Summer 2021

# Numerical Approximation of Lyapunov Exponents and its Applications in Control Systems

Nakita K. Andrews

NUMERICAL APPROXIMATION OF LYAPUNOV EXPONENTS AND ITS APPLICATIONS IN CONTROL SYSTEMS

by

NAKITA K. ANDREWS

(Under the Direction of Yan Wu)

ABSTRACT

The progression of state trajectories with respect to time, and its stability properties can be described by a system of nonlinear differential equations. However, since most nonlinear dynamical systems cannot be solved by hand, one must rely on computer simulations to observe the behavior of the system. This work focuses on chaotic systems. The Lyapunov Exponent (LE) is frequently used in the quantitative studies of a chaotic system. Lyapunov exponents give the average rate of separation of nearby orbits in phase space, which can be used to determine the state of a system, e.g. stable or unstable. The objective of this research is to provide control engineers with a convenient toolbox for studying the stability of a large class of control systems. This toolbox is implemented in MatLab with structured programming so that it can be easily adapted by users.

INDEX WORDS: Bass-Gura, Chaotic/Hyperchaotic systems, Controllability, Gram-Schmidt orthogonalization, Jacobian matrix, Lyapunov exponents, Separation of state trajectories, State feedback control

2009 Mathematics Subject Classification: 15A15, 41A10

# NUMERICAL APPROXIMATION OF LYAPUNOV EXPONENTS AND ITS APPLICATIONS IN CONTROL SYSTEMS

by

NAKITA K. ANDREWS

B.S., Georgia Southern University, 2014

M.S., Georgia Southern University, 2021

A Thesis Submitted to the Graduate Faculty of Georgia Southern University in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

NUMERICAL APPROXIMATION OF LYAPUNOV EXPONENTS AND ITS

APPLICATIONS IN CONTROL SYSTEMS

by

NAKITA K. ANDREWS

Major Professor: Yan Wu
Committee: Zhan Chen
Jiehua Zhu

Electronic Version Approved:
July 2021

# DEDICATION

This thesis is dedicated to my family, friends, and the faculty of the Department of Mathematical Sciences at Georgia Southern University that supported me through this program.

## ACKNOWLEDGMENTS

I wish to express my gratitude and appreciation to my advisor, Dr. Yan Wu. He has taught me so much during both my undergraduate and graduate studies at Georgia Southern University. Dr. Wu's guidance and assistance has played a major part in the completion of the most challenging part of my educational career. I also would like to show my appreciation to my committee members, Dr. Zhan Chen and Dr. Jiehua Zhu, as well as all of the other professors and faculty members that have helped me get this far.

# TABLE OF CONTENTS

Page

# LIST OF TABLES

Table Page

# LIST OF FIGURES

Figure Page

# LIST OF SYMBOLS

| | |
|---|---|
| $\mathbb{R}$ | Real Numbers |
| $\mathbb{C}$ | Complex Numbers |
| $\mathbb{Z}$ | Integers |
| $\mathrm{rank}(T)$ | Rank of a linear map |
| $\lambda$ | Lyapunov Exponent |
| $t$ | Time |
| $u$ | Controller |
| $K$ | Feedback Gain Matrix |
| $J$ | Jacobian Matrix |
| $E^*$ | Equilibrium of System |

# CHAPTER 1

# INTRODUCTION

## 1.1 PRELIMINARIES

In this thesis we discuss the development of a universal toolbox to compute Lyapunov exponents of nonlinear dynamical systems, using the Gram-Schmidt Reorthonormalizaton method (GSR). Furthermore, we will discuss how the program will serve as an automatic observer, mainly to control systems without human interaction or interference via the computed Lyapunov exponents, then applying said toolbox to a state feedback control system to detect if it is stabilized. Before we get into that, let us first introduce some basic concepts that will be useful in the next few sections.

**Definition 1.1.1.** A dynamical system is a system in which a function describes the time dependence of a point in a geometrical space.

An $n$ dimensional continuous-time (autonomous) smooth dynamical system is defined by the differential equation

$$\dot{x} = F(x), \tag{1.1}$$

where $\dot{x} = \frac{dx}{dt}$, $x(t) \in \mathbb{R}^n$ is the state vector at time $t$ and $F : U \to \mathbb{R}^n$ is a $\mathbb{C}^r$ function $(r \geq 1)$ on an open set $U \subset \mathbb{R}^n$ [1].

**Definition 1.1.2.** In dynamical systems, a trajectory is the set of points in state space that are the future states resulting from a given initial state.

Here is a formal definition of Lyapunov exponents [2].

**Definition 1.1.3.** Lyapunov exponents or Lyapunov characteristic exponents of a dynamical system characterize the average rate of separation of nearby state trajectories in phase space. Quantitatively, two trajectories in phase space with initial separation vector $\delta\mathbf{Z}_0$

diverge at a rate given by

$$|\delta \mathbf{Z}(t)| \approx e^{\lambda t} |\delta \mathbf{Z}_0|, \tag{1.2}$$

where $\lambda$ is the Lyapunov exponent

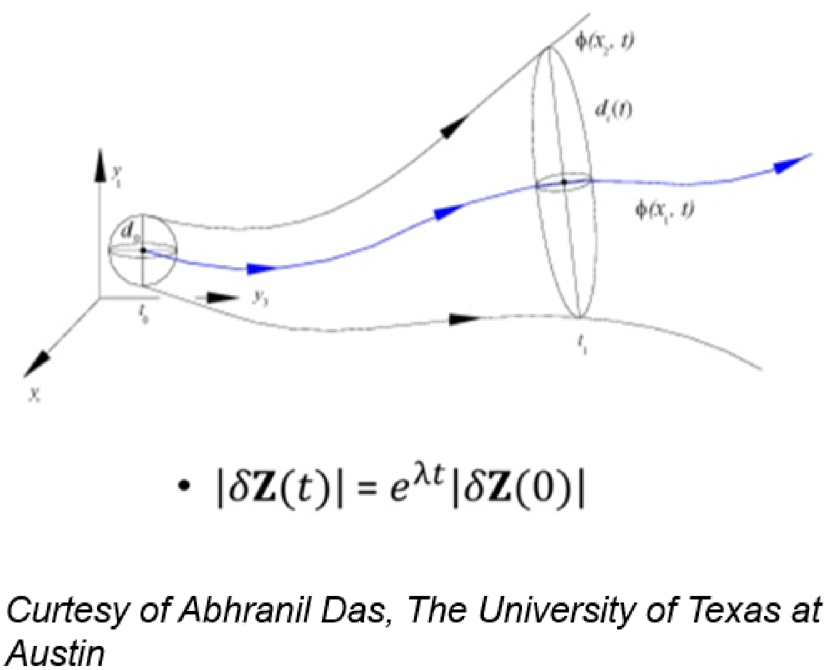


Figure 1.1: Lyapunov spherical divergence

A dynamical system of dimension $n$ has $n$ Lyapunov exponents and $n$ principal directions or eigenvectors, corresponding to a set of nearby trajectories. Or equivalently there are $n$ state variables used to describe the system. The Lyapunov spectra gives an estimate of the rate of entropy production and of the fractal dimension of the attractor of the dynamical system [2].

The state of a dynamical system can be classified by the sign of Lyapunov exponents.

- A positive Lyapunov exponent (+) is adequate for recognising chaos (the system (1.1) must be at least third order) and represents instability;
- Negative Lyapunov exponents (-) corresponds to contracting axes;
- While a zero Lyapunov exponent conveys that the axis is slow varying.

The $i^{th}$ Lyapunov exponent is defined in terms of the length of the $i^{th}$ ellipsoidal principal axis, $p_i(t)$:

$$\lambda_i = \lim_{t \to \infty} \frac{1}{t} \log \frac{\|p_i(t)\|}{\|p_i(0)\|}. \tag{1.3}$$

In a three dimensional dynamical system, for example, the possible signs of Lyapunov spectra are (+, 0, -) for a strange attractor, (0, 0, -) for a two-torus, (0, -, -) for a limit cycle, and finally (-,-,-) for a fixed point. While in a four dimensional system, the combination of (+,+,0,-) corresponds to a hyperchaotic system.

The formula (1.3) is not practical for computing Lyapunov exponents. It is hard to implement and not convenient for finding all Lyapunov exponents. There are numerical methods for estimating the largest Lyapunov exponent. However, the largest Lyapunov exponent doesn't produce enough information for a hyperchaotic system. For instance, in the Rossler four dimensional system, we have the following Lyapunov exponents (+,+,0,-), finding just the largest Lyapunov exponent would not tell us if the system is hyperchaotic.

## 1.2 Known Methods for Finding Lyapunov Exponents

There are different discrete and continuous methods for computing the Lyapunov exponents. All methods are either based on the QR or Singular Value Decomposition (SVD). In this section, these methods are compared for their efficiency and accuracy. Let's first look at the difference between discreet and continuous methods [3].

**Definition 1.2.1.** The discrete methods iteratively approximate the Lyapunov exponents in a finite number of (discrete) time steps and therefore apply to iterated maps and continuous dynamical systems where the linearized flow map is evaluated at discrete times.

**Definition 1.2.2.** A method is called continuous when all relevant quantities are obtained as solutions of certain ordinary differential equations, i.e., continuous methods can only be formulated for continuous dynamical systems, not for discrete maps.

A continuous dynamical system is given by an ordinary differential equation [3]

$$\dot{y} = f(y), y = y(x;t) \in M, t \in R. \tag{1.4}$$

where $M$ is a state space of dimension size $m$. Let an invertible $m \times m$ flow matrix be denoted by $Y$ where $Y = Y(x;t)$. Let $J$ denote the Jacobian matrix. In continuous systems the fundamental matrix is:

$$\dot{Y} = JY, Y(x;0) = I. \tag{1.5}$$

There are multiple methods to compute Lyapunov exponents. However, many methods have one of the following disadvantages:

1. computationally intensive,

2. relatively difficult to implement, or

3. unreliable for small data sets

These disadvantages motivated our search for an algorithm that estimates all Lyapunov exponents that is not impaired by the above mentioned setbacks. We shall go over three popular methods, the QR method, the Singular Value Decomposition (SVD) method, and the Gram-Schmidt Reorthonormalizaton method (GSR), and choose the most efficient method that is easy to implement numerically.

### 1.2.1 QR METHOD

This method is based on the 'QR' factorization for the decomposition of the tangent map (where Q is an orthogonal matrix and R is an upper triangular matrix). It utilizes orthogonal matrices applied to the tangent map and does not require the GSR procedure. This algorithm computes Lyapunov exponents and yields the Lyapunov spectra.

If $Y$ in the matrix variational equations (1.5) is replaced by the product QR we obtain

$$\dot{Q}R + Q\dot{R} = JQR. \tag{1.6}$$

Then multiplying (1.6) with $Q^T$ on the left where $Q^{-1} = Q^T$ and $R^{-1}$ on the right side gives us

$$Q^T\dot{Q} - Q^T JQ = -\dot{R}R^{-1} \tag{1.7}$$

Now the right side of (1.7) is an upper triangular matrix. The skew symmetric matrix components

$$S := Q^T\dot{Q} \tag{1.8}$$

are given by the equation

$$S := \begin{cases} (Q^T JQ)_{ij} & i > j \\ 0 & i = j \\ -(Q^T JQ)_{ji} & i < j \end{cases} \tag{1.9}$$

The matrix $S$ can be used to define the desired differential equation for $Q$:

$$\dot{Q} = QS. \tag{1.10}$$

By (1.7) and (1.9) the equations for the diagonal elements of $R$ are given by

$$\frac{\dot{R}_{ii}}{R_{ii}} = (Q^T JQ)_{ii}, (1 \leq i \leq m). \tag{1.11}$$

To determine the Lyapunov exponents $\lambda_i$ only the logarithms $\rho_i := ln(R_{ii})$ of the diagonal elements of $R$ are important to find. According to (1.11) they fulfill the equations

$$\dot{\rho}_i = (Q^T JQ)_{ii}.(1 \leq i \leq m). \tag{1.12}$$

Thus to compute the spectrum of Lyapunov exponents, we have to simultaneously solve equations (1.10) and (1.12) with the continuous dynamical system (1.4). The quantities $\frac{\rho_i(t)}{t}$ converge to the Lyapunov exponents $\lambda_i$ $(1 \leq i \leq m)$ in the limit $t \to \infty$. It is said to have several advantages over the existing methods, as it involves a minimum number of equations. The errors of this method decrease over time.

### 1.2.2 Singular Value Decomposition (SVD)

Similar to the continuous QR method we will formulate differential equations that are needed to compute the Lyapunov spectrum in terms of the Singular Value Decomposition (SVD). To avoid computational difficulties with the exponential diagonal elements $\phi_i$ $(1 \leq i \leq m)$ of the matrix $F$, let's consider the diagonal matrix

$$E := ln(F) = diag(\varepsilon_1, \cdots, \varepsilon_m) \tag{1.13}$$

with elements $\varepsilon_i := ln(\phi_i)(1 \leq i \leq m)$. When differentiating with respect to time we get

$$\dot{E} = F^{-1}\dot{F} = F^{-1}\dot{U}^T F + F^{-1}U^T JUF + V^T\dot{V}, \tag{1.14}$$

where the derivative of $F$, $\dot{F}$, is given by substituting the flow matrix $Y$ in the matrix variational equations in (1.5) by its singular value decomposition $Y = UFV^T$. To eliminate $V$ in (1.14), the sum $\dot{E} + \dot{E}^T = 2\dot{E}$ is computed where the term $V^T\dot{V} + \dot{V}^T V$ is then eliminated due to the orthogonality of $V$. With the abbreviations

$$\begin{aligned} A &:= U^T\dot{U} = -\dot{U}^T U, \\ B &:= -F^{-1}AF, \\ C &:= U^T JU, \\ D &:= F^{-1}CF, \end{aligned} \tag{1.15}$$

we are able to obtain the following differential equation for $E$:

$$2\dot{E} = B + B^T + D + D^T. \tag{1.16}$$

The right side of (1.16) depends on the matrices $J$, $F = exp(E)$, $U$, and $\dot{U}$. To separate the time derivatives of $E$ and $U$, $\dot{E}$ and $\dot{U}$ respectively, the components of the matrices $B$ and $D$ have to be considered. They are given by the equations

$$\begin{aligned} B_{ij} &= -A_{ij}\frac{\sigma_j}{\sigma_i}, \\ D_{ij} &= C_{ij}\frac{\sigma_j}{\sigma_i}. \end{aligned} \tag{1.17}$$

