D0*opi-3300.0D0*opi*opi+375.0D0*opi*opi*opi) &
                                            / (-43264.0D0+13860.0D0*opi)
  REAL(kind=2), parameter        :: optb2 =   (32224.0D0-7575.0D0*opi-1320.0D0*opi*opi+150.0D0*opi*opi*opi) &
                                            / (43264.0D0-13860.0D0*pi)
  REAL(kind=2), parameter        :: optb3 =   (53744.0D0-13095.0D0*opi-1980.0D0*opi*opi+225.0D0*opi*opi*opi) &
                                            / (-389376.0D0+124740.0D0*opi)

  TYPE (centered_scheme)                  :: &
     cent_expopt4_1  = centered_scheme( 0.D0, opta1 , opta2, opta3 , 0.D0    ),       &
     cent_expopt4_2  = centered_scheme( 0.D0, optb0 , optb1, optb2 , optb3      )
```

A.6 Error Analysis

The convergence properties for the Padé sixth-order, explicit sixth-order, and explicit-optimized fourth-order schemes are computed and compared. The model problem for this comparison is the computation of the function $f(x) = \cos(4x) - 1$ and its first- and second-derivatives over the domain $x \epsilon [0, 4]$. The primary and boundary stencils used are as described previously. The number of grid points used for the computation ranges from 21 points to 2001 points. For this comparison, the error is defined as the absolute-value of the difference between the computational and actual value at $x = 2.0$. In other words, the first-derivative error is computed as

$$\left(\frac{\partial f}{\partial x}\right)_{error} = \left| \left(\frac{\partial f}{\partial x}\right)_{computed} - \left(\frac{\partial f}{\partial x}\right)_{actual} \right|_{x=2.0} . \tag{A.53}$$

and the second-derivative error is computed as

$$\left(\frac{\partial^2 f}{\partial x^2}\right)_{error} = \left| \left(\frac{\partial^2 f}{\partial x^2}\right)_{computed} - \left(\frac{\partial^2 f}{\partial x^2}\right)_{actual} \right|_{x=2.0} . \tag{A.54}$$

The results of these computations are shown in Figure A.5.

As shown in Figure A.5, the order of the schemes is apparent from the slopes of the error functions. The increase in error at high resolution is attributed to roundoff error.

Comparison of *diffuser* results Finally, a comparison is made with computations from the *diffuser* code. The computations are of a two-dimensional flow through a straight duct. The

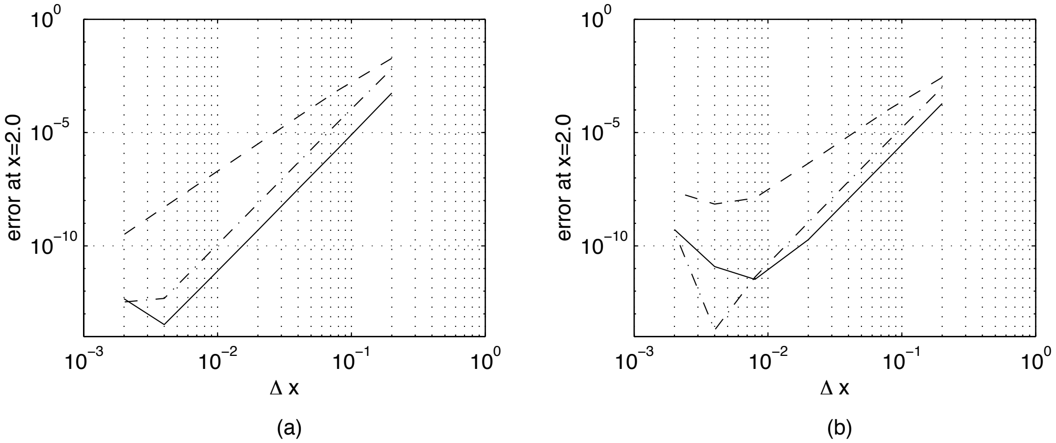


Figure A.5: Error analysis: (a) first derivative; (b) second derivative. (- - -) Optimized fourth-order finite difference scheme. (- · -) Explicit sixth-order finite-difference scheme. (-) Padé sixth-order finite-difference scheme.

domain is rectangular, with a streamwise length-to-transverse width ratio of 2:1 and a grid size of 41 x 21 points. The flow conditions are as follows: $M = 0.4$, $Re_{width} = 1000$, $Pr = 0.7$, and $\gamma = 1.4$. The boundary conditions are as follows: left inflow, right outflow, symmetry along the bottom, and wall conditions along the top. A Blasius boundary layer is included with an inlet displacement thickness of $\delta = 0.25$. Details of the simulation are found in the documentation for the code (Suzuki et al., 2004). It is noted that the extra features of the code, including acoustic forcing, suction, and upstream boundary-layer noise, are turned off for this comparison. The metric used for the comparison is a residual computed during the time-stepping routine. Computational results are included for cases with and without buffer zones and for cases with various differencing schemes. The cases studied are, for primary streamwise/transverse differencing, as follows: Padé/Padé; Padé/explicit; explicit/explicit; explicit/Padé/explicit. The Padé scheme corresponds to the sixth-order implicit scheme and the explicit scheme referes to the optimized fourth-order scheme. These results are shown in Figure A.6.

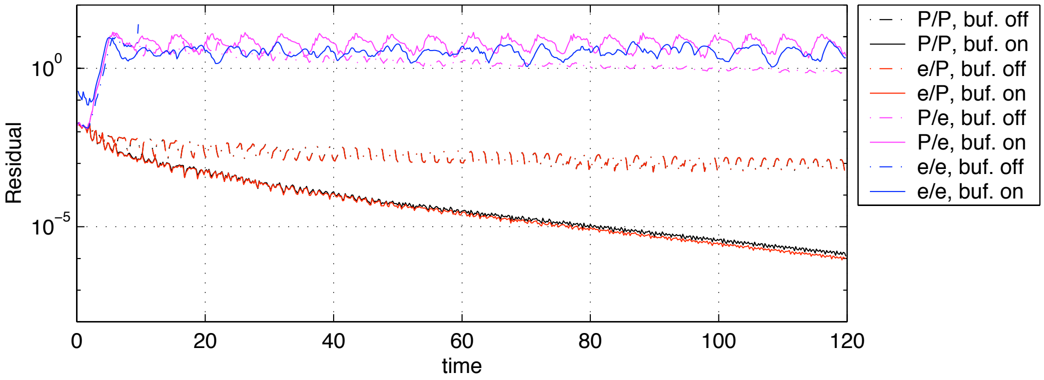


Figure A.6: Residual time-histories for the *diffuser* results. The streamwise / transverse differencing schemes are referenced as: P/P = Padé / Padé; e/P = explicit / Padé; P/e = Padé / explicit; e/e = explicit / explicit.

From Figure A.6, it is apparent that the full Padé and streamwise explicit/transverse Padé schemes compute the lowest residuals compared to the other cases. The use of the buffer is also correlated with lower computed residuals. The computation using the full explicit formulation diverged at an early time. Also note the high correlation between the full Padé and streamwise explicit/transverse Padé computations. From this comparison, the latter formulation is acceptable for computing flows with similar convergence to those computed from a fully implicit Padé scheme. This streamwise explicit/transverse Padé formulation will be used for future parallel implementation of the *diffuser* code, as the explicit formulation admits to a parallel, domain-splitting framework.

A.7 Concluding Remarks

Prior to parallelization of the *diffuser* code, new differencing schemes must be implemented that allow for domain-splitting. Classical and optimal explicit finite-differencing schemes were described, and a formulation using the optimal scheme is implemented into the code. Results from *diffuser* computations show that the new formulation compares well with the original implicit formulation. With this local (and accurate) formulation, the *diffuser* code is now ready for parallelization.

Appendix B

Parallel Implementation of the *diffuser* Code

Parallelization was performed according to the Message Passing Interface (MPI) standard because of the popularity and ease-of-use of MPI in applications requiring distributed processing or memory. This programming environment is well documented by Pacheco and Ming (1997) and at Argonne National Laboratories (2001). As the original `diffuser` code is written in Fortran 90, the Fortran 90 flavor of MPI is used.

B.1 Domain-Splitting Framework

The original spatial integration algorithms for the `diffuser` code were implicit in both the streamwise and transverse directions. To facilitate parallelization, explicit algorithms were chosen to replace the original algorithms in one direction (Fung, 2001). The direction chosen is the streamwise direction. Parallelization is now straightforward; individual processors are made responsible for a streamwise portion of the entire computational domain. Because the streamwise integration scheme is explicit, each processor can almost compute its portion of the flow independently. Each processor must communicate information across the boundaries of their local domains to maintain continuity in computing the flow physics. A schematic of this framework is shown in Figure B.1.

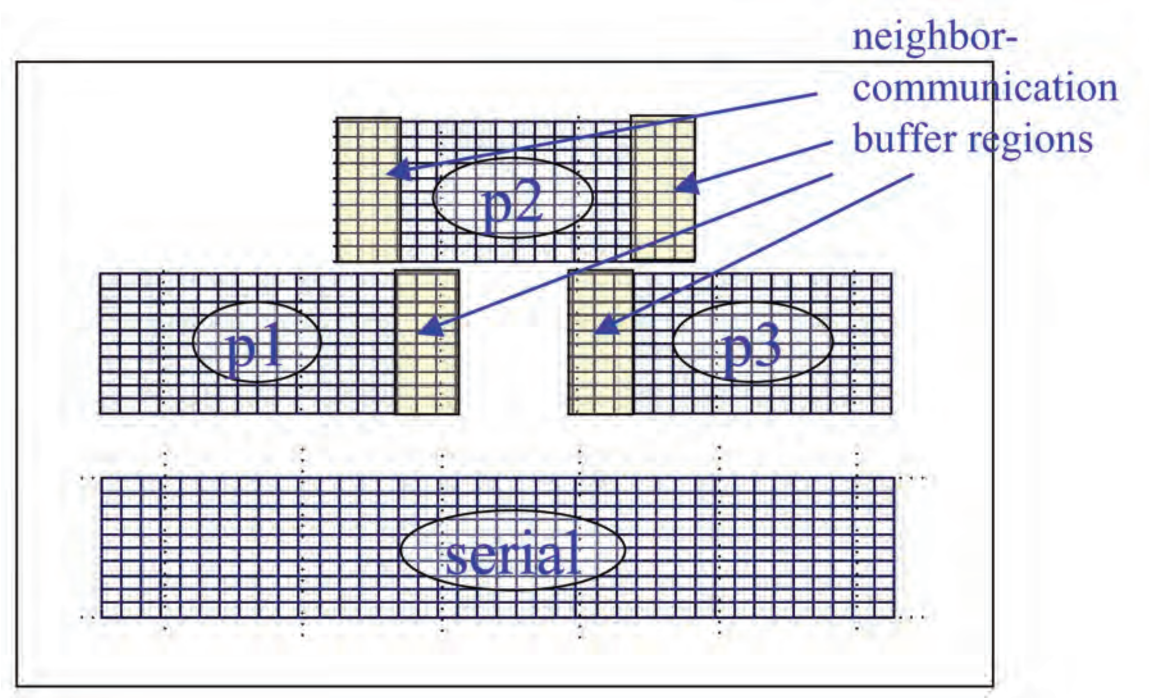


Figure B.1: Domain-splitting framework. The total computational task is distributed to the parallel processors in a streamwise fashion.

As shown in this figure, the local domain for each processor is extended so that processors responsible for neighboring regions can communicate their boundary information. At each step in the spatial integration routine, these neighbor-communication regions are synchronized. There are communications and computational costs associated with computing the flow in these communication regions. Currently, the benefits of having a simple parallel algorithm, whose implementation is straightforward, outweigh these costs.

B.2 Setup: the `mpi_basic` Module

This module contains the relevant code for initializing and terminating the processing environment.

`initialize_mpi` routine Three system calls are made here:

