THE PENNSYLVANIA STATE UNIVERSITY
SCHREYER HONORS COLLEGE

DEPARTMENT OF ENGINEERING SCIENCE AND MECHANICS

STABILITY ANALYSIS OF A SIMPLE CPG-BASED WALKING MODEL

PATRICK J. DEVLIN

Spring 2011

A thesis
submitted in partial fulfillment
of the requirements
for a baccalaureate degree
in Engineering Science
with honors in Engineering Science

Reviewed and approved* by the following:

Joseph P. Cusumano
Professor of Engineering Science and Mechanics
Thesis Supervisor

Corina S. Drapaca
Assistant Professor of Engineering Science and Mechanics
Honors Adviser

Judith A. Todd
P.B. Breneman Department Head Chair
Professor, Department of Engineering Science and Mechanics

* Signatures are on file in the Schreyer Honors College and Engineering Science and Mechanics Office.

ABSTRACT

Human locomotion, one of the most common of everyday occurrences, still eludes complete understanding. While it is simple to measure people's movements, it is very difficult to understand how they control them. When conducting experiments on humans, researchers do not have access to the variables that drive the control systems of the subject, which limits the range of methods that can be applied. To bypass this inhibitor, many researchers use computer simulations utilizing differential equations. These studies provide access to all necessary variables, however the models are often crude and unrealistic. These different branches of studies have both contributed to the understanding of locomotion, however their effectiveness is limited because of the difficulty of comparison. This paper attempts to bridge the gap between the two different types of studies by exploring an experimental method which, if validated, could study the stability properties of both humans and computer models. This process relies on delay reconstruction to extend the application of the limited laboratory variables to a higher dimensional analysis of the stability properties and gait transitions.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

This research project would not have been possible without the outstanding support of Dr. Joseph Cusumano. Thank you for providing constant insight and guidance throughout my project. I could not have been successful without your help.

I would also like to thank the Engineering Science and Mechanics Department at Penn State University. The faculty members have been extremely helpful throughout my studies and have prepared me for a fulfilling career.

Chapter 1

Introduction

Research towards further understanding of human locomotion is needed now as much as ever. Many different medical applications rely on the use of locomotion research to provide a better life to the injured or diseased. These applications include the design of prosthetics (Donker & Beek 2002, Culham et al. 1986), the diagnosis and treatment of neurological diseases such as Parkinson's Disease (Blin et al. 1990, Azulay et al. 1999), and many others. With such important aspects of life depending on locomotion studies, it is critical that these studies are of the highest possible quality. In this paper, possible methods for an experimental approach to studying the stability of human locomotion are tested for their validity. These methods provide insight into the stability properties of walking models and actual human walking.

Stability Analysis in Human Locomotion

Stability analysis remains one of the most important focuses of research in locomotion. Many of the medical applications of locomotion studies depend on the stability analysis of various gait variables. The absence of access to neurological control variables makes this analysis more difficult, as only biomechanical variables can be studied through actual human experimentation. Fortunately these tests are still useful, as a strong correlation has been drawn between the stability of these biomechanical variables and the risk of falling while walking (Menz et al. 2003). On the other hand, when studying locomotion through computer models and simulations, the neurological control variables can be studied. While there are many different methods to these studies, almost all depend on quantifying the variability of the state variables

from patterns (Dingwell & Cusumano 2000). The variability can be studied in an effort to determine how the observable correlates both quantitatively and qualitatively to physical gait.

Methods of Assessing Stability

Approaches for analyzing stability vary depending on the objective and the available observables. As mentioned, human experimentation is limited to the use of biomechanical variables. For these studies, many researchers choose to study stride lengths and stride times and their variability. Recently, some have begun to use these variables to study neurological control by measuring how the body reacts to any variability from the intended walking pattern (Dingwell et al. 2010). When using computer models, the researcher has access to all of the variables, including those simulating the neurological controls. In these studies, it is possible to change the neurological parameters and study their effect on the biomechanical system (Raftery et al. 2007). Furthermore, the availability of all of the variables allows the Jacobian to be studied. The Jacobian of a system is a nonlinear operator composed of partial derivatives that acts on a set of initial conditions and outputs the new state at a later time. This quantity is critical to stability analysis because the eigenvalues of this operator contain information about the stability properties of the system.

Analysis of Gait

The analysis of gait involves studying qualitatively different types of walking. Bifurcation theory provides an approach for conducting such analysis. By plotting a biomechanical variable of interest versus a control parameter, a bifurcation diagram demonstrates the transitions between different types of motion (Tufillaro et al. 1992). In gait these transitions

can represent the difference between walking and running. A sample variable used for bifurcation analysis in locomotion studies is the stiffness of a muscle. For instance, ankle stiffness has been studied for its effect on posture (Verdaasdonk et al. 2004). Bifurcation analysis is often, including in this paper, conducted by measuring data at a Poincare section of a limit cycle. This process will be better explained in Chapter 2.

Central Pattern Generator Models

Many realistic walking models today contain some form of a Central Pattern Generator. CPGs are control networks that rely on oscillators to drive some physical function of the model. For applications in locomotion, the CPG generally receives feedback, such as reaction forces with the ground, which then affects the control variables (Van de Crommert et al. 1997). Bipedal walking models that contain CPGs can become very complex. They often utilize a combination of several oscillators, rely on multiple sources of feedback, and contain dozens of differential equations that drive the system (Taga 1995). Less realistic CPG models exist as well and play a vital role in the study of locomotion. These models are much simpler to understand and despite their crude appearance, still demonstrate many important qualities of CPG based models. One such model will be the starting point for analysis in this paper.

Chapter 2

Stability Analysis

The purpose of this project is to validate experimental analysis methods of the stability of human locomotion. The methods to be studied involve using linear regression from a series of data points to estimate a matrix that maps the state of a periodic system to its state after an additional cycle. The matrix will then be analyzed for its stability properties. These methods are experimental in nature, and their application helps to bridge the disconnection between computer simulations and actual human experimentation. Validation would support the previous work of many researchers and will provide future researchers with additional tools and the confidence to use them.

Bifurcation Analysis

In order to justify that the methods in question can identify the stability properties of a system, a proven method must be conducted to provide target results. A Bifurcation analysis will be applied to find qualitative transition points in the behavior of the system. These points represent stability transition points, or 'bifurcation points', and carry great significance in the study of mechanical systems. Ideally, the proposed methods will be able to accurately locate the known bifurcation points of a system.

Poincare Section

The bifurcation analysis will be conducted at a Poincare section of the solution to the system of equations. A Poincare section is a surface of one less dimension than the system and corresponds to a plane where a certain variable is fixed. Every time the solution completes one cycle and crosses the section, the values of the other variables are recorded and this represents one data point. This can be seen in Figure 2-1, where x represents one data point, and the governing equations combined effect is summarized in P(x), a function that outputs the state of the next point on the surface based on the previous state.

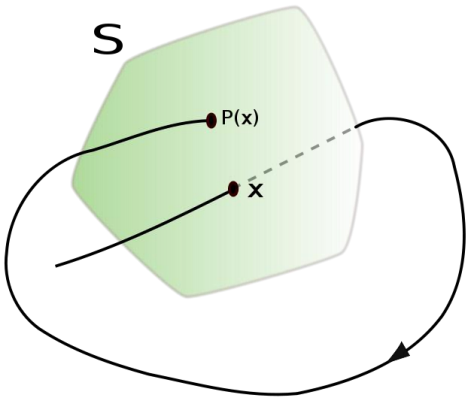

Figure 2-1 Poincare Map (Ocioso 2008).

In this paper, the Poincare section will represent when a biomechanical variable is at a peak and thus the velocity of this variable will be zero. Experimentally, a Poincare section could be set up to record the state of a person every time their heel strikes the ground. Relationships between how the data changes each time the solution reaches the Poincare section is very useful in locomotion analysis.

