New Methods for Sensitivity Analysis of Chaotic Dynamical Systems

by

Patrick Joseph Blonigan

B.S., Cornell University (2011)

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

Master of Science in Aeronautics and Astronautics

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2013

Author .....
Department of Aeronautics and Astronautics
May 10, 2013

Certified by .....
Qiqi Wang
Assistant Professor of Aeronautics and Astronautics
Thesis Supervisor

Accepted by .....
Eytan H. Modiano
Professor of Aeronautics and Astronautics
Chair, Graduate Program Committee

New Methods for Sensitivity Analysis of Chaotic Dynamical Systems

by

Patrick Joseph Blonigan

Submitted to the Department of Aeronautics and Astronautics
on May 10, 2013, in partial fulfillment of the
requirements for the degree of
Master of Science in Aeronautics and Astronautics

Abstract

Computational methods for sensitivity analysis are invaluable tools for fluid dynamics research and engineering design. These methods are used in many applications, including aerodynamic shape optimization and adaptive grid refinement.

However, traditional sensitivity analysis methods break down when applied to long-time averaged quantities in chaotic dynamical systems, such as those obtained from high-fidelity turbulence simulations. Also, a number of dynamical properties of chaotic systems, most notably the "Butterfly Effect", make the formulation of new sensitivity analysis methods difficult.

This paper will discuss two chaotic sensitivity analysis methods and demonstrate them on several chaotic dynamical systems including the Lorenz equations and a chaotic Partial Differential Equation, the Kuramoto-Sivshinsky equation. The first method, the probability density adjoint method, forms a probability density function on the strange attractor associated with the system and uses its adjoint to find gradients. This was achieved using a novel numerical method in which the attractor manifold, instead of a region of phase space, is discretized. The second method, the Least Squares Sensitivity method, finds some "shadow trajectory" in phase space for which perturbations do not grow exponentially. This method is formulated as a quadratic programing problem with linear constraints. Several multigrid-in-time methods to solve the KKT system arising from this optimization problem will be discussed in depth.

While the probability density adjoint method is better suited for smaller systems and reduced order models, least squares sensitivity analysis, solved with a multigrid-in-time method could be applied to higher dimensional systems such as high fidelity fluid flow simulations.

Thesis Supervisor: Qiqi Wang
Title: Assistant Professor of Aeronautics and Astronautics

Acknowledgements

Firstly, I would like to thank my advisor, Professor Qiqi Wang. I am very grateful for all the help and support I have recieved from him over the past two years and I look forward to working with him on my PhD research and beyond. The research presented in this thesis was conducted at MIT, NASA Langley, and the 2012 summer program at Stanford University's Center for Turbulence Research (CTR). During my time at NASA Langley, I received a great deal of advive and guidance from the research scientists of the Computational AeroSciences group. I would like to thank Dr. Boris Diskin of the National Institute of Aerospace for his insights into multigrid methods and Dr. Eric Nielsen for his all help during my time in Virginia, especially for running the airfoil simulations presented in this thesis with NASA's FUN3D CFD code. I would also like to thank Dr. Johan Larsson for hosting Professor Wang, Rui Chen and I for the CTR summer program. Finally, I would like to thank my parents, Margaret and Gregg, and my brothers, Andrew and Charlie, for all their love and support during my first two years at MIT.

Contents

1 Introduction **19**

1.1 Motivation 19

1.2 Past Work 21

1.3 Thesis Outline 21

2 Computing Sensitivities of Chaotic Systems **23**

2.1 Traditional Sensitivity Analysis Methods 23

2.1.1 Forward/Tangent Sensitivity Analysis 23

2.1.2 Backward/Adjoint Sensitivity Analysis 24

2.2 Breakdown of traditional sensitivity analysis for chaotic systems . . . 25

2.2.1 Sensitivity analysis of chaotic fluid flows 28

2.3 Smoothness of the Mean and Stationary Density Distribution of Chaos 29

3 Probability Density Adjoint Method for Chaotic 1D Maps **37**

3.1 Introduction 37

3.2 Computing Stationary Density 37

3.3 Computing gradients using the density adjoint 40

3.4 Algorithm Summary 42

3.5 Density adjoint for the cusp map 43

4 Probability Density Adjoint Method for Continuous Chaotic Systems **45**

4.1 Introduction 45

4.2 Computing Stationary Density 46
4.3 Computing the Density Adjoint 52
4.4 Algorithm Summary 54
4.5 Density adjoint for the Lorenz equations 55
4.6 Discussion 57

5 Least Squares Sensitivity Method **59**
5.1 Lyapunov Exponents and the Shadowing Lemma 59
5.2 Computing the Shadow Trajectory 63
5.3 Solving the KKT system numerically 64
5.4 LSS Examples 65
5.4.1 The Lorenz Equations 65
5.4.2 The Kuramoto-Sivashinsky Equation 66

6 Multigrid-in-time for Least Squares Sensitivity Analysis **69**
6.1 Solving the KKT System numerically 70
6.2 Classic Multigrid 71
6.3 Cyclic Reduction 72
6.4 Higher Order Averaging/Krylov Subspace Scheme 74
6.4.1 Matrix restriction multigrid 75
6.4.2 Solution restriction multigrid 78

7 Conclusions **85**

A Probability Density Adjoint Method **87**
A.1 Probability density adjoint for 1D maps 87
A.1.1 Deriving the continuous density adjoint equation 87
A.1.2 Derivation of the gradient equation 89
A.1.3 Deriving the discrete density adjoint equation 90
A.2 Probability density adjoint for continuous chaos 91
A.2.1 Deriving the continuous adjoint equation 91
A.2.2 Deriving the discrete Adjoint Equation 93

A.2.3 Computing Attractor Surface Areas 94
A.2.4 Computing gradients on the attractor surface 95

B Least Squares Sensitivity Analysis **97**
B.1 Deriving the KKT System 97
B.2 Computing Sensitivities using a Shadow Trajectory 98
B.3 Cyclic Reduction 99
B.3.1 Conducting cyclic reduction without inverting main diagonal matrices 99
B.3.2 Estimating the operation count for cyclic reduction 100

List of Figures

2-1 Lorenz equation sensitivities. The objective function is time-averaged z. The derivatives are with respect to the parameter r. [12] 27

2-2 Vorticity contours for Mach 0.1, Reynolds number 10000 flow past a NACA 0012 airfoil at an angle of attack of 20 degrees. 28

2-3 Drag coefficient C_d versus time step for the NACA 0012 airfoil. The aperiodic nature of drag is one indication that the air flow is chaotic. 29

2-4 L2 norm of the adjoint residual versus time step. Note the exponential increase in magnitude as the adjoint is propagated further back in time. 29

2-5 The shape of the logistic, tent and Cusp maps. 31

2-6 Smoothness of $\overline{x}$ as a function of parameter ξ. 31

2-7 Stationary density 32

2-8 $\overline{z}$ and $\overline{x^2}$ of the Lorenz attractor as r varies. 33

2-9 $\overline{x}$ and $\overline{z}$ of the Rössler attractor as c varies. 33

2-10 The stationary density of the Lorenz attractor and the Rössler attractor projected into the (x, z) plane. 34

3-1 Density Mapping for the cusp map 38

3-2 Cusp map transition matrix P_n structure for $\xi = 0.5$. 40

3-3 Cusp map density distribution ρ_s for $\xi = 0.5$. Generated with 256 nodes. 40

3-4 Adjoint ϕ for the cusp map with $\xi = 0.5$, generated using 1024 nodes. 43

3-5 Comparison of gradients computed using the adjoint method and the finite difference method. 1D space between 0 and 1 was discretized using 256 nodes. 44

3-6 Convergence of the residual r of $\frac{\partial \overline{J}}{\partial \xi}$ with the number of nodes n for $\xi = 0.5$. The residual was calculated by taking the L2 norm of the difference between the gradient for n nodes and 8096 nodes. 44

4-1 Poincaré Section at z = 27 for Lorenz attractor trajectories with $\frac{\partial z}{\partial t} > 0$. 46

4-2 Three dimensional view of the 2D surface approximating the Lorenz Attractor and the Poincaré section at z = 27. 46

4-3 Node distribution corresponding to a 64 streamline by 64 streamwise mesh for the Lorenz attractor. It was found that distributing the streamline starting positions so that there were more streamlines near the bifurcation increased the rate of convergence to the true density distribution. 49

4-4 Transition Matrix P_n Structure for a roughly uniform streamline distribution. Note the similarity of this matrix to that for the Cusp map 50

4-5 Transition Matrix P_n Structure for a non-uniform streamline distribution with more streamlines starting near $y_0 = 18$. 50

4-6 Density ρ_s versus x on the Poincare Section at z = 27. 512 streamlines were used to form P_n. 50

4-7 Density distribution on the surface of the Lorenz attractor for a 512 by 128 mesh. 50

4-8 Convergence of $\overline{z}$ for two different streamline start point distributions, where M is the number of streamlines. The clustered distribution has streamlines clustered near the bifurcation of the attractor. 52

4-9 Adjoint ϕ versus x on the Poincaré Section at z = 27. 1024 streamlines were used to form P_n. 55

4-10 Adjoint distribution on the surface of the Lorenz Attractor for a 512 by 128 mesh. 55

4-11 Sensitivity of $\overline{z}$ with respect to the parameters s, b, r and z_0 for difference grid sizes M. M is number of the streamlines, N ($= \frac{1}{4}M$) is the number of nodes along a streamline. The level lines correspond to sensitivities computed using finite differences of ensemble averaged data and the dotted lines are the 3σ confidence bounds [24]. 56

5-1 Schematic of Lyapunov exponents and covariant vectors 60

5-2 Phase space trajectory of a chaotic dynamical system. The unstable manifold, in red, is the space of all Lyapunov covariant vectors corresponding to positive exponents. The stable manifold, in green, corresponds to the space of all covariant vectors associated with negative exponents. A perturbation to the system (in red) has components in both manifolds, and the unstable component causes the perturbed trajectory (pink) to diverge exponentially from the unperturbed trajectory (in black). LSS chooses a perturbed trajectory with a different initial condition (in blue) that does not diverge from the unperturbed trajectory. 61

5-3 LEFT: Original and shadow phase space trajectories without any time transformation ($d\tau/dt = 1$). RIGHT: Original and shadow phase space trajectories with a time transformation $d\tau/dt = 1 + \eta$ that minimizes the distance between the two trajectories in phase space for all time. 62

5-4 Lorenz equation phase space trajectory ($u(t)$) for $r = 28$ (blue) and a corresponding approximate shadow trajectory ($u(t) + v(t)$) (red). Integration time was $T = 20$ in dimensionless time units. 66

5-5 Gradient of long-time averaged z with respect to the parameter r. Gradients computed with trajectory length $T = 20$ are shown as black diamonds. Those computed using $T = 1000$ are shown as a red line. 66

5-6 Time and space averaged $u(x,t)$ versus the parameter c. The error bars show the standard deviation computed from 20 solutions from random initial conditions for $t = 1024$ time units. The slope of the objective function is approximately -0.842. 68

5-7 Gradient of time and space averaged u with respect to the parameter c. Gradients computed with trajectory length $T = 100$ are shown as black diamonds. Those computed using $T = 1000$ are shown as a red line. 68

5-8 LEFT: KS equation solution ($u(x,t)$) for $c = -0.5$. RIGHT: Corresponding approximate shadow trajectory ($u(x,t)+v(x,t)$). Integration time was $T = 100$ in dimensionless time units. Note the transformed time scale τ for the shadow trajectory. 68

6-1 Convergence of the gradient of time-averaged z with respect to r, as computed using multigrid in time with 10 relaxation iterations before restriction and after prolongation on each level. 72

6-2 L2 norm of the residual while solving for the gradient of time-averaged z with respect to r using multigrid in time. Similar behavior was observed when computing other gradients 72

6-3 Lorenz equation solution; x,y,z are blue, green and red respectively. . 76

6-4 Convergence of matrix restriction multigrid with $dt_f = 0.01$ for different orders of averaging. 76

6-5 Condition Number κ (solid line) and maximum eigenvalue λ_{max} (dashed line) versus time step dt for coarsened grids corresponding to a fine grid with $dt = 0.001$ for the LSS system associated with the Lorenz equations 77

6-6 Convergence of Block Gauss-Seidel and Conjugate Gradient on the fine grid (dt=0.01,$\alpha^2 = 40$) and a coarsened grid (dt=0.02,$\alpha^2 = 40$). . . . 77

6-7 Convergence of matrix restriction multigrid for different fine grid time steps dt_f. 79

6-8 Convergence rate of matrix restriction multigrid versus α^2. The convergence rate is $-\gamma$, from the curve fit $\log_{10}\|r\|_{L2} = \gamma \log_{10} N_V + \log_{10} C$ of the residual L2 norm $\|r\|_{L2}$ versus the number of V-cycles N_V. . . 79
6-9 Convergence of matrix restriction multigrid for different coarsening thresholds dt_c. 79
6-10 Convergence of solution restriction multigrid for different values of dt_f. 80
6-11 Convergence rate of solution restriction multigrid versus α^2. The convergence rate is $-\gamma$, from the curve fit $\log_{10}\|r\|_{L2} = \gamma \log_{10} N_V + \log_{10} C$ of the residual L2 norm $\|r\|_{L2}$ versus the number of V-cycles N_V. . . 80
6-12 Convergence of solution restriction multigrid with 1st, 3rd and 5th order averaging. 81
6-13 Convergence plots for matrix restriction multigrid with MINRES smoothing, $dt_c = 5$, and 4th order averaging and solution restriction multigrid. 81
6-14 Convergence of MINRES for an LSS system for the Lorenz equations with $dt_f = 0.004$ and $\alpha^2 = 40$. The dashed line shows the gradient computed at a given iteration, which should be roughly 1.01 ± 0.04 [24] 82
6-15 Convergence of solution restriction multigrid for an LSS system for the Lorenz equations. The dashed line shows the gradient computed at a given iteration, which should be roughly 1.01 ± 0.04 [24] 82

A-1 The effect of a perturbation on the mapping function. 90

List of Tables

5.1 Comparison of sensitivities computed using linear regression with 10 samples [24] (LR), the probability density adjoint (PDA) method with a 256 by 512 grid for the attractor discretization, and LSS. 66

B.1 Estimate of Operation Count per iteration for cyclic reduction and for the Jacobi method for comparison. 101

Chapter 1

Introduction

1.1 Motivation

Sensitivity analysis of systems governed by ordinary differential equations (ODEs) and partial differential equations (PDEs) is important in many fields of science and engineering. Its goal is to compute sensitivity derivatives of key quantities of interest to parameters that influence the system. Applications of sensitivity analysis in science and engineering include design optimization, inverse problems, data assimilation, and uncertainty quantification.

Adjoint based sensitivity analysis is especially powerful in many applications, due to its efficiency when the number of parameters is large. In aircraft design, for example, the geometric parameters that define the aerodynamic shape is very large. As a result, the adjoint method of sensitivity analysis has proven to be very successful for aircraft design [10], [19]. Similarly, the adjoint method has been an essential tool for adaptive grid methods for solving PDE's [22], solving inverse problems in seismology, and for assimilating observation data for weather forecasting.

Sensitivity analysis for chaotic dynamical systems is important because of the prevalence of chaos in many scientific and engineering fields. One example is highly turbulent gas flow of mixing and combustion processes in jet engines. In this example, and in other applications with periodic or chaotic characteristics, statistical averaged quantities such as mean temperature and mean aerodynamic forces are of interest.

Therefore, the general problem this paper seeks a solution to is:

$$\text{Given } \frac{du}{dt} = f(u,\xi), \quad \overline{J} = \lim_{T\to\infty} \frac{1}{T}\int_0^T J(u,\xi)dt, \quad \text{Compute } \frac{\partial \overline{J}}{\partial \xi} \tag{1.1}$$

Sensitivity analysis for chaotic dynamical systems is difficult because of the high sensitivity of these systems to the initial condition, known as the "Butterfly Effect". Slightly different initial conditions will result in very different solutions, which diverge exponentially with time [14]. This also results in exponential growth of sensitivities and therefore the sensitivity of long-time averaged quantities is not equal to the long-time average sensitivities of chaotic systems [12]. Because the derivative and long-time average do not commute, the traditional adjoint method computes sensitivities that diverge, as shown in the work done by Lea et al. [12].

This paper presents two new methods for computing sensitivities of mean quantities in chaotic dynamical systems. The two methods require that the chaotic system is *ergodic*; that long time behavior of the system is independent of initial conditions.

The key idea of the first method, the probability density adjoint method, is to describe the objective function $\overline{J}$ as an average in phase space as in [21]:

$$\overline{J} = \int_{R^n} J(u)\rho_s(u)du \tag{1.2}$$

The probability density function or the invariant measure of the chaotic system $\rho_s(u)$ is governed by a probability density equation, whose adjoint equation can be solved to compute the desired sensitivities.

The second method, the least squares sensitivity (LSS) method, finds a perturbed trajectory that does not diverge exponentially from some trajectory in phase space. This non-diverging trajectory, called a "shadow trajectory", has its existence guaranteed by the shadowing lemma [16] for a large number of chaotic systems and can be used to compute sensitivities. The shadow trajectory is found by solving a quadratic programing problem with linear constraints, and is more readily extendible to high dimensional systems than the probability density adjoint method.

1.2 Past Work

Prior work in this area includes the ensemble-adjoint method proposed by Lea et al. [12]. This method has been applied to an ocean circulation model with some success [13]. Eyink et al. then went on to generalize the ensemble-adjoint method [6]. The ensemble-adjoint method involves averaging over a large number of ensemble calculations. It was found by Eyink et al. that the sample mean of sensitivities computed with the ensemble adjoint approach only converges as $N^{-0.46}$, where N is the number of samples, making it less computationally efficient than a naive Monte-Carlo approach [6].

More recently, a Fokker-Planck adjoint approach for climate sensitivity analysis has been derived [21]. This approach involves finding a probability density function which satisfies a Fokker-Planck equation to model the climate. The adjoint of this Fokker-Planck equation is then used to compute derivatives with respect to long time averaged quantities. However, this method requires the discretization of phase space, making it computationally infeasible for systems with a high number of dimensions. Also, the method requires adding diffusion into the system, potentially making the computed sensitivities inaccurate.

1.3 Thesis Outline

The rest of this paper is organized as follows: chapter 2 discusses the breakdown of traditional sensitivity analysis methods for chaotic systems and the well-posedness of the problem (1.1) by analyzing the differentiability of the time averaged quantities $\overline{J}$ in both discrete and continuous chaotic dynamical systems. Chapter 3 presents the probability density adjoint method for chaotic, 1D iterated maps. Chapter 4 extends the probability density adjoint method to continuous dynamical systems, with the Lorenz equations as an example. Chapter 5 discusses least squares sensitivity analysis (LSS) and demonstrates it on a system of ordinary differential equations (ODEs), the Lorenz equations, and a partial differential equation (PDE), the

Kuramoto-Sivashinsky (KS) equation. Chapter 6 discusses several multigrid-in-time schemes that could be used to solve the KKT system associated with LSS. Chapter 7 concludes this paper.

Chapter 2

Computing Sensitivities of Chaotic Systems

2.1 Traditional Sensitivity Analysis Methods

2.1.1 Forward/Tangent Sensitivity Analysis

Traditional sensitivity analysis for the system governed by equation (1.1) is conducted by solving the following equation:

$$\frac{\partial \overline{J}}{\partial \xi} = \frac{1}{T}\int_0^T \frac{\partial J}{\partial \xi}\, dt$$

The term $\frac{\partial J}{\partial \xi}$ can be computed by solving the linearization or tangent equation of the ODE of interest:

$$\frac{dv}{dt} = \frac{\partial f}{\partial u}v + \frac{\partial f}{\partial \xi}, \qquad v \equiv \frac{\partial u}{\partial \xi} \tag{2.1}$$

And using the chain rule:

$$\frac{\partial J}{\partial \xi} = \frac{1}{T}\int_0^T \left\langle \frac{\partial J}{\partial u}, v \right\rangle\, dt \tag{2.2}$$

To compute sensitivities using forward sensitivity analysis:

1. Compute the forward solution u by integrating the ODE in equation (1.1) forward in time.

2. Compute the tangent v by integrating equation (2.1) forwards in time (starting from $t = 0$). Note that $\frac{\partial f}{\partial u}$ is computed using the forward solution u.

3. Use equation (2.2) to compute gradients.

Forward sensitivity analysis is best for situations with few parameters, ξ, and many objective functions, $\overline{J}$.

2.1.2 Backward/Adjoint Sensitivity Analysis

The adjoint method allows us to efficiently compute the sensitivity of $\overline{J}$ to many different parameters. Using the adjoint, the computational cost to compute sensitivities to all parameters of interest is roughly that of two forward integrations [10].

Using forward sensitivity analysis would require solving for v for every parameter ξ. To avoid this, start by adding the inner product of the adjoint variable ψ and equation (2.1):

$$\frac{\partial \overline{J}}{\partial \xi} = \frac{1}{T}\int_0^T \left\langle \frac{\partial J}{\partial u}, v \right\rangle + \left\langle \psi, \left(-\frac{dv}{dt} + \frac{\partial f}{\partial u} v + \frac{\partial f}{\partial \xi} \right) \right\rangle \, dt$$

Where the inner product $\langle \cdot, \cdot \rangle$ is a dot product for vectors for an ODE or the discretization of a PDE. Using integration by parts:

$$\frac{\partial \overline{J}}{\partial \xi} = \frac{1}{T}\int_0^T \left\langle \frac{\partial J}{\partial u}, v \right\rangle + \left\langle v, \left(\frac{d\psi}{dt} + \frac{\partial f}{\partial u}^* \psi \right) \right\rangle + \left\langle \frac{\partial f}{\partial \xi}, \psi \right\rangle \, dt + \langle \psi, v \rangle \, |_0^T$$

Where x^* denotes the transpose or complex conjugate of x. Rearranging to collect terms including v:

$$\frac{\partial \overline{J}}{\partial \xi} = \frac{1}{T}\int_0^T \left\langle v, \left(\frac{d\psi}{dt} + \frac{\partial f}{\partial u}^* \psi + \frac{\partial J}{\partial u} \right) \right\rangle + \left\langle \frac{\partial f}{\partial \xi}, \psi \right\rangle \, dt + \langle \psi, v \rangle \, |_0^T$$

To eliminate any dependence of the gradient on the tangent solution v, we choose the adjoint variable to satisfy:

$$\frac{d\psi}{dt} + \frac{\partial f}{\partial u}^* \psi + \frac{\partial J}{\partial u} = 0 \qquad \psi(0) = \psi(T) = [0] \tag{2.3}$$

Where [0] is a vector of all zeros. Now sensitivities can be computed using the following equation:

$$\frac{\partial J}{\partial \xi} = \frac{1}{T} \int_0^T \left\langle \frac{\partial f}{\partial \xi}, \psi \right\rangle \, dt \tag{2.4}$$

For different parameters ξ, $\frac{\partial f}{\partial \xi}$ will change, but not ψ. Therefore, to compute sensitivities to ξ, just compute $\frac{\partial f}{\partial \xi}$ and substitute it into equation (2.4).

To compute sensitivities:

1. Compute the forward solution u by integrating the ODE in equation (1.1) forward in time.

2. Compute the adjoint ψ by integrating equation (2.3) *backwards* in time (from $t = T$ to $t = 0$). Note that $\frac{\partial f}{\partial u}$ is computed using the forward solution u.