The orthogonality of $U$ implies that $A$ is skew symmetric and thus $B_{ii} = -A_{ii} = 0$ $(1 \leq i \leq m)$. The diagonal elements $\dot{\varepsilon}_i = \frac{\dot{\sigma_j}}{\sigma_i}$ of $\dot{E}$ therefore fulfill the equation

$$\dot{\varepsilon}_i = C_{ii} \tag{1.18}$$

which can be used to compute the quantities $\frac{\dot{\varepsilon}(t)}{t} \xrightarrow{t\to\infty} \lambda_i(1 \leq i \leq m)$. By means of the off-diagonal elements in (1.16) the $\frac{m(m-1)}{2}$ equations

$$\begin{aligned} 0 &= B_{ij} + B_{ji} + D_{ij} + D_{ji} \\ &= -A_{ij}\frac{\sigma_j}{\sigma_i} - A_{ji}\frac{\sigma_i}{\sigma_j} + C_{ij}\frac{\sigma_j}{\sigma_i} + C_{ji}\frac{\sigma_i}{\sigma_j} \end{aligned} \tag{1.19}$$

where $i > j$ for the components of A can be derived. To remove exponentially growing quantities that are causing problems, equation (1.19) is multiplied by $\frac{\sigma_i}{\sigma_j}$ and the critical terms $\frac{\sigma_i^2}{\sigma_j^2}$ are replaced by

$$h_{ij} := exp(2(\varepsilon_i - \varepsilon_j))(1 \leq i, j \leq m, i \neq j) \tag{1.20}$$

to get

$$A_{ij} = \begin{cases} \frac{C_{ji}+C_{ij}h_{ji}}{h_{ji}-1} & i < j \\ 0 & i = j \\ \frac{C_{ij}+C_{ji}h_{ij}}{1-h_{ij}} & i > j \end{cases} \tag{1.21}$$

By matrix $A$, the desired differential equation for $U$ can be formulated as

$$\dot{U} = UA. \tag{1.22}$$

For non-degenerate Lyapunov spectra $\{\lambda_i\}$ the singular values $\sigma_i(t) \sim exp(\lambda_i t)$ compose of a strictly monotonically decreasing sequence $\sigma_1 > \cdots > \sigma_m$ for $t \to \infty$ and the quantities $h_{ij}\ exp(2t(\lambda_i - \lambda_j))$ quickly converge to zero for $i > j$. This means that the skew symmetric matrix $A$ in (1.22) tends to the matrix $S$ of (1.8) in the limit $t \to \infty$. It should be noted that (1.22) becomes singular for attractors with degenerate Lyapunov spectra because $\lambda_i = \lambda_j$ $(1 \leq i, j \leq m, i \neq j)$ implies $lim_{t\to\infty} h_{ij}(t) = 1$. For this reason and the fact that the continuous SVD method needs even more operations than the continuous QR method we have not investigated the $m \times k$ case although the Singular Value Decomposition is well defined for rectangular matrices.

### 1.2.3 Gram–Schmidt Reorthonormalizaton (GSR)

It seems that using the Gram–Schmidt Reorthonormalizaton (GSR) of tangent vectors is the most feasible method to compute the Lyapunov spectrum of a dynamical system. A differential version has been devised which corresponds to a continuous GSR of the tangent vectors. With the introduction of a stability parameter and a modification the method makes it reliable for numerical computations, applicable to systems with degenerate spectra, and dynamically stable.

To carry out this method, we must have some knowledge of the trajectory of a dynamical system integrating the equations using, for instance, the Runge-Kutta method. This will provide us with the state samples. The state samples are then plugged into the Jacobian system with the solutions being orthogonalized using the Gram-Schmidt method. This then provides some intermediate results of the Lyapunov exponents that are stored and normalized. These steps are repeated for how many runs the user stated. After taking the average of all successive approximations of the Lyapunov exponents over time, we find the final approximation of the Lyapunov exponents. We will give more details of the GSR method in chapter 2.

## 1.3 Chaotic and Hyper-chaotic Systems

Lyapunov exponents (LEs) can be used to identify a control system's stability at its equilibrium and distinguish between chaotic and hyperchaotic systems. When an attractor is chaotic, the trajectories diverge at an exponential rate represented by the largest Lyapunov exponent. If there is one positive Lyapunov exponent produced, the system is said to be chaotic. While if there are two or more positive Lyapunov exponents, the system is said to be hyperchaotic. Let's look at an example of both.

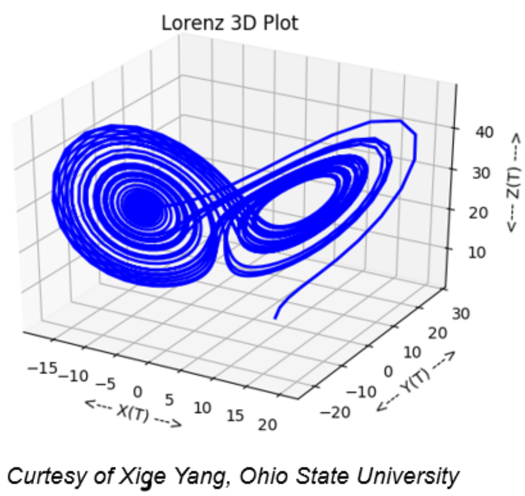


Figure 1.2: Lorenz strange attractor

The Lorenz system was first studied by Edward Lorenz and Ellen Fetter. This is a system of ordinary differential equations known for having chaotic solutions for certain parameter values. Specifically, the Lorenz attractor is the set of chaotic solutions. The equations describe the rate of change of three quantities with respect to time: $x$ is proportional to the rate of convection, $y$ to the horizontal temperature variation, and $z$ to the vertical temperature variation [4]. The shape of the Lorenz attractor, when plotted graphically, resembles a butterfly (Figure 1.2). The model is a system of three ordinary differential equations known as the Lorenz equations:

$$\begin{cases} \dot{x} = \sigma(y - x) \\ \dot{y} = x(\rho - z) - y \\ \dot{z} = xy - \beta z \end{cases}$$

where the constants $\sigma$, $\rho$, and $\beta$ are system parameters proportional to the Prandtl number, Rayleigh number, and certain physical dimensions of the layer itself. The Lorenz system is nonlinear, non-periodic, three-dimensional, and deterministic [4]. The Lorenz attractor is a typical landmark for a chaotic system. It corresponds to one positive Lyapunov exponents, one zero exponent, and one negative exponent.

**Definition 1.3.1.** A deterministic system is a system in which no randomness is involved in the development of future states of the system. A deterministic model will thus always produce the same output from a given starting condition or initial state.

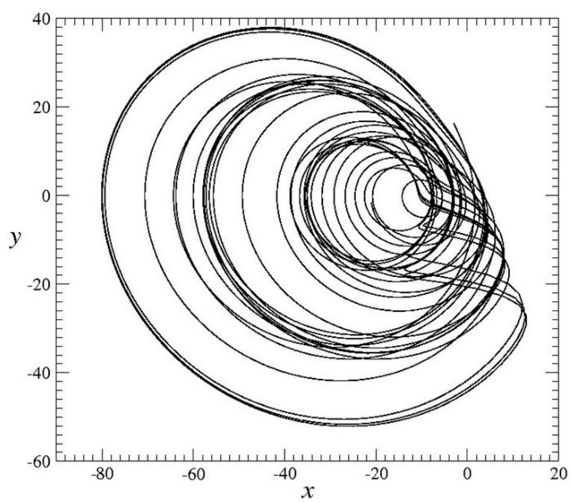


Figure 1.3: Rossler strange attractor

This hyper-chaotic system has a minimum of four dimensions. It is typically defined as a hyperchaotic system because it has at least two positive Lyapunov exponents. Combined with one null exponent along the flow and one negative exponent to ensure the boundness of the solution [5]. The model is a system of four ordinary differential equations:

$$\begin{cases} \dot{x} = -y - z \\ \dot{y} = x + ay + w \\ \dot{z} = b + xz \\ \dot{w} = -cz + dw \end{cases}$$

The Rossler attractor (Figure 1.3) was intended to behave similarly to the Lorenz attractor, but also be easier to analyze qualitatively. An orbit within the attractor follows an outward spiral close to the $x, y$ plane around an unstable fixed point. Once the graph spirals out enough, a second fixed point influences the graph, causing a rise and twist in the $z$-dimension.

The main difference between chaotic and hyperchaotic systems, is the dynamics of the hyperchaotic system expands in more than one direction and generate a much more complex attractor compared with the chaotic system with only one positive Lyapunov exponent. It means that hyperchaotic systems yields more complicated dynamical behaviors compared with chaotic systems.

## 1.4 Control Systems

Chaos control has received more attention due to its potential applications in physics, chemical reactor, biological networks, artificial neural networks, telecommunications and secure communication [11]. Many methods have been used to control dynamical systems. In this thesis we will discuss the Bass-Gura method in state feedback control. There are several methods used to suppress hyper-chaos to unstable equilibrium: speed feedback control, nonlinear doubly-periodic function feedback control, and nonlinear hyperbolic function feedback control. However, the one that will be discussed in this thesis will be linear state feedback control.

A control system regulates the behavior of dynamical systems using closed control loops. It can range from a heating controller using a thermostat to control a boiler to large industrial control systems which are used for controlling machines. For continuously modulated control, a feedback controller is used to automatically control a process or operation. The control system compares the value of the process variable (PV) being controlled with the desired value or set-point (SP), and applies the difference as a control signal to bring the process variable output of the plant to the same value as the set-point [6].

**Definition 1.4.1.** A process variable is the current measured value of a particular part of a process which is being monitored or controlled. An example of this would be the temperature of a furnace.

**Definition 1.4.2.** A set-point is the desired or target value for an essential variable or process variable of a system.

There are two common classes of control action: open loop and closed loop. In an open-loop control system, the action from the controller is independent of the process variable. For instance, think of a central heating boiler being controlled by a timer. The control action is the switching on or off of the boiler. The process variable is the building's temperature. The controller operates the heating system for a constant time regardless of the building's temperature.

In a closed-loop control system, the controller action is dependent on the process variable. In the boiler analogy, a closed-loop control system would utilise a thermostat to monitor the building's temperature, and feedback a signal to ensure the controller output maintains the building's temperature close to the set thermostat temperature. A closed loop controller has a feedback loop which ensures the controller initiates a control action to control a process variable at the same value as the set-point. For this reason, closed-loop controllers are also called feedback controllers.

In a linear feedback system, a control loop including sensors, control algorithms, and actuators are arranged in an attempt to regulate a variable at a set-point (SP). An everyday example would be the cruise control in vehicles, where outside influences, like hills, cause speed changes, and that the driver has the ability to change the desired set speed.

**Definition 1.4.3.** Feedback occurs when outputs of a system are routed back as inputs as part of a chain of cause-and-effect that forms a circuit or loop.

By using feedback properties, the behavior of a system can be modified to meet the needs of an application. One of these applications is stabilizing a systems. Dynamical systems with a feedback experience a modification of its chaos. If we are given a dynamical system

$$\dot{x} = Ax + Bu$$
$$\dot{y} = Cx$$

where $x \in \mathbb{R}^n, u \in \mathbb{R}^n$, $A$ is an $n \times n$ matrix , and $B$ is an $n \times 1$ matrix. The system poles are given by eigenvalues of $A$. We want to use the controller $u$ to move the system poles. Assuming the form of linear state feedback with a gain vector $K$,

$$u = r - Kx, K \in \mathbb{R}^n$$

linear control systems use negative feedback to produce a control signal to maintain the controlled process variable at the desired set-point [6]. There are several types of linear control systems with different capabilities. We will only discuss proportional control.

**Definition 1.4.4.** Negative state feedback occurs when some function of the output of a system, process, or mechanism is fed back in a manner that tends to reduce the fluctuations in the output, whether caused by changes in the input or by other disturbances.

Proportional control happens when a correction is applied to the controlled variable and it is proportional to the difference between the desired value (set-point) and the measured value (process variable). The proportional control system is more complex than an on–off control system, but simpler than a proportional-integral-derivative (PID) control system used in cruise control. An on–off control is not effective for quick corrections and responses. Proportional control overcomes this by modifying the manipulated variable (MV) at a gain level which avoids instability, as well as applying corrections as fast as possible by applying the optimal quantity of proportional correction. A drawback of proportional control is that it cannot eliminate the residual SP–PV error, as it requires an error to generate a proportional output.

## 1.5 Organization of Thesis

In Chapter 2, we present the theoretical results that are necessary for the numerical computation of Lyapunov exponents. Also, in this chapter, we will go into further detail about why Lyapunov exponents are important and its applications. We will then formulate and implement the Gram–Schmidt Reorthonormalizaton method for building a Lyapunov exponent toolbox. In Chapter 3, we will discuss controllability and techniques used to stabilize a chaotic or hyper-chaotic system. We will discuss how to determine the controllability of a dynamical system and then implementing the Bass-Gura method to design feedback control to stabilize the system. In the case studies, we apply the implemented Lyapunov exponent toolbox to monitor the stabilization process of a chaotic system (Lorenz) and a hyperchaotic system (Coupled Lorenz). In Chapter 4, I will make a few concluding remarks.

# CHAPTER 2

# GSR ALGORITHM FOR COMPUTING LYAPUNOV EXPONENTS

Lyapunov exponents are a quantitative measure of the state of a dynamical system from a fixed point, periodic orbits, chaos to hyperchaos. These exponents are important because they allow us to define chaos in a definitive way. If we were to only base chaotic behavior on a picture, there is no telling exactly where a system is chaotic. Hence everyone would have their own opinion about chaos. So introducing a measure of chaos allows us to strictly define chaos. Having this measure allows us to compare different systems, determining if one system is more chaotic than the other.

As mentioned previously, Lyapunov exponents characterize the average rate of separation of nearby state trajectories in phase space. It is calculated for each dimension and it is dependent on the length of the principal axis of the ellipsoid.

$$\lambda = \lim_{t \to \infty} \frac{1}{t} \log \frac{p_i(t)}{p_i(0)}$$

There are three applications of Lyapunov exponents that I will discuss. These exponents can:

- automatically identify control system's stability;
- estimate the dimension of a strange attractor associated with a chaotic system; and
- distinguish between chaotic and hyperchotic systems.

Lyapunov exponents measure the level of chaos in a system, as well as the sensitivity of the system to its initial conditions. These exponents give us an idea of whether a specific direction in the phase space is contracting or expanding. Figure 2.1 illustrates the three behaviors that trajectories may demonstrate.

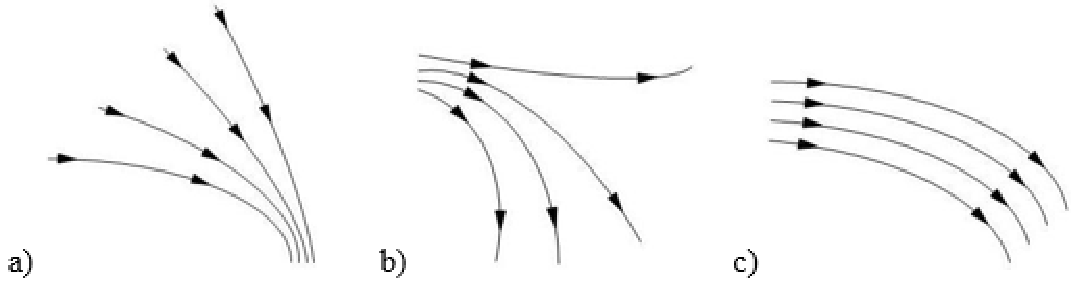


Figure 2.1: Neighboring trajectories with a) stable, b) unstable, and c) marginally stable behavior.