Design of Accurate Peak Detector

In order to perform regression testing, the data points need to be as accurate as possible. While it is easy to select a data point with the largest local value, a true maximum will actually occur between measured data points. This inaccuracy called for the design of a peak detector. Rather than using the local maximum data point, this point and the two on either side of it were fit to a parabola. The maximum of the function of the parabola was then calculated along with the time at which the peak occurred. Furthermore, parabolas were fit to the corresponding five data points for the other state variables. Then, these equations were used to calculate the values of the other state variables at the same precise time that the peak occurred. This new data set, consisting of the accurate peaks of one variable and the values of the other variables at these peaks, is the state of the system every time the velocity of the variable of interest is zero. This selection represents the data every time the zero velocity Poincare section is crossed.

Numerical Solution of the Jacobian

Because these tests are being conducted on computer models, it is possible to set the initial conditions of the system in any desired way. The desired product of these methods is a Jacobian that can map initial conditions on the Poincare section to the state at the next intersection of the Poincare section. This matrix's eigenvalues will provide insight into the stability properties of the whole system. While the Jacobian is a non-linear operator, with the right adjustments, it can be assumed linear. Rather than using the true values of the data points, the deviations of the data from the steady state limit cycle were calculated. The true Jacobian of this new system is still nonlinear; however it can be represented by a Taylor series expansion about the limit cycle. If the deviations are kept small, the higher order terms of the Taylor series

can be approximated as zero, leaving the Jacobian to be represented by a linear first order term. The freedom granted by the accessibility of the computer model allows a simple trick to find this estimated Jacobian quite accurately. The initial conditions can be set with a finite perturbation of only one variable with the other variables equaling the steady state limit cycle. Because of the zero deviation values of the other variables, this initial condition will map to a state which is directly proportional to a corresponding column of the Jacobian. This will be easier to understand using a specific model as an example, which will be done later in the paper. This numerical estimation of the Jacobian will serve as a baseline for comparison of the other methods.

Regression Jacobian Method

In reality, initial conditions cannot be set to precise desired states. There are uncontrollable variables involved in actual human locomotion that make finding the numerical solution of the Jacobian impossible. An experimentally useful method involves the regression of the deviations from a periodic solution as a result of random noise. A system is set initially to a periodic solution and then allowed to run with a small amount of noise added. As was done in the numerical solution, the data ($\underline{x}$) is collected from the crossing of the Poincare section and then converted to a system of deviations ($\underline{u}$) from the solution ($\underline{x}^*$) so that the Jacobian (**A**) may be estimated as linear as seen in equations 2.1 and 2.2.

$$\underline{u} = \underline{x} - \underline{x}^* \tag{2.1}$$

$$\underline{u}_{n+1} = \mathbf{A}\underline{u}_n \tag{2.2}$$

However now, in order to perform regression, instead of just collecting data for the next time the system crosses the Poincare section, the system will run for a substantial period of time, crossing the section many times and creating a large data set. The resulting algebra will take the

form of equation 2.3, where $\underline{u}_i$ is a column vector of deviations from the periodic solution and **A** is a square matrix estimating the Jacobian.

$$\underline{u}_{i+1} = \mathbf{A}\underline{u}_i \tag{2.3}$$

If N is the number of peaks measured, i ranges from 1 to N-1.

Delay Reconstruction Method

The third approach to analyzing the stability properties of a system presented in this paper is the Delay Reconstruction Method. The purpose is to allow the use of only one variable to determine the stability of a multi-variable system. If validated, the impact of this method could be tremendous as this can easily be applied to actual human experimentation. As described earlier, the lack of access to neurological control variables limits the ability to study the stability of human locomotion. However, if just a single biomechanical variable could be used to study the stability, then researchers could better compare computer models and human experimentation.

The basis for this method comes from the following algebra. Because the system is estimated as linear, based on the Taylor series expansion argument presented before, a set of operators map a set of initial conditions to the new state. This is symbolized in three dimensions in equation 2.4.

$$\underline{x}_1 = \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} = \begin{pmatrix} f_x(\mathrm{x}_0, \mathrm{y}_0, \mathrm{z}_0) \\ f_y(\mathrm{x}_0, \mathrm{y}_0, \mathrm{z}_0) \\ f_z(\mathrm{x}_0, \mathrm{y}_0, \mathrm{z}_0) \end{pmatrix} = \underline{F}(\underline{x}_0) \tag{2.4}$$

Therefore, x_1 will be a function of x_0, y_0, and z_0.

$$\mathrm{x}_1 = f_x(\mathrm{x}_0, \mathrm{y}_0, \mathrm{z}_0) \tag{2.5}$$

Likewise, x_2 will be a function of x_1, y_1, and z_1.

$$\mathrm{x}_2 = f_x(\mathrm{x}_1, \mathrm{y}_1, \mathrm{z}_1) \tag{2.6}$$

If x_1 is expressed as in equation 2.5, and y_1 and z_1 in a similar manner, then x_2 can be expanded as in equation 2.7.

$$x_2 = f_x(f_x(x_0, y_0, z_0), f_y(x_0, y_0, z_0), f_z(x_0, y_0, z_0)) \tag{2.7}$$

The combined effect of these functions can be expressed as a new function (g_x) of x_0, y_0, and z_0.

$$x_2 = g_x(x_0, y_0, z_0) \tag{2.8}$$

Likewise, x_3 can then be expressed as a new function (h_x) of x_0, y_0, and z_0.

$$x_3 = h_x(x_0, y_0, z_0) \tag{2.9}$$

Using these new relationships, the first three instances of x can be written as a set of operators acting on x_0, y_0, and z_0.

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} f_x(x_0, y_0, z_0) \\ g_x(x_0, y_0, z_0) \\ h_x(x_0, y_0, z_0) \end{pmatrix} \tag{2.10}$$

These functions represent a matrix operating on the initial conditions. This is significant because it shows that a single matrix can map three variables of one state into the next three instances of one of those variables. It is not important what the value of this matrix is, but rather the fact that it exists. Stability properties such as bifurcation points are inherent of the system. Any matrix that maps the full initial state to a new state should exhibit these same stability properties. Because the vector on the left hand side of equation 2.10 can be mapped from the initial conditions of all three variables, this new vector represents a state of the entire system that will demonstrate the same stability properties. Therefore by finding the matrix that maps this vector to the next such state, as in equation 2.11, important information can be gained.

$$\begin{pmatrix} x_{i+1} \\ x_{i+2} \\ x_{i+3} \end{pmatrix} = \mathbf{B} \begin{pmatrix} x_i \\ x_{i+1} \\ x_{i+2} \end{pmatrix} \tag{2.11}$$

To estimate the mapping matrix (**B**), the same techniques of linear regression will be applied for i = 1 to N-3. Although **B** is not the Jacobian, its eigenvalues should provide similar insight into the stability properties as those from the Regression Jacobian Method. This would

then allow researchers to use a single variable to analyze the stability properties of system for which they cannot access all of the variables, such as in human locomotion.

Stability Analysis

Once the matrices from the three different methods are constructed, they can be analyzed and compared. As described before, the eigenvalues of these matrices determine the stability properties. Of particular interest in this paper, is the fact that when a bifurcation point occurs, the magnitude of one of the eigenvalues approaches one. Specifically, when one of the eigenvalues approaches negative one, a period doubling occurs. By using the above methods over a range of control parameters, the eigenvalues can be plotted, and should demonstrate trends that would identify where bifurcation points occur. This would be very useful for studying gait transitions in the laboratory.