```
   call MPI_INIT(mpi_ierr)

   call MPI_COMM_RANK(MPI_COMM_WORLD, mpi_rank, mpi_ierr)

   call MPI_COMM_SIZE(MPI_COMM_WORLD, mpi_size, mpi_ierr)
```

This subroutine is called at the beginning of the program execution. The first call initializes the environment. The second call identifies the particular processor by its rank (the first processor is rank 0, the second processor is rank 1, and so on). The third call identifies the total number of processors available for the computation. The arguments in the calls are as follows: *MPI_COMM_WORLD*, the address for the entire processing environment, *mpi_rank*, the local rank (integer), *mpi_size*, the system size (integer), and *mpi_ierr*, an error return variable (integer array).

`finalize_mpi` routine Only one system call is made here:

```
  CALL MPI_Finalize(mpi_ierr)
```

This call simply terminates the entire environment, and the call is made at the end of the program execution.

`mpi_indices(n)` **routine** Each processor calls this routine to identify its local processing domain. The argument n refers to the global domain size. First, the domain is divided up according to the number of available processors. Currently, the division is based on the first dimension (streamwise) of the domain. The algorithm is very simple and may display pathological behavior depending on the domain size; please examine this subroutine if the computed local domain sizes are nonsensical. The variables *mpi_imin_nopad* and *mpi_imax_nopad* contain the limits of the local domain:

```
test=DBLE(n(1))/(DBLE(mpi_size))
divlow = floor(test)
divhigh = ceiling(test)
mpi_imin_nopad = mpi_rank*divhigh+1
mpi_imax_nopad = (mpi_rank+1)*divhigh
```

Now, the (communication) padding is added to the local domain size. Note that padding is only added to the local domain boundaries that are not coincident with the global domain boundaries. This should make sense, because communication is not required across the global domain boundary; communication is only required between processors that require neighboring (in the parallel and local domain sense) information. The following code documents this algorithm and the definition of the variables *mpi_imin* and *mpi_imax* for the domain local to the processor.