Trajectories that converge, moving closer together, have negative Lyapunov exponents. Having all exponents be negative means that the system is stable. They may also diverge, separating from each other over time, which means there is at least one positive Lyapunov exponent indicating chaotic behavior (the system is unstable). However, if they neither converge nor diverge, but maintain steady distance between each other in a stable cycle, they usually have a Lyapunov exponent close to zero.

Lyapunov exponents also have the ability to distinguish between chaotic and hyperchaotic systems. If the system produces one positive Lyapunov exponent, then the system is chaotic. However, if two or more are positive, the system is hyperchaotic.

In a three dimensional dynamical system, the Lyapunov spectra are (+,0,-), (0,0,-), (0,-,-), and (-,-,-). While in a four dimensional system, there are three types of chaotic attractors. Their Lyapunov spectra are (+,+,0,-), (+,0,0,-), and (+,0,-,-). An example of the first four dimensional attractor would be Rossler's hyperchaotic system.

Lyapunov exponents can also estimate the dimension of a strange attractor associated with a chaotic system. The Lyapunov dimension, $d_f$, is related to the Lyapunov spectrum by the equation:

$$d_f = j + \frac{\sum_{i=1}^{j} \lambda_i}{|\lambda_{j+1}|} \tag{2.1}$$

The concept of finite-time Lyapunov dimension and related definition of the Lyapunov dimension is convenient for the numerical experiments where only finite time can be observed.

## 2.1 MATHEMATICS BEHIND ALGORITHM

The algorithm for computing Lyapunov exponents from differential equations is inspired by the techniques of A. Wolf [7]. If we recall, Lyapunov exponents are mathematical attributes of a chaotic orbit that determine the exponential rate at which nearby trajectories diverge. Let's describe these calculations in some detail. The procedure for finding Lyapunov exponents could be implemented by defining the principal axes with initial conditions whose separations are extremely small and transforming these with nonlinear equations.

**Definition 2.1.1.** The principal axes defined by the linear system are always infinitesimal relative to the attractor.

**Definition 2.1.2.** The fiducial trajectory (center of the sphere) is defined by nonlinear equations with an initial condition. Trajectories of these points are defined by linearized equations on points separated from the fiducial trajectory.

The principal axes are defined by the evolution made by the linearized equations of an orthonormal vector. Even in the linear system, principal axis vectors diverge in magnitude, but this is a problem only because computers have a limited dynamic range for storing numbers [7]. This divergence is easily avoided.

To implement this procedure the fiducial trajectory is created by integrating the nonlinear equations of motion for some post-transient initial condition. Simultaneously, the linearized equations are integrated for $n$ different initial conditions defining an arbitrarily oriented frame of $n$ orthonormal vectors. It has already been pointed out that each vector will diverge in magnitude. This can be avoided by the repeated use of the Gram Schmidt reorthonormalization procedure on the vector. If we recall, the Gram Schmidt reorthonormalization:

**Definition 2.1.3.** Gram-Schmidt orthogonalization, also called the Gram-Schmidt process, is a procedure which takes a nonorthogonal set of linearly independent functions and constructs an orthogonal basis over an arbitrary interval with respect to an arbitrary interval.

Recall the linearized equations that act on the initial frame of orthonormal vectors is a set of vectors, $\{v_1, \cdots, v_n\}$, then Gram Schmidt reorthonormalization provides the orthonormal set, $\{v'_1, \cdots, v'_n\}$, where

$$\begin{aligned} v'_1 &= \frac{v_1}{\|v_1\|} \\ v'_2 &= \frac{v_2 - \langle v_2, v'_1\rangle v'_1}{\|v_2 - \langle v_2, v'_1\rangle v'_1\|} \\ &\vdots \\ v'_n &= \frac{v_n - \langle v_n, v'_{n-1}\rangle v'_{n-1} - \cdots - \langle v_n, v'_1\rangle v'_1}{\|v_n - \langle v_n, v'_{n-1}\rangle v'_{n-1} - \cdots - \langle v_n, v'_1\rangle v'_1\|} \end{aligned}$$

where $\langle v_i, v_j\rangle$ denotes the Euclidean inner product of $v_i$ and $v_j$.

During this process, we see that Gram Schmidt reorthonormalization never affects the direction of the first vector in a system. This vector tends to seek out the direction in tangent space which is most rapidly growing. The second vector has its component along the direction of the first vector removed, and then it's normalized. Since we are changing its direction, vector $v_2$ is not able to find the most rapidly growing direction. Because of the way it is being changed, it is also not able to seek out the second most rapidly growing direction. However, the vectors $v'_1$ and $v'_2$ span the same two-dimensional subspace as the vectors $v_1$ and $v_2$.

The area defined by these vectors is proportional to $2^{(\lambda_1+\lambda_2)t}$. The length of vector $v_1$ is proportional to $2^{\lambda_1 t}$.The monitoring of the length and area growth allows us to determine both exponents. In practice, as $v_1'$ and $v_2'$ are orthogonal, we may determine $\lambda_2$ directly from the mean rate of growth of the projection of vector $v_2$ on vector $v_2'$.

In general, the subspace spanned by the first $k$ vectors is not affected by the Gram Schmidt reorthonormalization. So the long-term evolution of the k-volume defined by these vectors is proportional to $2^{\mu}$, where $\mu = \sum_{i=1}^{k} \lambda_i t$. The projection of the developed vectors onto the new orthonormal frame updates the rates of growth of each of the first $k$-principal axes in turn providing estimates of the $k$ largest Lyapunov exponents. Thus Gram Schmidt reorthonormalization allows the integration of the vector frame for as long as is required for spectral convergence [7].

The definition of the Lyapunov exponent in equation (1.3) states that each Lyapunov exponent gives an asymptotic measure of the variation of the corresponding principal axis. This measure is the result of a cumulative effect on the stretching and compressing of the axes by the velocity field over time. For the purpose of approximating an Lyapunov exponent, consider the interval $(t_0, t_0 + k\Delta t), t_0 \geq 0), 0 < \Delta t < 1$, and $k \in \mathbb{Z}^+$. Using the following formula we can approximate the Lyapunov exponents:

$$\lambda_i \approx \frac{1}{N\Delta t} \sum_{k=1}^{N} log\, \|p_i(t_k)\| \tag{2.2}$$

where $t_k = t_0 + k\Delta t$ and $p_i(t_k)$ is the vector for the $i^{th}$ principal axis. This is the solution of a linearized system associated with the original dynamical system which will be explained in the following discussion. Given the following autonomous system,

$$\dot{x} = f(x, u) \tag{2.3}$$

where $f : \mathbb{R}^{n+1} \to \mathbb{R}^n$ and $u$ is a control input, let $x_k = x(t_k)$ that satisfies equation (2.3). The associated Jacobian system of equation (2.3) obtained from the linearization at $x_k$ is given by

$$\delta\dot{x}_k^{(i)} = \Delta f(x_k, u), \delta x_k^{(i)}(0) = v_i, \tag{2.4}$$

where $v_i$, $i = 1, 2, \ldots, n$, are orthonormal vectors from the Gram-Schmidt orthonormalization process. As such, the principal axis vector is as follows, $p_i(t_k) = \delta x_k^{(i)}$. Also, we let $p_i(t_o) = e_i$ be the standard unit vector. The diagram shown in figure 2.2, illustrates the general flow of the Gram Schmidt Reorthonormalization technique given by equations (2.2) $\sim$ (2.4).

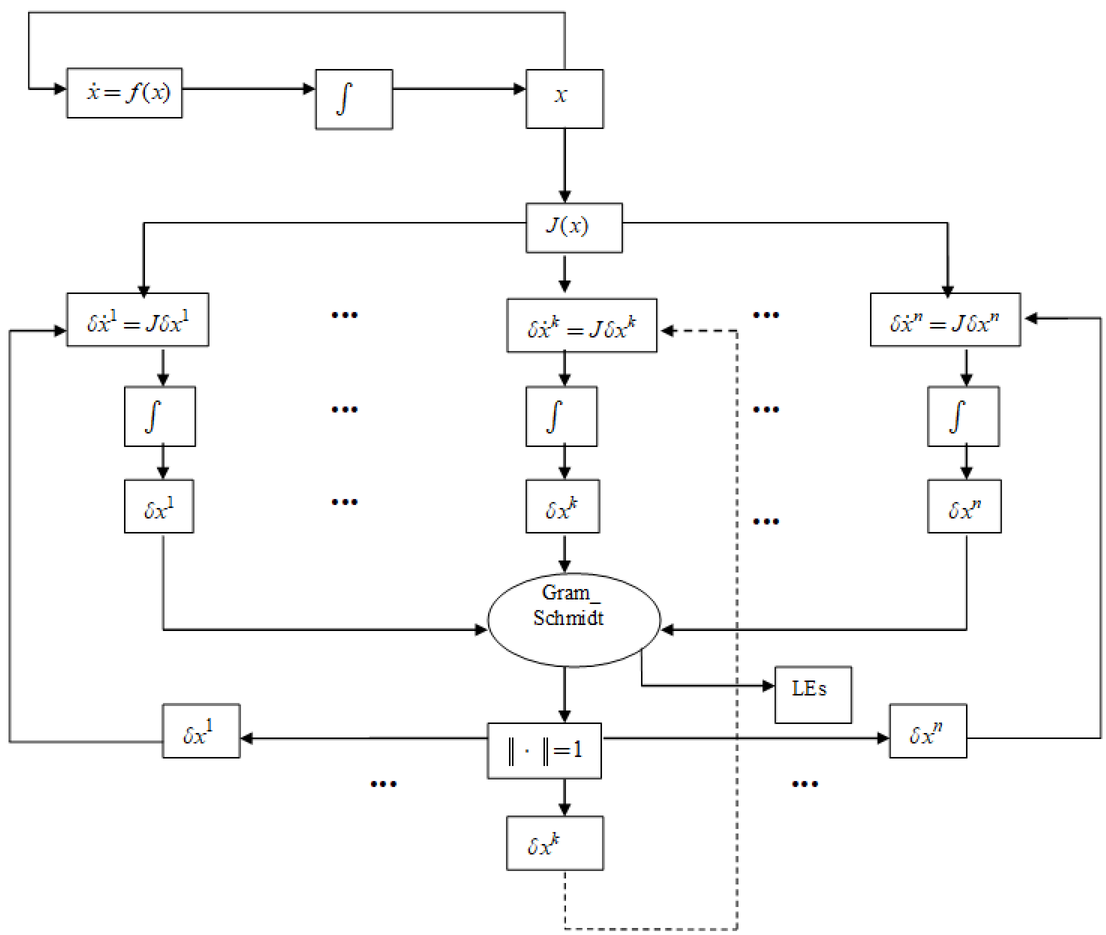


Figure 2.2: Flow chart for computing Lyapunov exponents

## 2.2 CASE STUDIES AND NUMERICAL RESULTS

Let's consider the Lorenz system.

$$\begin{cases} \dot{x} = \sigma(y - x) \\ \dot{y} = x(\rho - z) - y \\ \dot{z} = xy - \beta z \end{cases}$$

To set up the corresponding linearized system for the above equations, we must first find the Jacobian matrix. Which is given by

$$J = \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} & \frac{\partial f_1}{\partial z} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} & \frac{\partial f_2}{\partial z} \\ \frac{\partial f_3}{\partial x} & \frac{\partial f_3}{\partial y} & \frac{\partial f_3}{\partial z} \end{bmatrix}$$

where $f_i$ is the right-hand side of the $i^{th}$ differential equation. For a $n$-dimensional system we would have an $n \times n$ matrix. For the Lorenz system

$$J = \begin{bmatrix} -\sigma & \sigma & 0 \\ \rho - z_n & -1 & -x_n \\ y_n & x_n & -\beta \end{bmatrix}$$

Now to set up the variational equations we need to describe the variations.

$$[\delta] = \begin{bmatrix} \delta_{x1} & \delta_{y1} & \delta_{z1} \\ \delta_{x2} & \delta_{y2} & \delta_{z2} \\ \delta_{x3} & \delta_{y3} & \delta_{z3} \end{bmatrix}$$

where $\delta_{xi}$ is the component of the $x$ variation that came from the $i^{th}$ equation.

The column sums are the lengths of the *x*, *y*, and *z* coordinates of the evolved variation. The rows are the vector coordinates into which the original *x*, *y*, and *z* components of the variation have evolved.

Thus the linearized equations are:

$$\begin{bmatrix} \dot{\delta}_{x1} & \dot{\delta}_{y1} & \dot{\delta}_{z1} \\ \dot{\delta}_{x2} & \dot{\delta}_{y2} & \dot{\delta}_{z2} \\ \dot{\delta}_{x3} & \dot{\delta}_{y3} & \dot{\delta}_{z3} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} & \frac{\partial f_1}{\partial z} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} & \frac{\partial f_2}{\partial z} \\ \frac{\partial f_3}{\partial x} & \frac{\partial f_3}{\partial y} & \frac{\partial f_3}{\partial z} \end{bmatrix} \begin{bmatrix} \delta_{x1} & \delta_{y1} & \delta_{z1} \\ \delta_{x2} & \delta_{y2} & \delta_{z2} \\ \delta_{x3} & \delta_{y3} & \delta_{z3} \end{bmatrix}.$$

In terms of the Lorenz example:

$$\begin{bmatrix} \dot{\delta}_{x1} & \dot{\delta}_{y1} & \dot{\delta}_{z1} \\ \dot{\delta}_{x2} & \dot{\delta}_{y2} & \dot{\delta}_{z2} \\ \dot{\delta}_{x3} & \dot{\delta}_{y3} & \dot{\delta}_{z3} \end{bmatrix} = \begin{bmatrix} -\sigma & \sigma & 0 \\ \rho - z_n & -1 & -x_n \\ y_n & x_n & -\beta \end{bmatrix} \begin{bmatrix} \delta_{x1} & \delta_{y1} & \delta_{z1} \\ \delta_{x2} & \delta_{y2} & \delta_{z2} \\ \delta_{x3} & \delta_{y3} & \delta_{z3} \end{bmatrix}$$

To implement this procedure we solve the new system of differential equations with any numerical differential equation algorithm. In this case, Runge-Kutta 4 for some initial conditions and a time range $[t, t+h]$ where $t$ denotes the initial time and $h$ denotes the time step.

In a chaotic system, each vector tends to fall along the local direction of most rapid growth. In addition, the finite precision arithmetic of computing, the collapse towards a common direction causes the tangent space orientation of all axis vectors to become indistinguishable. To overcome this, Wolf et.al.[7] use repeated Gram-Schmidt reorthonormalization procedure on the vector frame.