Chapter 3

Hatsopoulos Oscillator

The methods described in Chapter 2 were applied to a simple CPG-based model. The model, proposed by Hatsopoulos (1996) and further developed by Raftery, Cusumano and Sternad (2007), consists of a van der Pol oscillator driving a simple pendulum. The oscillator represents a CPG, or the neurological aspect of the model, while the pendulum represents the biomechanical aspect of the model and can be thought of as a swinging arm. The model was chosen to test these methods due to its simplicity, yet fundamental similarities to more complex walking models.

Description of Model

The Hatsopoulos oscillator follows the set of differential equations found below.

$$\ddot{y} + \varepsilon(y^2 - 1)\dot{y} + (\omega + Bx)^2 y = 0 \tag{3.1}$$

$$\ddot{x} + \lambda^2 x = Gy \tag{3.2}$$

Like was done in the 2007 Raftery, Cusumano, and Sternad paper, the substitutions in equations 3.3 and 3.4 were made to yield what was used in this study, equations 3.5 and 3.6.

$$\bar{t} = \lambda t \qquad \bar{x} = \frac{\lambda^2}{G} x \qquad \bar{y} = y \tag{3.3}$$

$$a = \frac{\varepsilon}{\lambda} \qquad b = \frac{\omega}{\lambda} \qquad c = \frac{BG}{\lambda^3} \tag{3.4}$$

$$\bar{y}'' + a(\bar{y}^2 - 1)\bar{y}' + (b + c\bar{x})^2 \bar{y} = 0 \tag{3.5}$$

$$\bar{x}'' + \bar{x} = \bar{y} \tag{3.6}$$

The system depends on three control parameters, a, b, and c. This study was conducted with fixed values for a and b of .0995 and .15 respectively. These are held constant in order to

study the effect of c, the coupling strength parameter, on the stability of the system. If c is set to .13, the system will approach a steady state limit cycle similar to what is seen in Figure 3-1.

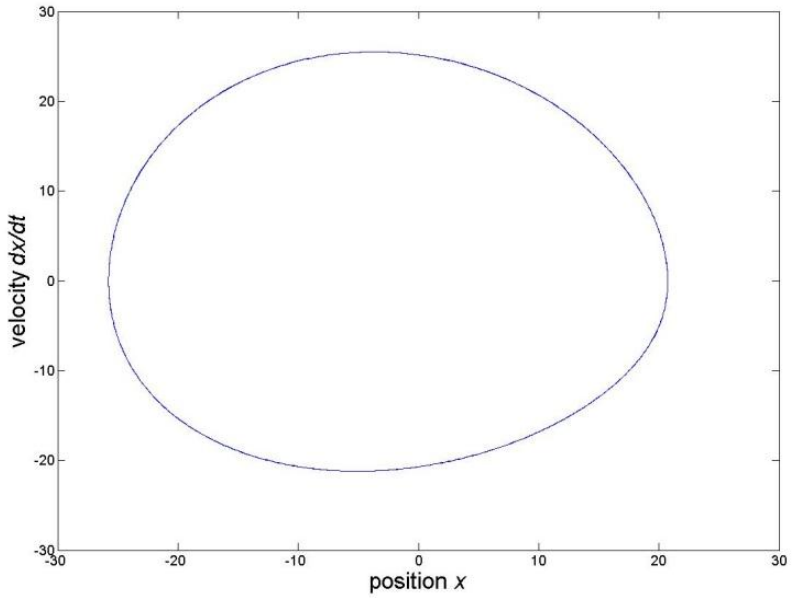

Figure 3-1 Position vs. Velocity at c = .13

The solution is fundamentally different when the coupling strength parameter is increased up to .15. Each cycle now follows two separate loops in the position versus velocity state space, as seen in Figure 3-3.

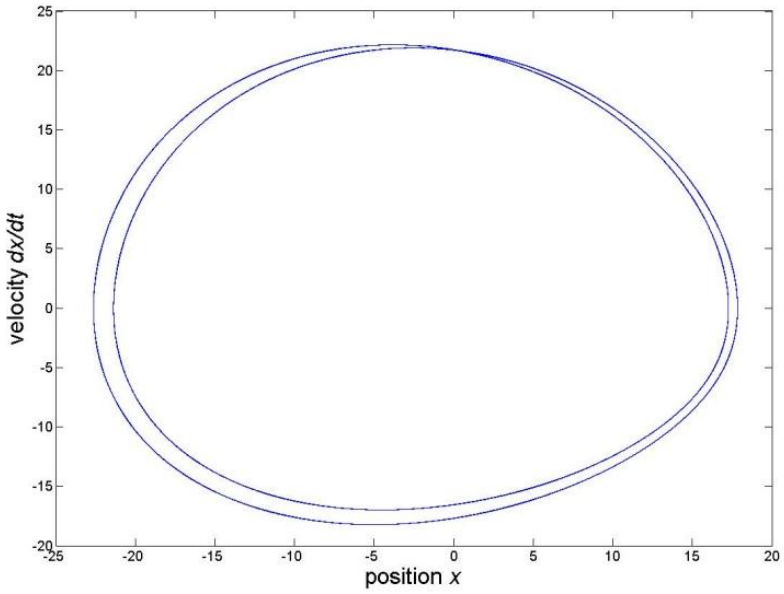

Figure 3-2 Position vs. Velocity at c = .15

As the parameter is further increased to .17, the loops are no longer distinct, but appear as a blur, as seen in Figure 3-4.

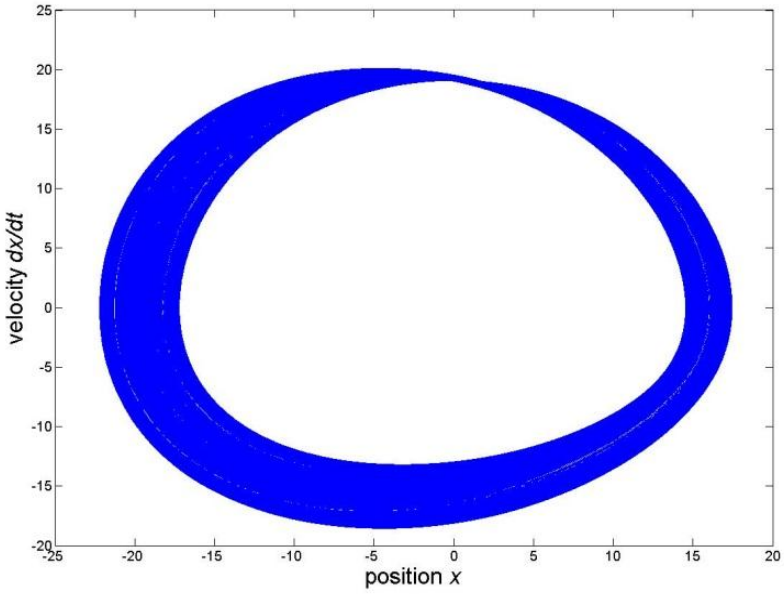

Figure 3-3 Position vs. Velocity at c = .17

Despite the appearance in the two dimensional space pictured, the system actually never overlaps itself. If you were able to plot these data points in the full four dimensional space of $(x, \dot{x}, y, \dot{y})$ there would be no intersecting lines. The system is deterministic, so each four dimensional data point yields one specific four dimensional data point after a certain time interval has passed.

Bifurcation Analysis

The analysis of the stability of the system begins with a bifurcation diagram. The x variable is tracked in this analysis because it is the biomechanical variable. This is the one that would be easily observable in a laboratory. In order to conduct the analysis described before, it was necessary to find the maxima of the x-data. For some c values, x maxima have almost exactly the same value as the system approaches steady state; yet for other c values, subsequent maxima can vary greatly. By plotting these maxima as a function of c, (Figure 3-4) the bifurcations of the system become easily observable.