3. Use equation (2.4) to compute gradients.

2.2 Breakdown of traditional sensitivity analysis for chaotic systems

Although traditional sensitivity analysis methods work for systems with initial transience and can be modified to work for systems with limit cycles [11], the traditional sensitivity analysis methods discussed in the previous section break down for chaotic systems.

Consider the general problem defined in chapter 1, equation (1.1). Define:

$$\bar{J}^T(\xi) = \frac{1}{T}\int_0^T J(u(t),\xi)dt, \quad \bar{J}^\infty(\xi) = \lim_{T\to\infty} \bar{J}^T(\xi)$$

Where $\bar{J}^\infty$ is the infinite time average. For non-chaotic systems, $\bar{J}^T$ will converge to $\bar{J}^\infty$ as $T \to \infty$. However, this is not the case for chaotic systems, for which:

$$\frac{d\bar{J}^\infty}{d\xi} \neq \lim_{T\to\infty} \frac{d\bar{J}^T}{d\xi} \tag{2.5}$$

In words, the derivative of a time averaged quantity is not equal to the time averaged derivative of that quantity. This inequality occurs because uniform convergence of the derivatives is not achieved:

$$\lim_{T\to\infty}\int_0^T \lim_{\epsilon\to\infty} \frac{J(\xi+\epsilon)-J(\xi)}{\epsilon}dt \neq \lim_{T\to\infty}\int_0^T \frac{\partial J}{\partial \xi}dt$$

Consider the Lorenz equations, a low order model of Rayleigh-Bénard convection [14]:

$$\frac{dx}{dt} = s(y-x), \quad \frac{dy}{dt} = x(r-z)-y, \quad \frac{dz}{dt} = x\,y - b\,z\ . \tag{2.6}$$

As shown in figure 2-1, the magnitude of local derivatives grows exponentially as T is increased. However, the overall trend of mean z versus r indicates that $\frac{\partial \overline{z}}{\partial r} \approx 1$. This shows that the traditional sensitivity analysis methods do not give useful sensitivity data for chaotic systems.

The discrepancy in equation (2.5) occurs because of the system's unstable mode associated with its positive Lyapunov exponent. The Lyapunov exponent is the rate at which two trajectories that are initially a short distance apart in phase space at some time converge (or diverge) in time. Unlike steady and periodic systems, chaotic systems have at least one positive Lyapunov exponent, meaning that two initially close trajectories will diverge. This means the sensitivity of $\overline{J}$ to some perturbation at time $t = 0$ will grow exponentially in time, resulting in the inequality of equation (2.5). This divergence is the manifestation of Edward Lorenz's "Butterfly effect" [14].

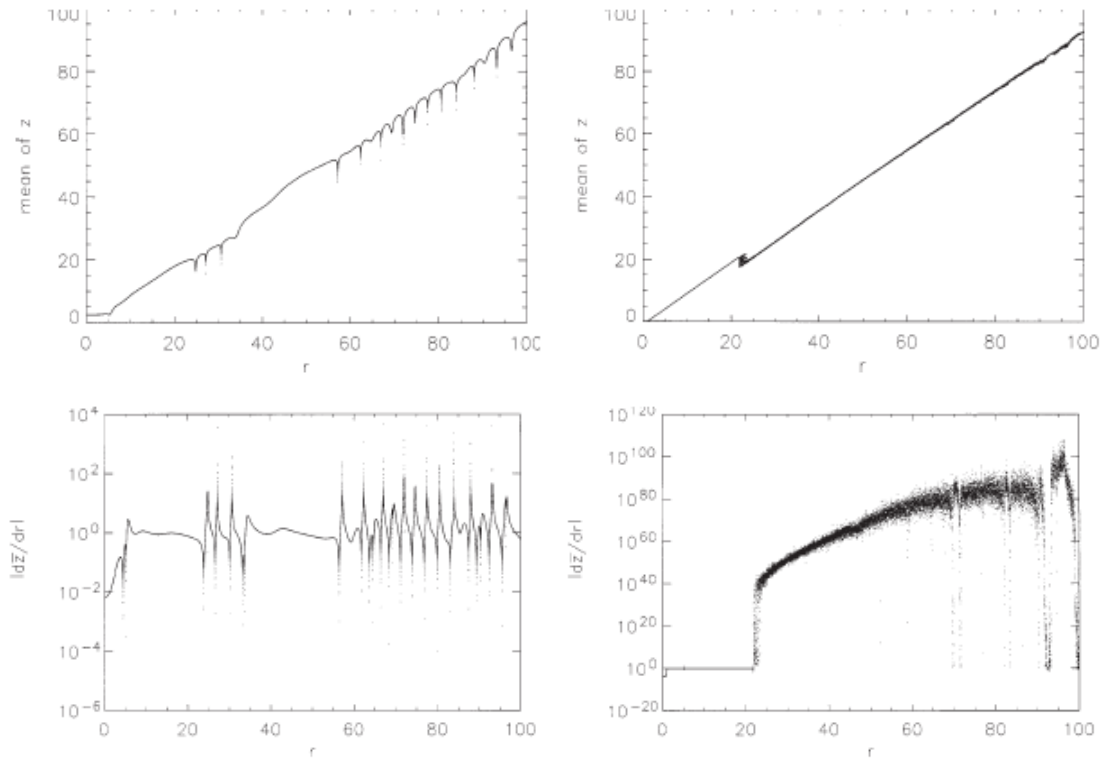


Figure 2-1: Lorenz equation sensitivities. The objective function is time-averaged z. The derivatives are with respect to the parameter r. [12]

2.2.1 Sensitivity analysis of chaotic fluid flows

The divergence of time averaged sensitivities has also been shown (indirectly) by the divergence of the magnitude of the drag-adjoint field of a cylinder in cross flow [25]. This section demonstrates the problem for a NACA 0012 airfoil at a high angle of attack, as this geometry and configuration is commonplace in aerospace applications, including low speed aircraft maneuvers, and air flow around rotor and propeller blades. The sensitivity of time averaged drag with respect to angle of attack was investigated. Work by Pulliam [17] indicates that the airflow around a NACA 0012 at 20 degrees angle of attack and free stream Mach number of 0.2 is chaotic when the Reynolds number is 3000 and above. It is important to note that it is not known if the chaos observed by Pulliam is a physical property of the flow or a purely numerical phenomenon.

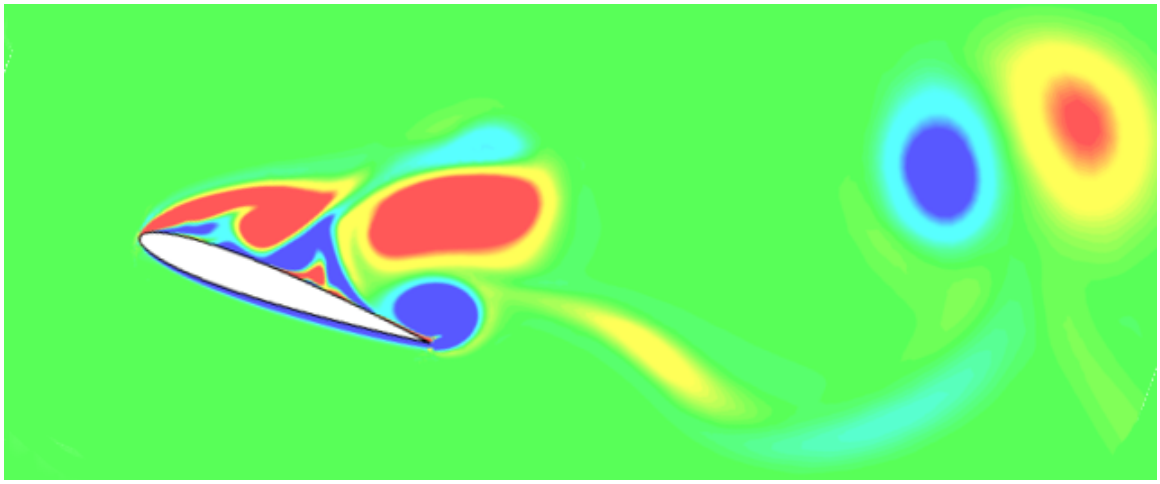


Figure 2-2: Vorticity contours for Mach 0.1, Reynolds number 10000 flow past a NACA 0012 airfoil at an angle of attack of 20 degrees.

All simulations were run using NASA's FUN3D CFD code, which includes a discrete adjoint solver. Divergence of the adjoint variable, which is used to compute sensitivity gradients, is observed by averaging the sensitivity over time intervals of different lengths. As the time interval of the average is increased in length, the magnitude of the sensitivity should increase exponentially, as observed for the Lorenz equation in the previous section. The adjoint is propagated backwards in time, and earlier perturbations will have an exponentially greater effect on the final state of the

system than later perturbations.

The simulation was a NACA 0012 at 20° angle of attack in $M = 0.1$, $Re = 10000$ flow. For these parameters the adjoint variables were observed to diverge exponentially with backwards time. This is seen from the divergence of the L2 norm of the adjoint residual shown in figure 2-4. This divergence shows that FUN3D's adjoint solver breaks down when applied to chaotic flow fields.

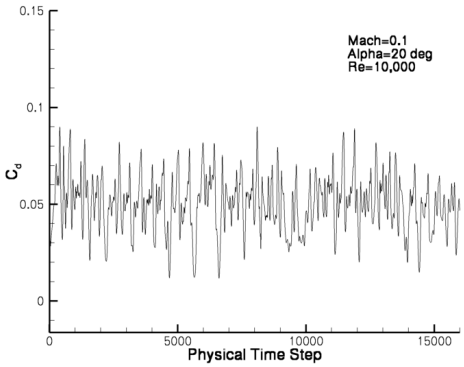


Figure 2-3: Drag coefficient C_d versus time step for the NACA 0012 airfoil. The aperiodic nature of drag is one indication that the air flow is chaotic.

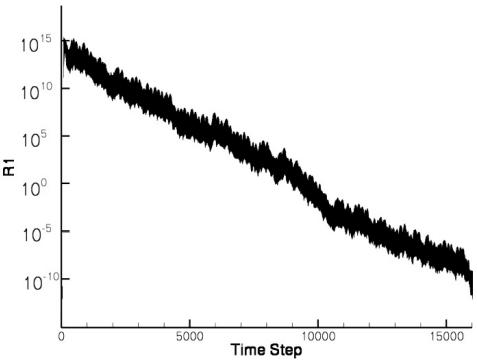


Figure 2-4: L2 norm of the adjoint residual versus time step. Note the exponential increase in magnitude as the adjoint is propagated further back in time.

As in Pulliam's work it is uncertain whether this chaos is a physical or numerical effect. Regardless of the source of the chaos, the divergence of the adjoint-based gradient poses a serious problem to engineers who wish to perform sensitivity analysis on simulations with chaotic flow fields.

2.3 Smoothness of the Mean and Stationary Density Distribution of Chaos

Not every chaotic dynamical systems has differentiable mean quantities $\overline{J}$. Hyperbolic chaos, a class of dynamical systems with ideal attractors, are known to have mean quantities that respond differentiably to small perturbations in its parameters [3]. Chaotic systems whose mean quantities are differentiable to perturbations are

generally classified as quasi-hyperbolic systems [3]. Other chaotic dynamical systems are known as non-hyperbolic. In these non-hyperbolic systems, the mean quantities are usually not differentiable, or even continuous as the parameters vary. In fact, the long time average for non-hyperbolic systems may have non-trivial dependence on the initial condition, leading to mean quantities that are not well-defined. This section numerically demonstrates this difference between quasi-hyperbolic chaos and non-hyperbolic chaos with examples both in 1D maps and in 3D continuous chaotic dynamical systems.

We first study three parameterized 1D chaotic maps:

1. The logistic map

$$x_{k+1} = F_{logistic}(x_k) = \left(4 - \frac{\xi}{4}\right) x_k(1 - x_k) \tag{2.7}$$

2. The tent map

$$x_{k+1} = F_{tent}(x_k) = \left(2 - \frac{\xi}{2}\right) \min(x_k, 1 - x_k) \tag{2.8}$$

3. The Cusp map

$$x_{k+1} = F_{lorenz}(x_k) = \left(1 - \frac{\xi}{4}\right) \left(1 - \left|\frac{1}{2} - x\right| - \sqrt{\left|\frac{1}{4} - \frac{x}{2}\right|}\right) \tag{2.9}$$

We also consider another "sharp" version of the Cusp map

$$x_{k+1} = F_{lorenz}(x_k) = \left(1 - \frac{\xi}{4}\right) \left(1 - \left|\frac{1}{2} - x\right| - \left(\left|\frac{1}{4} - \frac{x}{2}\right|\right)^{0.3}\right) \tag{2.10}$$

In all three maps, the parameter ξ controls the height of the maps. Figure 2-5 shows the logistic map for $\xi = 0.8$, the tent map for $\xi = 0.2$ and the two Cusp maps for $\xi = 0.2$.

Although the logistic map, the tent map and the Cusp map have the same mono-

tonic trends in $[0, 0.5]$ and $[0.5, 1]$, the smoothness of their mean

$$\overline{x} = \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} x_k \tag{2.11}$$

with respect to the parameter ξ are very different. Figure 2-6 plots the mean $\overline{x}$ of the three chaotic maps against the parameter ξ. The mean is approximated as

$$\overline{x} \approx \frac{1}{N\,M} \sum_{i=1}^{M} \sum_{k=n_0}^{N+n_0} x_{i,k}, \quad x_{i,k+1} = F(x_{i,k}), \qquad M = 1000, N = 50000, n_0 = 1000, \tag{2.12}$$

and $x_{i,0}$ are uniformly randomly sampled in $[0.25, 0.75)$.

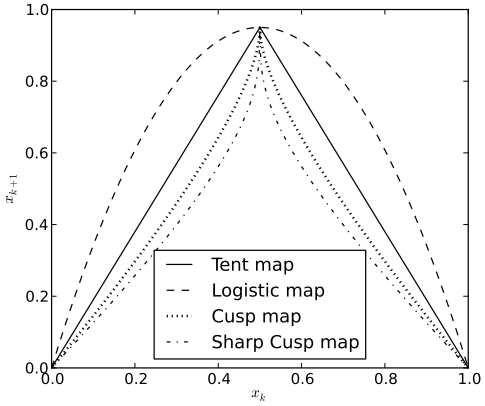


Figure 2-5: The shape of the logistic, tent and Cusp maps.

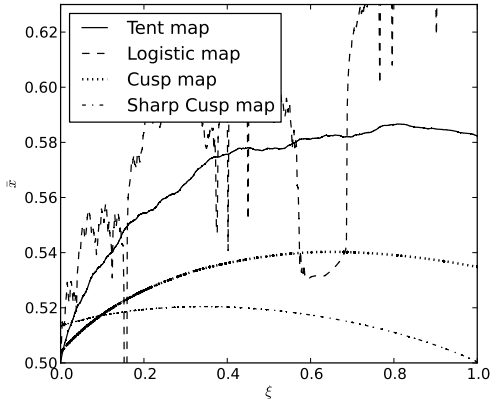


Figure 2-6: Smoothness of $\overline{x}$ as a function of parameter ξ.

The mean $\overline{x}$ of the logistic map appears to be discontinuous with respect to ξ. The mean of the tent map appears to be continuous and differentiable with respect to ξ, but it is difficult to assess its higher order smoothness. The mean of the two Cusp maps appears to be smoother than the tent map.

The smoothness of $\overline{x}$ with respect to the parameter ξ is correlated to the smoothness of the stationary density distribution. The stationary density distribution is a density distribution in the state space that is invariant under the dynamical system. For hyperbolic and quasi-hyperbolic systems, it can be rigorously characterized as the Sinai-Ruelle-Bowen (SRB) measure [20]. The stationary density can be computed by

evolving the dynamical system, with the initial condition drawn from an arbitrary, continuous density distribution in phase space.

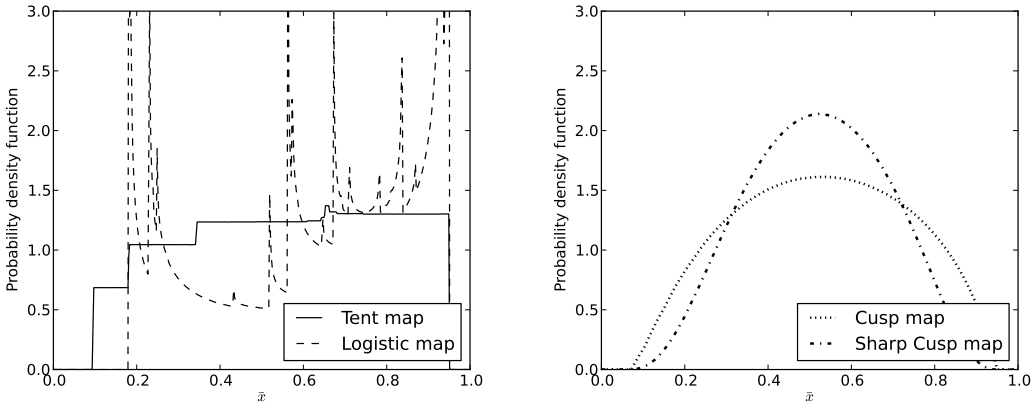


Figure 2-7: Stationary density

Figure 2-7 shows the stationary density distribution of the logistic map, the tent map and the two Cusp maps. The logistic map has a stationary density function that concentrates at discrete points, as evident from the peaks in the density function. The stationary density function of the tent map is bounded, but appears to contain discontinuities. The density of the Cusp map is continuous; while the density of the sharp Cusp map appears to be the most smooth. We find that the maps with smoother mean quantities tend to have smoother stationary distributions.

The same conclusion can be drawn from continuous dynamical systems. Here, we analyze the mean quantities and the stationary density distributions of the two most well known chaotic attractors: the Rössler attractor

$$\frac{dx}{dt} = -y - z, \quad \frac{dy}{dt} = x + a\,y, \quad \frac{dz}{dt} = b + z(x - c) \tag{2.13}$$

and the Lorenz attractor (see 6.1).

For the Rössler attractor, we analyze how $\overline{x}$ and $\overline{z}$ change as the parameter c varies. For the Lorenz attractor, we know that $\overline{x} \equiv \overline{y} \equiv 0$ due to symmetry of the governing equation. Therefore, we focus on the non-trivial quantities $\overline{z}$ and $\overline{x^2}$ as the Rayleigh number r varies.

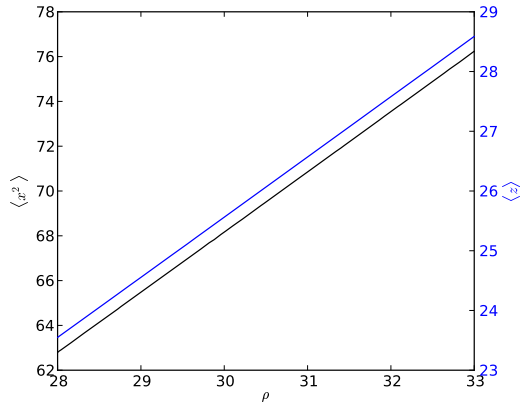


Figure 2-8: $\overline{z}$ and $\overline{x^2}$ of the Lorenz attractor as r varies.

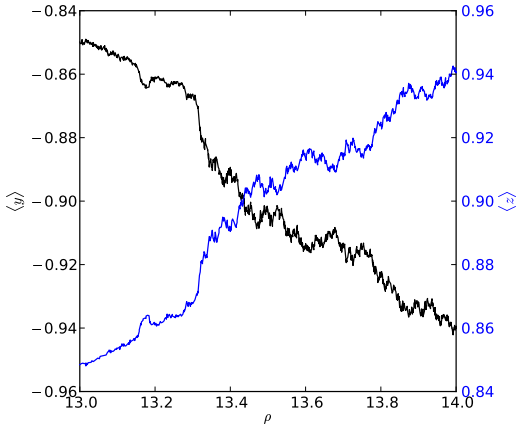
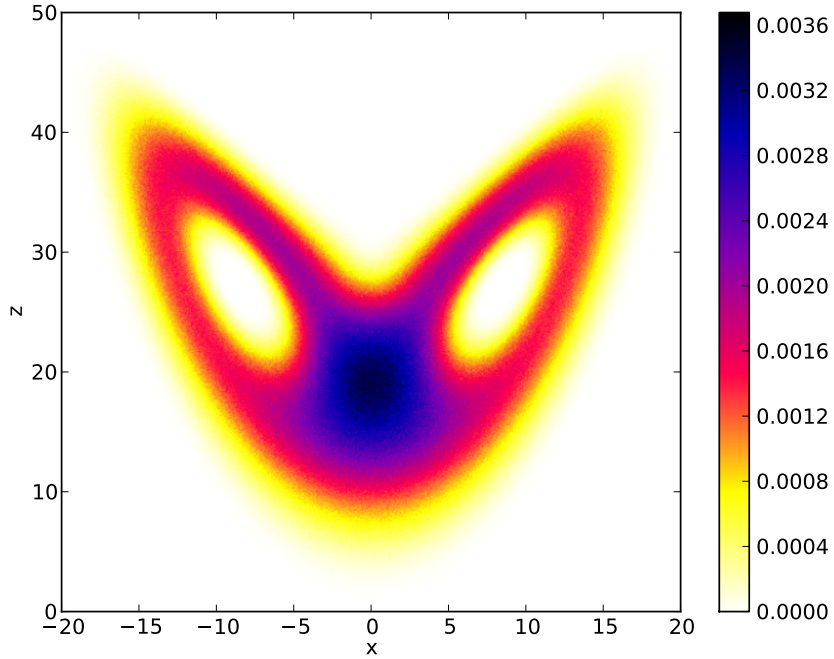


Figure 2-9: $\overline{x}$ and $\overline{z}$ of the Rössler attractor as c varies.

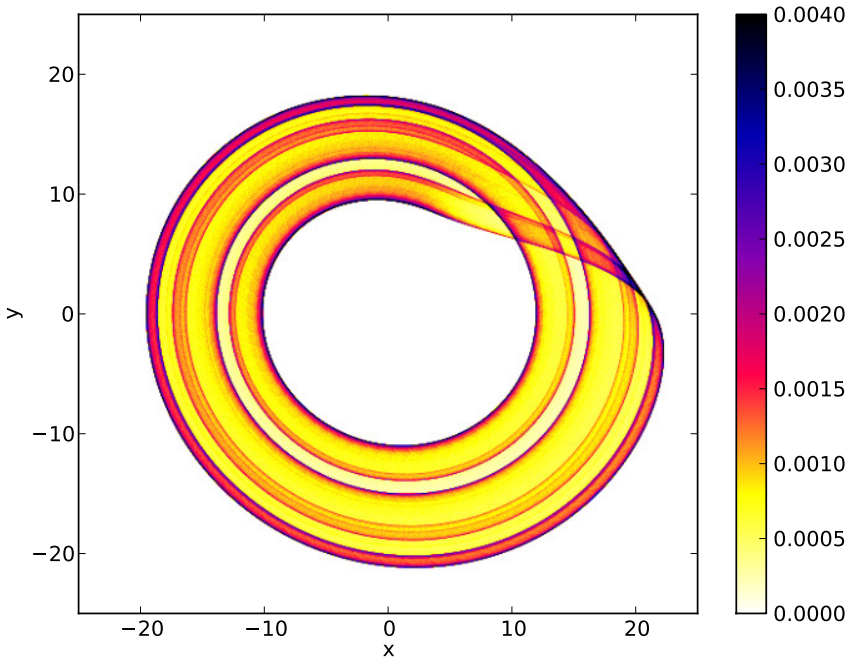
Figures 2-8 and 2-9 show how the mean quantities respond to parameter change for the Lorenz attractor and the Rössler attractor. The Rössler attractor has similar behavior to the logistic map. The mean quantities are not smooth functions of the parameter c. The Lorenz attractor has mean quantities that are smooth functions of its parameter.