```
IF (mpi_rank == 0) THEN
 padminus = 0
 padplus  = padding
```

```
END IF

IF (mpi_rank == mpi_size-1) THEN
 mpi_imax_nopad = n(1)                 ! avoids some pathological behavior
                                       ! from domain division
 padminus = padding
 padplus  = 0
END IF

IF (mpi_rank .gt. 0 .and. mpi_rank .lt. mpi_size-1) THEN
 padminus = padding
 padplus  = padding
END IF

 mpi_imin = mpi_imin_nopad - padminus  !subtract stencil length (communication)
 mpi_imax = mpi_imax_nopad + padplus   !add the stencil length (communication)
```

Finally, the domain-size variable n is redefined to reference the local domain size:

```
n(1) = mpi_imax - mpi_imin + 1
```

Note that each processor is only aware of its local domain size. This information is not communicated to the other processors.

B.3 Communication: the `mpi_advanced` Module

This module contains the relevant subroutines for transferring information from one processor to another. The communication style implemented for this code is very simple and straightforward. The processor of rank i communicates only with processors $i-1$ and $i+1$. The communication tasks are separated accordingly.

Communication using MPI is performed with synchronous sending. This type of sending is a communication where a processor starts to send a message regardless of the status of the recepient. However, the message sending will not complete unless a receive is reported by the recipient. Note that this method of communication is nonlocal; communication is dependent on the status of more than the local processor. The communication is buffered, but the synchronous mode is beneficial in that the buffer is guaranteed to be cleared before new messages are sent.

the `mpi_transfer(q, nvars)` routine This routine performs the necessary communication between processors. The argument q is the state variable to be communicated, and the argument $nvars$ is the dimension of q. Note, dimension does not refer to the domain size, but to the type of information (density, momentum, for instance). Observe that processors of rank 0 and mpi_size-1 each perform communication with one other processor (rank 1 and mpi_size-2, respectively). The other processors i communicate with processors $i-1$ and $i+1$, as shown in the following code:

```
if(mpi_rank .ne. mpi_size-1) THEN
 CALL mpi_transfer_NandNplus1(q, nvars)
```

```
end if


if(mpi_rank .ne. 0) THEN
 CALL mpi_transfer_NandNminus1(q, nvars)
end if
```

The first call is to a subroutine that handles communication between processors of rank i and $i+1$, and the second call is to a subroutine that handles communication between processors or rank i and $i-1$.

the `mpi_transfer_NandNplus1(q, nvars)` subroutine This handles communication from processors i to processors $i+1$. The arguments are the same as those in the `mpi_transfer(q, nvars)` routine. First, the data (for the variables q, over the local subdomain reserved for neighboring communication) is reorganized into a one-dimensional array:

```
DO i1 = 1,nvars
 DO i2 = 1,padding
  qtemp(padding*(i1-1)*n(2)+(i2-1)*n(2)+1: padding*(i1-1)*n(2)+i2*n(2)) &
        = q(i1)%f(n(1)-2*padding + i2,1:n(2))
 END DO
END DO
```

In this routine, processor i first posts a send message to the processor $i + 1$, and then processor i posts a receive message to the processor $i + 1$. Therefore, this routine governs half of two distinct communications. The system calls are as follows:

```
  tag1  = mpi_rank                          !identifier for outgoing message
  tag2  = 2*mpi_size+mpi_rank               !identifier for incoming message


!SEND AT N TO N+1
  CALL MPI_SSEND(qtemp(1),nvars*padding*n(2), MPI_DOUBLE_PRECISION, &
                mpi_rank + 1, tag1, MPI_COMM_WORLD, mpi_ierr)


!RECEIVE AT N FROM N+1
  CALL MPI_RECV(qtemp(1), nvars*padding*n(2), MPI_DOUBLE_PRECISION, &
                mpi_rank + 1, tag2+1, MPI_COMM_WORLD, status, mpi_ierr)
```

The arguments are as follows: $qtemp(1)$ refers to the address of the first value in the $qtemp$ array; $nvars * padding * n(2)$ is the size of the message (integer); $MPI\_DOUBLE\_PRECISION$ is the data type for the message; $mpi\_rank + 1$ is the rank of processor $i + 1$ (integer), $tag1$ and $tag2 + 1$ are tag identifiers for the messages (to keep track of multiple messages in the system) (integer); MPI_COMM_WORLD is the global system address; $status$ and mpi_ierr are integer arrays corresponding to error and status checking.

Finally, the incoming data is redistributed to the corresponding state variables q (over the local communication subdomain):

```
  DO i1 = 1,nvars
   DO i2 = 1,padding
    q(i1)%f(n(1)-padding+i2,1:n(2)) &
        = qtemp(padding*(i1-1)*n(2)+(i2-1)*n(2)+1: padding*(i1-1)*n(2)+i2*n(2))
   END DO
  END DO
```

the `mpi_transfer_NandNminus1(q, nvars)` **subroutine** This handles communication from processors i to processors $i-1$. Communication is very similar to the `mpi_transfer_NandNminus1(q, nvars)` routine. Observe that, complementary to the `mpi_transfer_NandNminus1(q, nvars)` routine, processor i first posts a receive to processor $i-1$, and then processor i posts a send to processor $i-1$. Reorganization and redistribution of the *qtemp* and q variables are performed accordingly. The main portion of the routine is listed as follows:

```
  tag1  = mpi_rank
  tag2  = 2*mpi_size+mpi_rank

!RECEIVE AT N FROM N-1
  CALL MPI_RECV(qtemp(1), nvars*padding*n(2), MPI_DOUBLE_PRECISION, &
                mpi_rank - 1, tag1-1, MPI_COMM_WORLD, status, mpi_ierr)

  DO i1 = 1,nvars
   DO i2 = 1,padding
```

```
    q(i1)%f(i2,1:n(2)) &
      = qtemp(padding*(i1-1)*n(2)+(i2-1)*n(2)+1: padding*(i1-1)*n(2)+i2*n(2))
   END DO
  END DO


! SEND AT N TO N-1
  DO i1 = 1,nvars
   DO i2 = 1,padding
    qtemp(padding*(i1-1)*n(2)+(i2-1)*n(2)+1: padding*(i1-1)*n(2)+i2*n(2)) &
         = q(i1)%f(i2+padding, 1:n(2))
   END DO
  END DO