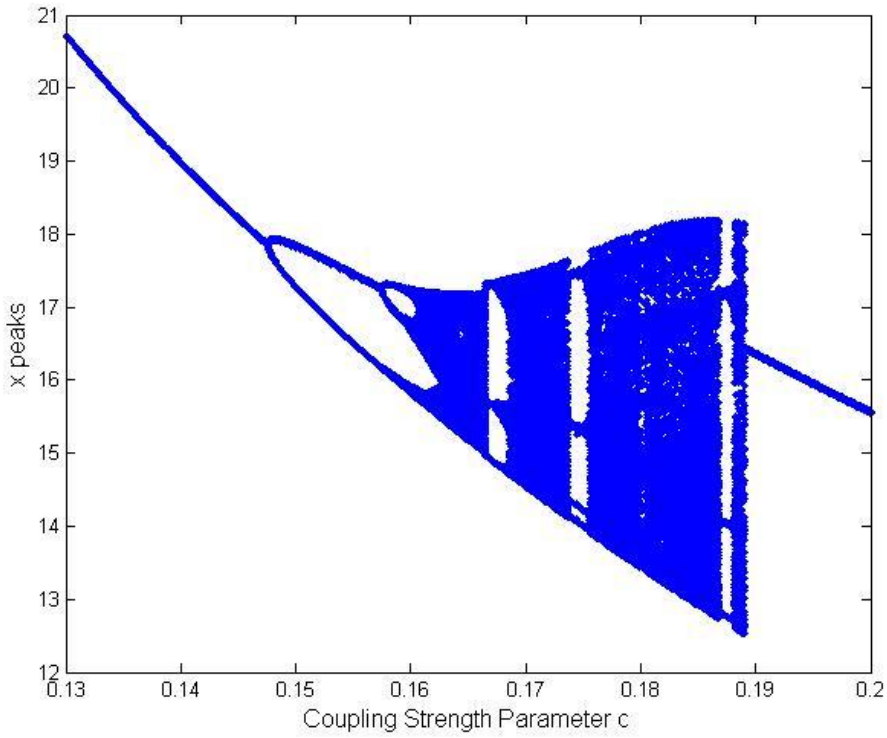

Figure 3-4 Bifurcation Diagram of Hatsopoulos Oscillator

This diagram agrees with what was seen in Figures 3-1 through 3-3. At a c value of .13, the system has a single maximum value. At .15, the system has two maximum values that it alternates between. Finally, at .17, the maxima are spread out and create a blur of data points. This diagram was created by collecting data over a t range of 1000 seconds, after the system had approached steady state, resulting in roughly 150 maxima at each c value. By using c values from .13 to .2 with a step of .0001, 700 c values were studied.

Because eigenvalues of the stability matrices described earlier will approach a magnitude of one as the system approaches a bifurcation point, the clear transition occurring between .14 and .15 was selected to test these methods. Here a clear period doubling occurs. Thus, if the

eigenvalues of the stability matrices are plotted over the c values leading up to this point, one of the eigenvalues should approach -1.

Numerical Solution of the Jacobian

The purpose of this method is to create an accurate estimate of the Jacobian of the deviations of the system from its steady state. By studying the x variable maxima, a Poincare Section of zero x velocity has been created. The resulting system is three dimensional in $(x, y, \dot{y})$. As seen below in equation 3.7, the desired Jacobian is a 3x3 matrix (**A**) that maps the deviations ($\underline{u}$) to the deviations of the next Poincare Section crossing ($\underline{u'}$).

$$\underline{u'} = \begin{pmatrix} u'_x \\ u'_y \\ u'_{\dot{y}} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix} \begin{pmatrix} u_x \\ u_y \\ u_{\dot{y}} \end{pmatrix} = \mathbf{A}(\underline{u}) \tag{3.7}$$

When only one of the three variables has a perturbation (ε), this equation becomes much simpler, as two columns of the Jacobian are irrelevant as proven in equation 3.8 below.

$$\begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix} \begin{pmatrix} \varepsilon \\ 0 \\ 0 \end{pmatrix} = \varepsilon \begin{pmatrix} A_{11} \\ A_{21} \\ A_{31} \end{pmatrix} \tag{3.8}$$

Therefore, equation 3.9 shows that the new state ($\underline{u'}$) found by using the one dimensional deviation above is equal to the first column of the Jacobian scaled by the magnitude of this deviation.

$$\begin{pmatrix} u'_x \\ u'_y \\ u'_{\dot{y}} \end{pmatrix} = \varepsilon \begin{pmatrix} A_{11} \\ A_{21} \\ A_{31} \end{pmatrix} \tag{3.9}$$

Thus, the first column of the Jacobian can be accurately estimated by finding $\underline{u'}$, and then dividing it by ε. The same method can be used with the other two variables to find the second and third columns of the Jacobian.

As discussed before, due to the results of the bifurcation diagram, it is expected that one of the eigenvalues of this Jacobian will approach -1 as the value of c approaches the bifurcation value of approximately .147. The Jacobian was found as described above, using an ε of .05, for every value of c between .13 and .147 with a Δc of .0001. The plot of the eigenvalues versus c for these Jacobians is displayed in Figure 3-5.

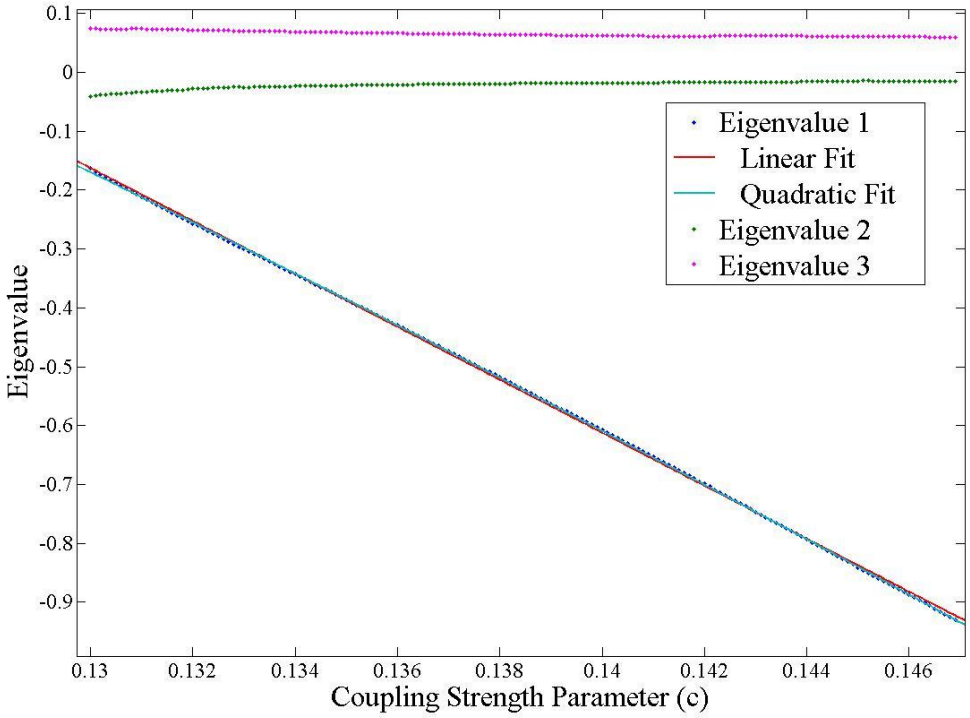

Figure 3-5 Eigenvalues vs. C of the Numerical Solution

As expected, one eigenvalue clearly approaches -1 as the system nears the known bifurcation point. The eigenvalues form a near linear plot towards this expected value, but both the linear and quadratic fit lines overlap extremely well with the data.