Let the linearized equations act on the initial frame of orthonormal vectors to give a set of vectors. After we solve the system of equations, consider the components corresponding to the variational equations. Then the Gram-Schmidt reorthonormalization provides an orthonormal set. The orthonormal set obtained serves as the new initial conditions for our linearized system. We then solve the system again now with these new initial conditions and a new time-range $[t, t+h]$ where $t$ has been changed to $t+h$. Then this is repeated $n$ times.

Below are the results produced by the program for each system.

1. Lorenz

| | |
|---|---|
| $\begin{cases} \dot{x} = \sigma(y - x) \\ \dot{y} = x(\rho - z) - y \\ \dot{z} = xy - \beta z \end{cases}$ | $\lambda_1 = 1.258$<br>$\lambda_2 = 0$<br>$\lambda_3 = -20.966$ |

Table 2.1: Dynamical system and Lyapunov exponents for the Lorenz system

As mentioned previously, the Lorenz system was first studied by Edward Lorenz and Ellen Fetter. It is a system of ordinary differential equations known for having chaotic solutions for certain parameter values. And looking at Figure 2.3, we can see that chaotic behavior. The model is a system of three ordinary differential equations known as the Lorenz equations (See Table 2.1). The equations relate the properties of a two-dimensional fluid layer uniformly heated from below and cooled from above. This figure shows the graphical output of Lyapunov exponents against time in seconds.

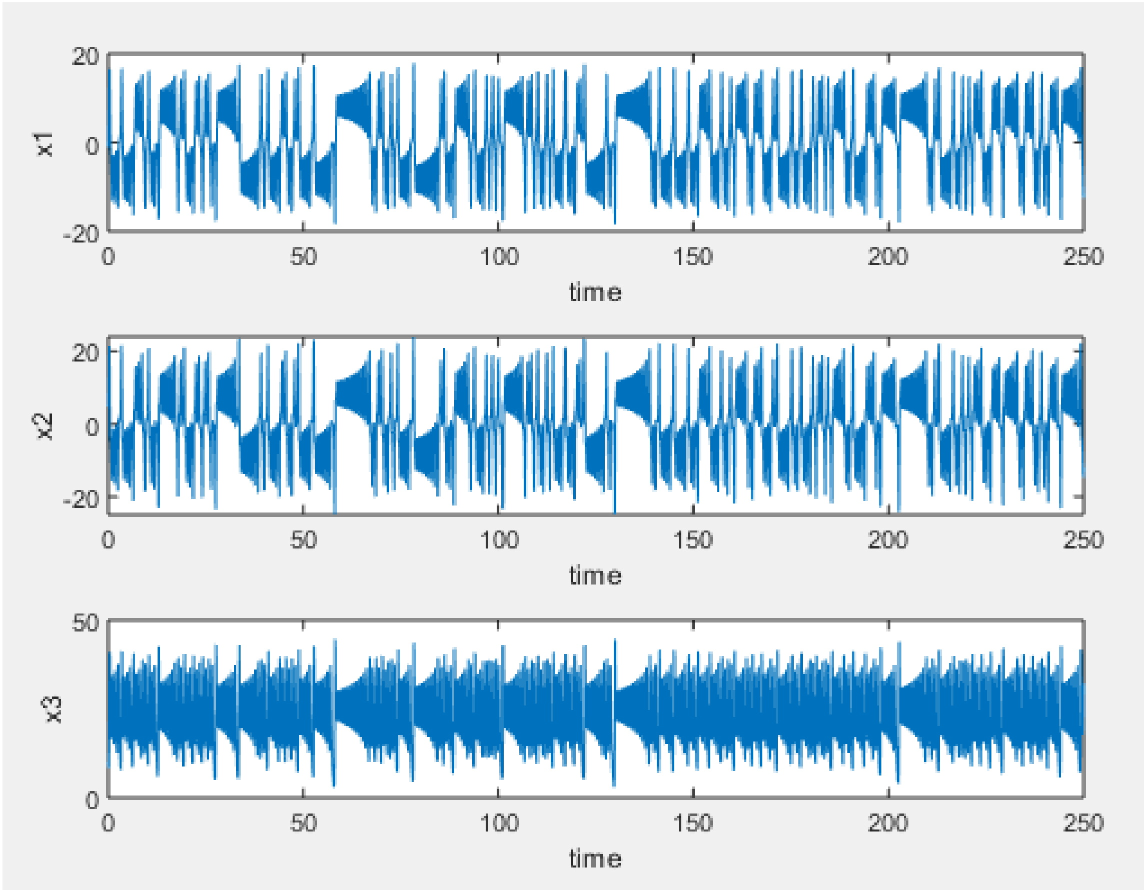


Figure 2.3: Solutions for the Lorenz system

2: Rossler-chaos

| | |
|---|---|
| $\begin{cases} \dot{x} = -y - z \\ \dot{y} = x + ay \\ \dot{z} = b + xz - cz \end{cases}$ | $\lambda_1 = .12$<br>$\lambda_2 = 0$<br>$\lambda_3 = -14.14$ |

Table 2.2: Dynamical system and Lyapunov exponents for the Rossler Chaotic system

The Rossler chaos system is a system of three non-linear ordinary differential equations. This system was originally studied by Otto Rossler in the 1970's. These differential equations define a continuous-time dynamical system that exhibits chaotic dynamics associated with the fractal properties of the attractor. This figure shows the graphical output of Lyapunov exponents against time in seconds.

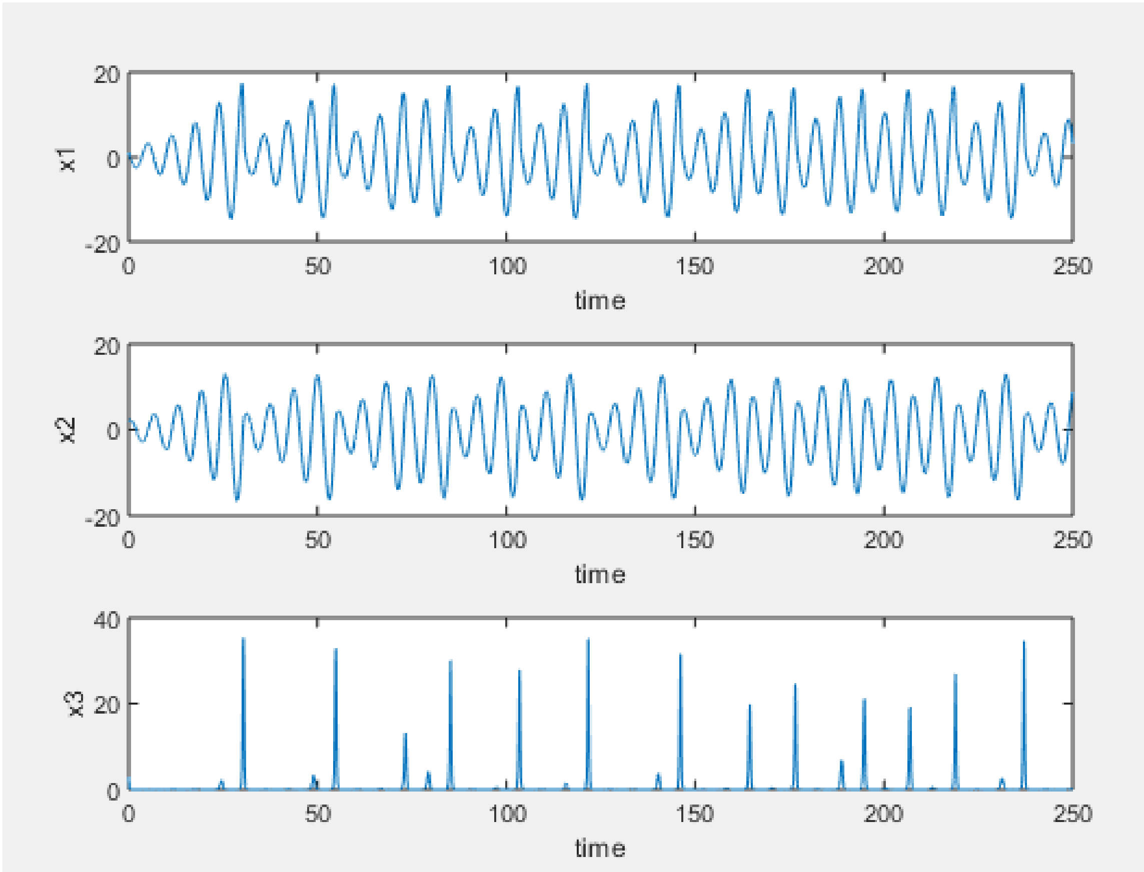


Figure 2.4: Solutions for the Rossler Chaotic system

3: Rossler-hyperchaos

$$\begin{cases} \dot{x} = -y - z \\ \dot{y} = x + ay + w \\ \dot{z} = b + xz \\ \dot{w} = -cz + dw \end{cases}$$

$$\lambda_1 = .17$$
$$\lambda_2 = .02$$
$$\lambda_3 = 0$$
$$\lambda_4 = -33.93$$

Table 2.3: Dynamical system and Lyapunov exponents for the Rossler Hyperchaotic system

This four dimensional system was proposed in 1979 by Rossler. We can see that there are two positive Lyapunov exponents. Therefore, the four dimensional system is a hyperchaotic system. The Lyapunov exponent spectrum is shown in Figure 2.5 against time in seconds.

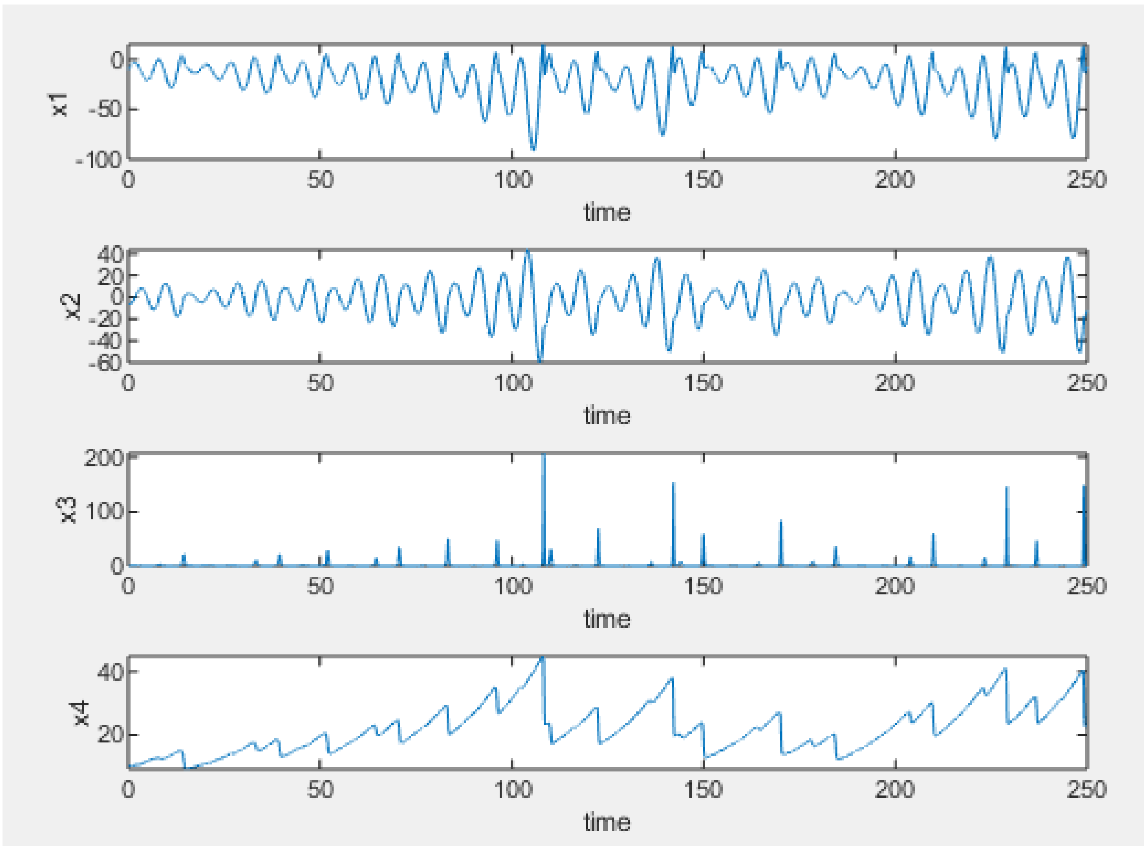


Figure 2.5: Solutions for the Rossler Hyperchaotic system

| 4: Coupled Lorenz | |
|---|---|
| $\dot{x_1} = \rho(y_1 - x_1) - \gamma(x_1 - x_2)$ | $\lambda_1 = 1.1062$ |
| $\dot{y_1} = R_1 x_1 - y_1 - x_1 z_1$ | $\lambda_2 = .84536$ |
| $\dot{z_1} = x_1 y_1 - z_1 - \eta(z_1 - z_2)$ | $\lambda_3 = -.013101$ |
| $\dot{x_2} = \rho(y_2 - x_2) - \gamma(x_2 - x_1)$ | $\lambda_4 = -.012153$ |
| $\dot{y_2} = R_2 x_2 - y_2 - x_2 z_2$ | $\lambda_5 = -18.366$ |
| $\dot{z_2} = x_2 y_2 - z_2 - \eta(z_2 - z_1)$ | $\lambda_6 = -19.051$ |

Table 2.4: Dynamical system and Lyapunov exponents for the Coupled Lorenz system

These equations are coupled via a term common to lasers with injected fields. The coupling constant $\beta$ controls the degree to which these systems interact and may be experimentally varied by reflection losses. The general problem of coupled lasers does not require that there be a single coupling constant for one systems' injection into the other[14]. This figure shows the graphical output of Lyapunov exponents against time in seconds.