  CALL MPI_SSEND(qtemp(1),nvars*padding*n(2), MPI_DOUBLE_PRECISION, &
                mpi_rank - 1, tag2, MPI_COMM_WORLD, mpi_ierr)}
```

B.4 Specifics

This section documents revisions of the original *diffuser* code to accomodate the parallel framework.

`bc` **module** Considering the boundary on a two-dimensional domain, not every processor is required to compute boundary information. In particular, only processors 0 and *mpi_size - 1* must compute certain quantities such as the $bc1$ and $bc2$ variables. Flow control statements are included in the `poinsotlele` subroutine and are not listed here.

`independents` module These revisions document a global domain definition for each processor. In addition to defining local grid parameters, as read from a preprocessing output file, global domain parameters are defined. In this sense, a processor i knows about the entire grid and grid metrics. The code for storing global grid information is almost identical to the code for storing the local grid information (which is in turn identical to the serial code) and is not included here. It is important to note that each processor requires knowledge of global geometric parameters like stagnation pressure measurement location. It is for this reason that any global information is stored.

`ddt` module The revisions to this subroutine are straightforward. First, the `redefine_met` subroutine contains global-local grid specification statements of the type found in the `independents` module. Algorithms for computing the stagnation pressure ratio and skin friction are included in a later subsection. The remaining comments in this subsection refer to the `ddt` subroutine.

Neighbor communication Calls to `mpi_transfer` are made at the beginning of the routine, after fluxes are computed, after the `divflux` subroutine call, and after the Laplacian viscous terms are computed.

Acoustic forcing An acoustic forcing amplitude parameter, based on the density at a downstream location, is communicated from processor *mpi_size - 1* to all other processors using a broadcast system call:

```
CALL MPI_BCAST(afconst, 1, MPI_DOUBLE_PRECISION, &
               mpi_size-1, MPI_COMM_WORLD, mpi_ierr)
```

Here, the arguments are similar in type and syntax to the synchronous send system call.

Reorganization The code for computing skin friction and stagnation pressure ratio have been removed from this subroutine and reorganized as distinct subroutines.

`dependents` **module** The revisions to this module contain revisions corresponding to local-domain grid specifications. These revisions are found in the `setup_ic` routine.

For the Blasius boundary-layer or Couette-flow solution, the pressure computation requires information about the local grid size. The code is included as follows:

```
 DO i = 1,n(1)
    q(4,1)%f(i,1:n(2)) = 1.D0/gamma + 5.D-1 * &
                 (mach**2-((upot(1)%f(i+mpi_imin-1,1:n_total(2)))**2 &
                           +(upot(2)%f(i+mpi_imin-1,1:n_total(2)))**2))
 END DO
```

Also, as the pressure change is computed and added, similar statements arise:

```
DO i = 1, n(1)
 DO j = 1, n(2)
```

```
          q(4,1)%f(i,j)=q(4,1)%f(i,j)+dp(i+mpi_imin-1)

         END DO

        END DO
```

Furthermore, the initial field for the acoustic pulse test case is computed from the global grid information:

```
             DO i = 1, n(1)

              DO j = 1, n(2)

               q(1,2)%f(i,j) = 1.0D-4 * &

                     DEXP( -50.D0*( (gorig%x(1)%f(i+mpi_imin-1,j)-xcen)**2 + &

                                    (gorig%x(2)%f(i+mpi_imin-1,j)-ycen)**2 ))

              END DO

             END DO
```

Computing the temperature and density near the end of the routine follows similarly, as it is computed using the potential flow solution, which is stored on a global domain-sized array:

```
! evaluation of temperature (which I temporarily store in q(3)) from T0=const.

   DO i=1,n(1)

     q(3,1)%f(i,1:n(2)) = tw-5.D-1 *((upot(1)%f(i+mpi_imin-1,1:n_total(2)))**2 +&

                                      (upot(2)%f(i+mpi_imin-1,1:n_total(2)))**2)

   END DO

! density
```

```
q(3,1)%f = gamma/gm1*q(4,1)%f/q(3,1)%f
DO i=1,n(1)
  q(4,1)%f(i,1:n(2)) =q(4,1)%f(i,1:n(2))/gm1 + 5.D-1*q(3,1)%f(i,1:n(2))*&
                     ( (upot(1)%f(i+mpi_imin-1,1:n_total(2)))**2 +&
                       (upot(2)%f(i+mpi_imin-1,1:n_total(2)))**2 )
  q(1,1)%f(i,1:n(2)) =upot(1)%f(i+mpi_imin-1,1:n_total(2))*q(3,1)%f(i,1:n(2))
  q(2,1)%f(i,1:n(2)) =upot(2)%f(i+mpi_imin-1,1:n_total(2))*q(3,1)%f(i,1:n(2))
END DO
```

Stagnation pressure ratio and skin friction

Revisions in the `ddt` module Stagnation pressure ratio and skin friction routines have been converted into distinct subroutines (the `prcv_compute` and `cf_compute` subroutines, respectively) within the `ddt` module. The routines are virtually identical to those found in the serial version of the code. Neighbor-communication system calls using the `mpi_transfer` subroutine are made at the beginning of these subroutines. This completes the revision for the code to compute skin friction.

Only the processors that are responsible for computing the stagnation pressure ratio are active in this subroutine. These processors are those of rank *prcv_upstream_proc* and *prcv_downstream_proc*. Determination of these processors is documented in the next paragraph. Computation on these processors goes as that of the serial code. At the end of the subroutine, the two processors broadcast the quantities $pt1$, $pt2$, $h1$, and $h2$ to the other processors using the `mpi_broadcast` system call. Then every processor computes the stagnation pressure ratio from these four quantities. This code was written for simplicity and readability and so it is recognized that there is redundancy in this algorithm.

Revisions in the `post_processing` module The algorithm for determining the processors responsible for computing stagnation pressure ratio is found in the `initialize_prcv` subroutine in this module. Each processor stores a logical array *up* and *down* (of dimension equal to the number of processors) and an IF statement is used to determine the respective processors:

```
IF (g%x(1)%f(padminus+1,1) .le. xtp1 .and. &
                      xtp1 .le. g%x(1)%f(n(1)-padplus,1) ) THEN
 up(mpi_rank+1) = .TRUE.
ELSE
END IF


IF(g%x(1)%f(padminus+1,1).le.xtp2.and.xtp2.le.g%x(1)%f(n(1)-padplus,1))THEN
 down(mpi_rank+1) = .TRUE.
ELSE
END IF
```

Here, the quantities $xtp1$ and $xtp2$ are the streamwise location of the upstream and downstream pressure probe locations that are defined in an input file. Each processor then reports their findings to the other processors:

```
   DO i = 1,mpi_size
    CALL MPI_BCAST(i-1, 1, MPI_INTEGER, i-1, MPI_COMM_WORLD, mpi_ierr)
```

```
 CALL MPI_BCAST(up(i), 1, MPI_LOGICAL, i-1, MPI_COMM_WORLD, mpi_ierr)
END DO


DO i = 1, mpi_size
 CALL MPI_BCAST(i-1, 1, MPI_INTEGER, i-1, MPI_COMM_WORLD, mpi_ierr)
 CALL MPI_BCAST(down(i), 1, MPI_LOGICAL, i-1, MPI_COMM_WORLD, mpi_ierr)
END DO
```

Following this communication, every processor is aware of which processors have their *up* and *down* settings as .true.; in other words, each processor is aware of which processors are now responsible for computing the stagnation pressure ratio:

```
DO i = 1, mpi_size+1
 IF(up(i) == .TRUE.) THEN
  prcv_upstream_proc = i-1
 END IF
 IF(down(i) == .TRUE.) THEN
  prcv_downstream_proc = i-1
 END IF
END DO
```

Probe output Each processor is responsible for the output of data for probes whose locations are found in its local domain. The selection process for this responsibility is similar to that for the stagnation pressure ratio upstream and downstream locations. The code for this selection process is found in the initialize_probes subroutine. A logical array stores values corresponding to whether a processor is responsible for a probe's data:

```
IF (i .ge. mpi_imin+padminus+1 .and. i .le. mpi_imax-padplus ) THEN
 prbflag(probei) = .TRUE.
END IF
```

A flow control statement using the *prbflag* variables is found in the write_probes subroutine for handling the actual data output. The actual write command is

```
WRITE(probe(i)%funit,fmt='(6E16.6)') time, &
     ( q(j,1)%f( prb(i)%ip(1)-mpi_imin+1, prb(i)%ip(2) ), j=1,5 )
```

Currently, the output filename for a probe file is prbNNN_MM.dat, where NNN is the probe number and MM is the processor rank.

Post-processing file IO The revisions to the post-processing routines are very straightforward. Each processor writes to its own restart, tecplot-compatible, and skin-friction files. Processors responsible for probe and stagnation pressure ratio data output to their own files. The data for tecplot and skin-friction files represents only the domain without the neighbor-buffer subdomains. A single example is given here, for the tecplot-file data output:

```
WRITE(plot3df%funit) n(1)-padminus-padplus,n(2),5
WRITE(plot3df%funit) (((SNGL(q(ivar,1)%f(i,j)),i=padminus+1, &
                                   n(1)-padplus),j=1,n(2)), ivar=1,5)
```

B.5 General Revisions to the *diffuser* Code

This section summarizes major revisions in the code that are not directly relevant to the parallelization effort. These include changes in the algorithms for input file processing and parameter definition, the *ddt* subroutine, and the introduction of a *testing* module.

B.6 `parameters` Module

The `initialize_parameters` routine has been rewritten to split file processing and parameter definition tasks as follows: general, runtime, buffer, suction, upstream noise, acoustic forcing, filtering, and compressor parameters. The parameter namelist has been divided into corresponding individual namelists. The `parameter.inp` file is also divided into the files `general.inp`, `runtime.inp`, `buffer.inp`, `suction.inp`, `upsnoise.inp`, `acforcing.inp`, `filter.inp`, and `compressor.inp`. Logical flags are now defined for each group. Routines in the other modules use these flags to determine whether algorithms for those groups should be computed.

A new subroutine called `additional_parameters` contains the algorithms for computing π, α, ν, $\gamma - 1$, wall temperature tw as found previously in the `initialize_parameters` routine. Also, this new subroutine contains algorithms for computing *conductivity*, $nuby3$, and *ttrd* as found originally in the `ddt` routine.

B.7 independents Module

The algorithms for providing the various functionality to the code, such as upstream boundary-layer noise, downstream acoustic forcing, buffer regions, filters, and compressors, are taken out of the **initialize_independents** subroutine and are implemented as individual subroutines. The new subroutine names are **setup_upsnoise**, and so on. The code for these algorithms is otherwise unchanged.

B.8 testing Module

This new module is defined to include most of the various debugging and testing subroutines written for the program. These include the existing **test_cfl** routine for computing the cfl condition and the maximum allowable timestep. New subroutines are included as follows:

test_mpi_indices Calls the **mpi_indices** routine to verify that the spatial discretization is valid.

test_mpi_packing Communications are made to verify the sending of 2-D array data as 1-D array data.

test_mpi_transfer Communications are made to verify the sending and receiving of 2-D data. Specifically, the q state variables are passed across processors to validate the accuracy of processor-to-processor communication.

test_print_q A simple routine for writing the state variables q from the original **2-D+State_Length** to a strictly 3-D format.

Please refer to the code for listings of these routines.

B.9 Results

B.10 Comparision with Serial *diffuser* Results

Computations were performed using the serial and parallel codes to simulate a flow. The flow parameters are as follows: $Re = 500$, $M = 0.65$. The domain is rectangular at a resolution of 41 x 21 with boundary conditions as follows: nonreflecting inflow at the upstream end, nonreflecting outflow at the downstream end, symmetry along one side of the flow and a wall boundary on the other side. The initial condition contains a Blasius boundary layer with inflow boundary-layer displacement thickness $\delta = 0.05$.

The computations were performed with a time step of 0.007 for ten time steps. At the end of the computation, the binary output files (containing the state variable information) were compared using the UNIX diff utility. The utility reported no differences in the output files over the entire computation. From this comparison, it is reasonable to conclude that the serial and parallel programs would compute identical flows given the same initial conditions, at least to the (single) precision of the written output files.

B.11 Benchmark Timing Study

The parallel program was tested using a Hewlett-Packard Exemplar V2500 system (a Caltech CACR facility). The program was run on this system using between 2-32 nodes to examine speedup behavior. The flow to be computed corresponds to a $Re = 10,000$ flow on a diffuser (area ratio of 1.4) at a Mach number of $M = 0.65$. The grid resolution is 1001x251. These results after computing 800 time steps are shown in Figure B.2.

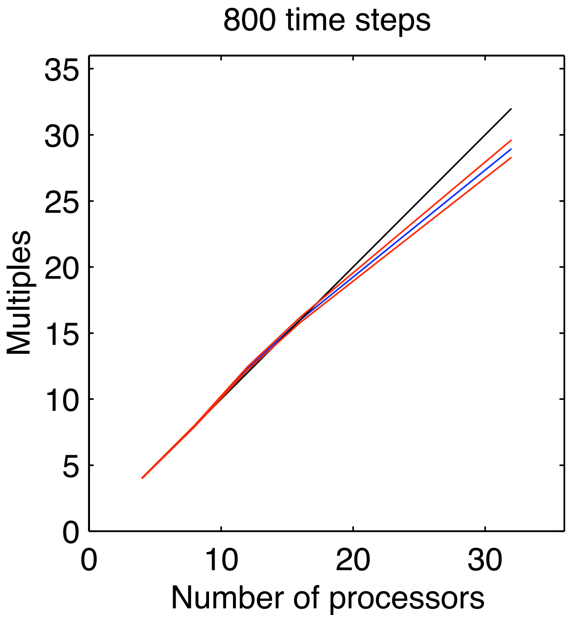


Figure B.2: Timing study on the V2500 for the parallel code. The measure is defined as the multiple of the required runtime for a 2-node computation.

The measure used in the timing study is defined as the multiple of the time required to perform a computation on two nodes. The error bars are included in the figure because the computational times were rounded to the nearest minute upon report to the user. As shown in this figure, the speedup behavior is very close to the theoretical limit. Note that extra computations are required for the neighbor-buffer regions. It is then clear that increasing the number of nodes for the computation would result in a larger total computational domain. This extra computational requirement may be responsible for the deviation of the speedup behavior away from the theoretical limit.

B.12 Program Compilation and Execution

Program compilation is performed using the `mpif90` utility usually offered in the MPI distribution.

Program execution is very straightforward in the MPI environment. The following are the main commands for either the HP V2500 or Beowulf cluster (mixing), found in the CACR network (see CACR, 2001).

HP V2500 mpirun -np *number of nodes* .../a.out

Beowulf (mixing) mpirun_rsh -np *number of nodes N*??? ... *N*??? .../a.out

Preparing the file directories and grid generation is similar to that for the serial version of the code (Suzuki et al., 2004).

The command given for mixing is for the MVICH (ServerNet) protocols, whose performance has not been thoroughly tested. A similar command is given that uses standard Ethernet protocols. Please consult the CACR documentation (see CACR, 2001) for further information.

B.13 Bugs

- Output to the `diffuser.out` file, for reporting initialization and post-processing steps, is not fully functional in the parallel implementation. The processors compete for access to the file, and so not all of the intended information is written to this file.

- Output to the probe files. Not all of the probe files are being written. There is probably an error in the flow control algorithm for selecting processor responsibilities upon reading the `probe.inp` file.

Bibliography

M. Amitay, D. Pitt, and A. Glezer. Separation in duct flows. *J. Aircraft*, 39:616–620, 2002.

D. D. Apsley and M. A. Leschziner. Advanced turbulence modelling of separated flow in a diffuser. *Flow Turbulence and Combustion*, 63:81–112, 1999.

Argonne National Laboratories. MPICH-a portable implementation of MPI, 2001. http://www-unix.mcs.anl.gov/mpi/mpich.

L. Back and R. Cuffel. Experimental investigation of turbulent wall-jets in the presence of adverse pressure gradients in a rectangular diffuser. *Intl. J. Heat Mass Transfer*, 25:871–887, 1982.

M. Ben Chiekh, J. C. Bera, and M. Sunyach. Synthetic jet control for flows in a diffuser: vectoring, spreading and mixing enhancement. *J. Turbulence*, 4, 2003.

M. Berger and J. Oliger. Adaptive mesh refinement for hyperbolic partial differential equations. *J. Comput. Phys.*, 53:484–512, 1984.

H. S. Bhat, R. C. Fetecau, J. E. Marsden, and M. West. Lagrangian averaged fluids. Submitted to SIAM J. Multiscale Modeling and Simulation, 2003.