Regression Jacobian Method

Next, the Regression Jacobian Method was applied to the Hatsopoulos oscillator. In order to create the variability necessary for regression testing, a pseudo-random noise was added to the system. This noise is of the same maximum magnitude as the finite perturbation in the previous method (σ = .05). As discussed earlier, by collecting a large sample of data points of the system's subsequent passes through the Poincare Section, linear regression can be applied to estimate the Jacobian. Equation 3.10 displays the algebra of this method applied to this model.

$$\underline{u}' = \begin{pmatrix} u'_x \\ u'_y \\ u'_{\dot{y}} \end{pmatrix} = \mathbf{A} \begin{pmatrix} u_x \\ u_y \\ u_{\dot{y}} \end{pmatrix} = \mathbf{A}\underline{u} \tag{3.10}$$

Here, A is the 3x3 Jacobian matrix that maps u_{n-1} to u_n. Equation 3.11 summarizes how the regression is set up.

$$\underline{u}_{i+1} = \mathbf{A}\underline{u}_i \tag{3.11}$$

Linear regression estimated the matrix A using i = 1 to N-1, where N is the number of peaks in the collected data. As before, one of the eigenvalues of the Jacobian should approach -1 as c approaches the known bifurcation point. Regression was applied to estimate the Jacobian at every c value between .13 and .147 using a Δc of .0001. The plot of the eigenvalues versus c can be seen in Figure 3-6.

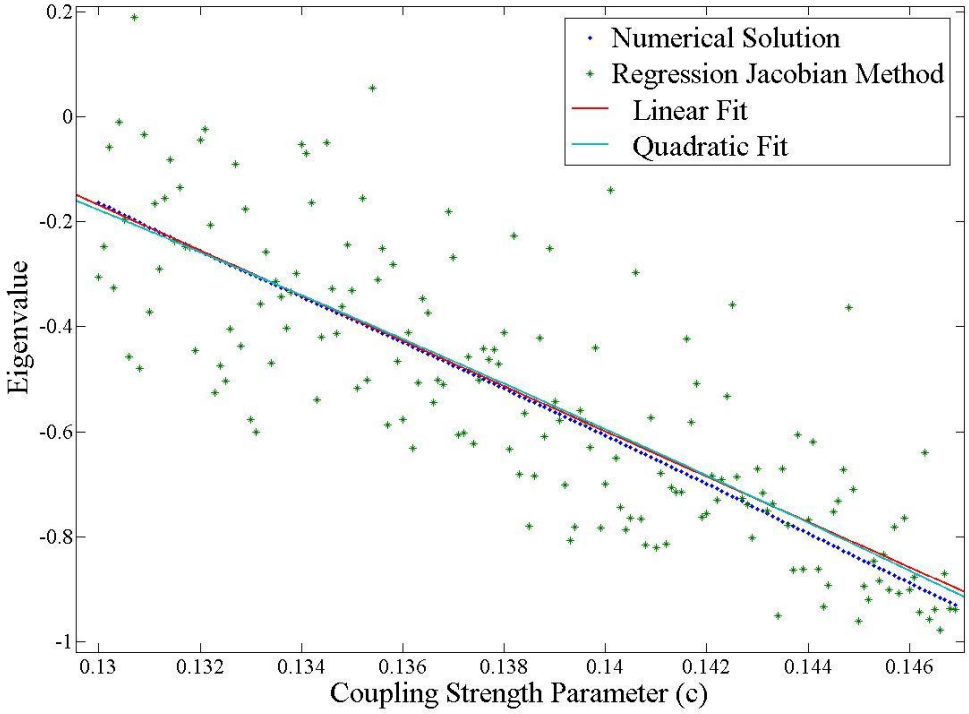

Figure 3-6 Eigenvalues vs. C by the Regression Jacobian Method

Though not as apparent as in the numerical solution, if a trend line is applied, one of the eigenvalues does still approach -1 near the bifurcation point. Neither the linear, nor the quadratic fit perfectly match the data from the numerical solution, however both match the data reasonably closely.

Delay Reconstruction Method

Finally, the Delay Reconstruction Method was applied to the Hatsopoulos system. The same pseudo-random noise of magnitude .05 was added to the system to conduct these tests. When applied to this particular system, the algebra is represented by equation 3.12.

$$\underline{u}_1 = \begin{pmatrix} u_{x1} \\ u_{x2} \\ u_{x3} \end{pmatrix} = \mathbf{B} \begin{pmatrix} u_{x0} \\ u_{x1} \\ u_{x2} \end{pmatrix} = \mathbf{B}\underline{u}_0 \tag{3.12}$$

The method involves the estimation of the matrix B by using linear regression as laid out in equation 3.13 with i ranging from 1 to N-3, if N is the number of peaks collected.

$$\underline{u}_{i+1} = \mathbf{B}\underline{u}_i \tag{3.13}$$

While B is not an approximation of the Jacobian of the system, it is hypothesized that the eigenvalues of this matrix can be used to find the stability transition points as the previous methods have. Regression was applied to estimate B at every c value between .13 and .147 using a Δc of .0001. The plot of eigenvalues versus c can be seen in Figure 3-7 below.

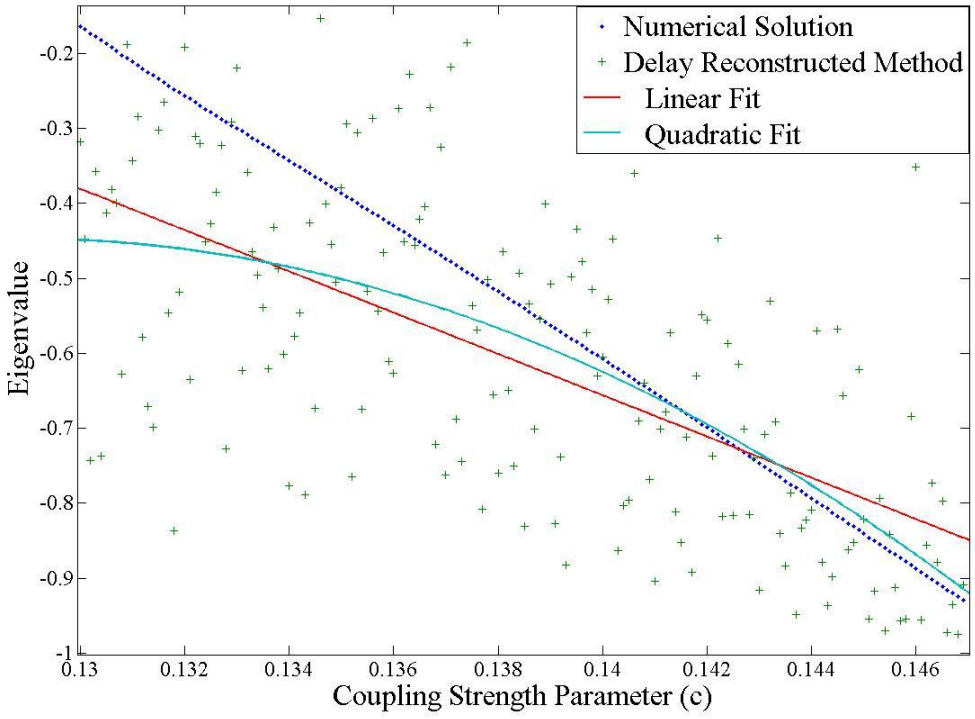

Figure 3-7 Eigenvalues vs. C by the Delay Reconstruction Method

This graph is significantly less definitive than the numerical solution, due to a high degree of spread in the data, but the eigenvalues still move towards -1 close to the correct value of c. While neither the linear, nor the quadratic fits match the numerical solution very closely, they both approach -1 in the vicinity of the bifurcation point. The quadratic fit does this slightly more accurately.

Chapter 4

Discussion and Conclusion

The results presented in Chapter 3 behaved in trends similar to what was expected, however there was more variability than anticipated. While the results of this study will not provide any definite conclusions, they do provide support for the hypotheses laid out earlier.