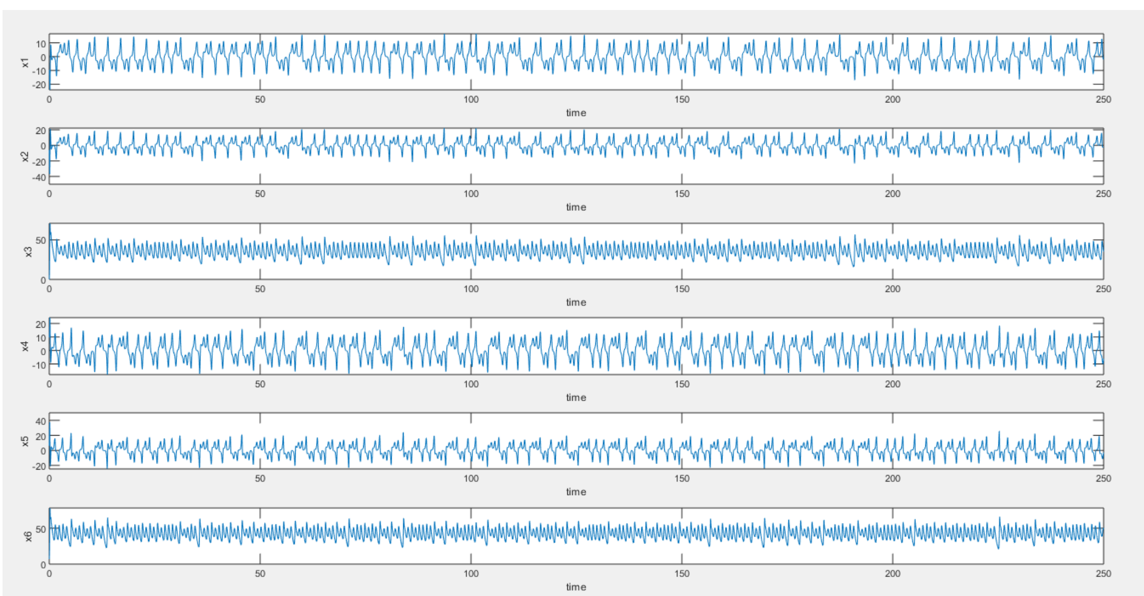


Figure 2.6: Solutions for the Coupled Lorenz system

5: Hyperchaotic Chen

$$\begin{cases} \dot{x} = a(y - x) \\ \dot{y} = 4x - 10xz + cy + 4w \\ \dot{z} = y^2 - bz \\ \dot{w} = -dx \end{cases}$$

$\lambda_1 = 1.9$

$\lambda_2 = .24$

$\lambda_3 = -.01$

$\lambda_4 = -26.70$

Table 2.5: Dynamical system and Lyapunov exponents for the Hyperchaotic Chen system

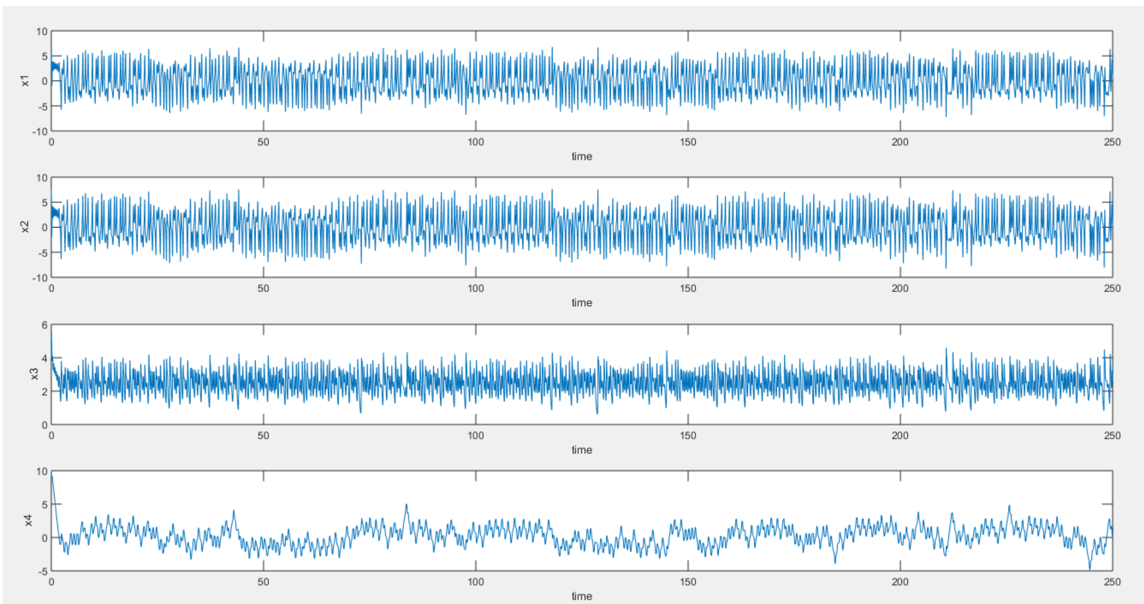


Figure 2.7: Solutions for the Hyperchaotic Chen system

6: Modified Hyperchaotic Chen system

$$\begin{cases} \dot{x} = a(y - x) + w \\ \dot{y} = dx - xz + cy \\ \dot{z} = xy - bz \\ \dot{w} = yz + rw \end{cases}$$

$\lambda_1 = .66$

$\lambda_2 = .22$

$\lambda_3 = 0$

$\lambda_4 = -37.82$

Table 2.6: Dynamical system and Lyapunov exponents for the Modified Chen System

The Chen system was found in 1999, based on Lorenz system. It was designed as a hyperchaos through adding a state-feedback controller to the first control input to drive a unified chaotic system to generate hyperchaos, and it was demonstrated by bifurcation analysis and an electronic circuit implementation. This figure shows the graphical output of Lyapunov exponents against time in seconds.

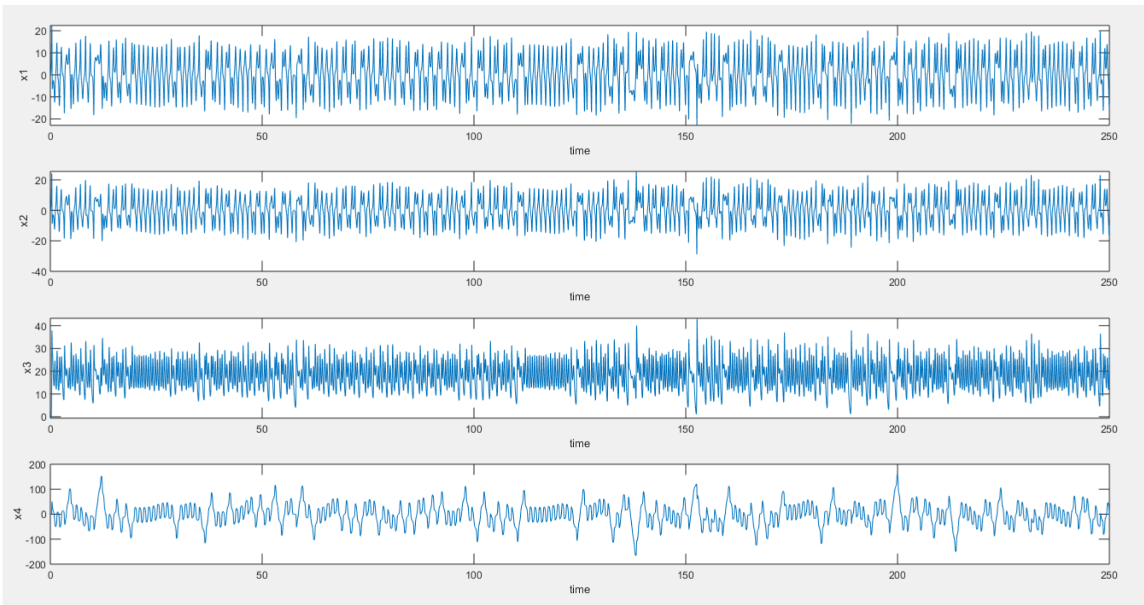


Figure 2.8: Solutions for the Modified Chen system

# CHAPTER 3

# APPLICATION IN CONTROL SYSTEMS

## 3.1 Background on State Feedback Control

The concepts of controllability and state feedback control were built around a linear time invariant system (LTI) with single input single output (SISO)

$$\begin{aligned}\dot{x} &= Ax + Bu \\ y &= Cx + Du\end{aligned} \tag{3.1}$$

where $A$ is an $n \times n$ matrix, $B$ is an $n \times 1$ column vector, $C$ is a $1 \times n$ row vector, and $D$ is $1 \times 1$.

**Definition 3.1.1.** A single-input and single-output (SISO) system is a simple single variable control system with one input and one output.

We know that the system dynamics is largely determined by eigenvalues of matrix $A$. Our goal is to design a control input, $u$, so that the system is stabilized. A proportional controller for a linear state feedback is given by

$$u = r - Kx \tag{3.2}$$

where $K \in \mathbb{R}^{1\times n}$ is the feedback gain matrix and $r$ is a reference signal (For simplicity, let $r = 0$). Since $x$ is known, we have the following closed loop system. See Figure 3.1 We can now substitute.

$$\begin{aligned}\dot{x} &= Ax + B(-Kx) \\ &= (A - BK)x \\ &= \tilde{A}x \\ y &= Cx + Du\end{aligned}$$

where $\tilde{A} = A - BK$, known as the closed loop state matrix

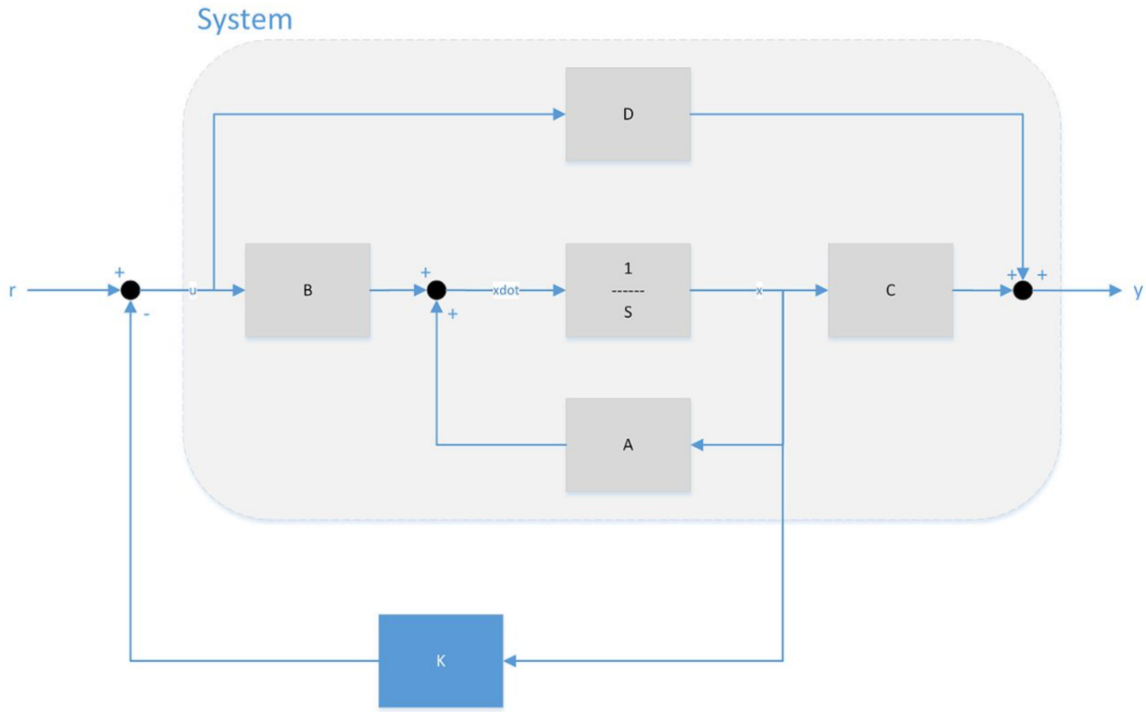


Figure 3.1: System with state feedback (closed loop)

Our objective is to find $K$ such that $\tilde{A}$ has the desired properties. For example, if $A$ is unstable then we must design $\tilde{A}$ to be stable, i.e: all the eigenvalues of $\tilde{A}$ are in the left half of the complex plane. Note that there are $n$ parameters in $K$ and $n$ eigenvalues in $A$. Let's look at an example. Consider the system:

$$\dot{x} = \begin{bmatrix} 0 & 2 \\ 0 & 3 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

then the determinant of matrix $A$ is

$$det(sI - A) = s(s - 3)$$

so our eigenvalues are $s_1 = 0$ and $s_2 = 3$. Because we have a positive eigenvalue, the system is unstable. Let

$$u = -Kx = \begin{bmatrix} k_1 & k_2 \end{bmatrix} x$$

and

$$\begin{aligned} \tilde{A} = A - BK &= \begin{bmatrix} 0 & 2 \\ 0 & 3 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} k_1 & k_2 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 2 \\ 0 & 3 \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ k_1 & k_2 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 2 \\ -k_1 & 3 - k_2 \end{bmatrix} \end{aligned}$$

then the characteristic polynomial of $\tilde{A}$,

$$det(sI - \tilde{A}) = s(s - 3 + k_2) + k_1 = s^2 + (k_2 - 3)s + 2k_1$$

Thus, by choosing $k_1$ and $k_2$, we can place the eigenvalues or poles of $\tilde{A}$ anywhere in the complex plane. For instance, if we want to place our closed loop poles at $s_1 = -2$ and $s_2 = -1$, then the desired characteristic polynomial is

$$(s + 1)(s + 2) = s^2 + 3s + 2$$

with the closed loop characteristic polynomial

$$det(sI - \tilde{A}) = s^2 + (k_2 - 3)s + 2k_1$$

By matching up the coefficients,

$$s^2 + 3s + 2 = s^2 + (k_2 - 3)s + 2k_1$$

we conclude that

$$\begin{aligned} k_2 - 3 = 3 &\rightarrow k_2 = 6 \\ 2k_1 = 2 &\rightarrow k_1 = 1 \end{aligned}$$

so the feedback gain matrix is $K = \begin{bmatrix} 1 & , & 6 \end{bmatrix}$. Of course, it is not always this easy, as lack of controllability might be an issue.

**Definition 3.1.2.** If for some initial state, $x_0$, and some final state, $x_f$, there exists an input sequence to transfer the system state from $x_0$ to $x_f$ in a finite time interval, then the system modeled by the state-space representation is *controllable*.

**Theorem 3.1.** *The $n \times n$ controllability matrix is given by*

$$\Omega_c = \begin{bmatrix} B & AB & A^2B & \cdots & A^{n-1}B \end{bmatrix}.$$

*The system is controllable if the controllability matrix has full row rank (i.e.$rank(\Omega_c) = n$).*

## 3.2 Bass Gura Method

If the system $\{A, B\}$ is controllable, then we can arbitrarily assign the eigenvalues of $\tilde{A}$. Given any polynomial,

$$s^n + \alpha_1 s^{n-1} + \cdots + \alpha_n,$$

there exists a unique gain matrix, $K \in \mathbb{R}^{1\times n}$, for a SISO system such that

$$det(sI - \tilde{A}) = s^n + \alpha_1 s^{n-1} + \cdots + \alpha_n,$$

where $\tilde{A} = A - BK$.