T. J. Bogar, M. Sabjen, and J. C. Kroutil. Characteristic frequencies of transonic diffuser flow oscillations. *AIAA Journal*, 21:1232–1240, 1983.

A. Brandt. Multi-level adaptive solutions to boundary value problems. *Math. Comput*, 31:333–390, 1977.

L. Brillouin. Remarques sur la mecanique ondulatoire. *J. Phys. Radium*, 7:353–368, 1926.

A. Brunn and W. Nitsche. Separation control by periodic excitation in a turbulent axisymmetric diffuser flow. *J. Turbulence*, 4, 2003.

CACR. Network and resource documentation, 2001. http://www.cacr.caltech.edu.

B. D. Coller. Vortex model for control of diffuser pressure recovery. In *Proceedings of the 39th IEEE Conference on Decision and Control*, Sydney, Australia, 2000.

B. D. Coller, B. R. Noack, S. Narayanan, A. Banaszuk, and A. I. Khibnik. Reduced-basis model for active separation control in a planar diffuser flow. In *Proceedings of the 30th AIAA Fluid Dynamics Conference and Exhibit*, Denver, Colorado, U.S.A., June 19–22, 2000. AIAA Paper 2000-2562.

T. Colonius and S. Pirozzoli. The *diffuser* simulation. Caltech internal document, 2002.

Commissariat à L'Energie Atomique, 2001. http://www-fusion-magnetique.cea.fr/gb/fusion/physique/trajectoire.htm.

H. S. Dou and S. Mizuki. Analysis of the flow in vaneless diffusers with large width-to-radius ratios. *Journal Of Turbomachinery—Transactions Of The ASME*, 120(1):193–201, 1998.

W. E and B. Engquist. The heterogeneous multiscale methods. *Comm. Math. Sci.*, 1:87–132, 2003.

S. H. Feakins, D. G. MacMartin, and R. M. Murray. Dynamic separation control in a low-speed asymmetric diffuser with varying downstream boundary condition. In *Proceedings of the 33rd*

AIAA Fluid Dynamics Conference and Exhibit, Orlando, Florida, June 23-26, 2003. AIAA Paper 2003-4161.

R. W. Fox and S. J. Kline. Flow regime data and design methods for curved subsonic diffusers. *ASME D: J. of Basic Engineering*, 84:303–312, 1962.

R. W. Fox, A. T. McDonald, and P. J. Pritchard. *Introduction to Fluid Mechanics.* Wiley, 2004.

J. B. Freund. Proposed inflow/outflow boundary condition for direct computation of aerodynamic sound. *AIAA Journal*, 35:740–742, 1997.

J. Fung. Modifications to derivative computation in the *diffuser* code. Caltech internal document, 2001.

C. W. Gear. *Numerical Initial Value Problems in Ordinary Differential Equations.* Prentice-Hall, 1971.

C. W. Gear and I. G. Kevrekidis. Projective methods for differential equations. *SIAM J. Sci. Comp.*, 24(4):1091–1106, 2003a.

C. W. Gear and I. G. Kevrekidis. Telescopic projective methods for parabolic differential equations. *J. Comput. Phys.*, 197(1):95–109, 2003b.

C. W. Gear and I. G. Kevrekidis. Computing in the past with forward integration. *Phys. Lett. A*, 321(5-6):335–343, 2004.

C. W. Gear, I. G. Kevrekidis, and C. Theodoropoulos. “Coarse” integration/bifurcation analysis via microscopic simulators: micro-galerkin methods. *Computers and Chemical Engineering*, 26 (7-8):941–963, August 15 2002.

C. W. Gear, J. Li, and I. G. Kevrekidis. The gap-tooth method in particle simulations. *Phys. Lett. A*, 316(3-4):190–195, 2003.

A. H. Gibson. On the resistance to flow of water through pipes or passages having divergent boundaries. *Royal Society of Edinburgh Trans.*, 48(1):97–116, 1911–1912.

A. Glezer and M. Amitay. Synthetic jets. *Ann. Rev. Fluid Mech.*, 34:503–529, 2002.

J. W. Hamstra, D. N. Miller, P. P. Truax, B. A. Anderson, and B. J. Wendt. Active inlet flow control technology demonstration. *Aeronaut. J.*, 104:473–479, 2000.

D. D. Holm, J. E. Marsden, and T. S. Ratiu. Euler-poincaré models of ideal fluids with nonlinear dispersion. *Phys. Rev. Lett.*, 349:4173–4177, 1998.

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

C. A. Holzhauser and L. P. Hall. Exploratory investigation of the user of area suction to eliminate air-flow separation in diffusers having large expansion angles. NACA TN 3793, NACA, 1956.

G. Hummer and I. G. Kevrekidis. Coarse molecular dynamics of a peptide fragment: Free energy, kinetics, and long-time dynamics computations. *J. Chem. Phys.*, 118(23):10762–10773, 2003.

G. Iaccarino. Predictions of a turbulent separated flow using commercial cfd codes. *Journal Of Fluids Engineering—Transactions Of The ASME*, 123(4):819–828, 2001.

H.-J. Kaltenbach, M. Fatica, R. Mittal, T. S. Lund, and P. Moin. Study of flow in a planar asymmetric diffuser using large-eddy simulation. *J. Fluid Mech.*, 390:151–185, 1999.

I. G. Kevrekidis, C. W. Gear, and G. Hummer. Equation-free: The computer-aided analysis of comptex multiscale systems. *AIChE Journal*, 50(7):1346–1355, 2004.

I. G. Kevrekidis, C. W. Gear, J. M. Hyman, P. G. Kevrekidis, O. Runborg, and C. Theodoropoulos. Equation-free multiscale computation: enabling microscopic simulators to perform system-level tasks. *Communications in the Mathematical Sciences*, 1:715–762, 2003.

A. I. Khibnik, S. Narayanan, C. A. Jacobson, and K. Lust. Analysis of low dimensional dynamics of flow separation. In D. Henry and A. Bergeon, editors, *Continuation Methods in Fluid Dynamics*, number 74 in Notes on Numerical Fluid Mechanics, pages 167–178. Vieweg, Wiesbaden, 2000.

P. Koumoutsakos. Active control of vortex-wall interactions. *Physics of Fluids*, 9(12):3808–3816, 1997.

H. A. Kramers. Wellenmechanik und halbzahlige quantisierung. *Z. Phys.*, 39:828–840, 1926.

A. H. M. Kwong and A. P. Dowling. Unsteady flow in diffusers. *ASME I: J. of Fluids Engineering*, 116:842–847, 1994.

S. K. Lele. Compact finite difference schemes with spectral-like resolution. *J. Comput. Phys.*, 103: 16 – 42, 1992.

J. Li, P. G. Kevrekidis, C. W. Gear, and I. G. Kevrekidis. Deciding the nature of the “coarse equation” through microscopic simulations: the baby-bathwater scheme. *SIAM J. Multiscale Modeling and Simulation*, 1(3):391–407, 2003.

S. Lim and H. Choi. Optimal shape design of a two-dimensional asymmetric diffuser in turbulent flow. *AIAA Journal*, 42(6):1154–1169, 2004.

R. G. Littlejohn. Variational principles of guiding centre motion. *J. Plasma Phys.*, 29(1):111–125, 1983.

R. G. Littlejohn. Geometry and guiding center motion. *Contemporary Mathematics*, 28:151–167, 1984.

D. G. MacMartin, R. M. Murray, A. Verma, and J. D. Paduano. Active inlet flow control technology demonstration. *Journal Of Fluids Engineering—Transactions Of The ASME*, FEDSM 2001-18275, 2001.

D. MacMynowski. Root locus computation of an El Niño model. private communiation., 2005.

A. G. Makeev, D. Maroudas, and I. G. Kevrekidis. ”coarse” stability and bifurcation analysis using stochastic simulators: Kinetic monte carlo examples. *J. Chem. Phys.*, 116(23):10083–10091, 2002a.

A. G. Makeev, D. Maroudas, and I. G. Kevrekidis. “Coarse” stability and bifurcation analysis using stochastic simulators: Kinetic monte carlo examples. *J. Chem. Phys.*, 116(23):10083–10091, 2002b.

A. G. Makeev, D. Maroudas, A. Z. Panagiotopoulos, and I. G. Kevrekidis. Coarse bifurcation analysis of kinetic monte carlo simulations: A lattice-gas model with lateral interactions. *J. Chem. Phys.*, 117(18):8229–8240, 2002c.

J. E. Marsden and T. S. Ratiu. *Introduction to Mechanics and Symmetry.* Springer-Verlag, 1994.

J. E. Marsden and S. Shkoller. Global well-posedness for the Lagrangian averaged Navier-Stokes (LANS-α) equations on bounded domains. *Phil. Trans. R. Soc. Lond. A*, 2001.

A. T. McDonald and R. W. Fox. An experimental investigation of incompressible flow in conical diffusers. *Int. J. Mechanical Sciences*, 8:125–139, 1966.

I. Moon. Distributed suction boundary layer control on conical diffusers. Research Report 17, Mississippi State College, Aerophysics Department, August 1958.