When conducting the numerical solution, the data resulted in smooth trends on the plot in Figure 3-5. This method involved a numerical estimate using finite perturbations from the steady state solution. Because of the absence of random noise in this method, the major question being asked is whether the system can be approximated as linear. Because they data formed smooth curves, and because one of the eigenvalues of the Jacobian approached -1 near the bifurcation point, the approximation of the Jacobian as a linear operator in this system of deviations from the limit cycle appears to be valid. The numerical solution both provides a benchmark for comparison, as well as supports the linear approximation used in the two regression methods.

The results for the Regression Jacobian Method were less conclusive. Figure 3-6 shows that the eigenvalues in this method had a significant spread when compared with the numerical solution. Some spread would be expected because the system contains random noise. This noise affects the values of the three measured variables at each x peak. Furthermore, and perhaps more importantly, the noise would also create a slight error in the identification of the Poincare section crossing, causing the other three variables to be measured at an incorrect time. Because the approximation that the Jacobian is linear for small perturbations was strongly supported by the numerical solution, the noise should be the primary reason for the poor precision. Despite the spread, the closeness with which the Regression Jacobian trend line follows the data from the numerical solution is very encouraging. These results suggest that individual measurements of the eigenvalues can be quite inaccurate; however, the trend of the Regression Jacobian

eigenvalues reflects the true behavior of the system. This implies that the method can be used to predict where bifurcations should occur, given enough data to draw trends.

Similarly, the results for the Delay Reconstruction Method were supportive, but not conclusive. Figure 3-7 shows that there is much greater spread in the data than in the Regression Jacobian Method. While similar inaccuracies would occur due to the noise added to the system, there is obviously something else affecting the results. One possible explanation is that the approximation of the stability matrix as a linear operator for a delay reconstructed system isn't as valid as the same approximation for the Jacobian. Regardless of the reason for the spread of this data, the trend lines still show similarity to the trend lines of the numerical solution. It is not nearly as accurate as the trend produced by the Regression Jacobian Method, but it still shows that the method has a degree of validity. The predicted bifurcation point by this method would be different; however this method point researchers close to the true bifurcation.

Looking forward, these methods need to be refined and then tested further to draw conclusive results about how useful they will be. All of the methods should be able to be improved by decreasing the time step in the data collection. A time step which allowed for test to finish in a reasonable amount of time was used. By using an infinitesimally small time step, the peaks will be precisely determined every time with the exception of the noise's effect. To keep the program execution speed but still improve results, the accurate peak detector could be revisited. The code fit parabolas to five data points to find the exact value of the peak. The number of data points used and the order of the polynomial used to fit the data could both be changed. Another key change that could be made is the use of a different type of noise. Different methods for how the noise is added could produce different results. One additional change that could be made is changing the order of the Delay Reconstruction Method. For this study, a 3x3 matrix was estimated because it was the same size as the Jacobians from the other two methods; but this doesn't have to be the case. The same method can be applied using a larger stability

matrix and vectors of a larger size, yet still following the same pattern. To summarize, there are several different possible approaches to improve these methods that have yet to be explored. This study merely provides a starting point.

Moreover, after the methods are better refined, they should be tested for different applications. The Hatsopoulos Oscillator explored in this study is far from a realistic model, it just displays similar properties. More realistic walking models, such as the one designed by Taga (1995) should be studied using the same methods. If the methods appear valid with those models as well, then the Delay Reconstruction Method should then be attempted on human data. The validation of these applications could drastically deepen the extent to which locomotion can be studied.

In conclusion, this study has laid a solid foundation for further work on this topic. Improvements need to be made to the implementation of the methods being studied, but initial results seem promising that useful results lie ahead. The validation of the Delay Reconstructed Method would allow researchers to further the understanding of how humans control their movement.

References

Azulay, J.-P., S. Mesure, et al. (1999). "Visual control of locomotion in Parkinson's disease." Brain(122): 111-120.

Blin, O., A. M. Ferrandez, et al. (1990). "Quantitatve analysis of gait in Parkinson patients: increased variability of stride length." Journal of the Neurological Sciences(98): 91-97.

Culham, E. G., M. Peat, et al. (1986). "Below-knee amputation: a comparison of the effect of the SACH foot and single axis foot on electromyographic patterns during locomotion." Prosthetics and Orhotics International(10): 15-22.

Cusumano, J. P. and P. Cesari (2006). "Body-goal variability mapping in an aiming task." Biological Cybernetics.

Dingwell, J. B. and J. P. Cusumano (2000). "Nonlinear time series analysis of normal and pathological human walking." Chaos 10(4): 848-863.

Dingwell, J. B. and J. P. Cusumano (2010). "Re-interpreting detrended fluctuation analyses of stride-to-stride variability in human walking." Gait and Posture.

Dingwell, J. B., J. John, et al. (2010). "Do Humans Optimally Exploit Redundancy to Control Step Variability in Walking?" PLoS Computational Biology 6(7).

Donker, S. F. and P. J. Beek (2002). "Interlimb coordination in prosthetic walking: effects of asymmetry and walking velocity." Acta Psychologica(110): 265-288.

Golubitsky, M., I. Stewart, et al. (1997). "A modular network for legged locomotion." Physica D(115): 56-72.

Hatsopoulos, N. G. (1996). "Coupling the Neural and Physical Dynamics in Rhythmic Movements." Neural Computation(8): 567-581.

Kantz, H. and T. Schreiber (2004). Nonlinear Time Series Analysis. Cambridge, Cambridge University Press.

Menz, H. B., S. R. Lord, et al. (2003). "Acceleration Patterns of the Head and Pelvis When Walking Are Associated With Risk of Falling in Community-Dwelling Older People." Medical Sciences 58A(5): 446-452.

Ocioso, G. (2008). Poincare Map. Poincare map.svg: Scheme of Poincare map. <http://en.wikipedia.org/wiki/File:Poincare_map.svg>

Raftery, A., J. Cusumano, et al. (2007). "Chaotic Frequency Scaling in a Coupled Oscillator Model for Free Rhythmic Actions." Neural Computation(20): 205-226.

Taga, G. (1994). "Emergence of bipedal locomtion through entrainment among the neuro-musculo-skeletal system and the environment." Physica D(75): 190-208.

Taga, G. (1995). "A model of the neuro-musculo-skeletal system for human locomotion." Biological Cybernetics(73): 97-111.

Taga, G., Y. Yamaguchi, et al. (1991). "Self-organized control of bipedal locomotion by neural oscillators in unpredictable environment." Biological Cybernetics(65): 147-159.

Tufillaro, N. B., T. Abbott, et al. (1992). An Experimental Approach to Nonlinear Dynamics and Chaos. Reading, Addison-Wesley Publishing Company.

Van de Crommert, H. W. A. A., T. Mulder, et al. (1997). "Neural control of locomotion: sensory control of the central pattern generator and its relation to treadmill training." Gait and Posture(7): 251-263.

Verdaasdonk, B. W., H. F. J. M. Koopman, et al. (2004). "Bifurcation and stability analysis in musculoskeletal systems: a study in human stance." Biological Cybernetics(91): 48-62.