We can prove this by solving for the state feedback gain matrix, $K$, in relation to the closed loop poles. This is where the Bass-Gura method comes into play. This method only works if the system $\{A, B\}$ is controllable. We can then transform $\{A, B, C\}$ into the controller canonical form. That is

$$R^{-1}\Omega_c^{-1}A\Omega_c R = A_c = \begin{bmatrix} -a_1 & -a_2 & \cdots & -a_n \\ 1 & 0 & \cdots & 0 \\ & \ddots & & \vdots \\ 0 & \cdots & 1 & 0 \end{bmatrix}$$

$$R^{-1}\Omega_c^{-1}B = B_c = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

where

$$R = \underbrace{\begin{bmatrix} 1 & a_1 & a_2 & \cdots & a_{n-1} \\ 0 & 1 & a_1 & \cdots & a_{n-2} \\ & & \ddots & & \\ 0 & & \cdots & & 1 \end{bmatrix}}_{\text{Toeplitz Matrix}},$$

$$\Omega_c = \begin{bmatrix} B & AB & A^2B & \cdots & A^{n-1}B \end{bmatrix}$$

$$det(sI - A_c) = s^n + a_1 s^{n-1} + \cdots + a_n$$

Note, $K_c = \begin{bmatrix} k_1 & \cdots & k_n \end{bmatrix}$ so,

$$B_c K_c = \begin{bmatrix} k_1 & k_2 & \cdots & k_n \\ 0 & & \cdots & 0 \\ \vdots & & & \vdots \\ 0 & \cdots & & 0 \end{bmatrix}$$

So the controlled matrix in canonical form:

$$\tilde{A} = A_c - B_c K_c = \begin{bmatrix} -(a_1+k_1) & -(a_2+k_2) & \cdots & -(a_n+k_n) \\ 1 & 0 & \cdots & 0 \\ & \ddots & & \vdots \\ 0 & \cdots & 1 & 0 \end{bmatrix}$$

to which the characteristic polynomial is

$$det(sI - \tilde{A}) = s^n + (a_1+k_1)s^{n-1} + \cdots + (a_n+k_n)$$

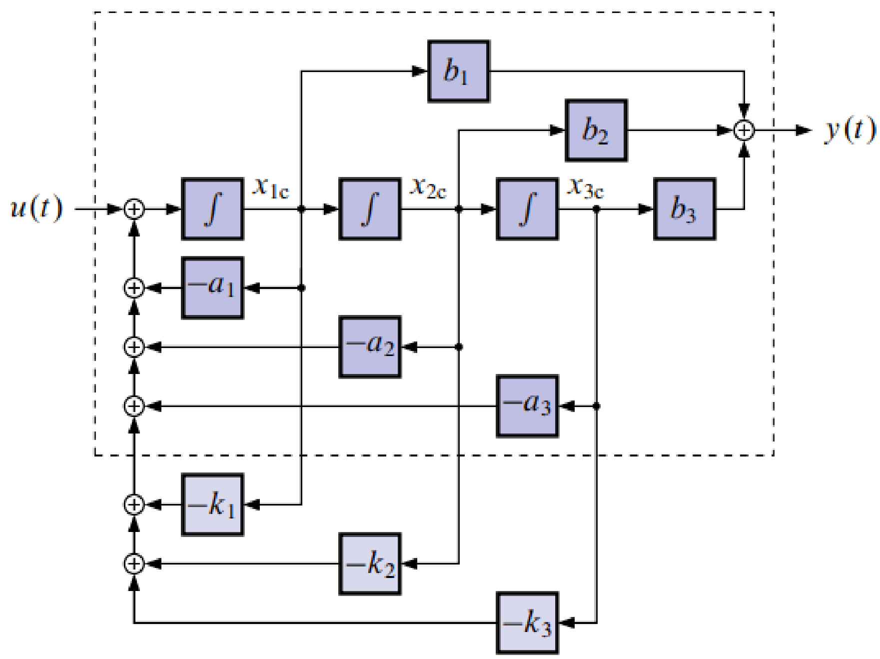


Figure 3.2: Three dimensional transfer function

The desired characteristic polynomial is

$$\tilde{p}(s) = s^n + \alpha_1 s^{n-1} + \cdots + \alpha_n$$

Setting both the original and desired characteristic polynomials equal to each other,

$$\alpha_1 = a_1 + k_{1c} \rightarrow k_1 = \alpha_1 - a_1$$
$$\alpha_2 = a_2 + k_{2c} \rightarrow k_2 = \alpha_2 - a_2$$
$$\vdots$$
$$\alpha_n = a_c + k_{nc} \rightarrow k_n = \alpha_n - a_n$$

we can conclude that, $k = \alpha - a$, where $\alpha = \begin{bmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_n \end{bmatrix}$ and $a = \begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix}$.

$$\begin{aligned}
\tilde{p}(s) &= det(sI - \tilde{A}) = det(sI - A + BK) \\
&= det((sI - A) + BK) = det((sI - A)[I + (sI - A)^{-1}BK]) \\
&= det(sI - A)det(I + (sI - A)^{-1}BK) \\
&= p(s)det(I + (sI - A)^{-1}BK)
\end{aligned}$$

If we were to move the vector $K$ in front of $(sI - A)^{-1}$, $K(sI - A)^{-1}B$ would become a scalar. Also due to the identity $det(I_m + PQ) = det(I_n + QP)$ where $P \in \mathbb{R}^{m \times n}$ and $Q \in \mathbb{R}^{n \times m}$, we have

$$\begin{aligned}
\tilde{p}(s) &= p(s)det(1 + K(sI - A)^{-1}B) \\
&= p(s)[1 + K(sI - A)^{-1}B] \\
&= p(s) + Kp(s)(sI - A)^{-1}B \\
&= p(s) + Kdet(sI - A)(sI - A)^{-1}B \\
&= p(s) + Kadj(sI - A)B
\end{aligned}$$

where $adj(\cdot)$ denotes the adjugate of a matrix. Note that

$$adj(sI - A) = s^{n-1}I + s^{n-2}(A + a_1 I) + s^{n-3}(A^2 + a_1 A + a_2 I) + \cdots$$

then,

$$\tilde{p}(s) - p(s) = K[s^{n-1}I + s^{n-2}(A + a_1 I) + s^{n-3}(A^2 + a_1 A + a_2 I) + \cdots]B$$

where

$$\begin{aligned}
\tilde{a}_1 - a_1 &= KB \\
\tilde{a}_2 - a_2 &= K(A + a_1 I)B = KAB + a_1 KB \\
\tilde{a}_3 - a_3 &= K(A^2 + a_1 A + a_2 I)B = KA^2B + a_1 KAB + a_2 KB \\
&\vdots
\end{aligned}$$

Let $\tilde{a} = \begin{bmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_n \end{bmatrix}$ and $a = \begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix}$.

So,

$$\begin{aligned}
\tilde{a} - a &= \begin{bmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_n \end{bmatrix} - \begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix} \\
&= \begin{bmatrix} KB & KAB & KA^2B & \cdots & KA^{n-1}B \end{bmatrix} \begin{bmatrix} 1 & a_1 & a_2 & \cdots & a_{n-1} \\ 0 & 1 & a_1 & \cdots & a_{n-2} \\ & & \ddots & & \\ 0 & & \cdots & & 1 \end{bmatrix}
\end{aligned}$$

Let

$$R = \underbrace{\begin{bmatrix} 1 & a_1 & a_2 & \cdots & a_{n-1} \\ 0 & 1 & a_1 & \cdots & a_{n-2} \\ & & \ddots & & \\ 0 & & \cdots & & 1 \end{bmatrix}}_{\text{Toeplitz Matrix}}, \Omega_c = \begin{bmatrix} B & AB & A^2B & \cdots & A^{n-1}B \end{bmatrix}.$$

Then, $\tilde{a} - a = K\Omega_c R$. So, $K = (\tilde{a} - a)R^{-1}\Omega_c^{-1}$. This is called the Bass-Gura formula for gain matrix $K$.

## 3.3 Simulation Results

Let's perform the Bass-Gura Method on the Lorenz system. Recall its system of equations:

$$\begin{cases} \dot{x} = \sigma(y - x) \\ \dot{y} = x(\rho - z) - y \\ \dot{z} = xy - \beta z \end{cases} \tag{3.3}$$

Recall that the Jacobian matrix for this system is

$$J = \begin{bmatrix} -\sigma & \sigma & 0 \\ \rho - z^* & -1 & -x^* \\ y^* & x^* & -\beta \end{bmatrix}, \tag{3.4}$$

where $E^* = (x^*, y^*, z^*)$ is the equilibrium of the system. For the Lorenz system, we will use the nonzero equilibrium at $E^* = (-\sqrt{(208/3)}, -\sqrt{(208/3)}, 26)$ based on the following parameter values: $\sigma = 10, \rho = 27, \beta = \frac{8}{3}$ and the control input matrix

$$B = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

So if we go ahead and plug the equilibrium in equation 3.4, we have the following Jacobian matrix:

$$J_{E^*} = \begin{bmatrix} -10 & 10 & 0 \\ 1 & -1 & \sqrt{(208/3)} \\ -\sqrt{(208/3)} & -\sqrt{(208/3)} & -\frac{8}{3} \end{bmatrix}$$

First we must determine if the system is controllable. If we recall Theorem 3.1, we must examine the controllability matrix and see if it is full rank.

The controllability matrix associated with the Lorenz system (3.3)

$$\Omega_c = \begin{bmatrix} B & JB & J^2B \end{bmatrix} = \begin{bmatrix} 0 & 10 & -110 \\ 1 & -1 & -58.333 \\ 0 & -8.3267 & -52.736 \end{bmatrix}$$

and

$$rank(\Omega_c) = 3.$$

Since $rank(\Omega_c)$ has a full rank, the system $\{J, B\}$ is controllable. Now we can use the Bass-Gura method to find the feedback gain matrix, which will be the controller used to stabilize the Lorenz system.

The original characteristic polynomial is

$$p(s) = s^3 + a_1 s^2 + a_2 s + a_3,$$

where

$$a_1 = 13.667$$

$$a_2 = 98.667$$

$$a_3 = 1386.7$$

and let

$$a = \begin{bmatrix} 13.667 & 98.667 & 1386.7 \end{bmatrix}.$$

Let the closed-loop eigenvalues be {-7,-3,-4}. Then the desired characteristic polynomial:

$$\tilde{p}(s) = (s+7)(s+3)(s+4) = s^3 + 14s^2 + 61s + 84$$

where

$$\tilde{a}_1 = 14$$

$$\tilde{a}_2 = 61$$

$$\tilde{a}_3 = 84$$

so $\tilde{a} = \begin{bmatrix} 14 & 61 & 84 \end{bmatrix}$.

Now to find the $R$ matrix

$$R = \begin{bmatrix} 1 & a_1 & a_2 \\ 0 & 1 & a_1 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 13.667 & 98.667 \\ 0 & 1 & 13.667 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\Omega_c = \begin{bmatrix} 0 & 10 & -110 \\ 1 & -1 & -58.333 \\ 0 & 8.3267 & -52.736 \end{bmatrix}$$

Finally, we compute the feedback gain matrix.

$$\begin{aligned} K &= (\tilde{a} - a)R^{-1}\Omega_c^{-1} \\ &= \begin{bmatrix} 2.73333 & .33 & 8.3133 \end{bmatrix} \end{aligned}$$

If we recall from Chapter 2, the Lyapunov exponents for the Lorenz system were

$\lambda_1 = 1.258$

$\lambda_2 = 0$

$\lambda_3 = -20.966$

After controlling the system, we now have the following Lyapunov exponents:

$\lambda_1 = -.29504$

$\lambda_2 = -.91674$

$\lambda_3 = -24.276$

which indicates that the chaotic system is stabilized to its equilibrium. Figure 3.3 is the graphical representation of the system being controlled to its equilibrium.

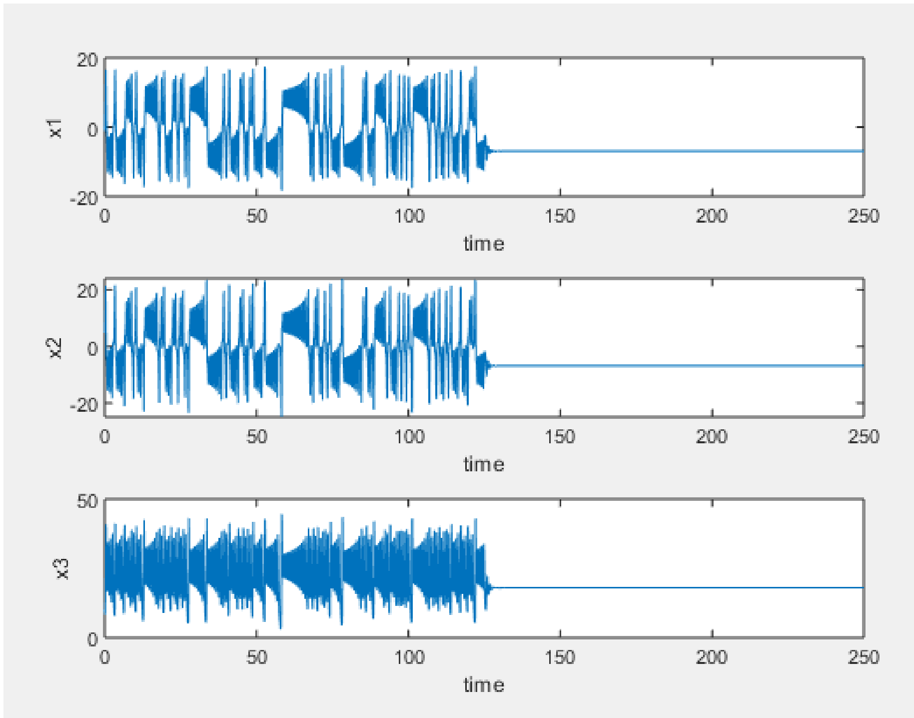


Figure 3.3: Lorenz system controlled after 12,500 runs

If we recall, for this system, we had only one positive exponents before we controlled the system. Which is the reason that it is classified as a chaotic system. In Figure 3.4, you will see the graduate Lyapunov exponent curve for the positive exponents against time in seconds. By the $12,500^{th}$ run, this positive Lyapunov exponent starts to take a dive to become negative. Notice that the curve ends a little below the zero mark, this is what we wanted. With the program after controlling the system, $\lambda_1$ became roughly $-.3$ and this lines up with Figure 3.4 nicely.