Figures 2-10 a and b show the stationary density distributions projected onto the xz and xy planes respectively for the Lorenz and Rössler attractors. As was the case for the 1D maps, the density of the Lorenz attractor, whose mean quantities vary smoothly with respect to parameter changes has a smooth density distribution. A number of discontinuities are present in the density distribution of the Rössler attractor, whose mean quantities do not exhibit smooth variation with respect to parameter changes.

The relationship between hyperbolicity and smoothness of stationary density provide an empirical justification for the methods developed in this paper. If the mean quantities are differentiable with respect to the parameters of a chaotic dynamical system, the stationary density function in the state space is likely smooth on its attractor manifold. This smooth density function can be accurately solved by discretizing its governing equation, namely the probability density equation on its attractor manifold. Sensitivity derivatives of the mean quantities with respect to the parameters can



(a) Lorenz attractor at $\rho = 28, \sigma = 10, \beta = 8/3$.



(b) Rossler attractor at $a = b = 0.1, c = 14$.

Figure 2-10: The stationary density of the Lorenz attractor and the Rössler attractor projected into the (x, z) plane.

then be computed via sensitivity analysis of the probability density equation. Also, the smooth variations of mean quantities with parameters indicates that computing the shadow trajectory required for LSS is a well posed problem.

Chapter 3

Probability Density Adjoint Method for Chaotic 1D Maps

3.1 Introduction

This chapter uses the parameterized cusp map as an example to illustrate the probability density adjoint method. This 1D map is defined as

$$x_{k+1} = F_{cusp}(x_k) = 1 - \xi|2x - 1| - (1 - \xi)\sqrt{|2x - 1|} \tag{3.1}$$

Where the parameter $0 \leq \xi \leq 1$ defines the shape of the map. When $\xi = 1$, the map is a tent map (equation (2.8)); when $\xi = 0.5$, the map is a cusp map (equation (2.9)). The method was used to compute the sensitivity of the mean $\bar{x}$ with respect to the parameter ξ.

3.2 Computing Stationary Density

The stationary density $\rho_s(x)$ is a one dimensional probability density distribution determined by a given mapping function $x_{k+1} = F(x_k)$. It is governed by the Frobenius-Perron equation,and defines the probability that an initial point x_0 will be mapped to some region Δx after infinitely many mappings. Consider a series of random vari-

ables X_k satisfying $X_{k+1} = F(X_k)$ for all $k \geq 0$. The distribution of X_k converges to the stationary distribution as $k \to \infty$ whenever X_0 has a finite distribution function. Denote the Frobenius-Perron operator P as the map from the probability distribution ρ_k of X_k to the probability distribution ρ_{k+1} of X_{k+1} [5]. Then $\rho_s = \lim_{k\to\infty} P^k(\rho_0)$ for any finite ρ_0. An equivalent statement is that $\rho_s(x)$ is an eigenfunction of the operator P, with an eigenvalue of one:

$$(P\rho_s)(x) = \rho_s(x) \ , \quad x \in [0,1] \tag{3.2}$$

The operator P in equation (3.2) is the Frobenius-Perron operator defined in [5] as:

$$\int_0^1 P\rho(x)dx = \int_0^1 \rho(F(x))dx \tag{3.3}$$

To derive P, recall that probability density is conserved in our domain, phase space, by the normalization axiom of probability.

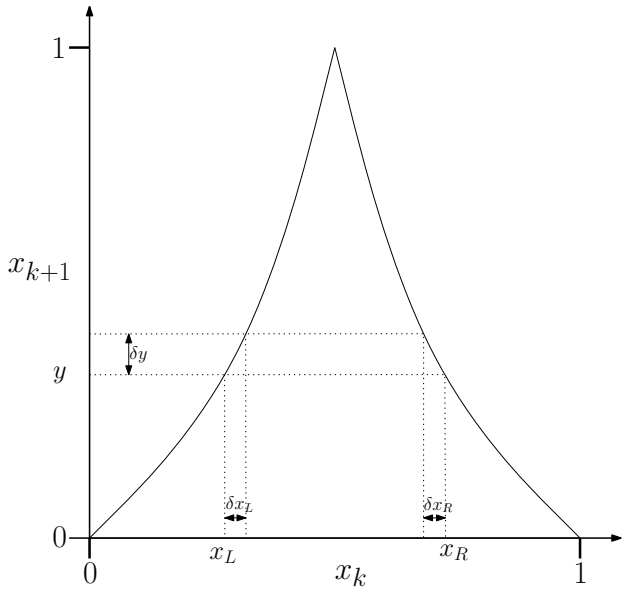


Figure 3-1: Density Mapping for the cusp map

In the case of the map shown in figure 3-1, the integral of the density contained in the small intervals δx_L and δx_R will be mapped into the interval δy. This can be written as follows, where $y = F(x_L) = F(x_R)$:

$$\int_{y}^{y+\delta y} \rho_{k+1}(s)\ ds = \int_{x_L}^{x_L+\delta x_L} \rho_k(s)\ ds + \int_{x_R}^{x_R+\delta x_R} \rho_k(s)\ ds$$

Differentiating with respect to s and dividing both sides by dy/ds, an expression for the mapping of density is obtained:

$$\rho_{k+1}(y) = \frac{1}{|F'(x_L)|}\rho_k(x_L) + \frac{1}{|F'(x_R)|}\rho_k(x_R) \tag{3.4}$$

Where $F'(x) = \frac{dF}{dx}$. Ding and Li [5] compute ρ_0 by using finite elements to construct a discrete approximate of P. Both linear and higher order elements were investigated and ρ_0 was correctly computed for a number of 1D maps including the tent map.

We construct a finite difference discretization of the Frobenius-Perron operator P based on equation (3.4). The interval $[0, 1]$ is discretized into n equally spaced nodes, with $y_i = \frac{i-1}{n-1}$. We represent the discretized version of the linear operator P as an n by n matrix P_n. From equation (3.4), for the discretized density distributions $\underline{\rho}_k \equiv (\rho_k(y_1), \rho_k(y_2), ..., \rho_k(y_n))$ and $\underline{\rho}_{k+1} \equiv (\rho_{k+1}(y_1), \rho_{k+1}(y_2), ..., \rho_{k+1}(y_n))$:

$$\underline{\rho}_{k+1} = P_n \underline{\rho}_k$$

The matrix P_n is constructed by finding $x_{Li}, x_{Ri} = F^{-1}(y_i)$. This is done by computing the inverse functions associated the left and right sides of $F(x)$ with Newton's method. Next, $F'(x)$ is determined at all x_{Li} and x_{Ri}. In most cases, x_{Li}, x_{Ri} will not be equal to any y_k from the discretization. To account for this, $\rho(x_{Li})$ and $\rho(x_{Ri})$ are found by linear interpolation between the two nearest nodes. This means that each row of P_n will typically contain two pairs of non-zero entries, one pair for the right side of $F(x)$, the other for the left side. For a uniform discretization of y_i, the non-zero entries will form the shape of $F(x)$ upside down in the matrix, as shown in figure 3-2. It is important to note that although equation (3.4) is derived assuming conservation of probability mass, P_n does not conserve probability mass. Unlike P, the largest eigenvalue λ of P_n is not exactly one, due to numerical error from the interpolation. To use P_n to compute $\rho_s(x)$ with a power iteration, $\rho_s(x)$ must be manipulated after each iteration such that its integral is equal to one.

As shown by figure 3-3, $\rho_s(x)$ for the cusp map is continuous, suggesting that the objective function is continuous with respect to ξ and the sensitivity with respect to ξ is defined.

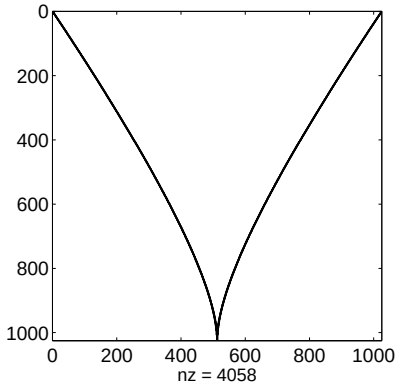


Figure 3-2: Cusp map transition matrix P_n structure for $\xi = 0.5$.

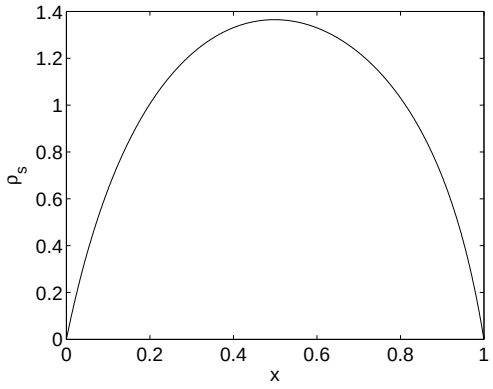


Figure 3-3: Cusp map density distribution ρ_s for $\xi = 0.5$. Generated with 256 nodes.

3.3 Computing gradients using the density adjoint

By the definition of the Frobenius-Perron Operator P, a perturbation to $F(x)$ leads to a perturbation to P. As $\rho_s(x)$ is the first eigenfunction of the Frobenius-Perron Operator, there is a density perturbation $\delta\rho_s(x)$ corresponding to a perturbation to the operator. A perturbation to a mean quantity $\delta\bar{J}$ can be computed from $\delta\rho_s(x)$ using the following expression, where $J(x)$ is the quantity of interest:

$$\delta\bar{J} = \int_0^1 J(x)\ \delta\rho_s(x)\ dx \tag{3.5}$$

$\delta\bar{J}$ can also be computed using the adjoint density ϕ

$$\delta\overline{J} = \int_0^1 \phi(x)\ \delta P\ \rho_s(x)\ dx \tag{3.6}$$

where ϕ satisfies the adjoint equation:

$$P^*\phi - \phi = \overline{J} - J \tag{3.7}$$

For a more detailed derivation of the adjoint equation, see appendix A.1.1

λ is the first eigenvalue of the operator and is equal to one.

The term $\delta P \rho_s$ in equation 3.6 can be found by considering the mapping of probability mass. First we define $\delta\rho_0 = \delta P \rho_s$. From equation (3.2):

$$\rho_s + \delta\rho_0 = (P + \delta P)\rho_s$$

Assuming a small perturbation δP (and therefore a small δF):

$$\int_0^y \delta\rho_0 \, ds = \rho_s(x_L)\delta x_L - \rho_s(x_R)\delta x_R \tag{3.8}$$

For a small perturbation δF, it can be shown that:

$$\frac{\delta F}{\delta x} \approx F'(F^{-1}(y))$$

Substituting into equation (3.8) and differentiating with respect to y:

$$\delta\rho_0 = \delta P \rho_s = \frac{\partial}{\partial y}\left(\frac{\rho_s(x_L)}{F'(x_L)}\delta F(x_L) - \frac{\rho_s(x_R)}{F'(x_R)}\delta F(x_R)\right) \tag{3.9}$$

Combining equations(3.9) and (3.6), an expression for $\delta\bar{J}$ in terms of a mapping function perturbation δF is obtained:

$$\delta\bar{J} = \int_0^1 \phi(y)\frac{\partial}{\partial y}\left(\frac{\rho_s(x_L)}{F'(x_L)}\delta F(x_L) - \frac{\rho_s(x_R)}{F'(x_R)}\delta F(x_R)\right) dy \tag{3.10}$$

If F and δF are symmetric, $F'(x_L)$ will be positive and $F'(x_R)$ will be negative, therefore (3.4) can be rewritten as:

$$\rho_s(F(x)) = \frac{1}{F'(x_L)}\rho_s(x_L) - \frac{1}{F'(x_R)}\rho_s(x_L) \tag{3.11}$$

Combining equations (3.10) and (3.11):

$$\delta\bar{J} = \int_0^1 \phi(y)\frac{\partial}{\partial y}(\rho_s(y)\delta F(F^{-1}(y)))dy \tag{3.12}$$

This is consistent with the equation for the density derivative in [20]. To compute the gradient with respect to some parameter ξ, substitute $\frac{\partial F}{\partial \xi}\delta\xi$ for δF in equation (3.12) and divide through by $\delta\xi$:

$$\frac{\partial \bar{J}}{\partial \xi} = \frac{\delta \bar{J}}{\delta \xi} = \int_0^1 \phi(y) \frac{\partial}{\partial y} \left(\frac{\rho_s(x_L)}{F'(x_L)} \frac{\partial F}{\partial \xi}|_{x_L} - \frac{\rho_s(x_R)}{F'(x_R)} \frac{\partial F}{\partial \xi}|_{x_R} \right) dy \tag{3.13}$$

Or for the symmetric case:

$$\frac{\partial \bar{J}}{\partial \xi} = \int_0^1 \phi(y) \frac{\partial}{\partial y} \left(\rho_s(y) \frac{\partial F}{\partial \xi}|_{F^{-1}(y)} \right) dy \tag{3.14}$$

Finally, care needs to be taken when discretizing the density adjoint equations. The first eigenvalue of the discrete operator P_n is not exactly one and can change when the system is perturbed. Because of this, an additional adjoint equation is required for λ to compute the discrete density adjoint (see appendix A.1.3 for a derivation):

$$\begin{bmatrix} P_n^T - \lambda I & -\underline{v} \\ -\underline{\rho}_s^T & 0 \end{bmatrix} \begin{bmatrix} \underline{\phi} \\ \eta \end{bmatrix} = \begin{bmatrix} \underline{J} \\ 0 \end{bmatrix} \tag{3.15}$$

Where η is the adjoint of λ and can be shown to be equal to $\overline{J}$ in the continuous limit.

3.4 Algorithm Summary

To compute some gradient $\frac{\partial \overline{J}}{\partial \xi}$, the following algorithm was used:

1. Compute the inverse of the mapping function $F(x)$ using Newton's Method.
2. Construct the matrix P_n using the equations outlined in section 3.2.
3. Determine the stationary density $\underline{\rho}_s$ using a power method. Also determine the left eigenvector v corresponding to the eigenvalue λ of $\underline{\rho}_s$.
4. Compute the adjoint variable $\underline{\phi}$ by solving (3.15). To solve (3.15), be sure to take advantage of the sparseness of P_n.

5. Compute the gradient using (3.13) or (3.14). Approximate the y-derivative with a 2nd order center finite difference scheme.

3.5 Density adjoint for the cusp map

The sensitivity of the mean $\bar{x}$ with respect to the parameter ξ for the cusp map was computed. For comparison, $\frac{\partial \bar{x}}{\partial \xi}$ was also computed using 1st order finite differences of equation (1.2) adapted for the 1D case and discretized in n nodes:

$$\bar{x} = (1/n)x^T \underline{\rho}_s \tag{3.16}$$

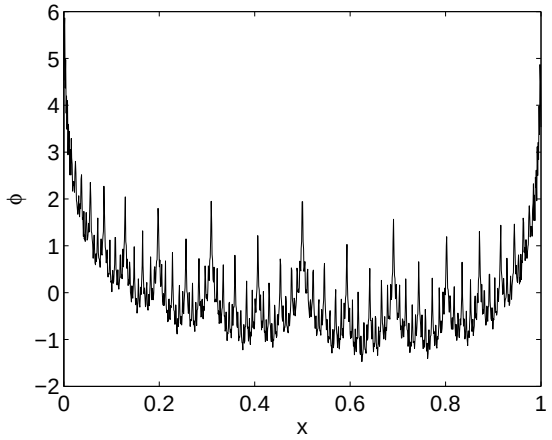


Figure 3-4: Adjoint ϕ for the cusp map with $\xi = 0.5$, generated using 1024 nodes.

Figure 3-4 shows the adjoint density distribution for the cusp map with $\xi = 0.5$. Sensitivities are almost discontinuous, so small perturbations to density can have large effects on the objective function, which is consistent with Lorenz's "butterfly effect". However, there is a fractal structure to the density adjoint. This arises from the adjoint being computed backwards in time with the operator P_n^T. P_n folds and stretches density distributions, so P_n^T duplicates and compresses features of adjoint density distributions. This fractal structure arises because of the cusp map's "peak" at $x = 0.5$, which causes the folding and stretching.

Despite the additional numerical dissipation, the 1D density adjoint computes accurate gradient values. Figure 3-5 shows the adjoint and finite difference computed

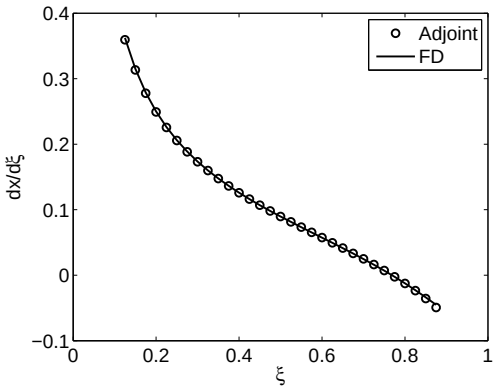


Figure 3-5: Comparison of gradients computed using the adjoint method and the finite difference method. 1D space between 0 and 1 was discretized using 256 nodes.

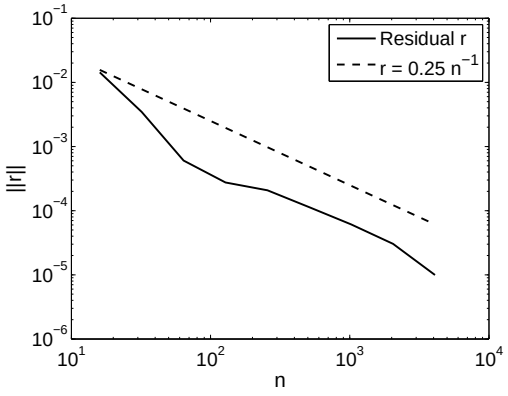


Figure 3-6: Convergence of the residual r of $\frac{\partial \overline{J}}{\partial \xi}$ with the number of nodes n for $\xi = 0.5$. The residual was calculated by taking the L2 norm of the difference between the gradient for n nodes and 8096 nodes.

gradients match up well visually. It was found that the adjoint method predicts the gradient within 5% of the finite difference calculation for most values of ξ. The highest error is just under 10%. The order of convergence of the gradient varied slightly with ξ and was typically around 1.15, as in figure 3-6.

Chapter 4

Probability Density Adjoint Method for Continuous Chaotic Systems

4.1 Introduction

The following chapter uses the Lorenz equations as an example to illustrate the probability density adjoint method. The method was used to compute the partial derivative of $\overline{z}$ with respect to the parameters s, r, b and z_0 in the slightly modified Lorenz equations:

$$\begin{aligned}\frac{dx}{dt} &= s(y-x)\\ \frac{dy}{dt} &= -x(z-z_0)+rx-y\\ \frac{dz}{dt} &= xy-b(z-z_0)\end{aligned}$$

The parameters were set to their canonical values of $s = 10$, $r = 28$, $b = 8/3$ and $z_0 = 0$. The Lorenz attractor with these parameters has a fractal dimension of roughly 2.05, so the attractor was approximated as a 2D surface in 3D phase space.

4.2 Computing Stationary Density

In multiple dimensions, one could build a discrete Frobenius-Perron operator P_n as in the 1D case. P_n would be an N by N matrix, where N is number of cells or nodes used to discretize the strange attractor. To reduce the size of the matrix P_n, the matrix is built for a Poincaré section. In this case, P_n is n by n, where n is the number of nodes in the Poincaré section, which is typically a small fraction of the total number of nodes N. For the Lorenz attractor, a good choice for the Poincaré section is a constant z-plane including the two non-zero unstable fixed points at $(\pm\sqrt{b(r-1)}, \pm\sqrt{b(r-1)}, r-1)$.

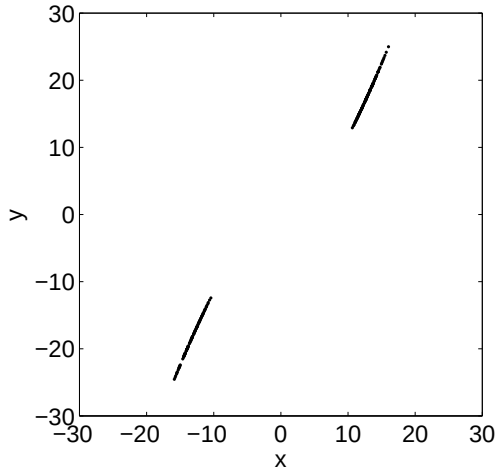


Figure 4-1: Poincaré Section at z = 27 for Lorenz attractor trajectories with $\frac{\partial z}{\partial t} > 0$.

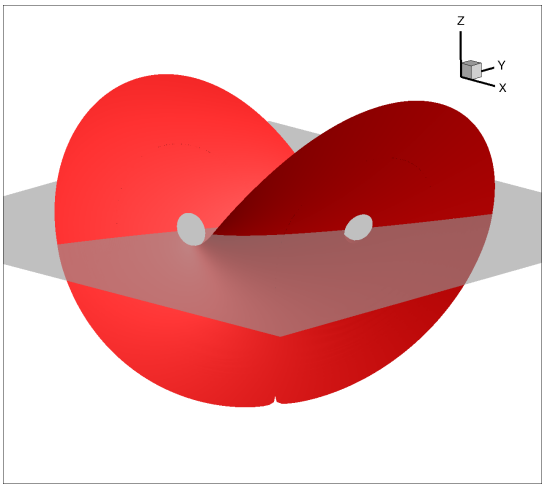


Figure 4-2: Three dimensional view of the 2D surface approximating the Lorenz Attractor and the Poincaré section at z = 27.

This Poincaré section has an attractor cross-section that can be well approximated as a function of either x or y. Therefore solving for the density distribution at the Poincaré section is a 1D map problem and the stationary density distribution ρ_s can be computed from this operator P_n as shown in chapter 3. For the Lorenz attractor, the starting positions of the streamlines were determined using a 7th order polynomial curve fit through a Poincaré section taken from a trajectory with length $t = 10000$.

As the Poincaré section can be modelled with a polynomial curve fit, the attractor itself can be approximately modelled as a 2D surface, as shown in figure 4-2.

As the Lorenz attractor lies in a three dimensional phase space, vector notation is used in this section. A lower case symbol is a scalar (i.e. stationary density ρ_s), a symbol with an arrow overhead is a column vector (i.e. phase space position $\vec{x} \equiv u$ from equation (1.1)) and a matrix/tensor is indicated by bold script (i.e. a Jacobian $\mathbf{J}$).

Unlike the 1D case, an explicit form of the mapping function is not available. Instead, a probability mass conservation equation is derived from the normalization and additivity axioms of probability. The probability mass conservation equation can be used to compute the ratio between densities for a given "mapping", which can be used instead of the mapping function slope in the density mapping equation.

A very helpful physical analogy to the conservation of probability on the attractor surface is the conservation of mass in a fluid flow. Like mass, probability cannot be created or destroyed according to the nonnegativity and normalization axioms. Therefore, the following equation holds:

$$\vec{\nabla}_s \cdot (\rho_s(\vec{x})\vec{f}(\vec{x})) = 0 \tag{4.1}$$

Where the gradient operator $\vec{\nabla}_s$ is over the surface of the attractor, $\vec{x}$ is a point in phase space R^n and:

$$\frac{d\vec{x}}{dt} = \vec{f}(\vec{x})$$

Is the system of equations governing the dynamical system of interest. The physical analog of $\vec{f}(\vec{x})$ is a velocity field in a fluid flow.