Appendix A

MATLAB Code

Find Peaks

```
function peaks = FindPeaks(x,t)

%INPUT PARAMETERS
% x is a 4 column vector
%   The first column is the variable for which the peaks are to be found
%   The 2nd-4th columns contain the other three variables for which values
%       will be found at the precise time of the peak occurrence
%   i.e.  x(1,:) = [20.7231   -0.0006    1.1853   11.3269]
% t is a 1 column vector representing the times at which the measurements
%   for each value of x was measured
%   i.e.  t = [.01  .02  .03  .04 ... 1000]'

%OUTPUT PARAMETERS
% peaks will be a 6 column vector
%   Column 1 will be the indices of x for the data point nearest the peaks
%   Column 2 will be the time for the data point nearest the peaks
%   Columns 3:6 will be the value of the four variables of x at the peaks
%   i.e. peaks(1,:) = [250  2.5  20.7231   -0.0006    1.1853   11.3269]


%finds peak estimates by selecting the largest data points
[max,min]=peakdet(x(1:end,1),.01);

%Defines the starting and ending point based on the number of peaks
start = 2;
finish = length(max)-1;

%Defines variable sizes
peaks = zeros(finish-start + 1,6);

for i = start:finish
    %Selects the index of estimated peak
    index = max(i,1);

    %Compiles a data set including the 2 points before and after the
    %   estimated peak
    xdata = [x(index-2,1),x(index-1,1),x(index,1),x(index+1,1),x(index+2,1)];
    tdata = [t(index-2),t(index-1),t(index),t(index+1),t(index+2)];

    %Fits a parabola to the data
    f = PolyFit(tdata, xdata, 2);

    %Solves for the actual time of the maximum based on the 2nd variable in
    %   x being the time derivative of the first
    time = -f(2)/(2*f(1));
```

```
    %Enters data into the output parameter
    peaks(i-start+1,1) = index;
    peaks(i-start+1,2) = time;
    peaks(i-start+1,3) = polyval(f,time);

    %Finds the value of the other 3 variables at the peaks
    for j = 2:4
        %Compiles a data set including the 2 points before and after the
        %   estimated peak
        xdata = [x(index-2,j),x(index-1,j),x(index,j)...
                 x(index+1,j),x(index+2,j)];
        tdata = [t(index-2),t(index-1),t(index),t(index+1),t(index+2)];

        %Fits a parabola to the data
        f = PolyFit(tdata, xdata, 2);

        %Enters data into the output parameter
        peaks(i-start+1,j+2) = polyval(f,time);
    end
end
```

Numerical Solution of the Jacobian

```
function [matrices,eigenvalues] = NumericalSolution(x0,epsilon)

%INPUT PARAMETERS
% x0 is a 5 column vector
%   The 1st column contains values of c.
%   The 2nd-5th columns contain a near steady state data point at the
%       corresponding value of c.
%   i.e.  [0.1300   20.7231   -0.0006    1.1853   11.3269]
% epsilon is the size of the perturbation from the mean used to estimate
%   the Jacobian.
%   i.e.  .01

%OUTPUT PARAMETERS
% matrices will be a 3 Dimensional variable
%   It will be best understood as a list of 3 by 3 matrices.
%   These will be the Jacobians estimated by mapping how perturbations of
%       each variable affect the state variables after one cycle.
%   i.e. [1-Length, 1-3, 1-3]
% eigenvalues will be a 10 column vector describing the eigenvalues of the
%   Jacobians
%   The 1st column contains values of c.
%   The 2nd-4th columns contain the real parts of the eigenvalues
%   The 5th-7th columns contain the imaginary parts of the eigenvalues
%   The 8th-10th columns contain the magnitudes of the eigenvalues

%Defines variable sizes
Length = length(x0(:,1));
eigenvalues = zeros(Length,10);
matrices = zeros(Length,3,3);

%The Jacobian and its eigenvalues are estimated for each row of x0
for i = 1:Length
    c = x0(i,1);
    xstar = x0(i,2:5);
```

```
    %Creates an estimate of the Jacobian, using perturbations of epsilon
    A = NumericalJacobian(c,xstar,epsilon);
    matrices(i,1:3,1:3)=A;

    %Computes the eigenvalues of the Jacobian
    Eigs = sort(eigs(A));

    %Places the real and imaginary components and the magnitudes of the
    %   eigenvectors in the proper columns of the output vector.
    eigenvalues(i,1) = c;
    eigenvalues(i,2:4) = real(Eigs);
    eigenvalues(i,5:7) = imag(Eigs);
    eigenvalues(i,8:10) = abs(Eigs);
end
```

Numerical Jacobian

```
function A = NumericalJacobian(c,xstar,epsilon)

%INPUT PARAMETERS
% c is the value of the Coupling Strength Parameter
%   i.e.  c = .14
% xstar is a 4 dimensional vector representing the steady state at the
%   current c value
%   i.e.  xstar = [17.888  0  0.797  11.322]
% epsilon is the size of the finite perturbation to be used
%   i.e.  epsilon = .05

%OUTPUT PARAMETERS
% A will be a 3x3 matrix estimating the Jacobian of the system's deviations
%   from xstar at the given c value.

%perturbs xstar in the x direction
x1 = xstar + [epsilon 0 0 0];
%runs the system and finds the next xpeak
[t,x]=runhats(x1,0,10,.01,c,0);
peak1 = FindFirstPeak(x,t);
%subtracts xstar to find the deviation
u1 = peak1 - xstar;

%perturbs xstar in the y direction
x2 = xstar + [0 0 epsilon 0];
%runs the system and finds the next xpeak
[t,x]=runhats(x2,0,10,.01,c,0);
peak2 = FindFirstPeak(x,t);
%subtracts xstar to find the deviation
u2 = peak2 - xstar;

%perturbs xstar in the ydot direction
x3 = xstar + [0 0 0 epsilon];
%runs the system and finds the next xpeak
[t,x]=runhats(x3,0,10,.01,c,0);
peak3 = FindFirstPeak(x,t);
%subtracts xstar to find the deviation
u3 = peak3 - xstar;
```

```
%Scales the deviations by epsilon to solve for the columns of the Jacobian
A1 = u1/epsilon;
A2 = u2/epsilon;
A3 = u3/epsilon;

%builds the Jacobian from the separate columns
A(1,1) = A1(1);
A(2:3,1) = A1(3:4);
A(1,2) = A2(1);
A(2:3,2) = A2(3:4);
A(1,3) = A3(1);
A(2:3,3) = A3(3:4);
```

Regression Jacobian Method

```
function [matrices,eigenvalues] = RegressionJacobianMethod(x0,sigma)

%INPUT PARAMETERS
% x0 is a 5 column vector
%   The 1st column contains values of c.
%   The 2nd-5th columns contain a near steady state data point at the
%       corresponding value of c.
%   i.e.  [0.1300   20.7231   -0.0006    1.1853   11.3269]
% sigma is the magnitude of the noise to be added to allow for regression.
%   i.e.  .01

%OUTPUT PARAMETERS
% matrices will be a 3 Dimensional variable
%   It will be best understood as a list of 3 by 3 matrices.
%   These will be the Jacobians estimated by regression on perturbations
%       from the mean caused by noise.
%   i.e. [1-Length, 1-3, 1-3]
% eigenvalues will be a 10 column vector describing the eigenvalues of the
%   Jacobians
%   The 1st column contains values of c.
%   The 2nd-4th columns contain the real parts of the eigenvalues
%   The 5th-7th columns contain the imaginary parts of the eigenvalues
%   The 8th-10th columns contain the magnitudes of the eigenvalues

%Defines variable sizes
Length = length(x0(:,1));
eigenvalues = zeros(Length,10);
matrices = zeros(Length,3,3);

%The Jacobian and its eigenvalues are estimated for each row of x0
for i = 1:Length
    c = x0(i,1);
    xstar = x0(i,2:5);

    %Runs the oscillator, adding noise of magnitude sigma
    [t,x]=runhats(xstar,0,3000,.001,c,sigma);
    peaks=FindPeaks(x,t);

    %Creates an estimate of the Jacobian, using regression of matrices  with
    %noise
    A = RegressionJacobian(peaks(:,3),peaks(:,5),peaks(:,6));
    matrices(i,1:3,1:3)=A;
```

```
    %Computes the eigenvalues of the Jacobian
    Eigs = ReorderEigenvalues(eigs(A));

    %Places the real and imaginary components and the magnitudes of the
    %   eigenvectors in the proper columns of the output vector.
    eigenvalues(i,1) = c;
    eigenvalues(i,2:4) = real(Eigs);
    eigenvalues(i,5:7) = imag(Eigs);
    eigenvalues(i,8:10) = abs(Eigs);
end
```