S. Narayanan and A. Banaszuk. Experimental study of a novel active separation control approach. In *Proceedings of the 41st AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, January 6–9, 2003. AIAA Paper 2003-0060.

S. Narayanan, A. I. Khibhik, C. A. Jacobson, I. G. Kevrekidis, R. Rico-Martinez, and K. Lust. Low dimensional models for active control of flow separation. In *Proceedings of the 1999 IEEE Int. Conference on Control Applications*, Kohala Coast, Hawaii, USA, 1999.

W. B. Nicoll and B. R. Ramaprian. Performance of conical diffusers with annular injection at inlet. *ASME D: J. of Fluids Engineering*, 92:827–835, 1970.

M. Nishi, K. Yoshida, and K. Morimitsu. Flow control in wide-angled conical diffusers. *ASME D: J. of Fluids Engineering*, 98:728–735, 1976.

M. Nishi, K. Yoshida, and K. Morimitsu. Control of separation in a conical diffuser by vortex generator jets. *JSME International Journal Series B—Fluids And Thermal Engineering*, 41: 233–238, 1998.

T. G. Northrop. *The adiabatic motion of charged particles.* Interscience Publishers, 1963.

P. S. Pacheco and W. C. Ming. Introduction to message passing programming: MPI user guide in FORTRAN, 1997.

H. Peters. Conversion of energy in cross sectional divergences under different conditions of inflow. NACA TM 737, NACA, 1934.

T. J. Poinsot and S. K. Lele. Boundary conditions for direct simulations of compressible viscous flows. *J. Comput. Phys.*, 101:104–129, 1992.

D. M. Rao. A method of flow stabilisation with high pressure recovery in short, conical diffusers. *J. of the Royal Aeronautical Soc. TN*, 75:336–339, 1971.

L. R. Reneau, J. P. Johnston, and S. J. Kline. Performance and design of straight, two-dimensional diffusers. *ASME D: J. of Basic Engineering*, 89:141–150, 1967.

R. Rico-Martinez, C. W. Gear, and I. G. Kevrekidis. Coarse projective kMC integration: forward/reverse initial and boundary value problems. *J. Comput. Phys.*, 196(2):474–489, 2004.

C. W. Rowley, T. Colonius, and R. M. Murray. Model reduction for compressible flows using POD and galerkin projection. *Physica D*, 189(1-2):115–129, 2004.

O. Runborg, C. Theodoropoulos, and I. G. Kevrekidis. Effective bifurcation analysis: A time-stepper based approach. *Nonlinearity*, 15, May 2002.

J. T. Salmon, T. J. Bogar, and M. Sabjen. Laser doppler velocimeter measurements in unsteady, separated, transonic diffuser flows. *AIAA Journal*, 21:1690–1697, 1983.

J. U. Schlüter, X. Wu, and H. Pitsch. Large-eddy simulations of a separated plane diffuser. In *Proceedings of the 43rd AIAA Aerospace Sciences Meeting and Exhibi*, Reno, Nevada, January 10-13, 2005. AIAA Paper 2005-0672.

A. Seifert, A. Darabi, and I. Wygnanski. Delay of airfoil stall by periodic excitation. *J. Aircraft*, 33:691–698, 1996.

A. Seifert and L. G. Pack. Oscillatory control of separation at high Reynolds numbers. *AIAA Journal*, 37:1062–1071, 1999.

L. F. Shampine and M. K. Gordon. *Computer Solution of ordinary differential equations: the initial value problem*. W. H. Freeman, 1975.

Y. H. Shin, K. H. Kim, and B. J. Son. An experimental study on the development of a reverse flow zone in a vaneless diffuser. *JSME International Journal Series B—Fluids And Thermal Engineering*, 41(3):546–555, 1998.

G. M. Shroff and H. B. Keller. Stabilization of unstable procedures: the recursive projection method. *SIAM J. Num. Anal.*, 30:1099–1120, 1993.

C. I. Siettos, A. Armaou, A. G. Makeev, and I. G. Kevrekidis. Microscopic/stochastic timesteppers and coarse control: a kmc example. *AiChE Journal*, 49(7):1922–1926, 2003a.

C. I. Siettos, M. D. Graham, and I. G. Kevrekidis. Coarse Brownian dynamics for nematic liquid crystals: Bifurcation, projective integration, and control via stochastic simulation. *J. Chem. Phys.*, 118(22):10149–10156, 2003b.

C. I. Siettos, I. G. Kevrekidis, and D. Maroudas. Coarse bifurcation diagrams via microscopic simulators: A state-feedback control-based approach. *International Journal of Bifurcation and Chaos*, 14(1):207–220, 2004.

C. I. Siettos, C. C. Pantelides, and I. G. Kevrekidis. Enabling dynamic process simulators to perform alternative tasks: A time-stepper based toolkit for computer-aided analysis. *Industrial and Engineering Chemistry Research*, 42(26):6795–6801, 2003c.

C. R. Smith. Transitory stall time-scale for plane-wall air diffuser. *ASME D: J. of Fluids Engineering*, 100:133–135, 1978.

C. R. Smith and S. J. Kline. An experimental investigation of the transitory stall regime in a two-dimensional diffuser. *ASME D: J. of Fluids Engineering*, 96:11–15, 1974.

T. Smith. Comparison of coarse time integration and pod/galerkin simulation of channel flow turbulence. private communiation., 2005.

T. R. Smith, J. Moehlis, and P. Holmes. Low-dimensional models for turbulent plane Couette flow in a minimal flow unit. submitted to Journal of Fluid Mechanics, 2005.

R. K. Sullerey and A. M. Pradeep. Secondary flow control using vortex generator jets. *Journal Of Fluids Engineering—Transactions Of The ASME*, 126(4):650–657, 2004.

T. Suzuki and T. Colonius. Inverse-imaging method for detection of a vortex in a channel. *AIAA Journal*, 41(9):1743–1751, 2003.

T. Suzuki, T. Colonius, and S. Pirozzoli. Vortex shedding in a two-dimensional diffuser: theory and simulation of separation control by periodic mass injection. *J. Fluid Mech.*, 520:187–213, 2004.

C. K. W. Tam and J. C. Webb. Dispersion-relation-preserving finite difference schemes for computational acoustics. *J. Comput. Phys.*, 107:262–281, 1993.

J. S. Tennant. A subsonic diffuser with moving walls for boundary-layer control. *AIAA Journal*, 11:240–242, 1973.

C. Theodoropoulos, Y.-H. Qian, and I. G. Kevrekidis. “Coarse” stability and bifurcation analysis using time-steppers: A reaction-diffusion example. *Proceedings of the National Academy of Sciences*, 97:9840–9843, August 29 2000.

C. Theodoropoulos, K. Sankaranarayanan, S. Sundaresan, and I. G. Kevrekidis. Coarse bifurcation studies of bubble flow Lattice Boltzmann simulations. *Chemical Engineering Science*, 59(12): 2357–2362, 2004.

T. Tsukiya, E. Tatsumi, T. Nishinaka, N. Katagiri, Y. Takewa, H. Ohnishi, M. Oshikawa, T. Mizuno, Y. Taenaka, H. Takano, and S. Kitamura. Design progress of the ultracompact integrated heart lung assist device-part 2: Optimization of the diffuser vane profile. *Artificial Organs*, 27(10):914–919, 2003a.

T. Tsukiya, E. Tatsumi, T. Nishinaka, N. Katagiri, Y. Takewa, H. Ohnishi, M. Oshikawa, K. Shioya, T. Mizuno, Y. Taenaka, H. Takano, and S. Kitamura. Design progress of the ultracompact integrated heart lung assist device-part 1: Effect of vaned diffusers on gas-transfer performances. *Artificial Organs*, 27(10):907–913, 2003b.

L. van Lier, S. Dequand, A. Hirschberg, and J. Gorter. Aeroacoustics of diffusers: an experimental study of typical industrial diffusers at Reynolds numbers of $o(10^5)$. *J. Acoust. Soc. Am.*, 109(1): 108–115, 2001.

M. Vinokur. Conservation equation of gas-dynamics in curvilinear coordinate systems. *J. Comput. Phys.*, 14:105–125, 1974.

M. R. Visbal and G. V. Gaitonde. High-order-accurate methods for complex unsteady subsonic flows. *AIAA Journal*, 37:1231–1239, 1999.

M. Vujičić and C. Crnojević. Calculation of the separation point for the turbulent flow in plane diffusers. *Facta universitatis: Mechanics, Automatic Control and Robotics*, 3(15):1001–1006, 2003.

G. Wentzel. Eine verallgemeinerung der quantenbedingung fur die zwecke der wellenmechanik. *Z. Phys.*, 38:518–529, 1926.

G. B. Whitham. *Linear and Nonlinear Waves*. John Wiley and Sons, 1974.