From (4.1), a partial differential equation (PDE) governing the density distribution on an attractor can be derived. From the chain rule:

$$\vec{f}(\vec{x}) \cdot \vec{\nabla}_s \rho(\vec{x}) + \rho(\vec{x}) \nabla_s \cdot \vec{f}(\vec{x}) = 0$$

The Lorenz attractor is approximated as a 2D surface, so two natural coordinates are used; l, which is in the direction of the "velocity field" defined by $\vec{f}(\vec{x})$ and s,

which is orthogonal to l but tangent to the attractor surface. l and s will be referred to as the streamwise and spanwise directions respectively. $\hat{l}$ and $\hat{s}$ are unit vectors along the streamwise and spanwise directions.

Therefore:

$$\vec{\nabla}_s \cdot \vec{f}(\vec{x}) = \frac{\partial f_l}{\partial l} + \frac{\partial f_s}{\partial s}$$

Where f_l and f_s are the $\hat{l}$ and $\hat{s}$ components of $\vec{f}(\vec{x})$.

Using the surface coordinates, the density PDE can be simplified:

$$|\vec{f}(\vec{x})|\frac{\partial \rho}{\partial l} = -\rho(\vec{x})\vec{\nabla}_s \cdot \vec{f}(\vec{x}) \qquad \Rightarrow \qquad \frac{\partial \rho}{\partial t} = -\rho(\vec{x})\vec{\nabla}_s \cdot \vec{f}(\vec{x})$$

To compute $\vec{\nabla}_s \cdot \vec{f}(\vec{x})$, the 3D Cartesian gradients of f_l and f_s are needed. By the definition of the Jacobian, $\mathbf{J}$:

$$\frac{\partial \vec{f_l}}{\partial l} = \mathbf{J}\hat{l}, \qquad \frac{\partial \vec{f_s}}{\partial s} = \mathbf{J}\hat{s}$$

To compute the magnitude of the derivatives in their respective directions:

$$\frac{\partial f_l}{\partial l} = \frac{\partial \vec{f_l}}{\partial l} \cdot \hat{l}, \qquad \frac{\partial f_s}{\partial s} = \frac{\partial \vec{f_l}}{\partial s} \cdot \hat{s}$$

Therefore:

$$\frac{\partial \rho}{\partial t} = -\rho(\vec{x})(\hat{l}^T\mathbf{J}\hat{l} + \hat{s}^T\mathbf{J}\hat{s}) \tag{4.2}$$

As ρ is invariant when multiplied by a constant, this equation can be integrated to find the ratio between density at different points on a given Poincaré section:

$$\log\frac{\rho(T)}{\rho_0} = -\int_0^T (\hat{l}^T\mathbf{J}\hat{l} + \hat{s}^T\mathbf{J}\hat{s})dt \tag{4.3}$$

Where $\rho_0(x)$ is defined as the density at the beginning of the streamline beginning at x. Equation (4.3) is numerically integrated to find the ratio between the density at the beginning and end of n streamlines. These ratios, along with the start and

end positions of each streamline can be used to form a Frobenius-Perron Operator P_n with a first eigenvector corresponding to the stationary density distribution at the Poincaré section. As the streamline starting and ending point will rarely match (i.e. $x(T)_i \neq x(0)_j$), linear interpolation is used as in the 1D case to compute the density "flow" between the starting and ending points.

By the symmetry of the Lorenz equations, the Poincaré plane intersections for the Lorenz attractor are 180 degree rotational translations of one another as is evident in figure 4-1, where it can be seen that $x = -x$ and $y = -y$. This symmetry of the attractor can be exploited for lower computational costs. If the attractor is discretized with streamlines starting along the Poincaré section in the first quadrant ($x > 0,\ y > 0$), a portion of the streamlines return to the first quadrant and a portion go to the third quadrant ($x < 0,\ y < 0$), as seen in figure 4-3. By symmetry, the streamlines running from the first to the third quadrant are the same as those from the third to the first rotated 180 degrees about the z-axis. This means that the density flux from the third quadrant is the same as the density flux to the third quadrant. The density flow from returning and incoming streamlines make up two sides of the transition matrix P_n, as shown in figures 4-4 and 4-5.

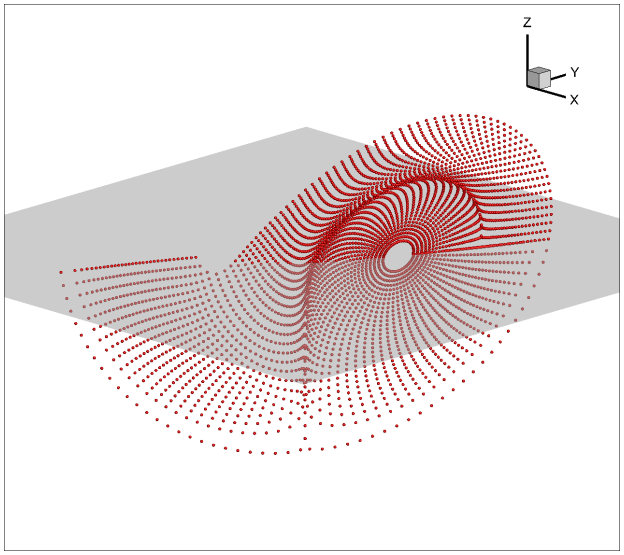


Figure 4-3: Node distribution corresponding to a 64 streamline by 64 streamwise mesh for the Lorenz attractor. It was found that distributing the streamline starting positions so that there were more streamlines near the bifurcation increased the rate of convergence to the true density distribution.

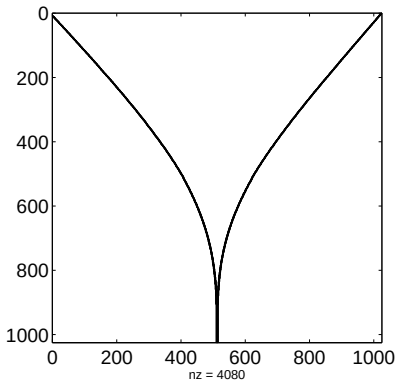


Figure 4-4: Transition Matrix P_n Structure for a roughly uniform streamline distribution. Note the similarity of this matrix to that for the Cusp map

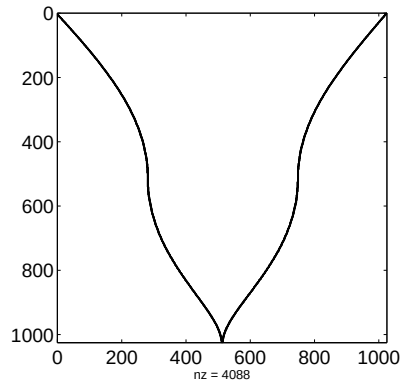


Figure 4-5: Transition Matrix P_n Structure for a non-uniform streamline distribution with more streamlines starting near $y_0 = 18$.

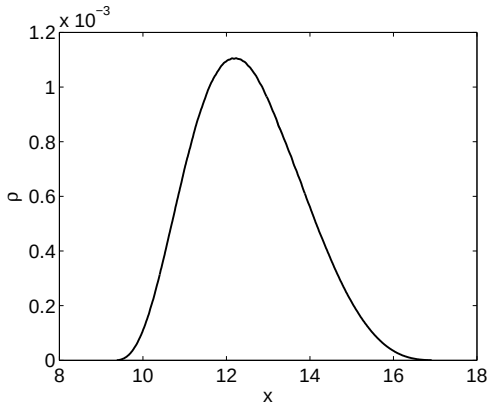


Figure 4-6: Density ρ_s versus x on the Poincare Section at z = 27. 512 streamlines were used to form P_n.

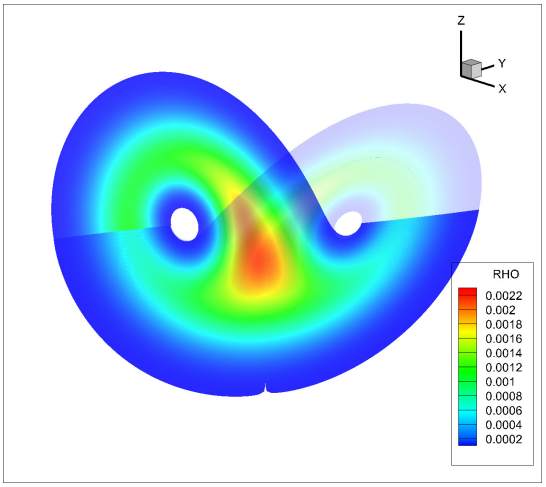


Figure 4-7: Density distribution on the surface of the Lorenz attractor for a 512 by 128 mesh.

Because the finite difference scheme does not ensure conservation of probability mass, the Poincaré stationary distribution $\rho_0(x_0)$ as computed using a power method is not properly normalized. This is because the first eigenvalue is not equal to one as it would be if probability mass was conserved. To normalize $\rho_0(x_0)$ begin with the density over the entire attractor and use the fact that $dl = |\vec{f}(\vec{x})|dt$:

$$\iint \rho \; dlds = \iint \rho |\vec{f}(\vec{x})| dtds = 1 \tag{4.4}$$

Conservation of probability mass along a streamline can be written as:

$$\rho|\vec{f}(\vec{x})|ds = \rho_0|\vec{f}(\vec{x}_0) \times \hat{s}_0|ds_0$$

Where ds is the width of the streamline at a given $\vec{x}$, ds_0 is its initial width, $\vec{f}(\vec{x}_0)$ is the "initial velocity" and $\hat{s}_0$ is the initial spanwise direction. Substituting into equation (4.4):

$$\iint_0^T \rho_0|\vec{f}(\vec{x}_0) \times \hat{s}_0|ds_0 dt = 1$$
$$\int \rho_0 \int_0^T dt|\vec{f}(\vec{x}_0) \times \hat{s}_0|ds_0 = 1$$
$$\int \rho_0 T|\vec{f}(\vec{x}_0) \times \hat{s}_0|ds_0 = 1$$

The discretized form of this equation, which can be used to normalize ρ_0, is:

$$\underline{\rho}_0^T \underline{v} = 1$$

Where:

$$\underline{v}_i = T|f(x_0) \times \hat{s}_0|ds_0 \tag{4.5}$$

and T is the total time a particle spends along streamline i. It can be shown that v is the leading eigenvector of P_n corresponding to $\lambda \approx 1$.

Figure 4-6 shows the Poincaré section stationary distribution $\rho_0(x_0)$ for the Lorenz attractor. Like the stationary distribution for the cusp map it is smooth and continuous.

Once the Poincaré stationary distribution $\rho_0(x_0)$ is computed and normalized, the stationary density distribution over the entire attractor is computed by integrating equation (4.2) for each streamline with $\rho_0(x_0)$ as the initial value of ρ for a streamline starting at x_0. The density distribution computed for the Lorenz attractor is shown in figure 4-7. The apparent discontinuity results from the intersection of the two branches of the attractor. The sum of the density distribution on the intersection of

these two branches is equal to the distribution on the Poincaré section.

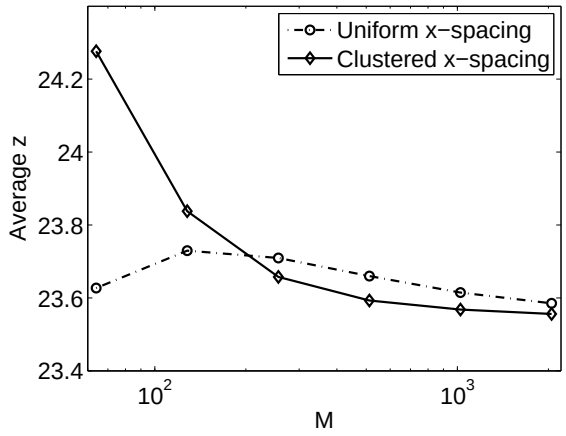


Figure 4-8: Convergence of $\overline{z}$ for two different streamline start point distributions, where M is the number of streamlines. The clustered distribution has streamlines clustered near the bifurcation of the attractor.

Figure 4-8 shows that $\overline{z} = 23.6$ from the density distribution, which is consistent with the value $\overline{z} = 23.550$ found using ensemble averages of long phase space trajectories.

4.3 Computing the Density Adjoint

As the negative Lyapunov exponent for the Lorenz attractor has a large magnitude relative to the positive Lyapunov exponent, it can be assumed that perturbations the long-time averaged quantity $\overline{J}$ arise mainly from perturbations to the stationary density $\delta\rho$ as opposed to perturbations to the attractor manifold:

$$\delta\overline{J} = \iint J(\vec{x})\delta\rho \; dlds \tag{4.6}$$

The adjoint density equation can be found using equation (4.6) and the linearization of equation (4.1) (see appendix A.2.1 for detailed derivation):

$$\frac{\partial\phi}{\partial t} = J(\vec{x}) - \overline{J} \tag{4.7}$$

Perturbations to $\overline{J}$ can then be computed using:

$$\delta\overline{J} = \iint \phi\, \vec{\nabla}_s \cdot (\rho_s \delta\vec{f})\; dlds \tag{4.8}$$

Therefore gradients with respect to some parameter ξ are:

$$\frac{\partial\overline{J}}{\partial\xi} = \iint \phi\, \vec{\nabla}_s \cdot \left(\rho_s \frac{\partial\vec{f}}{\partial\xi}\right) dlds$$

To derive the discrete adjoint equations for a numerical scheme, first consider equation

(1.2):

$$\overline{J} = \iint J(\vec{x})\rho(\vec{x}) dlds$$

This can be rewritten as:

$$\overline{J} = \iint_0^T J(t)dt\rho_0 |\vec{f}(\vec{x}_0) \times \hat{s}_0| ds_0$$

Defining $\underline{\mathcal{J}}_i = \int_0^T J(t)dt$ for streamline i "flowing" from the Poincaré section, the above equation has the discretized form:

$$\overline{J} = \underline{\mathcal{J}}^T D \underline{\rho}_0 \tag{4.9}$$

where D is a diagonal matrix with $|\vec{f}(\vec{x}_0) \times \hat{s}_0| ds_0$ for the ith streamline along the main diagonal. Using D to rescale P_n, the adjoint equation for $\underline{\rho}_0$ can be derived for the 1D Poincaré map as in chapter 3 (see appendix A.2.2 for a detailed derivation):

$$\begin{bmatrix} (D^{-1}P^T D - \lambda I) & -D^{-1}\underline{v} \\ \underline{\rho}_s^T D & 0 \end{bmatrix} \begin{bmatrix} \underline{\phi}_0 \\ \overline{J} \end{bmatrix} = \begin{bmatrix} \underline{\mathcal{J}} \\ 0 \end{bmatrix} \tag{4.10}$$

λ is included in equation (4.10) because it is not exactly one in practice. The adjoint density along the Poincaré section $\phi_0(x_0)$ is computed using equation (4.10). Then equation (4.7) is integrated to compute the adjoint along each streamline, using $\phi_0(x_0)$ as the initial value.

Gradients can be computed by discretizing equation (4.8):

$$\frac{\partial \overline{J}}{\partial \xi} \approx \sum_{k=0}^{N} \phi_k \, [\vec{\nabla}_s \cdot \left(\rho_s \frac{\partial \vec{f}}{\partial \xi} \right)]_k dA_k \tag{4.11}$$

Where dA_k is the area corresponding to the kth node. This can be computed by integrating a differential equation formed using conservation of probability mass (see appendix A.2.3). The quantity $[\vec{\nabla}_s \cdot \left(\rho_s \frac{\partial f}{\partial \xi} \right)]_k$ can be computed by finite differences as ρ_s is known for each node and $\frac{\partial \vec{f}}{\partial \xi}$ can be found analytically for each node (see appendix A.2.4 for a detailed derivation).

4.4 Algorithm Summary

To compute some gradient $\frac{\partial \overline{J}}{\partial \xi}$, the following algorithm was used:

1. Find a Poincaré Section for the attractor such that the intersections trace an approximately one to one function, as seen in in figure 4-1. Find a curve fit for these intersections.

2. Construct the matrix P_n with a loop, by integrating (4.2) along a set of streamlines originating and terminating at the Poincaré Plane from step 1.

3. Determine the stationary density ρ_0 on the Poincaré plane using a power method. Smooth this distribution using a low-pass filter if necessary.

4. Compute $\overline{J}$ using the following equation:

$$\overline{J} = \rho_0^T D \mathcal{J}_s$$

5. Determine the left eigenvector v corresponding to the eigenvalue λ of ρ_0 using (4.5).

6. Compute the Poincaré Plane adjoint variable ϕ_0 by solving (3.15). To solve (4.10), be sure to take advantage of the sparseness of P_n.

7. Using ρ_0 and ϕ_0 as initial values, integrate (4.2) and (4.7) along each streamline to find ρ_s and ϕ for the entire attractor.

8. Find $\frac{\partial \vec{f}}{\partial \xi}$ analytically and calculate its value at all nodes.

9. Compute the gradient using (4.11).

4.5 Density adjoint for the Lorenz equations

Figures 4-9 and 4-10 show the adjoint density distribution on the Poincaré section and on the entire attractor surface. As for the cusp map, the adjoint has a fractal structure. Starting from the Poincaré plane, a given distribution is duplicated along both branches of the attractor and is propagated backward in time towards the origin, where it is squeezed and merged with the distribution from the other side of the attractor. This merged, squeezed distribution then propagates back to the Poincaré section. As the sensitivity is for long time averages, this process is repeated many times, resulting in the fine fractal structures shown in figures 4-9 and 4-10.

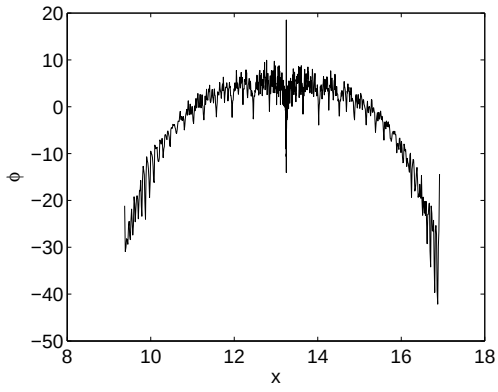


Figure 4-9: Adjoint ϕ versus x on the Poincaré Section at z = 27. 1024 streamlines were used to form P_n.

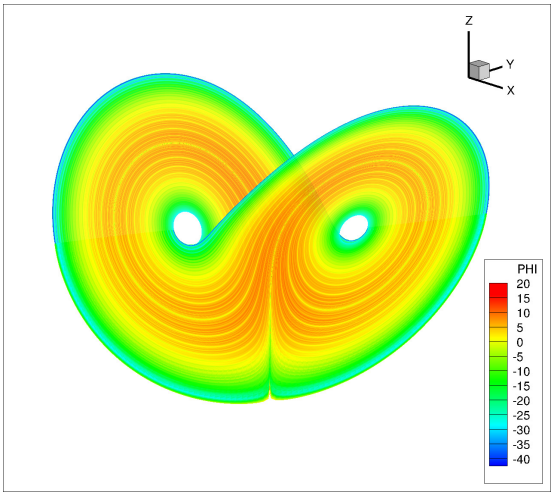


Figure 4-10: Adjoint distribution on the surface of the Lorenz Attractor for a 512 by 128 mesh.

Equation (4.11) can be used to compute the sensitivity of the average z position of the Lorenz attractor with respect to a number of parameters.

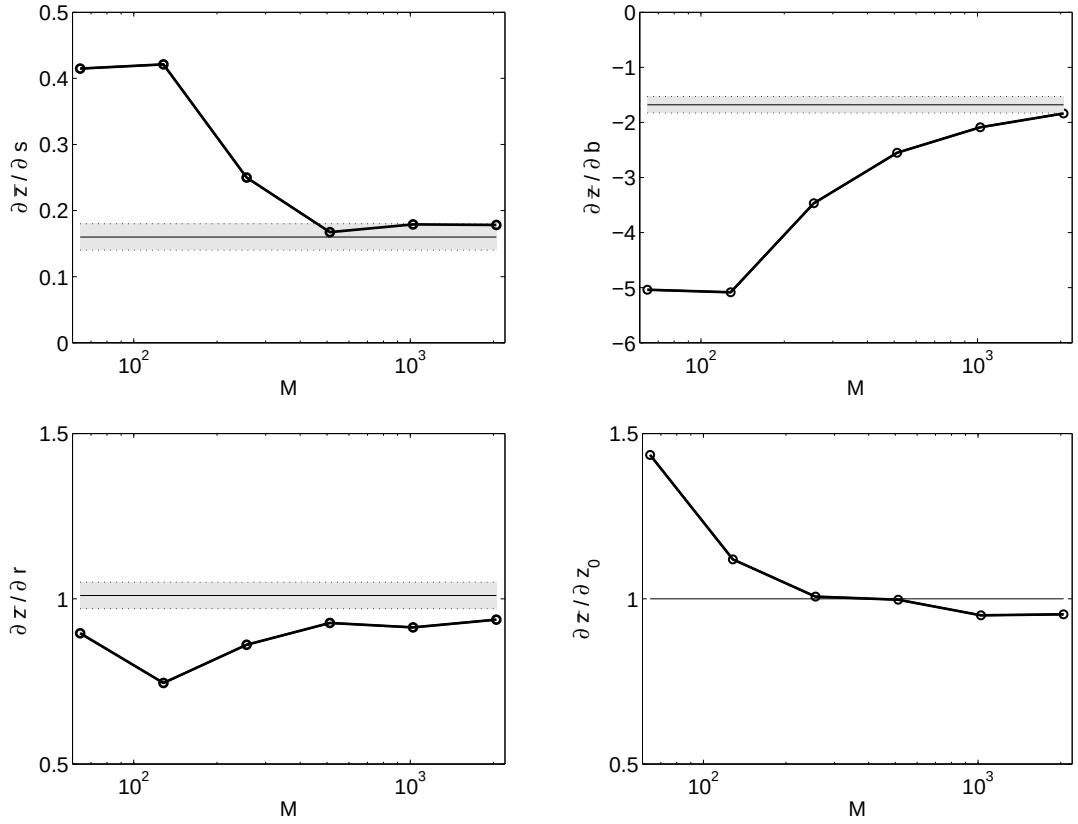


Figure 4-11: Sensitivity of $\overline{z}$ with respect to the parameters s, b, r and z_0 for difference grid sizes M. M is number of the streamlines, N ($= \frac{1}{4}M$) is the number of nodes along a streamline. The level lines correspond to sensitivities computed using finite differences of ensemble averaged data and the dotted lines are the 3σ confidence bounds [24].

From figure 4-11 it can be seen that the density adjoint method predicts the gradients quite well. $\frac{\partial \overline{z}}{\partial z_0}$ is the most accurately predicted (under-predicted by 5% for the denser grids). $\frac{\partial \overline{z}}{\partial r}$ converges to values roughly 10% under the gradient predicted using finite differences. $\frac{\partial \overline{z}}{\partial r}$ is over-predicted by around 12%. Finally $\frac{\partial \overline{z}}{\partial b}$ converges slower than the other gradients and is over-predicted by around 9% by the densest grid (2048 by 512).

Larger errors are present in the continuous, two dimensional implementation of the density adjoint because changes in density distribution are not solely responsible for changes in the objective function. Perturbations to the Lorenz equation parameters displace and deform the attractor manifold as well, and these changes are not accounted for by the density adjoint method outlined in the this paper.

4.6 Discussion

As discussed in the previous two chapters, the probability density adjoint method computes the sensitivity of long time averaged quantities to input parameters for ergodic chaotic dynamical systems if a few conditions are met. Firstly, the system must have a smooth invariant measure (probability density function). Secondly, the manifold of the strange attractor associated with the system must be correctly approximated and discretized in phase space. For 1D chaotic maps discretization is trivial. For continouous chaotic systems, such as the Lorenz equations, discretization of the attractor manifold is more involved but achievable.