Regression Jacobian

```
function A = RegressionJacobian(var1, var2, var3)


%INPUT PARAMETERS
% var1, var2, and var 3 are all column vectors representing the values of
% x, y, and ydot at every xpeak
%   i.e.  var 1 = [17.888  17.886 ... 17.889]'
%   i.e.  var 2 = [.797    .793   ...   .799]'
%   i.e.  var 3 = [11.322  11.324 ... 11.322]'

%OUTPUT PARAMETERS
% A will be a 3x3 matrix estimating the Jacobian of the system's deviations
%   from xstar at the given c value.

%finds the mean of each variable to represent the steady state solution
mean1 = mean(var1);
mean2 = mean(var2);
mean3 = mean(var3);

%defines the length of the deviation variables
Length = length(var1);
u1 = zeros(Length,1);
u2 = zeros(Length,1);
u3 = zeros(Length,1);

%calculates the deviations by subtracting the means
for i = 1:Length
    u1(i) = var1(i) - mean1;
    u2(i) = var2(i) - mean2;
    u3(i) = var3(i) - mean3;
end

%defines the size of the matrices to perform regression on
U0 = zeros(Length-1,3);
U1 = zeros(Length-1,3);

%builds the matrices  to perform regression on
for i = 1:(Length-1)
    U0(i,1) = u1(i);
    U0(i,2) = u2(i);
    U0(i,3) = u3(i);

    U1(i,1) = u1(i+1);
```

```
    U1(i,2) = u2(i+1);
    U1(i,3) = u3(i+1);
end

% The linear regression of U0\U1 is the Jacobian estimate A
A = (U0\U1)';
```

Delay Reconstruction Method

```
function [matrices,eigenvalues] = DelayReconstructionMethod(x0,sigma)

%INPUT PARAMETERS
% x0 is a 5 column vector
%   The 1st column contains values of c.
%   The 2nd-5th columns contain a near steady state data point at the
%       corresponding value of c.
%   i.e.  [0.1300   20.7231   -0.0006    1.1853   11.3269]
% sigma is the magnitude of the noise to be added to allow for regression.
%   i.e.  .01

%OUTPUT PARAMETERS
% matrices will be a 3 Dimensional variable
%   It will be best understood as a list of 3 by 3 matrices.
%   These will be the stability matrices estimated by the regression of
%       the delay reconstructed matrices.
%   i.e. [1-Length, 1-3, 1-3]
% eigenvalues will be a 10 column vector describing the eigenvalues of the
%   stability matrices
%   The 1st column contains values of c.
%   The 2nd-4th columns contain the real parts of the eigenvalues
%   The 5th-7th columns contain the imaginary parts of the eigenvalues
%   The 8th-10th columns contain the magnitudes of the eigenvalues

%Defines variable sizes
Length = length(x0(:,1));
eigenvalues = zeros(Length,10);
matrices = zeros(Length,3,3);

%The stability matrix and its eigenvalues are estimated for each row of x0
for i = 1:Length
    c = x0(i,1);
    xstar = x0(i,2:5);

    %Runs the oscillator, adding noise of magnitude sigma
    [t,x]=runhats(xstar,0,3000,.001,c,sigma);
    peaks=FindPeaks(x,t);

    %Creates a stability matrix using the regression of delay reconstructed
    %matrices of the variable x.
    A = DelayReconstruction3DRegression(peaks(:,3));
    matrices(i,1:3,1:3)=A;

    %Computes the eigenvalues of the Jacobian
    Eigs = ReorderEigenvalues(eigs(A));

    %Places the real and imaginary components and the magnitudes of the
    %   eigenvectors in the proper columns of the output vector.
```

```
    eigenvalues(i,1) = c;
    eigenvalues(i,2:4) = real(Eigs);
    eigenvalues(i,5:7) = imag(Eigs);
    eigenvalues(i,8:10) = abs(Eigs);
end
```

Delay Reconstruction Regression

```
function B = DelayReconstruction3DRegression(X)

%INPUT PARAMETERS
% X is a column vector of all of the x peaks

%OUTPUT PARAMETERS
% B will be a 3x3 matrix mapping the delay reconstruction of the deviations
%   from the steady state solution

%Finds the length and mean of the X vector
Length = length(X);
meanx = mean(X);

%Defines the size of the matrices on which to perform linear regression
X0 = zeros(Length-3,3);
X1 = zeros(Length-3,3);

%Builds the matrices on which to perform linear regression
for i = 1:Length-3
    X0(i,1)= X(i) - meanx;
    X0(i,2)= X(i+1) - meanx;
    X0(i,3)= X(i+2) - meanx;

    X1(i,1)= X(i+1) - meanx;
    X1(i,2)= X(i+2) - meanx;
    X1(i,3)= X(i+3) - meanx;
end
%uses linear regression to estimate the matrix that maps the delay
%   reconstruction
B = (X0\X1)';
```

Reorder Eigenvalues

```
function neweigs = ReorderEigenvalues(eigs)

%INPUT PARAMETERS
% eigs is the 3 dimensional vector of the eigenvalues of the Jacobian or
%   stability matrix being studied.
%   i.e. eigs = [-.675  .234  -.106]

%OUTPUT PARAMETERS
% new eigs will be the same eigenvalues but sorted.
%   If any of theeigenvalues have an imaginary part, the all real
%       eigenvalue will be first followed by the two with imaginary parts,
%       which are complex conjugates.
```

```
%   If there are not imaginary parts, the eigenvalues are sorted with the
%       most negative one being first and the most positive one last.
%   i.e. neweigs = [-.675  -.106  .234]

%defines the length of neweigs
neweigs = [0 0 0];

%Tests for imaginary parts to put the real value first, otherwise sorts
%   placing the most negative first.
for i=1:3
    if imag(eigs(i))==0 && neweigs(1)==0
        neweigs(1)=eigs(i);
    elseif imag(eigs(i))<0
        neweigs(2)=eigs(i);
    elseif imag(eigs(i))>0
        neweigs(3)=eigs(i);
    else
        neweigs = sort(eigs);
    end
end

%If the real part of the imaginary eigenvalues is greater than the all real
%   eigenvalue, then the all real value is placed last.
if real(neweigs(2))<real(neweigs(1))
    temp = neweigs(1);
    neweigs(1)=neweigs(2);
    neweigs(2)=neweigs(3);
    neweigs(3)=temp;
end
```

ACADEMIC VITA of Patrick J. Devlin

Patrick Devlin
5367 Highgrove Road
Pittsburgh, PA 15236
pjd5070@gmail.com

Education:

Bachelor of Science Degree in Engineering Science
The Pennsylvania State University
Schreyer Honors College
Graduation date: May 13, 2011
Thesis Title: Stability Analysis of a Simple CPG-Based Walking Model
Thesis Supervisor: Joseph P. Cusumano

Related Experience:

Internship with Westinghouse Electric Company
Managers: Robert Laubham, Michael Ruben
Summers of 2009 and 2010

Teaching Internship in the Department of Engineering Science and Mechanics
Supervising Professors: Osama O. Awadelkarim, Gary L. Gray
Fall 2010 and Spring 2011

Awards:

Patrick J. Dixon Scholarship in Engineering
Eileen & Vernon H. Neubert Award
Dean's List

Activities:

Society of Engineering Science
President 2010 – 2011
Member 2009 – 2011

Penn State IFC/Panhellenic Dance Marathon
Merchandise Captain 2010 – 2011
OPPerations Committee Member 2009 – 2010

Student Space Programs Laboratory
Science Team Member 2009 – 2011

Tang Soo Do Karate
Awarded 2nd Degree Black Belt 2006
Student 2001 – 2007