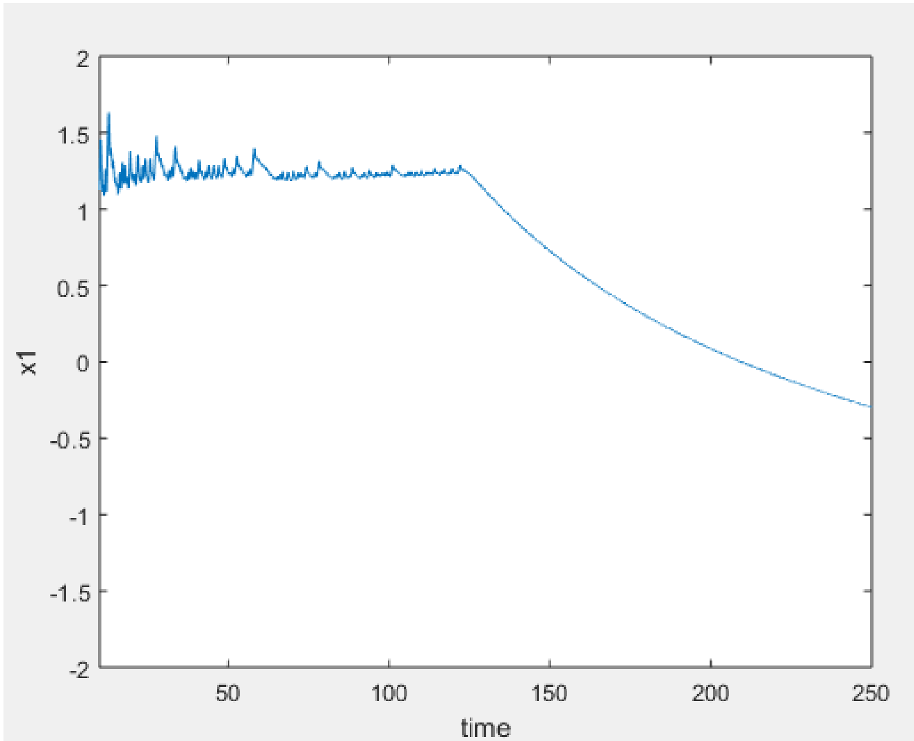


Figure 3.4: Intermediate values for the positive LE

Let's run the simulation again with a system of a higher degree. The sixth order hyperchaotic Coupled Lorenz system. Recall its system of equations:

$$\begin{cases} \dot{x_1} = \rho(y_1 - x_1) - \gamma(x_1 - x_2) \\ \dot{y_1} = R_1 x_1 - y_1 - x_1 z_1 \\ \dot{z_1} = x_1 y_1 - z_1 - \eta(z_1 - z_2) \\ \dot{x_2} = \rho(y_2 - x_2) - \gamma(x_2 - x_1) \\ \dot{y_2} = R_2 x_2 - y_2 - x_2 z_2 \\ \dot{z_2} = x_2 y_2 - z_2 - \eta(z_2 - z_1) \end{cases}$$

and the Jacobian matrix,

$$J = \begin{bmatrix} -\rho-\gamma & \rho & 0 & \gamma & 0 & 0 \\ R_1 - z_1 & -1 & -x1 & 0 & 0 & 0 \\ y_1^* & x_1^* & -1-\eta & 0 & 0 & \eta \\ \rho & 0 & 0 & -\rho-\gamma & \rho & 0 \\ 0 & 0 & 0 & R_2 - z_2^* & -1 & -x_2^* \\ 0 & 0 & \eta & y_2^* & x_2^* & -1-\eta \end{bmatrix},$$

where $E^* = (x_1^*, y_1^*, z_1^*, x_2^*, y_2^*, z_2^*)$ is the equilibrium of the system. For the Lorenz system, we will use the zero equilibrium, $E^* = (0, 0, 0, 0, 0, 0)$. We have the following parameters: $\rho = 10, \gamma = .2, R_1 = 38, \eta = .1, R_2 = 45$ and the control matrix

$$B = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

So if we go ahead and plug in our equilibrium, we have the following Jacobian:

$$J_{E^*} = \begin{bmatrix} -10.2 & 10 & 0 & 0.2 & 0 & 0 \\ 38 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1.1 & 0 & 0 & 0.1 \\ 0.2 & 0 & 0 & -10.2 & 10 & 0 \\ 0 & 0 & 0 & 45 & -1 & 0 \\ 0 & 0 & 0.1 & 0 & 0 & -1.1 \end{bmatrix}$$

First we must determine if the system is controllable. If we recall Theorem 3.1, we must examine the controllability matrix and see if it will produce a full rank.

Let

$$\Omega_c = \begin{bmatrix} B & JB & J^2B & J^3B & J^4B & J^5B \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 10 & -112 & 4952.8 & -96897 & 297300 \\ 1 & -1 & 381 & -4637 & 192840 & -3874900 \\ 0 & 0.1 & -.22 & .364 & -.5368 & 0.74416 \\ 0 & 0 & 2 & -42.8 & 2327.1 & -63276 \\ 0 & 0 & 0 & 90 & -2016 & 106740 \\ 1 & -1.1 & 1.22 & -1.364 & 1.5368 & -1.7442 \end{bmatrix}$$

then to determine the rank of matrix $\Omega_c$.

$$rank(\Omega_c) = 6$$

Since $rank(\Omega_c)$ has a full rank, the system {J,B} is controllable. Now we can use the Bass-Gura method to find the feedback gain matrix, which will be the controller used to stabilize the Coupled Lorenz system.

The original characteristic polynomial is

$$p(s) = s^6 + a_1 s^5 + a_2 s^4 + a_3 s^3 + a_4 s^2 + a_5 s + a_6,$$

where

$$a_1 = 24.6$$

$$a_2 = -633.72$$

$$a_3 = -10546$$

$$a_4 = 141870$$

$$a_5 = 346920$$

$$a_6 = 195170$$

then

$$a = \begin{bmatrix} 24.6 & -633.72 & -10546 & 141870 & 346920 & 195170 \end{bmatrix}.$$

Let the closed-loop eigenvalues be {-7,-3,-4,-2,-1,-6}. Then the desired characteristic polynomial:

$$\tilde{p}(s) = (s{+}7)(s{+}3)(s{+}4)(s{+}2)(s{+}1)(s{+}6) = s^6{+}23s^5{+}207s^4{+}925s^3{+}2144s^2{+}2412s{+}1008,$$

where

$$\tilde{a}_1 = 23$$

$$\tilde{a}_2 = 207$$

$$\tilde{a}_3 = 925$$

$$\tilde{a}_4 = 2144$$

$$\tilde{a}_5 = 2412$$

$$\tilde{a}_6 = 1008$$

then $\tilde{a} = \begin{bmatrix} 23 & 207 & 925 & 2144 & 2412 & 1008 \end{bmatrix}.$

Now to find the $R$ matrix

$$R = \begin{bmatrix} 1 & a_1 & a_2 & a_3 & a_4 & a_5 \\ 0 & 1 & a_1 & a_2 & a_3 & a_4 \\ 0 & 0 & 1 & a_1 & a_2 & a_3 \\ 0 & 0 & 0 & 1 & a_1 & a_2 \\ 0 & 0 & 0 & 0 & 1 & a_1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 23 & 207 & 925 & 2144 & 2412 \\ 0 & 1 & 23 & 207 & 925 & 2144 \\ 0 & 0 & 1 & 23 & 207 & 925 \\ 0 & 0 & 0 & 1 & 23 & 207 \\ 0 & 0 & 0 & 0 & 1 & 23 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\Omega_c = \begin{bmatrix} 0 & 10 & -112 & 4952.8 & -96897 & 297300 \\ 1 & -1 & 381 & -4637 & 192840 & -3874900 \\ 0 & 0.1 & -.22 & .364 & -.5368 & 0.74416 \\ 0 & 0 & 2 & -42.8 & 2327.1 & -63276 \\ 0 & 0 & 0 & 90 & -2016 & 106740 \\ 1 & -1.1 & 1.22 & -1.364 & 1.5368 & -1.7442 \end{bmatrix}.$$

Finally, we compute the feedback gain matrix.

$$\begin{aligned} K &= (\tilde{a} - a)R^{-1}\Omega_c^{-1} \\ &= \begin{bmatrix} 87.848 & -1.6007 & -.00065074 & -372.07 & 2422.6 & .00065074 \end{bmatrix} \end{aligned}$$

If we recall from Chapter 2, the Lyapunov exponents for the Coupled Lorenz system were

$$\lambda_1 = 1.1062$$

$$\lambda_2 = .84536$$

$$\lambda_3 = -.013101$$

$$\lambda_4 = -.012153$$

$$\lambda_5 = -18.366$$

$$\lambda_6 = -19.051$$

After controlling the system, we now have the following Lyapunov exponents:

$$\lambda_1 = -.12591$$

$$\lambda_2 = -3.1460$$

$$\lambda_3 = -19.202$$

$$\lambda_4 = -22.275$$

$$\lambda_5 = -31.469$$

$$\lambda_6 = -62.587$$

which indicates that the chaotic system is stabilized to its equilibrium. Figure 3.5 is the graphical representation of the system being controlled to its equilibrium.

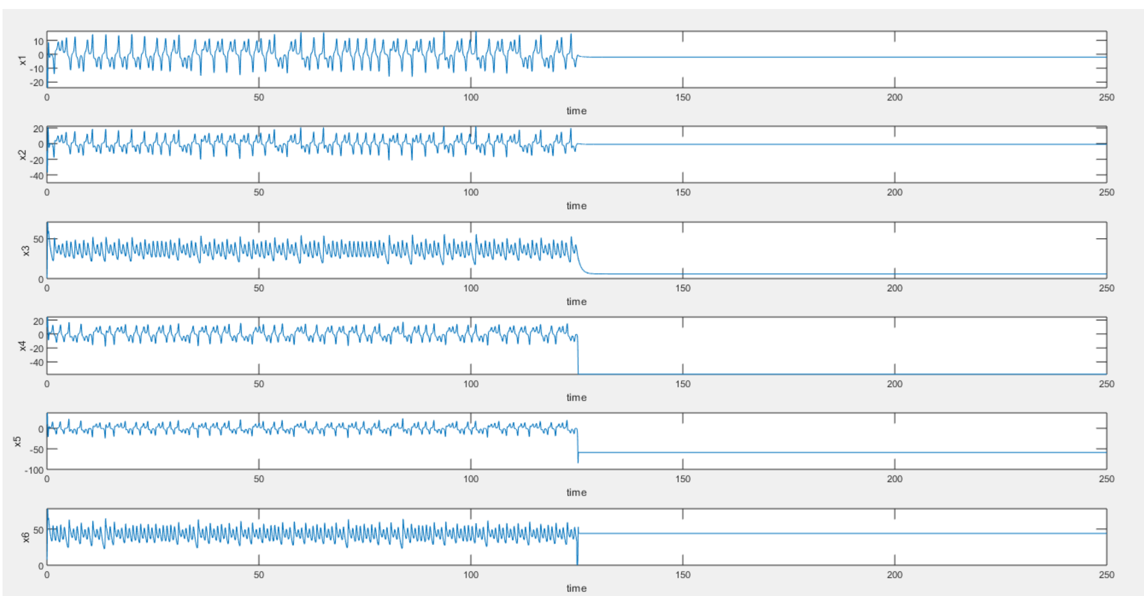


Figure 3.5: Coupled Lorenz system controlled after 12,500 runs

If we recall, for this system, we had two positive exponents before we controlled the system. Which is the reason it is classified as a hyperchaotic system. In Figures 3.6 and 3.7, you will see both graduate Lyapunov exponent curves for the positive exponents againg time in seconds. By the $12,500^{th}$ run, the positive Lyapunov exponents start to fall and negative. Notice that the first curve, which is associated with the first positive Lyapunov exponent ($\lambda_1$) ends a little below the zero mark, this is what we wanted. After controlling the system, $\lambda_1$ became roughly $-.1$ and this lines up with Figure 3.6 nicely. As for the second positive Lyapunov exponent, $\lambda_2$, the curve ends a little under the $-3$ mark which coincides with $\lambda_2$ being roughly $-3.1$.

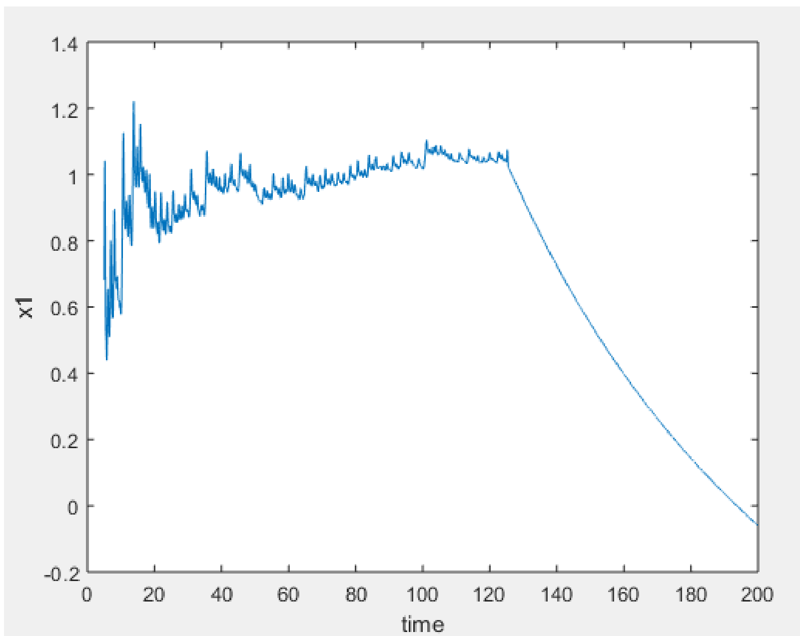


Figure 3.6: Intermediate values for the first positive LE

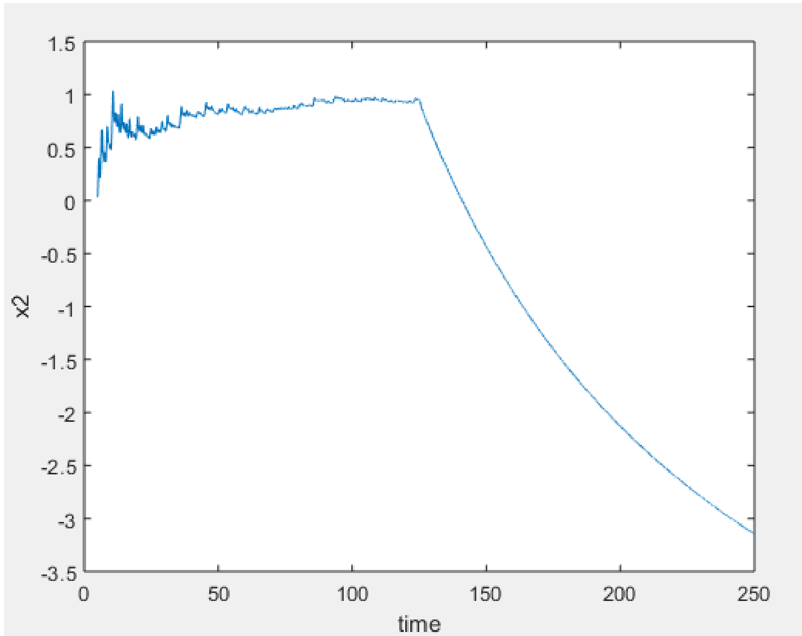


Figure 3.7: Intermediate values for the second positive LE

# CHAPTER 4

# CONCLUSION

The objective of this thesis was to development a universal toolbox to compute Lyapunov exponents of arbitrary nonlinear dynamical systems, using the dynamical equation method. We wanted to remove the need of human interaction or interference during the process of stabilizing a dynamical system. We believe that this toolbox is beneficial to control engineers.

Since the Lyapunov exponents are effective indicators for determining if a system is stable, we incorporated the Lyapunov exponent toolbox with different control systems to test its instant feedback on stability. The results are consistent enough that the computed Lyapunov exponents can be used to monitor the behavior of a control system effectively.