Although some additional work needs to be done on computing the contribution of the displacement of the attractor manifold to sensitivities, the density adjoint method has computed accurate gradients for the 1D cusp map and the approximately 2D Lorenz attractor. The probability density adjoint method could be used to analyze low-dimensional chaotic systems. However, the need to discretize the attractor could make it difficult to compute the density adjoint for high-dimensional chaotic systems, such as computational simulations of turbulent airflow. The probability density adjoint also provides insight into adjoint sensitivities of chaotic systems. The adjoint density is observed to be fractal in structure, an illuminating result given that the stretching and folding of density distributions forwards in time becomes compressing and duplicating adjoint density distributions backwards in time. This fractal structure is due to the peak of the cusp map and the bifurcation of the Lorenz attractor and further work is needed to see if fractal adjoints are specific to these systems or if all chaotic dynamical systems have fractal adjoint densities. The probability density adjoint formulation also shows that accurate gradients can be computed in the presence of some numerical dissipation.

Chapter 5

Least Squares Sensitivity Method

As for the probability density adjoint method, the least squares sensitivity (LSS) method requires the assumption of ergodicity. Unlike the probability density adjoint method, LSS does not require the discretization of the attractor associated with the dynamical system of interest. This makes LSS more readily extendible to higher dimensional chaotic systems than the probability density adjoint method.

5.1 Lyapunov Exponents and the Shadowing Lemma

Before LSS and the shadowing lemma from which it is derived is discussed, a more in-depth discussion of Lyapunov exponents and Lyapunov covariant vectors is required. For some system $\frac{du}{dt} = f(u)$, there exist Lyapunov covariant vectors $\phi_1(u), \phi_2(u), ..., \phi_i(u)$ corresponding to each Lyapunov exponent Λ_i, which satisfy the equation:

$$\frac{d}{dt}\phi_i(u(t)) = \frac{\partial f}{\partial u} \cdot \phi_i(u(t)) - \Lambda_i \phi_i(u(t))$$

To understand what Λ_i and ϕ_i represent, consider a sphere comprised of perturbations δf to a system $\frac{\partial u}{\partial t} = f(u)$ at some time, as shown in the far left of figure 5-1. As this system evolves in time, this sphere expands in some directions, contracts in some, and remains unchanged in others. The rate at which the sphere expands or contracts corresponds to the Lyapunov exponent Λ_i and the corresponding direction

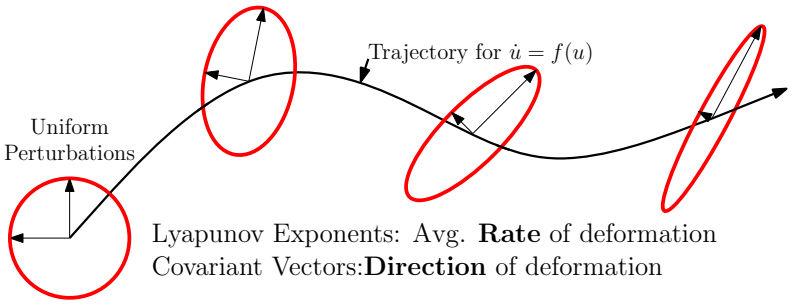


Figure 5-1: Schematic of Lyapunov exponents and covariant vectors

of expansion or contraction is the Lyapunov covariant vector ϕ_i. It is important to note that the ϕ_i are not the same as local Jacobian eigenvectors, ϕ_i depends on all Jacobians along a trajectory. Also, the ϕ_i are not necessarily orthogonal, but the number of Lyapunov covariant vectors is the same as the number of dimensions of the system.

A strange attractor, the type of attractor associated with chaotic dynamical systems, has at least one positive and one zero Lyapunov exponent. To illustrate the effect of the positive exponent, we consider figure 5-2. We see that if the perturbed trajectory has the same initial condition as the unperturbed trajectory, the two trajectories diverge exponentially, leading to the issues with traditional sensitivity analysis discussed in chapter 2.

However, the assumption of ergodicity means that it is not necessary to compare two trajectories with the same initial condition. Therefore, an initial condition can be chosen such that the perturbed and unperturbed trajectories do not diverge, resulting in the blue trajectory in figure 5-2. The existence of this trajectory, called a "shadow trajectory", follows from the shadowing lemma [16] : *Consider a reference solution* u_r *to*

$$\frac{du}{dt} = f(u, \xi)$$

If this system has a Hyperbolic strange attractor and if some system parameter ξ *is slightly perturbed:*

For any $\delta > 0$ *there exists* $\varepsilon > 0$*, such that for every* u_r *that satisfies* $\|du_r/dt -$

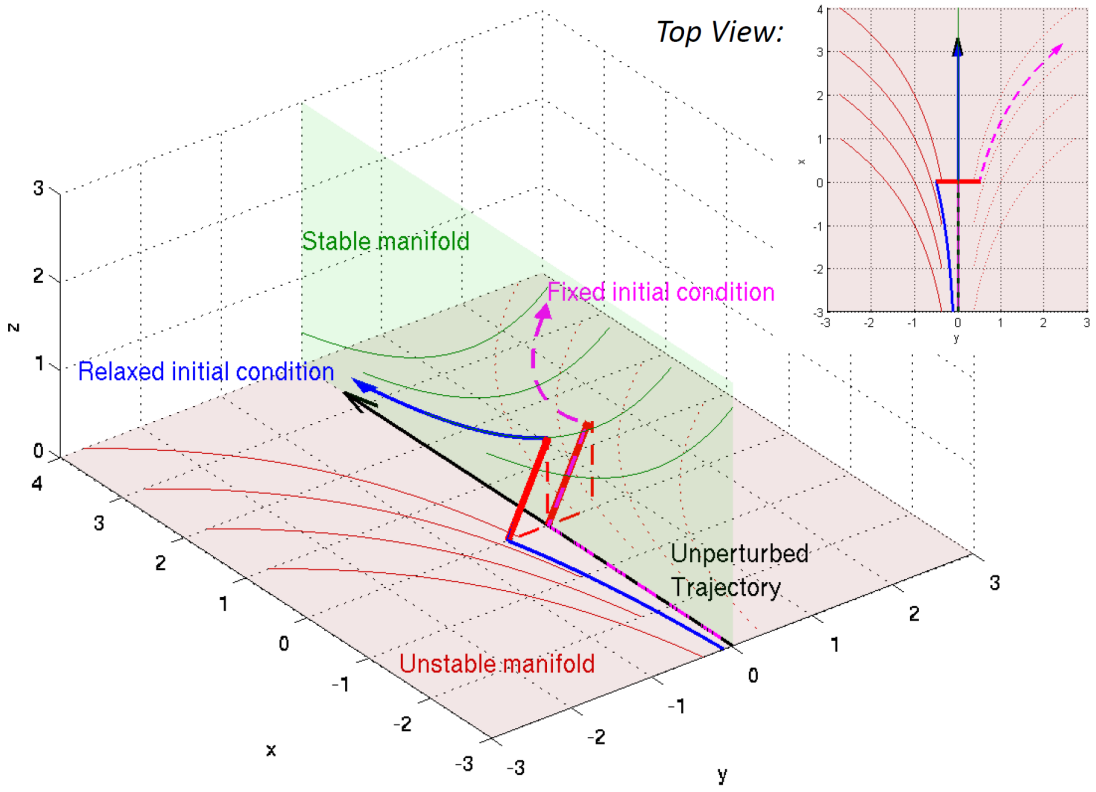


Figure 5-2: Phase space trajectory of a chaotic dynamical system. The unstable manifold, in red, is the space of all Lyapunov covariant vectors corresponding to positive exponents. The stable manifold, in green, corresponds to the space of all covariant vectors associated with negative exponents. A perturbation to the system (in red) has components in both manifolds, and the unstable component causes the perturbed trajectory (pink) to diverge exponentially from the unperturbed trajectory (in black). LSS chooses a perturbed trajectory with a different initial condition (in blue) that does not diverge from the unperturbed trajectory.

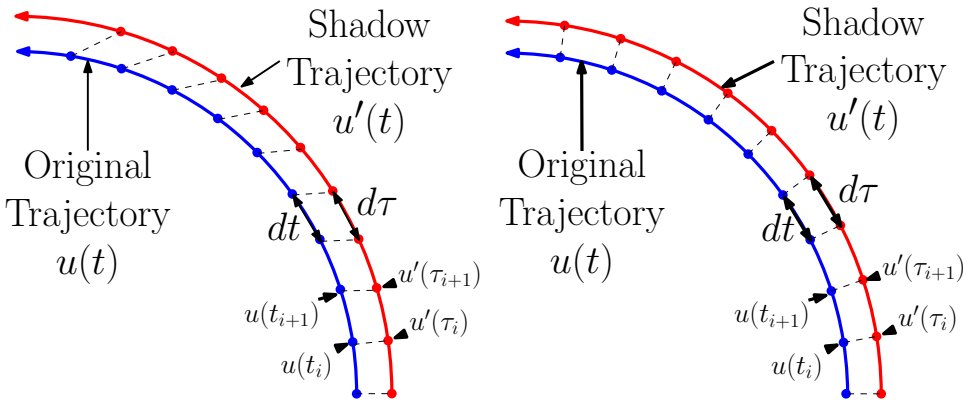


Figure 5-3: LEFT: Original and shadow phase space trajectories without any time transformation ($d\tau/dt = 1$). RIGHT: Original and shadow phase space trajectories with a time transformation $d\tau/dt = 1 + \eta$ that minimizes the distance between the two trajectories in phase space for all time.

$f(u_r)\| < \varepsilon$, $0 \leq t \leq T$, *there exists a true solution* u_s *and a time transformation* $\tau(t)$, *such that* $\|u_s(\tau(t)) - u_r(t)\| < \delta$, $|1 - d\tau/dt| < \delta$ *and* $du_s/d\tau - f(u_s) = 0$, $0 \leq \tau \leq \mathcal{T}$.

Note that $\|\cdot\|$ *refers to distance in phase space*

Therefore, relaxing the initial condition allows us to find the shadow trajectory $u_s(\tau)$. The key assumption of the shadowing lemma is that the attractor associated with the system of interest is *hyperbolic*. The key property of hyperbolic attractors for the shadowing lemma is that tangent space can be decomposed into stable, neutrally stable and unstable components everywhere on the attractor manifold [8]. Another way to state this property is that the Lyapunov covariant vectors make up a basis for phase space at all points on the attractor. Although this not the case for many attractors, including the Lorenz attractor, there is a *Chaotic Hypothesis* which states that many high-dimensional chaotic systems will behave as if they were hyperbolic [6]. For example, since the single point on the Lorenz attractor that is not hyperbolic is the unstable fixed point at the origin, most phase space trajectories do not pass through it and the shadowing lemma holds.

The time transformation alluded to in the shadowing lemma is required to deal with the zero (neutrally stable) Lyapunov exponent, whose covariant vector is simply $f(u)$. The need for this transformation can be seen by considering figure 5-3.

The time transformation, referred to as "time dilation" in this paper and other

LSS literature, is required to keep a phase space trajectory and its shadow trajectory close (in phase space) for all time.

5.2 Computing the Shadow Trajectory

Although LSS could be implemented by computing Lyapunov exponents and covariant vectors (as in [24]), this is not necessary. To find the shadow trajectory, an optimization problem is solved, where the objective function is the L2 norm of the tangent solution [26]. That is, for some system of equations $\frac{du}{dt} = f(u, \xi)$, the tangent equations are solved, where $v = \frac{\partial u}{\partial \xi}$:

$$\min_{v,\eta} \frac{1}{2} \int_0^T v^2 + \alpha^2 \eta^2 dt, \quad s.t. \quad \frac{dv}{dt} = \frac{\partial f}{\partial u} v + \frac{\partial f}{\partial \xi} + \eta f \,, \quad 0 < t < T, \tag{5.1}$$

where η is the time dilation term, corresponding to the time transformation from the shadowing lemma discussed in the previous section.

This optimization problem is a linearly constrained least-squares problem, with the following KKT equations, derived using calculus of variations (see appendix B.1):

$$\frac{\partial w}{\partial t} = -\frac{\partial f}{\partial u}^* w - v \quad w(0) = w(T) = 0 \tag{5.2}$$

$$\alpha^2 \eta = -\langle f, w \rangle \tag{5.3}$$

$$\frac{dv}{dt} = \frac{\partial f}{\partial u} v + \frac{\partial f}{\partial s} + \eta f \tag{5.4}$$

Equations (5.2), (5.3) and (5.4) can be combined to form a single second order equation for the Lagrange multiplier w, as in [18]:

$$-\frac{d^2 w}{dt^2} - \left(\frac{d}{dt} \frac{\partial f}{\partial u}^* - \frac{\partial f}{\partial u} \frac{d}{dt} \right) w + \left(\frac{\partial f}{\partial u} \frac{\partial f}{\partial u}^* + \frac{1}{\alpha^2} f f^* \right) w = \frac{\partial f}{\partial \xi}, \quad w(0) = w(T) = 0 \tag{5.5}$$

Equation (5.5) shows that the LSS method has changed the tangent equation from an initial value problem to a boundary value problem. The LSS solution can be used to compute gradients for some quantity of interest J (i.e. Drag):

$$\frac{\partial \bar{J}}{\partial \xi} = \overline{\left\langle \frac{\partial J}{\partial u}, v \right\rangle} + \overline{\eta J} - \overline{\eta}\,\overline{J}$$

Where $\overline{x} \equiv \frac{1}{T}\int_0^T x\ dt$. See appendix B.2 for a derivation of this.

5.3 Solving the KKT system numerically

Equations (5.2), (5.3) and (5.4) are discretized using finite differences and combined to form the following symmetric system:

$$\begin{pmatrix}
I & & & & & & F_0^T & & & & & & v_0 & & 0 \\
& I & & & & & G_1^T & F_1^T & & & & & v_1 & & 0 \\
& & \ddots & & & & & G_2^T & \ddots & & & & \vdots & & \vdots \\
& & & I & & & & & \ddots & F_{m-1}^T & & & \vdots & & \vdots \\
& & & & I & & & & & G_m^T & & & v_m & & 0 \\
\hline
& & & & & & \alpha^2 & & & & f_1^T & & \eta_1 & & 0 \\
& & & & & & & \alpha^2 & & & & f_2^T & \eta_2 & = - & \\
& & & & & & & & \ddots & & & \ddots & \vdots & & \vdots \\
& & & & & & & & & \alpha^2 & & & \eta_m & & 0 \\
\hline
F_0 & G_1 & & & & & f_1 & & & & & & w_1 & & b_1 \\
& F_1 & G_2 & & & & & f_2 & & & & & w_2 & & b_2 \\
& & \ddots & \ddots & & & & & \ddots & & & & \vdots & & \vdots \\
& & & & F_{m-1} & G_m & & & & f_m & & & w_m & & b_m
\end{pmatrix}$$

$$F_i = \frac{I}{\Delta t} + \frac{1}{2}\frac{\partial f}{\partial u}(u_i, \xi), \quad G_i = -\frac{I}{\Delta t} + \frac{1}{2}\frac{\partial f}{\partial u}(u_i, \xi),$$
$$b_i = \frac{1}{2}\left(\frac{\partial f}{\partial \xi}(u_i, \xi) + \frac{\partial f}{\partial \xi}(u_{i+1}, \xi)\right), \quad f_i = \frac{1}{2}\left(f(u_i) + f(u_{i+1})\right), \quad i = 0, ..., m$$

This KKT system is a block matrix system, where each block is n by n, where n is the number of states (i.e. the number of nodes in a CFD simulation). w_i and v_i

are length n vectors, and η_i is a scalar. The blocks highlighted in red correspond to equation (5.2), white to equation (5.3), and yellow to (5.4).

Note that as the system is symmetric, the adjoint is computed simply by changing the right hand side, allowing many gradients to be computed simultaneously.

The KKT system is quite large, with $2mn+n+1$ by $2mn+n+1$ elements for m time steps. For a discretization with a stencil of five elements, the matrix would have approximately $23mn$ non-zero elements. Consider a CFD simulation with 1×10^5 nodes ($n = 1 \times 10^5$) and 16000 time steps. For this simulation, the KKT matrix would be 3.2×10^9 by 3.2×10^9 with 3.7×10^{10} non-zero elements. Therefore, to use the LSS method on CFD problems, a method is needed to solve the KKT system without forming the entire matrix.

5.4 LSS Examples

LSS has been successfully applied to both Ordinary Differential Equations (ODEs) and a Partial Differential Equation (PDE) [26]. Results for the Lorenz equations (an ODE system) and the Kuramoto-Sivashinsky (KS) equation (a PDE) are presented in this paper.

5.4.1 The Lorenz Equations

The Lorenz equations were solved forward in time using a 4th order Runge-Kutta time stepping scheme. LSS was conducted by forming and inverting the KKT matrix shown in section 5.3. Solutions were integrated from a random initial condition and run for 100 time units before LSS was applied, to ensure that the portion of the solution being used was on the attractor manifold.

Figure 5-4 shows an approximate shadow trajectory for the Lorenz equations. Note how the trajectories stay close for all times, as governed by the shadowing lemma. The shadow trajectory allows the computation of accurate sensitivities for relatively short integration times as shown in table 5.1. Figure 5-5 shows gradients computed for a number of r values and two integration time lengths.

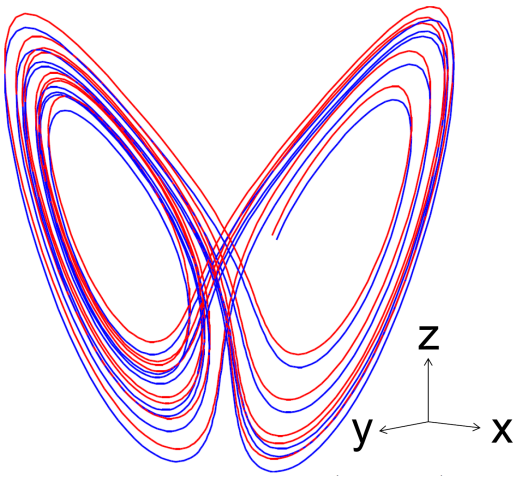


Figure 5-4: Lorenz equation phase space trajectory ($u(t)$) for $r = 28$ (blue) and a corresponding approximate shadow trajectory ($u(t) + v(t)$) (red). Integration time was $T = 20$ in dimensionless time units.

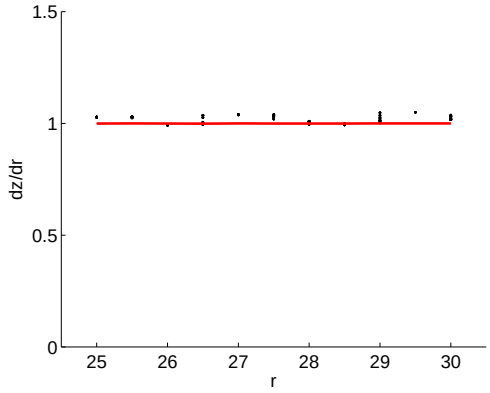


Figure 5-5: Gradient of long-time averaged z with respect to the parameter r. Gradients computed with trajectory length $T = 20$ are shown as black diamonds. Those computed using $T = 1000$ are shown as a red line.

Method	Int. Time	$d\bar{z}/ds$	$d\bar{z}/dr$	$d\bar{z}/db$
LR	1000000	0.16 ± 0.02	$1.01 \pm 0.04^*$	-1.68 ± 0.15
PDA	1000	0.1782 (11%)	0.9369 (7%)	-1.8374 (9%)
LSS	20	0.1558 (2%)	1.0094 ($< 1\%$)	-1.6043 (5%)

Table 5.1: Comparison of sensitivities computed using linear regression with 10 samples [24] (LR), the probability density adjoint (PDA) method with a 256 by 512 grid for the attractor discretization, and LSS.

5.4.2 The Kuramoto-Sivashinsky Equation

The KS equation is a 4th order, chaotic PDE, that can be used to model a number of physical phenomena including turbulence in the Belouzov-Zabotinskii reaction and thermal diffusive instabilities in laminar flame fronts [9]:

$$\frac{\partial u}{\partial t} = -(u + c)\frac{\partial u}{\partial x} - \frac{\partial^2 u}{\partial x^2} - \frac{\partial^4 u}{\partial x^4}$$

The c term is added to make the system ergodic [18]. In addition, the KS equation

is made ergodic by using the boundary conditions:

$$u\Big|_{x=0,100} = \frac{\partial u}{\partial x}\Big|_{x=0,100} = 0$$

and $u(x,0)$ is randomly chosen at each node x_i from $u \in [-0.5, 0.5]$. The objective function was chosen to be u averaged over both space and time:

$$J = \frac{1}{128T}\int_0^T \int_0^1 128u(x,t)\ dx\ dt$$

The system was discretized using a 2nd order finite difference scheme in space and solved forward in time using a 4th order Runge-Kutta time stepping scheme. The system was integrated for 300 time units before LSS was applied to ensure the solution was on the attractor, as was done for the Lorenz equations.

Even if relatively short integration times are used, LSS computes accurate gradients of time and space averaged u (or $\overline{J}$), as shown by how closely the approximate slope in figure 5-6 is matched by the LSS gradients in figure 5-7.

Finally, figure 5-8 shows an approximate shadow trajectory for a KS equation solution with $c = -0.5$. Note that although the values of u are different, the oscillatory structures in space and time are similar for the solution and its shadow. This means that a "shadow trajectory" in the context of fluid flows would be a flow field corresponding to some altered parameter or boundary condition with similar flow structures, but different velocity, pressure and temperature fields.

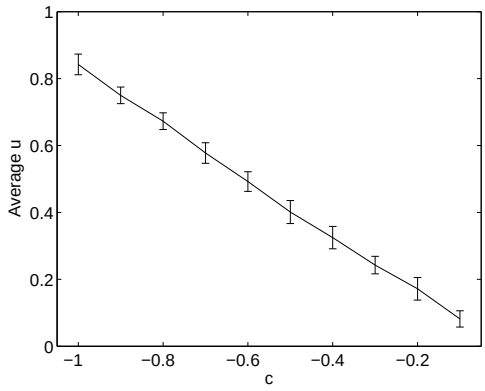


Figure 5-6: Time and space averaged $u(x,t)$ versus the parameter c. The error bars show the standard deviation computed from 20 solutions from random initial conditions for $t = 1024$ time units. The slope of the objective function is approximately -0.842.

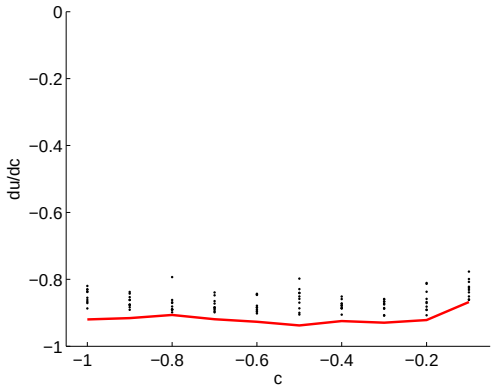


Figure 5-7: Gradient of time and space averaged u with respect to the parameter c. Gradients computed with trajectory length $T = 100$ are shown as black diamonds. Those computed using $T = 1000$ are shown as a red line.

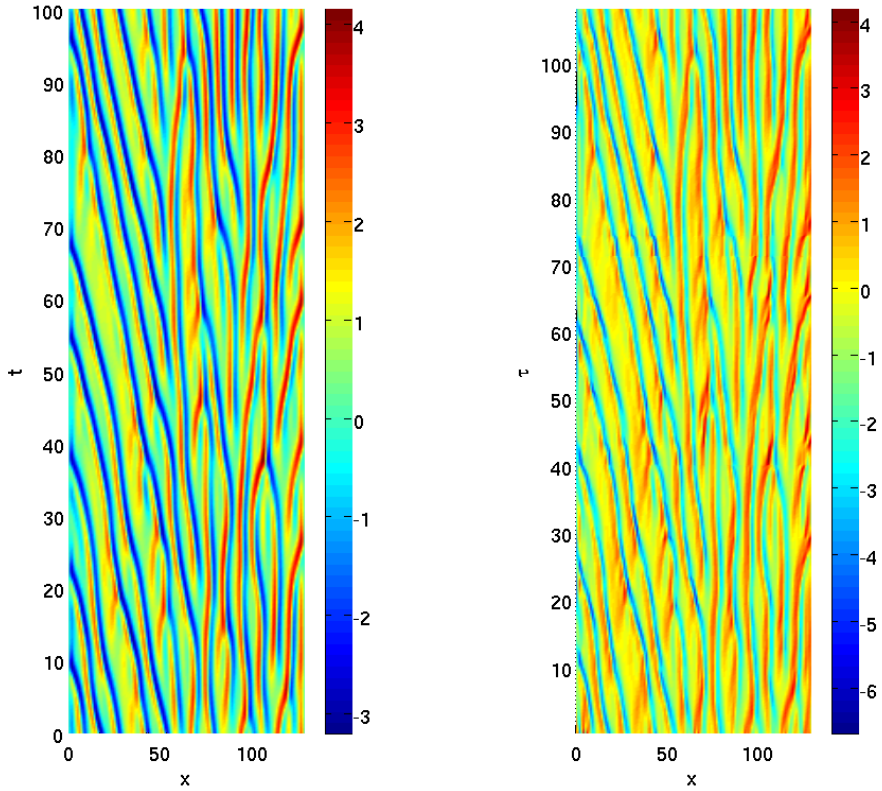


Figure 5-8: LEFT: KS equation solution $(u(x,t))$ for $c = -0.5$. RIGHT: Corresponding approximate shadow trajectory $(u(x,t)+v(x,t))$. Integration time was $T = 100$ in dimensionless time units. Note the transformed time scale τ for the shadow trajectory.

Chapter 6

Multigrid-in-time for Least Squares Sensitivity Analysis

When choosing a method to solve LSS numerically, we consider equation (5.5). As this equation is elliptic in time, a multigrid-in-time scheme is attractive because of its fast convergence relative to other iterative methods for many elliptic PDEs [23].

Several multigrid schemes were used to solve the LSS KKT system, with vastly varying convergence rates and computational cost. The first scheme considered, referred to as "classic" multigrid, is based off of the geometric multigrid scheme outlined by Briggs et al. [23]. This scheme leads to very slow convergence of the KKT system's residual. The second scheme considered was cyclic reduction. This scheme converges in one cycle if the system is solved exactly on the coarsest grid. However, the implementation of this scheme would use too much memory or require too many floating point operations to be viable for solving large scale systems with the computational resources that are currently available to most engineers and scientists. Finally, it was found that using a scheme with a Krylov subspace solver as a smoother and higher order averaging between the KKT systems on the fine and coarse grids led to textbook multigrid convergence rates. This was found to be the case for schemes which satisfied the variational condition and those that did not, although schemes that did not satisfy the variational condition converged slightly slower than those that did.

This chapter is comprised of a detailed discussion of the above results. Each

section will outline each scheme. Also, each section will present the performance of each scheme when used to apply the LSS method to the Lorenz equations. Finally, the implications of these results for the computational efficiency of each scheme will be discussed.

6.1 Solving the KKT System numerically

Rather than directly discretizing equation (5.5), we solve the Schur complement of the KKT system shown in section 5.3. The KKT system can be written as:

$$\begin{pmatrix} I & 0 & B^T \\ 0 & \alpha^2 I & C^T \\ B & C & 0 \end{pmatrix} \begin{pmatrix} \underline{v} \\ \underline{\eta} \\ \underline{w} \end{pmatrix} = - \begin{pmatrix} 0 \\ 0 \\ \underline{b} \end{pmatrix}$$

where B is a $mn \times (m+1)n$ matrix and C is a $mn \times m$ matrix. Conducting block Gaussian elimination, the Schur complement is found to be:

$$\underbrace{(BB^T + \frac{1}{\alpha^2} CC^T)}_{A} \underline{w} = \underline{b}$$

Written in terms of the block matrices in section 5.3:

$$\underbrace{\begin{pmatrix} F_0F_0^T + G_1G_1^T + \frac{1}{\alpha^2} f_1 f_1^T & G_1F_1^T & \\ F_1G_1^T & F_1F_1^T + G_2G_2^T + \frac{1}{\alpha^2} f_2 f_2^T & G_2F_2^T \\ \ddots & \ddots & \ddots \\ & F_{m-1}G_{m-1}^T & F_{m-1}F_{m-1}^T + G_mG_m^T + \frac{1}{\alpha^2} f_m f_m^T \end{pmatrix}}_{A}$$

From this form we see that the Schur complement is a $mn \times mn$ SPD and block tridiagonal matrix.

6.2 Classic Multigrid

The first multigrid scheme implemented was a simple geometric scheme, referred to in this paper as "classic multigrid". Restriction is achieved by eliminating every second equation and prolongation is carried out by linear interpolation. A "V" cycle was used, in which the system is coarsened until only one equation remains, then prolongated back to the full fine grid. Gauss-Seidel iterations were was used for relaxation, with 4-10 cycles before restriction and after prolongation on each level.

The scheme was found to be unstable if the under-relaxation factor is kept the same on the coarser grid. An empirical formula was used to reduce the under-relaxation factor as the time-step Δt increases in length on the coarse grid.

The method was tested on the Lorenz equations. The gradient of time-averaged z with respect to the parameters b, r, and s was computed:

$$\frac{d\bar{z}}{ds} = 0.122, \quad \frac{d\bar{z}}{dr} = 1.00, \quad \frac{d\bar{z}}{db} = -1.67. \tag{6.1}$$

The gradients with respect to r and b are within the error bounds of the finite difference obtained by [24] using linear regression, while the gradient with respect to s was slightly over-predicted.

The gradients converged within 20-30 cycles, as shown in figure 6-1. However, the residual of the system did not converge as quickly, as in figure 6-2. In fact, the residual was not observed to converge to machine precision until around 10^4 cycles. Subsequent analysis explored the causes of the slow convergence of the residual. Firstly, the method was analyzed by conducting Ideal Coarse Grid (ICG) iterations, as defined in [1]. The convergence of the ICG iterations was found to be satisfactory, suggesting that the relaxation scheme was working well and the convergence issues arose from the grid coarsening scheme. Also, convergence on individual grids was analyzed. It was found that the residual did not decrease in magnitude on the coarsest grids. Additionally, the method was carried out on the KS equation and the same convergence issues were observed.

Although this implementation of the multigrid in time method found correct gra-

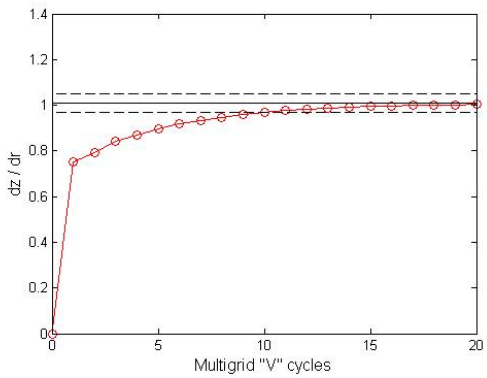


Figure 6-1: Convergence of the gradient of time-averaged z with respect to r, as computed using multigrid in time with 10 relaxation iterations before restriction and after prolongation on each level.

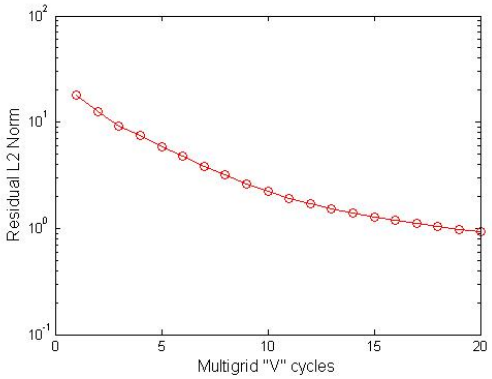


Figure 6-2: L2 norm of the residual while solving for the gradient of time-averaged z with respect to r using multigrid in time. Similar behavior was observed when computing other gradients

dient values, the residual convergence issues make the robustness of the method questionable. The multigrid analysis methods carried out indicate that different grid coarsening techniques need to be explored to ensure textbook multigrid convergence to a near-zero residual.

6.3 Cyclic Reduction

In classic multigrid, linear interpolation is used for prolongation, but it is not the best method when the coefficients of the equation being solved are highly oscillatory or discontinuous [4]. If this is the case, the coarse grid correction is a poor approximation for the low order error, and textbook multigrid convergence is not achieved. In this case, interpolation to some fine grid point with index i should be carried out by solving (5.5) with $\frac{\partial f}{\partial \xi} = 0$ between the two nearest coarse grid points, $i-1$ and $i+1$, with the boundary conditions $w(t_{i-1}) = w_{i-1}$ and $w(t_{i+1}) = w_{i+1}$. This interpolation scheme can be proved to lead to a multigrid scheme that converges independently of grid size for 1D problems [4]. Furthermore, it can be shown that this scheme is equivalent to cyclic reduction [4]. Cyclic reduction is carried out as follows: defining the lower, main and upper diagonal blocks as L_i, D_i and U_i, elimination is conducted as follows for three given rows:

$$L_{i-1}w_{i-2} + D_{i-1}w_{i-1} + U_{i-1}w_i = b_{i-1} \tag{6.2}$$

$$L_i w_{i-1} + D_i w_i + U_i w_{i+1} = b_i \tag{6.3}$$

$$L_{i+1}w_i + D_{i+1}w_{i+1} + U_{i+1}w_{i+2} = b_{i+1} \tag{6.4}$$

Equations (6.2) and (6.4) give expressions for w_{i-1} and w_{i+1}, which can be substituted into equation (6.3):

$$L_I w_{i-2} + D_I w_i + U_I w_{i+2} = b_I$$

Where:

$$\begin{aligned} L_I &= -L_i D_{i-1}^{-1} L_{i-1} \\ D_I &= -L_i D_{i-1}^{-1} U_{i-1} + D_i - U_i D_{i+1}^{-1} L_{i+1} \\ U_I &= -U_i D_{i+1}^{-1} U_{i+1} \\ f_I &= -L_i D_{i-1}^{-1} f_{i-1} + f_i - U_i D_{i+1}^{-1} f_{i+1} \\ b_I &= -L_i D_{i-1}^{-1} b_{i-1} + b_i - U_i D_{i+1}^{-1} b_{i+1} \end{aligned} \tag{6.5}$$

If the system is solved exactly on the coarsest grid, cyclic reduction will converge in one cycle. Also, the algorithm can be implemented in parallel, as each coarse grid equation only depends on three adjacent fine grid equations.

Although equation (6.5) involves inverting the main diagonal matrices D, which contains products of Jacobians $\frac{\partial f}{\partial u}$, in practice these inversions do not need to be carried out (see appendix B.3.1). However, the operation count of this scheme does not scale very well with the size of the KKT system. It can be shown (see appendix B.3.2) that the number of floating point operations, N, required for matrix multiplication with the coarse grid matrix scales approximately as:

$$N \sim \mathcal{O}(2p(2q)^l)$$

Where p is the number of operations required to multiply a vector by an F_i or G_i matrix, q is the number of iterations needed by an iterative solver to solve any system

involving D_i^{-1}, and l is the number of levels, assuming the coarsest grid is $n \times n$ (one time step). This is very large relative to the approximate operation count for a single Jacobi iteration on the LSS KKT system:

$$N \sim \mathcal{O}(2^{(l+2)}p)$$

To summarize, the advantages of cyclic reduction are in its memory efficiency and potential for parallel implementation, not its operation count.

6.4 Higher Order Averaging/Krylov Subspace Scheme

Taking into account the performance of classic multigrid and cyclic reduction, a new multigrid scheme was designed for solving the KKT system associated with LSS. The poor performance of classic multigrid and the excellent performance of cyclic reduction indicate that the accuracy of the coarse grid correction has a large effect on the convergence rate of the scheme. To obtain a more accurate coarse grid correction, higher order averaging was used to coarsen the KKT system. Higher order averaging ensures that the coarse grid non-linear solution $u(t)$ from which the KKT system is constructed is smooth, which leads to better multigrid performance [4].

Higher order averaging schemes are formed as follows: consider a two point (first order) average of some value x at time step i:

$$\overline{x}_i = \frac{1}{2}x_{i-1/2} + \frac{1}{2}x_{i+1/2}$$

A first order scheme is formed by setting $x_{i-1/2} = x_{i-1}$ and $x_{i+1/2} = x_{i+1}$. For a second order scheme, set $x_{i-1/2} = \frac{1}{2}x_{i-1} + \frac{1}{2}x_i$ and $x_{i+1/2} = \frac{1}{2}x_i + \frac{1}{2}x_{i+1}$:

$$\overline{x}_i = \frac{1}{4}x_{i-1} + \frac{1}{2}x_i + \frac{1}{4}x_{i+1}$$

The third order scheme can be derived by substituting first order averages into the second order scheme, and so on:

$$\overline{x}_{i+1/2} = \frac{1}{8}x_{i-1} + \frac{3}{8}x_i + \frac{3}{8}x_{i+1} + \frac{1}{8}x_{i+2} \quad \text{3nd Order}$$

$$\overline{x}_i = \frac{1}{16}x_{i-2} + \frac{1}{4}x_{i-1} + \frac{3}{8}x_i + \frac{1}{4}x_{i+1} + \frac{1}{16}x_{i+2} \quad \text{4th Order}$$

$$\overline{x}_{i+1/2} = \frac{1}{32}x_{i-2} + \frac{5}{32}x_{i-1} + \frac{10}{32}x_i + \frac{10}{32}x_{i+1} + \frac{5}{32}x_{i+2} + \frac{1}{32}x_{i+3} \quad \text{5th Order}$$

Higher order averaging was applied to multigrid-in-time in two ways. *Matrix restriction multigrid* uses higher order averaging on the KKT matrix itself, using McCormick et al's variational conditions [15]. *Solution restriction multigrid* uses higher order averaging on the non-linear solution $u(t)$. The KKT system is formed on the coarse grid from the restricted solution $u(t)$.

In addition, it was observed that stationary iterative methods such as Block Gauss Seidel converge very slowly for the KKT system. Krylov subspace methods such as conjugate gradient and MINRES[1] were examined as smoothers as an alternative to the stationary methods used in classic multigrid. Other parameters of the KKT system and the multigrid solver were also examined and some were found to have a considerable effect on convergence rates, especially the parameter α^2 from equation (5.5), the weighting of the time dilation term in the minimization statement.

The following sections discuss matrix restriction multigrid and its application to LSS for the Lorenz equations, followed by a discussion of solution restriction multigrid. Finally, the cost of solution restriction multigrid is compared to MINRES.

6.4.1 Matrix restriction multigrid

Matrix restriction multigrid was designed to satisfy the variational conditions [15]:

$$I_h^{2h} = c^h (I_{2h}^h)^T, \qquad A^{2h} = I_h^{2h} A^h I_{2h}^h$$

Where A^h is the fine grid matrix, A^h is the coarse grid matrix, I_h^{2h} is the restriction matrix, I_{2h}^h is the prolongation matrix and c^h is some constant that could depend on

[1]These methods were chosen because the KKT system is Symmetric Positive Definite.

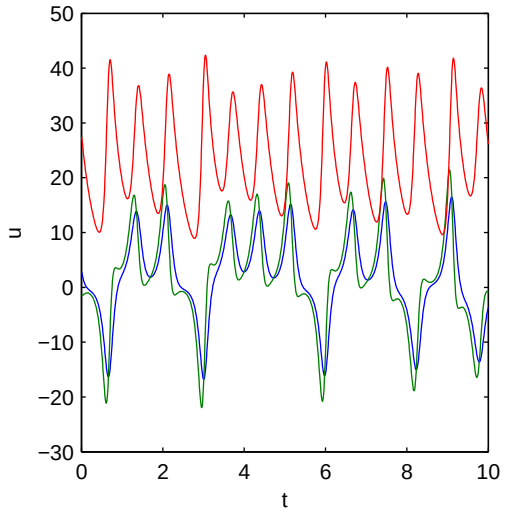


Figure 6-3: Lorenz equation solution; x,y,z are blue, green and red respectively.

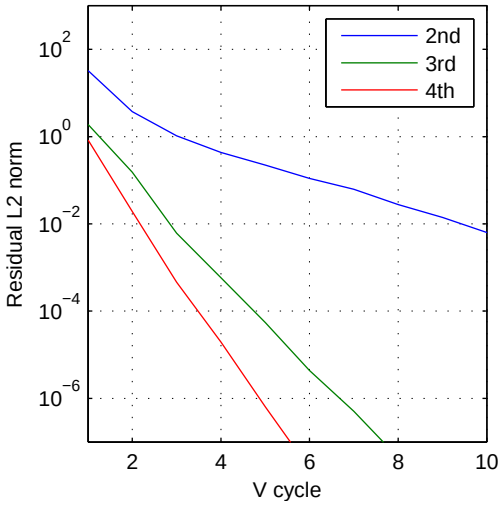


Figure 6-4: Convergence of matrix restriction multigrid with $dt_f = 0.01$ for different orders of averaging.

the grid size.

Satisfying the variational condition ensures that the error of the solution will decrease monotonically, assuming the smoother decreases or does not change the magnitude of the error on all grids [15].

Matrix restriction multigrid has been demonstrated on LSS for the Lorenz equations, in particular the solution shown in figure 6-3. Unless otherwise stated, the results presented correspond to a multigrid scheme with conjugate gradient (CG) smoothing with $\nu_1 = 30$ presmoothing iterations, $\nu_2 = 30$ postsmoothing iterations, 4th order averaging, $dt = dt_f = 0.0012$ on the finest grid and $dt = 0.08$ on the coarsest grid.

Since the typical Lorenz equation solution in figure 6-3 is oscillatory, a higher order averaging scheme can be used to improve the coarse grid correction by ensuring that the components of the block matrices (which depend on $u(t)$) that make up KKT matrix vary smoothly even on very coarse grids. Because of this, the use of higher order averaging drastically improves the convergence rate of multigrid-in-time, as shown by figure 6-4.

Block Gauss-Seidel and other stationary solvers that are used for smoothing in

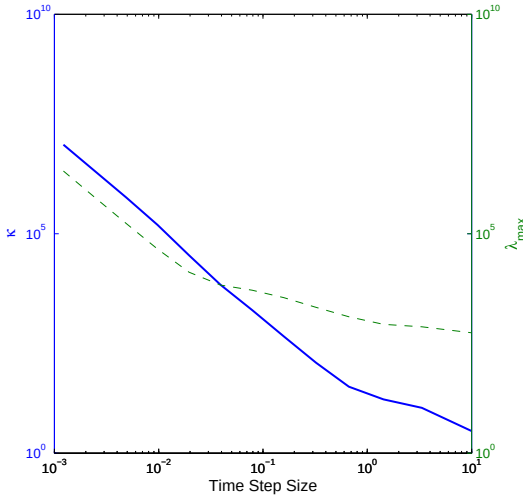


Figure 6-5: Condition Number κ (solid line) and maximum eigenvalue λ_{max} (dashed line) versus time step dt for coarsened grids corresponding to a fine grid with $dt = 0.001$ for the LSS system associated with the Lorenz equations

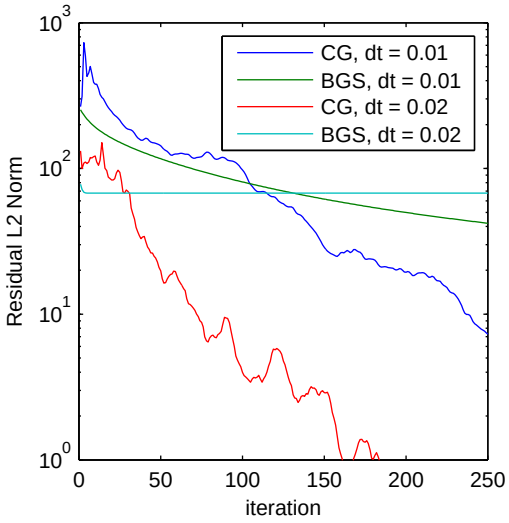


Figure 6-6: Convergence of Block Gauss-Seidel and Conjugate Gradient on the fine grid (dt=0.01,$\alpha^2 = 40$) and a coarsened grid (dt=0.02,$\alpha^2 = 40$).

classic multigrid schemes were found to converge very slowly. Also, the convergence of these solvers worsen as the grid is coarsened, as shown in figure 6-6. This is because the largest eigenvalue of the KKT system, which can be shown to set the convergence rate of a stationary solver [7], decreases at a slower rate as the KKT system is coarsened, as shown in figure 6-5. This is in contrast to the behavior of the finite difference (or finite element) matrix of the Poisson equation, whose largest eigenvalue decreases exponentially as the grid is coarsened, resulting in much faster convergence of Jacobi or Gauss-Seidel solvers on the coarser grids [7].

To accelerate convergence of multigrid-in-time, conjugate gradient (CG) is used as a smoother. With a CG smoother, the rate of convergence is bounded by $((\kappa - 1)/\kappa)^N$, where κ is the condition number of the KKT matrix A_h and N is the number of smoothing iterations [2]. The condition number decreases quickly as the grid is coarsened (see figure 6-5), leading to faster CG convergence on coarser grids, as seen in figure 6-6.

A matrix restriction scheme with a CG smoother and 4th order averaging has

been observed to converge independently of the number of fine grid points, as shown in figure 6-7.

In addition, a number of tuning parameters were observed to affect this convergence rate. The parameter α^2 from equation (5.5), the weighting of the time dilation term in the minimization statement, has a strong effect on multigrid convergence, as shown in figure 6-8. There is an optimal α^2, found to be equal to around 40 in the case of our sample problem. α^2 has an effect on convergence because it affects the condition number and the eigenvalue spectrum of A on all grids.

We define the coarsening threshold dt_c is defined as the time step size below which multigrid is not called recursively. Figure 6-9 shows that there is an optimal value for dt_c. Below this, the coarse grid correction actually slows convergence in some cases, because it is a poor approximation for low order errors.

6.4.2 Solution restriction multigrid

Although matrix restriction multigrid performs very well, using the variational condition to form the coarse grid matrix makes the matrix restriction multigrid more expensive than classic multigrid. To conduct matrix multiplication on a coarse grid, the coarse grid solution is prolongated to the fine grid, multiplied by the fine grid matrix A_h and then restricted to the coarse grid. This means that conducting smoothing on the coarse grid(s) is slightly more expensive than smoothing on the fine grid.

By restricting the non-linear solution $u(t)$ instead of the matrix A_h, solution restriction multigrid results in reduced smoothing costs on the coarse grid, because the KKT system formed on the coarse grid from the restricted $u(t)$ is half the size of the fine grid system. As shown by figure 6-10, the solution restriction scheme also leads to convergence rates with little dependence on dt_f for a given dt_c for the Lorenz equations [2].