# REFERENCES


[1] Marco Sandri *Numerical Calculation of Lyapunov Exponents* The Mathematica Journal (1996) 78-84

[2] Lyapunov Exponent
https://en.wikipedia.org/wiki/Lyapunov_exponent

[3] Karlheinz Geist, Ulrich Parlitz, and Werner Lauterborn *Comparison of Different Methods for Computing Lyapunov Exponents* Progress of Theoretical Physics, Vol.83, No.5 (1990)

[4] Lorenz System
https://en.wikipedia.org/wiki/Lorenz_system

[5] Christophe Letellier and Otto E. Rossler *Hyperchaos* Scholarpedia, 2(8):1936 (2007)

[6] Control System
https://en.wikipedia.org/wiki/Control_system

[7] A. Wolf, J. B. Swift, H. L. Swinney, and J. A. Vastano *Determining Lyapunov Exponents From a Time Series* Physica 16D (1985) 285-317

[8] T. Okushima, *New Method for Computing Finite-Time Lyapunov Exponent*, Phys. Rev. Lett. 25 254101 (2003)

[9] Benettin, Giancarlo, Galgani, L., Giorgilli, Antonio, Strelcyn, Marie. *Lyapunov Characteristic Exponents for smooth dynamical systems and for Hamiltonian systems; A method for computing all of them. Part 2: Numerical application.*, Meccanica.(1980). 15. 21-30. 10.1007/BF02128237.

[10] Rossler Attractor
https://en.wikipedia.org/wiki/R%C3%B6ssler_attractor

[11] Zhenya Yan *Controlling hyperchaos in the new hyperchaotic Chen system* Applied Mathematics and Computation 168 (2005) 1239–1250

[12] Michael T. Rosenstein, James J. Collins, and Carlo J. De Luca *A Practical Method for Calculating Largest Lyapunov Exponents from Small Data Sets* NeuroMuscular Research Center, Boston University (1992)

[13] Jo Bovy *Lyapunov Exponents and Strange Attractors in Discrete and Continuous Dynamical Systems* Theoretical Physics Project (2004)

[14] N. M. Lawandy and Kayee Lee *Stability Analysis of Two Coupled Lorenz Lasers and the Coupling-Induced Periodic-Chaotic Transition* Optics Communications (1986)

[15] Runge-Kutta Methods
`https://en.wikipedia.org/wiki/Runge%E2%80%93Kutta_methods`

[16] K. Ramasubramanian and M. S. Sriram *A Comparative Study of Computation of Lyapunov Spectra with different Algorithms* Department of Theoretical Physics, University of Madras (2018)

[17] Benettin, Giancarlo, Galgani, L., Giorgilli, Antonio, and Strelcyn, Marie. *Lyapunov Characteristic Exponents for smooth dynamical systems and for Hamiltonian systems; A method for computing all of them. Part 1: Theory.*, Meccanica.(1980)

[18] Raju K George, IIST, and Thiruvananthapuram
`http://www.math.iitb.ac.in/~neela/CIMPA/notes/CIMPA_RKG.pdf`

[19] Guoyuan Qi, Michael Antonie van Wyk, Barend Jacobus van Wyk, and Guanrong Chen *On a New Hyperchaotic System* Physics Letter A (2008)

[20] Jiang Xu, Gouliang Cai, and Song Zheng *A Novel Hyperchaotic System and its Control* Journal of Uncertain Systems, Vol.3, No.2, (2009)

[21] P. Ehrharda, Ch. Marchera, and U. Mullera *Dynamical Behavior oof Natural Convection in a Double-Loop System* Hemisphere Publication, Vol.2, No.1, (1987)

[22] Hubertus F. von Bremen, Firdaus E. Udwadia, and Wlodek Proskurowski *An Efficient QR Based Method for the Computation of Lyapunov Exponents* Physica D (1997)

[23] Abhranil Das
`http://informationtransfereconomics.blogspot.com/2016/05/lyapunov-exponents-and-information.html`

[24] Xige Yang
https://people.math.osu.edu/yang.2677/Files/Lorenz.html

# APPENDIX A

# COMPLETE MATLAB CODE FOR FINDING LYAPUNOV EXPONENTS

The algorithm for computing Lyapunov exponents from differential equations are inspired by the techniques of A. Wolf [7]. This program is used for computing the Lyapunov exponents of nonlinear dynamical systems from experimental data done from a set of differential equations. The toolbox consists of four different programs: three subroutines and a main program. The first subroutine is a code for Runge-Kutta 4 used to solve $\frac{dy}{dt} = f(t, y)$ along with a linearized system. The second lists the parameters and functions for each dynamical system. The end result will be in the form of a column vector, which will be processed in the main program. The third is on the Gram Schmidt orthogonalization method. The main program ties everything together. It's where we set our initial conditions. Parts of this program requires user-input. The user is able to decide which of the six pre-selected systems they would like to compute Lyapunov exponents for. The user can add new systems of differential equations to the program.

## Runge-Kutta 4 Method

```
% RK method is used to solve dy/dt=f(t,y) along with a linearized system. In this program, the sysnf corresponds to the f(t,y)and the linearized system

function [nx,ndv]=rk4n(t,x,dv,u,h)

xx=[x;dv];
k1=h*sysnf(t,xx,u);
k2=h*sysnf(t+h/2,xx+k1/2,u);
```

```
k3=h*sysnf(t+h/2,xx+k2/2,u);
k4=h*sysnf(t+h,xx+k3,u);
xx=xx+(k1+2*k2+2*k3+k4)/6;
nx=xx(1:length(x));
ndv=xx(length(x)+1:2*length(x));
```

SYSTEM DIFFERENTIAL EQUATIONS FUNCTIONS

```
%This is the f-function in the DE dy/dt=f(t,y). This is
    where you list all of your parameters. You would list all
     of your functions as well. Your end result will be in
    the form of a column vector.

function dx=sysnf(t,x,u)
    global picksys a b c d r K1 K2 acu

    if picksys==1
        p=10; R=27; b=8/3;  % system parameters for the
            Lorenz system
        x1=x(1); x2=x(2); x3=x(3);

        dx1=p*(x2-x1);
        dx2=R*x1-x2-x1*x3+u*x2;
        dx3=x1*x2-b*x3;
        dx=[dx1;dx2;dx3];       %column vector

        JM=[-p p 0; R-x3 -1+u -x1; x2 x1 -b];  % Jacobian
```

```
            matrix for linear syst
        delx=x(4:6);
        ddx=JM*delx;

        dx=[dx;ddx];
    end

    if picksys==2
        a=.1; b=.1; c=9;              % system parameters for
            the Rossler Chaos system
        x1=x(1); x2=x(2); x3=x(3);

        dx1= -(x2+x3);
        dx2= x1 + a*x2;
        dx3= b + x3*(x1-c);
        dx=[dx1;dx2;dx3];       %column vector

        JM=[0 -1 -1; 1 a 0; x3 0 x1-c];
        delx=x(4:6);
        ddx=JM*delx;

        dx=[dx;ddx];
    end

    if picksys==3
        a=.25; b=3; c=0.05; d=0.5;              % system
```

```
            parameters for the Rossler Hyper-Chaos system
        x1=x(1); x2=x(2); x3=x(3); x4=x(4);

        dx1= -(x2+x3);
        dx2= x1 + a*x2 + x4;
        dx3= b + x3*x1;
        dx4= c*x4 - d*x3;
        dx=[dx1;dx2;dx3;dx4];      %column vector

        JM=[0 -1 -1 0; 1 a 0 1; x3 0 x1 0; 0 0 -d c];
        delx=x(5:8);
        ddx=JM*delx;

        dx=[dx;ddx];
    end

    if picksys==4
        p=10; y=.2; R1=38; n=.1; R2=45;      % system
            parameters for the Lorenz Coupled system
         x1=x(1); x2=x(2); x3=x(3); x4=x(4); x5=x(5); x6=x
             (6);

         dx1=p*(x2-x1)-y*(x1-x4);
         dx2= R1*x1-x2-x1*x3-acu*K1*x2;
         dx3= x1*x2-x3-n*(x3-x6);
         dx4= p*(x5-x4)-y*(x4-x1);
```

```
        dx5= R2*x4 - x5-x4*x6;
        dx6= x4*x5-x6-n*(x6-x3)-acu*K2*x6;
        dx=[dx1;dx2;dx3;dx4;dx5;dx6]; %column vector

        JM=[-p-y p 0 y 0 0; R1-x3 -1-acu*K1 -x1 0 0 0;x2 x1
            -1-n 0 0 n;y 0 0 -p-y p 0;0 0 0 R2-x6 -1 -x4; 0
            0 n x5 x4 -1-n-acu*K2];
        delx=x(7:12);
        ddx=JM*delx;

        dx=[dx;ddx];
    end

if picksys ==5                          % system
    parameters for the Chen system
      a=35; b=3; c=21; d=2;
      x1=x(1); x2=x(2); x3=x(3); x4=x(4);

      dx1=a*(x2-x1);
      dx2=4*x1-10*x1*x3+c*x2+4*x4;
      dx3=x2^2-b*x3;
      dx4= -d*x1;
      dx=[dx1;dx2;dx3;dx4];     %column vector

      JM=[-a a 0 0; 4-10*x3 c -10*x1 4; 0 2*x2 -b 0; -d 0
         0 0];
```

```
            delx=x(5:8);
            ddx=JM*delx;

            dx=[dx;ddx];
end

if picksys ==6                                        % system
    parameters for the Chen system
        % a= 35; b = 3; c = 12; d = 7; r = 0.4;
            x1=x(1); x2=x(2); x3=x(3); x4=x(4);

            dx1= a*(x2-x1) + x4;
            dx2= d*x1 - x1*x3 + c*x2;
            dx3= x1*x2 - b*x3+u;
            dx4= x2*x3 + r*x4+u;
            dx=[dx1;dx2;dx3;dx4];        %column vector

            JM=[-a a 0 1; d-x3 c -x1 0; x2 x1 -b 0; 0 x3 x2 r];
            delx=x(5:8);
            ddx=JM*delx;

            dx=[dx;ddx];
    end
```

## Gram Schmidt Orthogonalization

```
%   This function is to perform Gram_schmidt orthogonalization process

function Q=ngrsch(A)
[M,N]=size(A);
rkn=rank(A);
if rkn~=N
    disp('The vectors are not linearly independent. Abort!')
    Q=[];
    return
end
v1=A(:,1);
%v1=v1/norm(v1);
Q=v1;
for k=2:N
    w=A(:,k);
    vsum=0;
    for j=1:k-1
        vsum=vsum+w'*Q(:,j)*Q(:,j)/norm(Q(:,j))^2;
    end
    P=w-vsum;
    % P=P/norm(P);
    Q=[Q,P];
end
```

## Main program

```
clear all
close all
format short e
global picksys a b c d r K1 K2 acu
a = 35; b = 3; c = 12; d = 7; r = .4;
%*****************************
% Initial set up
%*****************************
picksys=1;    % Which nonlinear system to use:
              % 1: Lorenz, 2: Rossler-chaos, 3: Rossler-hyperchaos,
              % 4: Lorenz Coupled, 5: Chen, 6: Modified Chen


% initial vector for the nonliear system tied up with picksys
if picksys==1  % Lorenz system
    x=[-1, 5, 10]';
end
if picksys==2 % Rossler 3D
    x=[1, 2, 3]';
end
if picksys==3 % Rossler 4D
    x=[-10, -6, 0, 10]';
```

```
end
if picksys==4 % Lorenz coupled
    x=[-10, -6, 0, 10, 6, 7]';
end
if picksys ==5 %Chen
    x = [3;7;4;10];
end
if picksys==6 % Modified Chen
    x = [-0.1 0.2 -0.6 0.4]';
end

%*****************************
% Parameters/Initial Conditions
%*****************************

t=0; % initial condition
h=0.01; %step size
tt=t;
xx=x';
total=25000; %total number of runs

dimN=length(x);
initialm=eye(dimN);
initm=[];

for k=1:dimN
```

```
        dx=initialm(:,k);
        initm=[initm,dx];
end
si=zeros(1,dimN);
%****************************
%Activate controller for Lorenz
stc=12500;
u=0; CK=8;
ple=[];  % record the intermediate values for the positive
    LE

%Activate controller for Coupled Lorenz
STC=12500;
acu=0;
K1=68;
K2=76;
ple1=[]; ple2=[];% record the intermediate values for the
    positive LEs

%***************************
% Simulation block
%***************************

for n=1:total
        if n>stc
                u=-CK;
```

```
        end
        if n>STC
            acu=1;
      end
        for k=1:dimN
            dv=initm(:,k);
            [nx,dv]=rk4n(t,x,dv,u,h); % numerically integrate the
                augmented system
            initm(:,k)=dv;
        end

 % apply the gram_schmidt subroutine to orthonormalize initm
       V=ngrsch(initm);    % not normalized

        for i=1:dimN
            if picksys==1
                if i==1
                    ple=[ple;log(norm(V(:,i)))/log(2)];
                end
            end
            if picksys==4
                if i==1
                    ple1=[ple1;log(norm(V(:,i)))/log(2)];
                end
                if i==2
                    ple2=[ple2;log(norm(V(:,i)))/log(2)];
```

```
                end
            end
            si(i)= si(i)+log(norm(V(:,i)))/log(2);
            V(:,i)=V(:,i)/norm(V(:,i));
        end
        initm=V;
        x=nx;
        t=t+h;
        tt=[tt;t];
        xx=[xx;x'];
end
%*****************************

lamv=[];
for k=1:dimN
        lambda = si(k)/t;
        lamv=[lamv,lambda];
end

if picksys==1
        plev=[];
        lenple=length(ple);
        for k=1:lenple
                intave=sum(ple(1:k))/(k*h);
                plev=[plev;intave];
        end
```

```
end

if picksys==4
    plev1=[]; plev2=[];
    lenple=length(ple1);
    for k=1:lenple
        intave1=sum(ple1(1:k))/(k*h);
        intave2=sum(ple2(1:k))/(k*h);
        plev1=[plev1;intave1];
        plev2=[plev2;intave2];
    end
end

disp('Here are the computed Lyapunov exponents:')

lamv
%****************************


%****************************
% Plot the state trajectories
%****************************
plotstate=[1:dimN];    % user defined states to plot
Np=length(plotstate);

for k=1:Np %plots on same
```

```
    figure(1)
    subplot(Np,1,k)
    plot(tt,xx(:,plotstate(k)))
    xlabel('time')
    ylabel(['x',num2str(plotstate(k))])
end


%***********************************************
% Plot the intermediate values for the positive LE(s)
%***********************************************

if picksys==1
    figure(2)
    axis([10,250,-2,2])
    plot(tt(2:25001),plev)
    xlabel('time')
    ylabel('x1')

end
if picksys==4
    figure(2)
    axis([0,200,-1,1.5])
    plot(tt(500:20000),plev1(500:20000))
    xlabel('time')
    ylabel('x1')
```

```
    figure(3)
    axis([0,200,-1,1.5])
    plot(tt(500:25000),plev2(500:25000))
    xlabel('time')
    ylabel('x2')
end
```