As observed for matrix restriction multigrid, the value of the parameter α^2 and the order of averaging both affect the convergence rate, as shown in figures 6-11 and

[2]Parameter values used for this plots in this section (unless otherwise stated): $dt_f = 0.004$ or $m = 4096$, $\nu_1 = \nu_2 = 30$, $dt_c = 0.2$, $\alpha^2 = 40$, MINRES smoothing, and 3rd order averaging

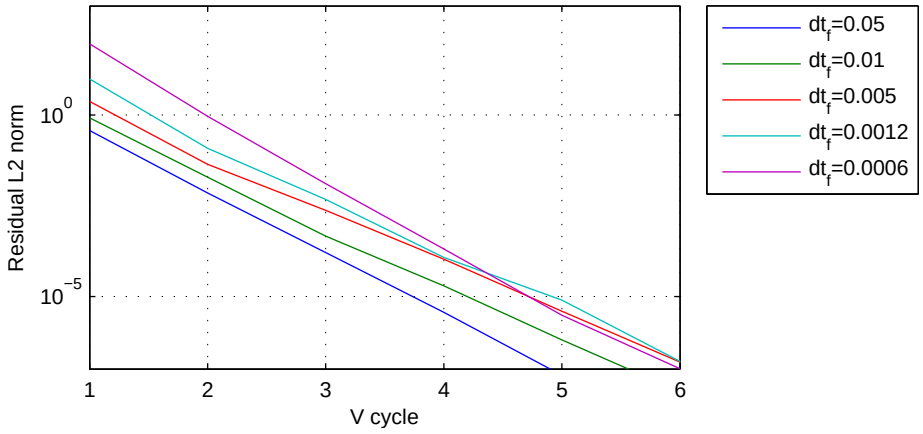


Figure 6-7: Convergence of matrix restriction multigrid for different fine grid time steps dt_f.

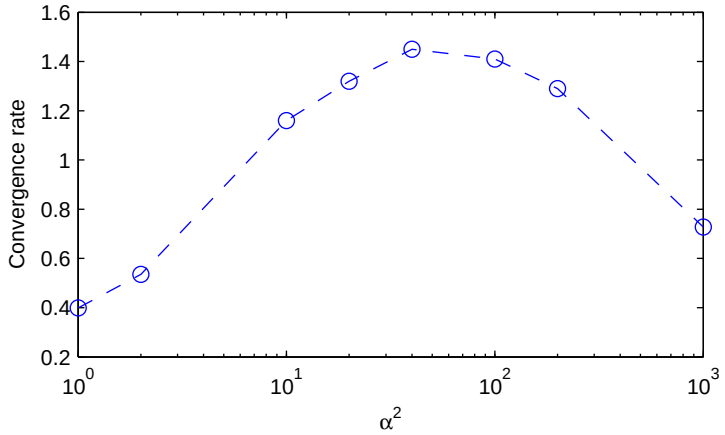


Figure 6-8: Convergence rate of matrix restriction multigrid versus α^2. The convergence rate is $-\gamma$, from the curve fit $\log_{10}\|r\|_{L2} = \gamma \log_{10} N_V + \log_{10} C$ of the residual L2 norm $\|r\|_{L2}$ versus the number of V-cycles N_V.

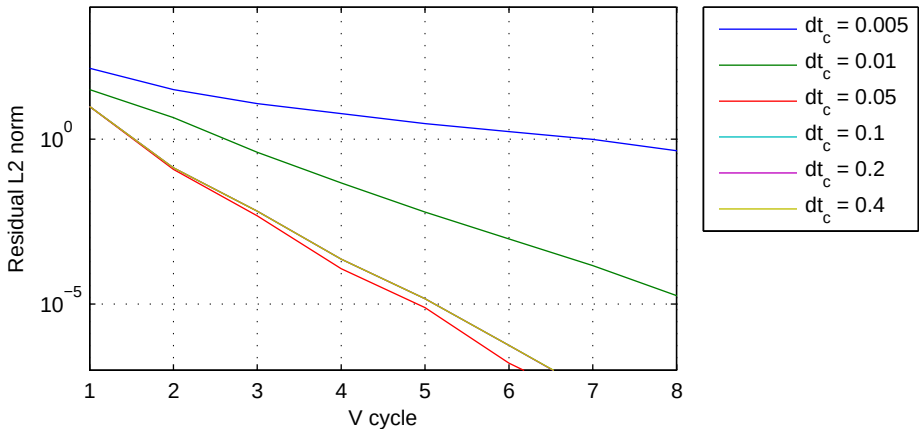


Figure 6-9: Convergence of matrix restriction multigrid for different coarsening thresholds dt_c.

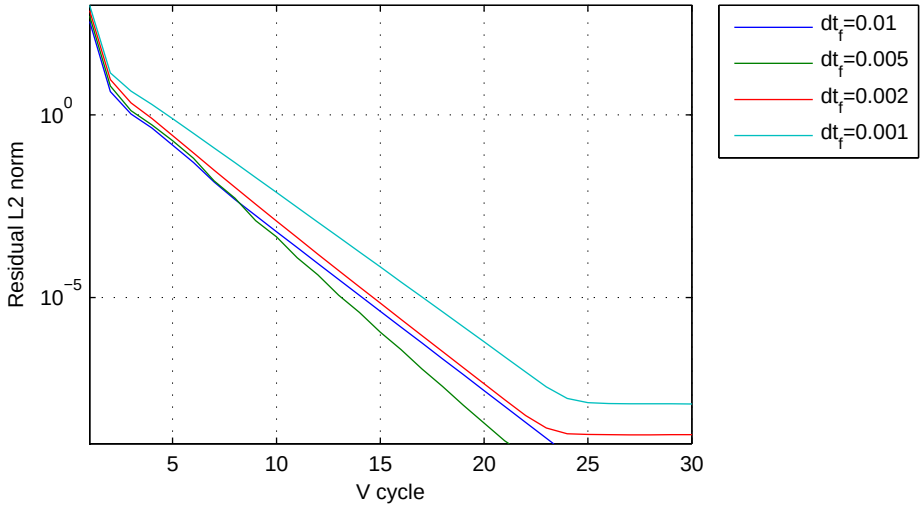


Figure 6-10: Convergence of solution restriction multigrid for different values of dt_f.

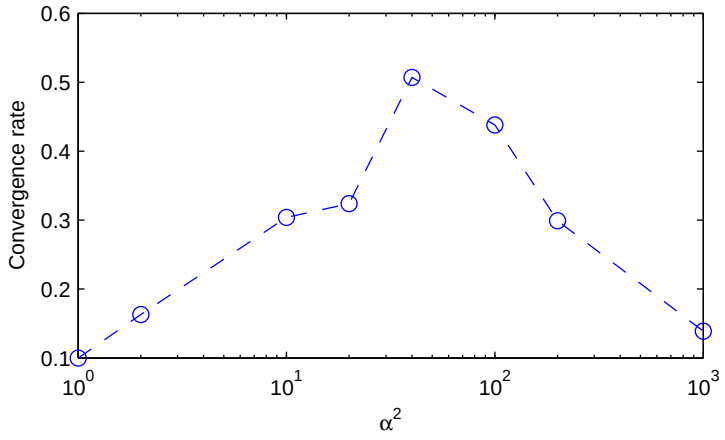


Figure 6-11: Convergence rate of solution restriction multigrid versus α^2. The convergence rate is $-\gamma$, from the curve fit $\log_{10}\|r\|_{L2} = \gamma \log_{10} N_V + \log_{10} C$ of the residual L2 norm $\|r\|_{L2}$ versus the number of V-cycles N_V.

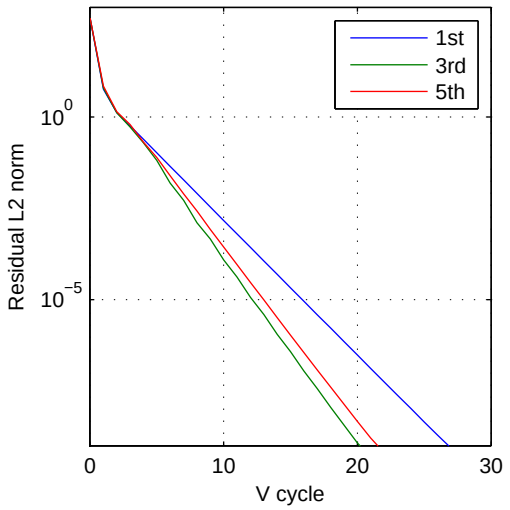


Figure 6-12: Convergence of solution restriction multigrid with 1st, 3rd and 5th order averaging.

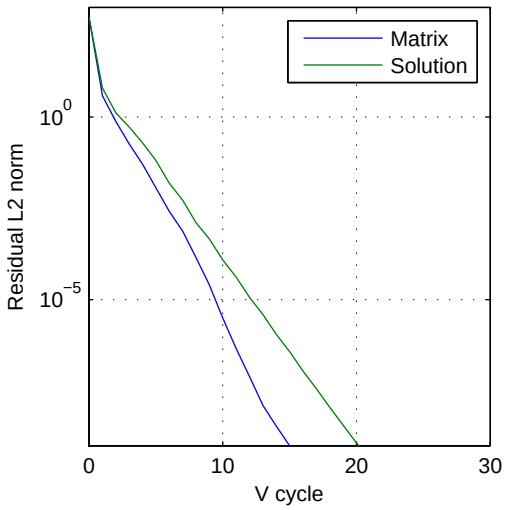


Figure 6-13: Convergence plots for matrix restriction multigrid with MINRES smoothing, $dt_c = 5$, and 4th order averaging and solution restriction multigrid.

6-12. However, higher order averaging is only beneficial to a certain degree, as the 5th order scheme leads to slower convergence than the 3rd order one (figure 6-12). This is because high order averaging could smooth $u(t)$ too much. If this is the case the course grid $u(t)$ is a poor approximation of the fine grid $u(t)$. Also, α^2 has a much greater effect than the averaging scheme on the convergence rate of solution restriction multigrid.

There is a slight trade-off for the lower costs of solution restriction multigrid. Figure 6-13 shows that solution restriction multigrid converges slightly slower than matrix restriction multigrid. However, this slower convergence does not outweigh the benefits of the lower cost of solution restriction multigrid.

The benefits of solution restriction multigrid can be seen by comparing its cost to that of using MINRES to solve the fine grid solution. Figure 6-14 shows that the solution of the KKT system converges after about 4700 iterations, and the gradient converges after about 1800 iterations. For a Krylov subspace method applied to a sparse matrix, the number of floating point operations for a single iteration, p_{MINRES}, is $p_{MINRES} \sim \mathcal{O}(mnN)$, where N is the number of previous iterations, m is the

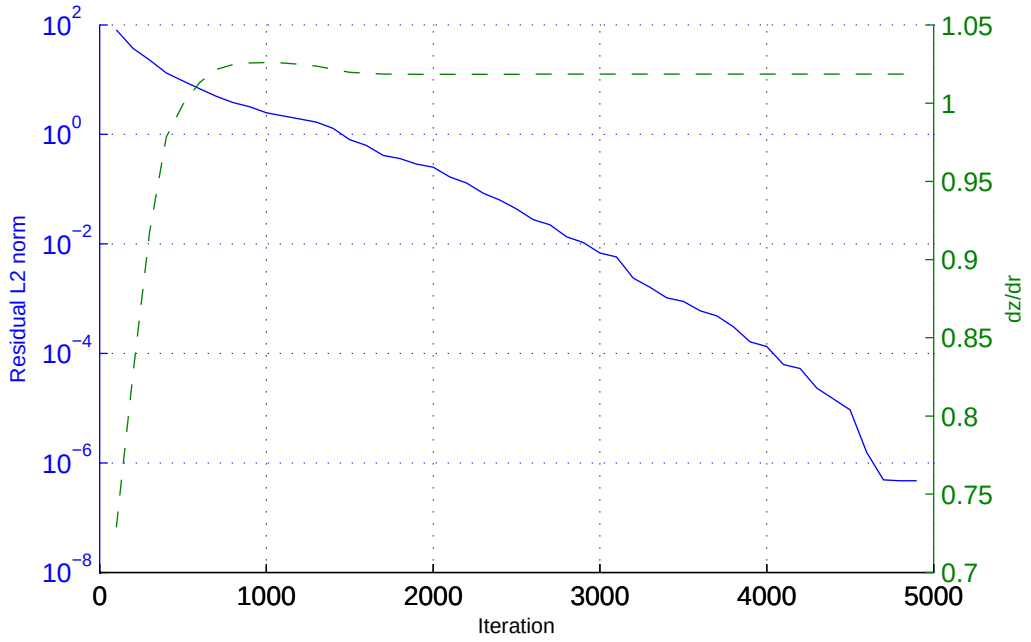


Figure 6-14: Convergence of MINRES for an LSS system for the Lorenz equations with $dt_f = 0.004$ and $\alpha^2 = 40$. The dashed line shows the gradient computed at a given iteration, which should be roughly 1.01 ± 0.04 [24]

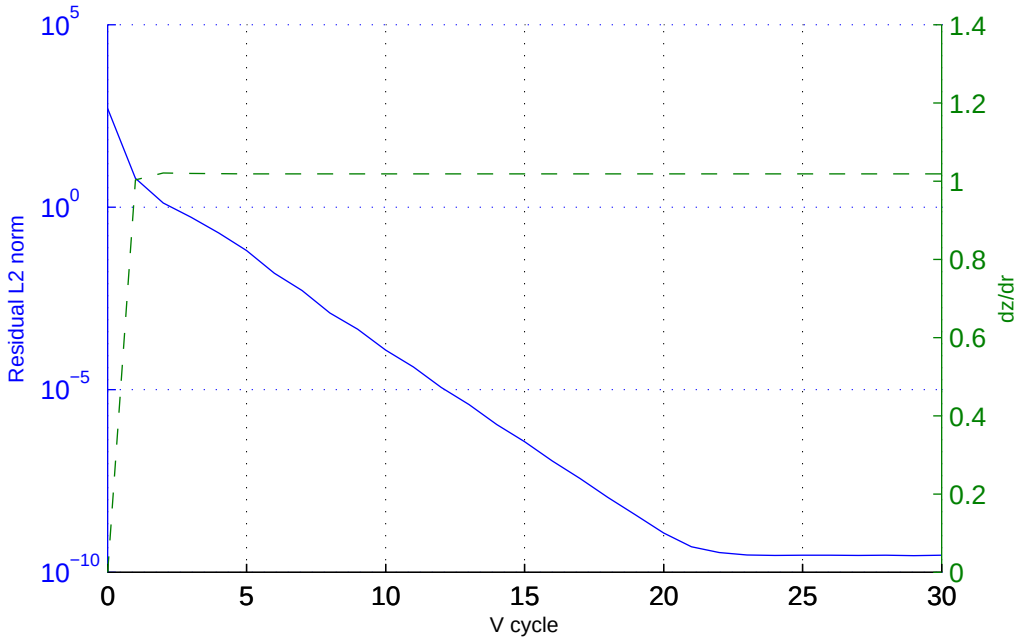


Figure 6-15: Convergence of solution restriction multigrid for an LSS system for the Lorenz equations. The dashed line shows the gradient computed at a given iteration, which should be roughly 1.01 ± 0.04 [24]

number of time steps and n is the number of dimensions of the dynamical system [7]. Therefore, for the solution in figure 6-14 the total number of operations, P_{MINRES}, is roughly

$$P_{MINRES} \sim 4700\mathcal{O}(mn)$$

For a fixed number of iterations $N = \nu_1 + \nu_2$, the cost of smoothing is halved for one coarsening as mn is halved when the system is coarsened. When a large number of grids are used for a V-cycle:

$$\begin{aligned} p_{MG} &\sim \mathcal{O}(mn(\nu_1+\nu_2)) + \frac{1}{2}\mathcal{O}(mn(\nu_1+\nu_2)) + \frac{1}{4}\mathcal{O}(mn(\nu_1+\nu_2)) + ... \\ &\approx 2\mathcal{O}(mn(\nu_1+\nu_2)) = 120\mathcal{O}(mn) \end{aligned}$$

Figure 6-15 shows that the solution of the KKT system converges after about 20 cycles with solution restriction multigrid, therefore:

$$P_{MG} \sim 2400\mathcal{O}(mn) \approx \frac{1}{2}P_{MINRES}$$

In addition to requiring only half of the floating point operations of MINRES, solution restriction multigrid computes the correct gradient after 2 cycles, which requires $240\mathcal{O}(mn)$ operations, an order of magnitude less than the roughly $1800\mathcal{O}(mn)$ operations required by MINRES.

To summarize, multigrid-in-time is found to work well when higher order averaging is used to construct the KKT system on coarse grids and a Krylov subspace method is used for smoothing. Matrix restriction multigrid converges quickly, but smoothing on the coarse grid requires more operations than smoothing on the fine grid . Solution restriction multigrid converges slightly slower than matrix restriction multigrid, but is less expensive. Because of its lower cost, solution restriction multigrid is currently the most promising numerical method for implementing LSS for large scale systems.

Chapter 7

Conclusions

In conclusion, traditional sensitivity analysis methods are unable to compute sensitivities of long-time averaged quantities in chaotic dynamical systems. Two new methods, the probability density adjoint method and the least squares sensitivity (LSS) method are able to compute these sensitivities if the system is ergodic.

The probability density adjoint method works well for 1D maps and low order ODEs, but it would be difficult to extend to the higher dimensional (10+) attractors associated with many ODEs and discretized PDEs of engineering interest. It does however provide an insight into sensitivities of low-dimensional chaotic dynamic systems. For instance, the highly oscillatory adjoint density solutions show how small perturbations which are relatively small distance apart in phase space can have radically different effects on long-time averaged quantities.

LSS is more readily extended to sensitivity analysis of higher dimensional ODEs and PDEs and has been shown to compute accurate gradients for the Lorenz equation and a modified version of the Kuramoto-Sivashinsky equation. To solve the large KKT system associated with LSS, several multigrid-in-time schemes have been investigated. Classic geometric multigrid with a Gauss-Seidel smoother was found to converge very slowly. Cyclic reduction converged in one cycle and can be run in parallel but it used a very large number of floating point operations to solve the system. A higher order averaging multigrid scheme with a Krylov subspace smoother was found to give textbook multigrid convergence. It was found that the parameter α^2, the weighting of

the time dilation term in equation (5.1), had a large effect on the rate of convergence of multigrid.

Future work will start with further investigation of solution restriction multigrid-in-time. In particular, a method to determine the α^2 value corresponding to the fastest rate of convergence needs to be found.

Once a robust, scalable numerical method for solving the KKT equation is found, LSS will be tested on chaotic fluid flows, such as air flow around the NACA 0012 airfoil presented in chapter 2. Eventually, LSS could be used to investigate more complicated flows such as flow around a lifting body, the fuel injection system in a jet engine or scramjet combustor, or a internal flow in a rocket engine.

Appendix A

Probability Density Adjoint Method

A.1 Probability density adjoint for 1D maps

A.1.1 Deriving the continuous density adjoint equation

To derive the adjoint equation, we define a function $v(x) = 1$ and an inner product:

$$\langle a, b \rangle = \int_0^1 a(x)b(x)dx$$

Consider a small perturbation to P. From equation (3.2) and conservation of probability mass:

$$\delta(P\rho_s) = \delta P\rho_s + P\delta\rho_s = \rho_s, \quad \langle v, \delta\rho_s \rangle = 0 \tag{A.1}$$

Define ϕ as the adjoint variable. Using integration by parts:

$$0 = \langle \phi, \delta P\rho_s + P\delta\rho_s - \rho_s \rangle = \langle P^*\phi - \phi, \delta\rho_s \rangle + \langle \phi, \delta P\rho_s \rangle \tag{A.2}$$

Combining equations (A.1) and (A.2) with equation (3.5) :

$$\begin{aligned}\delta\overline{J} &= \langle e, J\rangle - \langle P^*\phi - \phi, \delta\rho_s\rangle + \langle \phi, \delta P\rho_s\rangle - \eta\langle v, \delta\rho_s\rangle = 0 \\ &= \langle J - \eta v - P^*\phi + \phi, \delta\rho_s\rangle + \langle \phi, \delta P\rho_s\rangle\end{aligned}$$

For ϕ and η such that:

$$\langle J - \eta v - P^*\phi + \phi, \delta\rho_s\rangle = 0$$

Gradients can be computed as follows:

$$\delta J = \langle \phi, \delta P\rho_s\rangle \tag{A.3}$$

We derive an expression to compute $\delta P\rho_s$ in section 3.3 and A.1.2.

To find η:

$$J + \delta J = \langle J, \rho_s + \delta\rho_s\rangle$$

For equation (A.3) to be consistent with this:

$$0 = \langle \rho_s, J - \eta v - P^*\phi + \phi\rangle = \langle \rho_s, J\rangle - \eta\langle \rho_s, v\rangle - \langle \phi, P\rho_s - \rho_s\rangle$$

The second and third inner products on the right hand side are by definition 1 and 0 respectively, therefore:

$$\eta = \langle \rho_s, J\rangle = \overline{J}$$

Therefore, the adjoint equation is:

$$P^*\phi - \phi = \overline{J} - J$$

A.1.2 Derivation of the gradient equation

To find $\delta P\rho_s$, we first define $\delta\rho_0 = \delta P\rho_s$. From equation 3.2:

$$\rho_s + \delta\rho_0 = (P + \delta P)\rho_s$$

Using the coordinates defined in figure A-1, density is mapped by P as follows:

$$\int_0^y \rho_s \; ds = \int_0^{x_L} \rho_s \; ds + \int_{x_R}^1 \rho_s \; ds \tag{A.4}$$

Similarly, $P + \delta P$ maps density as this:

$$\int_0^y \rho_s + \delta\rho_0 \; ds = \int_0^{x_L} \rho_s \; ds + \int_{x_R}^1 \rho_s \; ds \tag{A.5}$$

Taking the difference of equations A.4 and A.5:

$$\int_0^y \delta\rho_0 \; ds = \int_{x_L}^{x'_L} \rho_s \; ds + \int_{x'_R}^{x_R} \rho_s \; ds$$

Assuming a small perturbation δP (and therefore a small δF):

$$\begin{aligned} \int_0^y \delta\rho_0 \; ds &= \rho_s(x_L)\delta x_L - \rho_s(x_R)\delta x_R \\ &= \rho_s(x_L)\delta F^{-1}(y)_L - \rho_s(x_R)\delta F^{-1}(y)_R \end{aligned} \tag{A.6}$$

From figure A-1, a local functional perturbation left of the peak moves $F^{-1}(y)$ by δx_L to the left. Also, note that $\delta F/\delta x_L$ is a first order approximation to the local slope, therefore $\delta F = -F'(F^{-1}(y))\delta x_L$ for small δF. A similar argument can be made to show that the same equation applies to the right of of the peak. Therefore equation 3.8 can be rewritten as:

$$\int_0^y \delta\rho_0 \; ds = \frac{\rho_s(x_L)}{F'(x_L)}\delta F(x_L) - \frac{\rho_s(x_R)}{F'(x_R)}\delta F(x_R)$$

Differentiating both sides with respect to y:

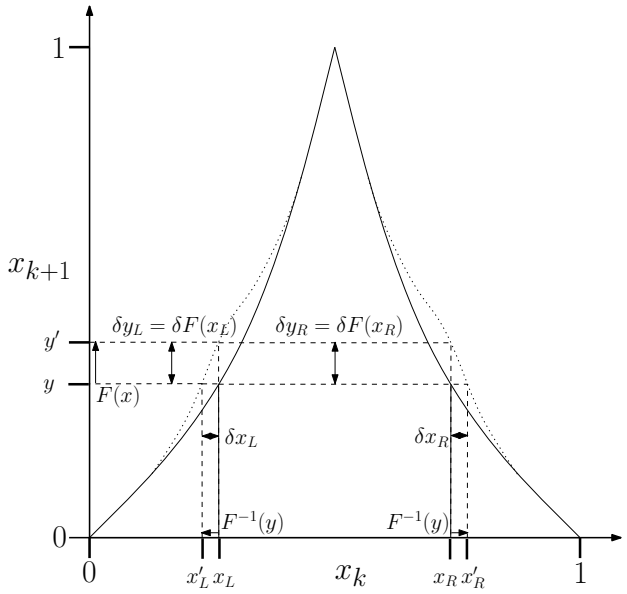


Figure A-1: The effect of a perturbation on the mapping function.

$$\delta\rho_0 = \delta P \rho_s = \frac{\partial}{\partial y}\left(\frac{\rho_s(x_L)}{F'(x_L)}\delta F(x_L) - \frac{\rho_s(x_R)}{F'(x_R)}\delta F(x_R)\right) \tag{A.7}$$

A.1.3 Deriving the discrete density adjoint equation

Recall that the first eigenvalue of the discrete operator P_n is not exactly one. Denoting λ as the first eigenvalue, which converges to one as $n \to \infty$:

$$P_n \underline{\rho}_s - \lambda \underline{\rho}_s = 0$$

Now consider the linearization:

$$\delta P_n \underline{\rho}_s - \delta\lambda \underline{\rho}_s + P_n \delta\underline{\rho}_s - \lambda\delta\underline{\rho}_s = 0, \quad -\underline{v}^T \delta\underline{\rho}_s = 0$$

Combine this with the discrete version of equation 3.5:

$$\delta\overline{J} = \frac{1}{n}[J^T \delta\underline{\rho}_s + \underline{\phi}^T(-\delta\lambda\underline{\rho}_s + P_n\delta\underline{\rho}_s - \lambda\delta\underline{\rho}_s) + \underline{\phi}^T \delta P_n \underline{\rho}_s - \eta v^T \delta\underline{\rho}_s]$$

Where η is the adjoint variable for the eigenvalue perturbation $\delta\lambda$. Rearrange to isolate $\delta\underline{\rho}_s$ and $\delta\lambda$:

$$\delta\overline{J} = \frac{1}{n}[\underline{\phi}^T \delta P_n \underline{\rho}_s + (J^T - \underline{\phi}^T P_n - \underline{\phi}^T \lambda - \eta \underline{v}^T)\delta\underline{\rho}_s + (-\underline{\phi}^T \underline{\rho}_s)\delta\lambda]$$

Equation (3.15) is obtained by eliminating any dependence of $\delta\overline{J}$ on $\delta\underline{\rho}_s$ and $\delta\lambda$. Also, from the continuous adjoint equation, it can be seen that $\eta = \overline{J}$:

$$\begin{aligned} P_n^T \underline{\phi} - \lambda\underline{\phi} - \underline{v}\overline{J} &= J \\ -\underline{\rho}_s^T \underline{\phi} &= 0 \end{aligned}$$

A.2 Probability density adjoint for continuous chaos

A.2.1 Deriving the continuous adjoint equation

First, linearize equation (4.1)

$$\vec{\nabla}_s \cdot (\delta\rho\vec{f} + \rho\delta\vec{f}) = 0 \tag{A.8}$$

As (A.8) is zero, it can be multiplied by some scalar variable ϕ and added to equation (4.6). By conservation of probability mass, a perturbation to ρ_s does not change the total probability:

$$\iint \delta\rho \; dlds = 0$$

Therefore, $\delta\rho$ can also be added to equation (4.6):

$$\delta\overline{J} = \iint \phi\vec{\nabla}_s \cdot (\delta\rho\vec{f} + \rho_s\delta\vec{f}) + J(\vec{x})\delta\rho + c\delta\rho \; dlds \tag{A.9}$$

Where c is some constant. Conducting integration by parts:

$$\begin{aligned}
\delta\overline{J} &= \iint \phi\nabla_s \cdot (\delta\rho\vec{f}) + J(\vec{x})\delta\rho + c\delta\rho + \phi\nabla_s \cdot (\rho_s\delta\vec{f})\ dlds \\
&= \iint -\delta\rho\vec{f} \cdot \nabla_s\phi + J(\vec{x})\delta\rho + c\delta\rho + \phi\nabla_s \cdot (\rho_s\delta\vec{f}))\ dlds \\
&= \iint (-\frac{\partial\phi}{\partial t} + J(\vec{x}) + c)\delta\rho + \phi\nabla_s \cdot (\rho_s\delta\vec{f}))\ dlds
\end{aligned}$$

In order to eliminate the dependence of $\delta\overline{J}$ on $\delta\rho$, the adjoint density equation is:

$$\frac{\partial\phi}{\partial t} = J(\vec{x}) + c \tag{A.10}$$

Multiplying both sides of (A.10) by ρ_s and integrating over the attractor surface shows that $c = -\overline{J}$:

$$\begin{aligned}
\iint \rho_s f(\vec{x})\nabla_s\phi\ dlds &= \iint \rho_s J(\vec{x}) + \rho_s c\ dlds \\
\iint \phi\nabla_s \cdot (\rho_s f(\vec{x}))\ dlds &= \iint \rho_s J(\vec{x}) + \rho_s c\ dlds \\
0 &= \iint \rho_s J(\vec{x})\, dlds + c\iint \rho_s\ dlds \\
c &= -\overline{J}
\end{aligned}$$

Therefore:

$$\frac{\partial\phi}{\partial t} = J(\vec{x}) - \overline{J}$$

If the above equation is satisfied, equation (A.9) reduces to

$$\delta\overline{J} = \iint \phi\,\vec{\nabla}_s \cdot (\rho_s\delta\vec{f})\ dlds$$

A.2.2 Deriving the discrete Adjoint Equation

Consider the eigenvalue equation for the Poincaré stationary density:

$$P_n \underline{\rho}_0 = \lambda \underline{\rho}_0$$

This can be modified using D, a diagonal matrix with $|\vec{f}(\vec{x}_0) \times \hat{s}_0| ds_0$ for the ith streamline along the main diagonal:

$$D P_n D^{-1} D \underline{\rho}_0 = \lambda D \underline{\rho}_0$$

Defining $D\underline{\rho}_0 = \underline{q}$ and $DP_nD^{-1} = A$:

$$A\underline{q} = \lambda \underline{q}$$

The adjoint is derived using A and $\underline{q}$ because perturbations to $\underline{q}$ correspond to density perturbations on the attractor surface. A perturbation $\delta\underline{q}$ can be written as follows:

$$\begin{aligned} \delta(\lambda \underline{q}) &= \delta(A\underline{q}) \\ \lambda \delta \underline{q} + \underline{q}\delta\lambda &= \delta A \underline{q} + A \delta \underline{q} \\ (\lambda I - A)\delta\underline{q} - \delta A \underline{q} + \underline{q}\delta\lambda &= 0 \end{aligned} \tag{A.11}$$

From equation (4.9), $\delta\overline{J}$ is related to a perturbation to $\underline{q}$ as follows

$$\delta \overline{J} = \underline{\mathcal{J}}^T \delta \underline{q} \tag{A.12}$$

where we define $\underline{\mathcal{J}}_i = \int_0^T J(t)dt$ for streamline i.
Also, it can be shown that:

$$\iint \delta\rho_s dl ds \quad \Rightarrow \quad \underline{v}^T D^{-1} \delta \underline{q} = 0 \tag{A.13}$$

Adding equation (A.12) to the product of equation (A.11) and the discrete density adjoint $\underline{\phi}_0$ as well as equation (A.13) and the adjoint eigenvalue η yields:

$$\delta\overline{J} = \underline{\mathcal{J}}^T\delta\underline{q} + \underline{\phi}_0^T((\lambda I - A)\delta\underline{q} + \eta(\underline{v}^T D^{-1}\delta\underline{q}) - \delta A\underline{q} + \underline{q}\delta\lambda)$$
$$\delta\overline{J} = (\underline{\phi}_0^T(\lambda I - A) + \eta\underline{v}^T D^{-1} + \underline{\mathcal{J}}^T)\delta\underline{q} + \underline{\phi}_0^T\underline{q}\delta\lambda - \underline{\phi}_0^T\delta A\underline{q}$$

To eliminate the dependence of $\delta\overline{J}$ on $\delta\underline{q}$ and $\delta\lambda$:

$$(A - I)^T\underline{\phi}_0 + \eta D^{-1}\underline{v} = \underline{\mathcal{J}}, \qquad \underline{q}^T\underline{\phi}_0$$

Therefore:

$$(DP_nD^{-1} - I)^T\underline{\phi}_0 + \eta D^{-1}\underline{v} = \underline{\mathcal{J}}, \qquad \underline{\rho}_s^T D\underline{\phi}_0$$

As in section 3.3, it can be shown that $\eta = -\overline{J}$, therefore:

$$\begin{bmatrix} (D^{-1}P^TD - \lambda I) & -D^{-1}\underline{v} \\ \underline{\rho}_s^T D & 0 \end{bmatrix} \begin{bmatrix} \underline{\phi}_0 \\ \overline{J} \end{bmatrix} = \begin{bmatrix} \underline{\mathcal{J}} \\ 0 \end{bmatrix}$$

A.2.3 Computing Attractor Surface Areas

Recall:

$$\rho(\vec{x})|\vec{f}(\vec{x})|ds = \rho_0|\vec{f_0} \times \hat{s}_0|ds_0$$

Where ds is the streamline width and the subscript 0 indicates values at the start of the streamline. ds_0 is set as the average distance from a streamline to its neighboring streamlines along the Poincaré section.

Noting that $|\vec{f}(\vec{x})| = \frac{\partial l}{\partial t}$:

$$|\vec{f}(\vec{x})|ds = \frac{\partial l}{\partial t}ds = \frac{\partial A}{\partial t}$$

Therefore:

$$\frac{\partial A}{\partial t} = \frac{\rho_0 |\vec{f_0} \times \hat{s}_0| ds_0}{\rho(\vec{x})} \tag{A.14}$$

Integrating this equation along a streamline yields the total area of that streamline. For a given node k, dA_k is found by taking the difference of A at the midpoint between nodes $k-1$ and k and the midpoint between nodes k and $k+1$. The area of the first node is computed as A at the first midpoint. The area of the last node is the difference between A for the entire streamline and the last midpoint.

A.2.4 Computing gradients on the attractor surface

The partial derivative in the l direction is found using a central difference when possible. At the beginning of a given streamline, a forward difference is used and a backward difference is used at the end of a given streamline. As $\rho_s \frac{\partial \vec{f}}{\partial \xi}$ is a three dimensional vector, three derivatives are obtained, corresponding to the x, y and z components.

To find the partial derivative in the s direction, first consider the forward difference between the jth node on streamline i and the jth node on streamline $i+1$. This difference can be used to approximate the derivative in the s' direction, which is not equal to the s direction. To find the s direction, the difference of some vector $\vec{X}$ in the s' direction, $\Delta_{s'}\vec{X}$ can be decomposed as follows:

$$\Delta_{s'}\vec{X} = \alpha \frac{\Delta_l \vec{X}}{\Delta l} + \beta \frac{\Delta_s \vec{X}}{\Delta s}$$

Where $\vec{X} = \rho_s \frac{\partial \vec{f}}{\partial \xi}$, $\alpha = \vec{s'} \cdot \hat{l}$ and $\beta = \vec{s'} \cdot \hat{s}$. It is important to note that $\vec{s'}$ is not a unit vector like $\hat{l}$ and $\hat{s}$. This expression can be rearranged to yield an expression for $\frac{\partial \vec{X}}{\partial s}$:

$$\frac{\partial \vec{X}}{\partial s} \approx \frac{\Delta_s \vec{X}}{\Delta s} = \frac{1}{\beta}\Delta_{s'}\vec{X} - \frac{\alpha}{\beta}\frac{\Delta_l \vec{X}}{\Delta l}$$

This same equation can be solved for the backwards difference and the average of the forward and backward differences can be taken to find the central difference.

Finally, to find the surface gradient:

$$\nabla_s \cdot \left(\rho_s \frac{\partial f}{\partial \xi} \right) = \frac{\partial \vec{X}}{\partial l} \cdot \hat{l} + \frac{\partial \vec{X}}{\partial s} \cdot \hat{s}$$

Therefore:

$$\frac{\partial \overline{J}}{\partial \xi} \approx \sum_{k=0}^{N} \phi_k \left[\frac{\partial \vec{X}}{\partial l} \cdot \hat{l} + \frac{\partial \vec{X}}{\partial s} \cdot \hat{s} \right]_k dA_k$$

Appendix B

Least Squares Sensitivity Analysis

B.1 Deriving the KKT System

First form the Lagrangian function for equation (5.1):

$$L = \int_0^T \frac{v^2 + \alpha^2\eta^2}{2} + \left\langle w, \left(-\frac{dv}{dt} + \frac{\partial f}{\partial u}v + \frac{\partial f}{\partial \xi} + \eta f \right) \right\rangle \, dt$$

Now consider the first variation, which can be rearranged and transformed by integration by parts:

$$\delta L = \int_0^T \left\langle \delta v, \left(v + \frac{dw}{dt} + \frac{\partial f^*}{\partial u} w \right) \right\rangle \, dt + \int_0^T (\alpha^2\eta + \langle f, w \rangle)\delta\eta \, dt + \langle \delta v, w \rangle |_0^T$$

For first order optimality $\delta L = 0$ for all δv and $\delta\eta$, therefore:

$$\boxed{v + \frac{dw}{dt} + \frac{\partial f^*}{\partial u} w = 0, \quad w(0) = w(T) = 0} \tag{B.1}$$

and

$$\boxed{\alpha^2\eta + \langle f, w \rangle = 0} \tag{B.2}$$

Equations (B.1) and (B.2), along with the tangent equation, make up the KKT system.

B.2 Computing Sensitivities using a Shadow Trajectory

For a trajectory $u(t)$ and shadow trajectory $u'(t)$:

$$\begin{aligned}\delta\bar{J} &= \frac{1}{\mathcal{T}}\int_0^{\mathcal{T}} J(u'(\tau))\ d\tau - \frac{1}{T}\int_0^T J(u(t))\ dt \\ &= \int_0^T \frac{1}{\mathcal{T}} J(u'(\tau(t)))\frac{d\tau}{dt} + \frac{1}{T}J(u(t))\ dt\end{aligned}$$

For some perturbation to the tangent equation $\delta f = \varepsilon\frac{\partial f}{\partial \xi}$, the time transformation is $\frac{d\tau}{dt} = 1 + \varepsilon\eta$:

$$\mathcal{T} = \int_0^T \frac{d\tau}{dt}dt = T + \varepsilon\underbrace{\int_0^T \eta\ dt}_{\equiv H} = T + \varepsilon H$$

Therefore:

$$\delta\bar{J} = \frac{1}{T+\varepsilon H}\int_0^T J(u'(\tau(t))) - J(u(t)) + \varepsilon\eta J(u'(\tau(t))) - \frac{\varepsilon H}{T}J(u(t))\ dt$$

Diving through by ε:

$$\begin{aligned}\frac{\partial\bar{J}}{\partial\xi} &= \lim_{\varepsilon\to 0}\left[\frac{1}{T+\varepsilon H}\int_0^T \frac{(J(u'(\tau(t))) - J(u(t))}{\varepsilon} + \eta J(u'(\tau(t))) - \frac{H}{T}J(u(t))\ dt\right] \\ &= \frac{1}{T}\int_0^T \left\langle\frac{\partial J}{\partial u}, v\right\rangle dt + \frac{1}{T}\int_0^T \eta J(u(t))\ dt - \frac{H}{T}\frac{1}{T}\int_0^T J(u(t))\ dt\end{aligned}$$

By the definition of H, and defining $\overline{x} \equiv \frac{1}{T}\int_0^T x\ dt$:

$$\boxed{\frac{\partial\bar{J}}{\partial\xi} = \overline{\left\langle\frac{\partial J}{\partial u}, v\right\rangle} + \overline{\eta J} - \overline{\eta}\,\overline{J}}$$

B.3 Cyclic Reduction

B.3.1 Conducting cyclic reduction without inverting main diagonal matrices

Consider the following system:

$$\begin{pmatrix} D_1 & U_1 & 0 \\ L_2 & D_2 & U_2 \\ 0 & L_3 & D_3 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

Applying equation (6.5), the following system is obtained:

$$Aw_2 = b$$

with:

$$\begin{aligned} A = &\quad -L_2 D_1^{-1} U_1 + D_2 - U_2 D_3^{-1} L_3 \\ b = &\quad -L_2 D_1^{-1} b_1 + b_2 - U_2 D_3^{-1} b_3 \end{aligned}$$

This system can be solved iteratively, using some preconditioner P:

$$P\Delta x = b - Ax_k, \quad x_{k+1} = x_k + \Delta x$$

Where x_k is the value of λ_2 after k iterations. To compute Ax_k, it is decomposed into three parts:

$$Ax_k = -L_2 D_1^{-1} U_1 x_k + D_2 x_k - U_2 D_3^{-1} L_3 x_k = \alpha + \beta + \gamma$$

α and γ include an inverted matrix, but the inversion can be avoided as follows:

$$-L_2 D_1^{-1} U_1 x_k = \alpha$$

Compute $y_k = U_1 x_k$:

$$-L_2 D_1^{-1} y_k = \alpha$$

Next, define $z_k = D_1^{-1} y_k$. Iteratively solve:

$$D_1 z_k = y_k$$

and use the result to compute α:

$$\alpha = -L_2 z_k$$

γ and the right hand side b can be computed using a similar method. This idea can be applied to a much larger system and allows cyclic reduction to be conducted without inverting any Jacobian matrices. Also, all the matrices are products of Jacobians and identities, so they can be formed when needed, so cyclic reduction is relatively memory efficient for solving such a large system.

B.3.2 Estimating the operation count for cyclic reduction

Define:

- p: the number of flops required to multiply a vector by a Jacobian matrix for the system of interest.
- q: the number of iterations required to carry out multiplication by an inverse matrix.
- n: the number of states in the system

As $L_i = F_{i-1} G_{i-1}^T$ and $U_i = G_i F_i^T$, multiplication by these matrices requires $2p$ flops. As $D_i = F_{i-1} F_{i-1}^T + G_i G_i^T + f_i f_i^T$, multiplication by these matrices requires $4p+2n$ flops. For a Jacobi solver, the number of flops for inverse matrix multiplication is qp.

An estimate of the operation count of cyclic reduction is shown for a few low order terms. These were computed using a symbolic calculator. Starting from a 1 by 1 coarse grid, the number of flops for multiplication by D, U or L were substituted into equation (6.5) recursively. The highest order term for a few different grids is shown in

table B.1, with the number of operations for a single Jacobi iteration (derived in the same way) for comparison. These estimates do not take into account fixed number of operations to backward substitute the coarse grid solution for the fine grid solution.

Time steps m	CR Flops	Jacobi Flops
3	$8pq^2$	$20p + 13n$
5	$16pq^3$	$36p + 23n$
9	$32pq^4$	$68p + 43n$
17	$64pq^5$	$132p + 83n$

Table B.1: Estimate of Operation Count per iteration for cyclic reduction and for the Jacobi method for comparison.

Bibliography

[1] R. Mineck B. Diskin and J. Thomas. Textbook multigrid efficiency for leading edge stagnation. Technical Report TM-2004-213037, NASA Langley Research Center, Hampton, Virginia, May 2004.

[2] R. E. Bank and C. C. Douglas. Sharp estimates for multigrid rates of convergence with general smoothing and acceleration. *SIAM journal on numerical analysis*, 22.4:617–633, 1985.

[3] C. Bonatti, L. Diaz, and M. Viana. *Uniform Hyperbolicity: A Global Geometric and Probabilistic Perspective*. Springer, 2005.

[4] T. F. Chan and W. L. Wan. Robust multigrid methods for nonsmooth coefficient elliptic linear systems. *Journal of Computational and Applied Mathematics*, 123.1:323–352, 2000.

[5] J. Ding and T. Li. Markov finite approximation of frobenius-perron operator. *Nonlinear Analysis, Theory, Methods & Applications*, 17:759–772, 1991.

[6] G. Eyink, T. Haine, and D. Lea. Ruelle's linear response formula, ensemble adjoint schemes and lévy flights. *Nonlinearity*, 17:1867–1889, 2004.

[7] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins Univ. Press, Baltimore, 1996.

[8] B. Hasselblatt. Hyperbolic dynamical systems. In *Handbook of Dynamical Systems 1A*, pages 239–319.

[9] J. M. Hyman and B. Nicolaenko. The kuramoto-sivashinsky equation: A bridge between pde's and dynamical systems. *Physica D: Nonlinear Phenomena*, 18:1-3:113–126, January 1986.

[10] A. Jameson. Aerodynamic design via control theory. *Journal of Scientific Computing*, 3:233–260, 1988.

[11] J.A. Krakos, Q. Wang, S.R. Hall, and D.L. Darmofal. Sensitivity analysis of limit cycle oscillations. *Journal of Computational Physics*, 231:8:3228–3245, 2012.

[12] D. Lea, M. Allen, and T. Haine. Sensitivity analysis of the climate of a chaotic system. *Tellus*, 52, 2000.

[13] D. Lea, T. Haine, M. Allen, and J. Hansen. Sensitivity analysis of the climate of a chaotic ocean circulation model. *Journal of the Royal Meteorological Society*, 128:2587–2605, 2002.

[14] E. Lorenz. Deterministic nonperiodic flow. *Journal of the Atmospheric Sciences*, 20:130–141, 1963.

[15] S. F. McCormick and J. W. Ruge. Multigrid methods for variational problems. *SIAM Journal on Numerical Analysis*, 19.5:924–929, 1982.

[16] S. Y. Pilyugin. Shadowing in dynamical systems. *Lecture Notes in Mathematics*, 1706, 1999.

[17] T. H. Pulliam. Low reynolds number numerical solutions of chaotic flow. In *Proc. AIAA 27th Aerospace Sciences Meeting, Reno, NV Symposium on the Theory of Computing*, 1989. AIAA 89-0123.

[18] S. Gomez Q. Wang and P. Blonigan. Towards scalable parallel-in-time turbulent flow simulations. Submitted to Physics of Fluids. Preprint available at arXiv:1211.2437, 2013.

[19] J.J. Reuther, A. Jameson, J. J. Alonso, M. J. Rimlinger, and D. Sanders. Constrained multipoint aerodynamic shape optimization using an adjoint formulation and parallel computers. *Journal of Aircraft*, 36:21–58, 2001.

[20] D. Ruelle. Differentiation of srb states. *Communications in Mathematical Physics*, 187:227–241, 1997.

[21] J. Thuburn. Climate sensitivities via a fokker-planck adjoint approach. *Quarterly Journal of the Royal Meteorological Society*, 131:73–93, 2005.

[22] D. Venditti and D. Darmofal. Grid adaptation for functional outputs: Application to two-dimensional inviscid flow. *Journal of Computational Physics*, 176:40–69, 2002.

[23] V. E. Henson W. L. Briggs and S. F. McCormick. *A multigrid tutorial*. Society for Industrial and Applied Mathematics, 2000.

[24] Q. Wang. Forward and adjoint sensitivity computation of chaotic dynamical systems. *Journal of Computational Physics*, 235:1–13, February 2013.

[25] Q. Wang and J. Gao. The drag-adjoint field of a circular cylinder wake at reynolds numbers 20, 100 and 500. *submitted to the Journal of Fluid Mechanics*, 2012.

[26] Q. Wang and R. Hui. Sensitivity computation of periodic and chaotic limit cycle oscillations. Submitted to SIAM J. Sci. Comp. Preprint availible at arXiv:1204.0159, 2013.