An Empirical Method for the Estimation of Eigenvalues of Cardiac Dynamical Systems From Time Series Data

2024-04-29
Luke Stinemetze, Andrew Patella, Josh Colgrove
Lecturer (all 3): Colin Mitchell

Abstract

Cardiac alternans are a specific class of cardiac arrhythmia characterized by beat-to-beat variation of the strength of a cardiac muscle contraction, detectable through variations in electrical impulses of heart muscle. These have been recognized as symptoms of ventricular fibrillation, a fatal heart condition. The behavior of the electrical impulses can be modeled as a dynamical system in which alternans appear as loss of stability of the system, indicated by the dominant eigenvalue. Due to the complex nature of cardiac muscle, there are potentially infinite variables that affect the system, making it difficult to predict cardiac behavior. This paper demonstrates how to predict future cardiac conditions by limiting the analysis to a few measurable variables and their respective eigenvalues. To implement this algorithm we calculated the stability of a cardiac system by modeling electrical impulses across gradually decreasing cycle lengths. Using our model we can show how the stability of the system changes with decreasing cycle lengths by analyzing these dominant eigenvalues. This analysis leads to the conclusion of either a healthy heart or a heart that is exhibiting cardiac alternans and entering into ventricular fibrillation.

Attribution

Andrew Patella: Lead programmer. Wrote the code to model varying cardiac cycles and predict the stability of the cardiac system.
Luke Stinemetze: Analyst. Provided essential information on the biological explanation of the cardiac system and helped write the report.
Josh Colgrove: Analyst. Supported in writing the report and provided mathematical explanations for the algorithms used.

Introduction

The beating of the human heart can be modeled as a dynamic system which remains stable in healthy individuals. When this system loses stability, cardiac arrhythmias–the irregular beating

of the heart–are exhibited and detectable. This work specifically examines a class of arrhythmia known as cardiac alternans, characterized by variation in the contraction strength of each beat at a constant heart rate. This condition is a known indicator of ventricular fibrillation, a heart rhythm condition that kills 350,000 people per year in America.

Cardiac rhythm instability is a result of the electrophysiology of the heart. When an electrical stimulus is applied to the heart cells, they respond by rapidly depolarizing the transmembrane voltage, called an action potential (AP), resulting in a muscular contraction. This is then followed by a repolarization of the voltage to the initial resting value. The time period when the transmembrane voltage is greater than the resting voltage is called the action potential duration (APD) and can be parsed according to the percent of repolarization (50%, 90%, etc.), as seen in Figure 1.

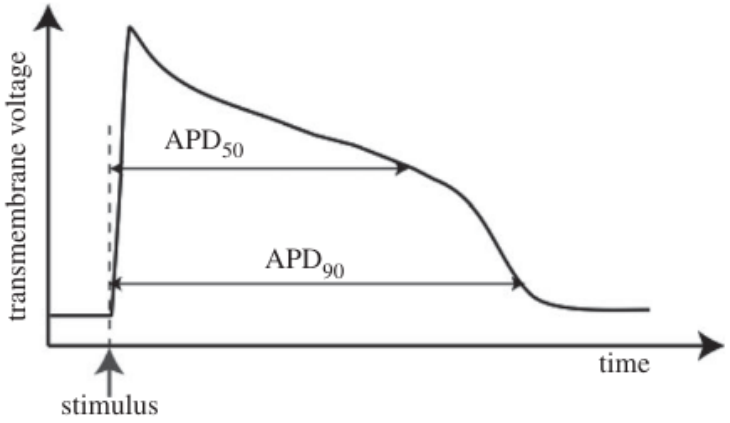

Figure 1: Graph demonstrating action potential cycle. APD_{50} corresponds to the duration of time spent at 50% percent repolarization, likewise for APD_{90}. (Petrie and Zhao. 2012.)

These parsed time periods are easily measurable and can accurately describe the behavior of the cardiac tissue. Data about the heart and its behavior comes primarily from electrocardiogram readings (ECG). This data comes as transmembrane voltage values over time, which does not incorporate the behavior of the dynamical system. The time between consecutive stimuli and the resulting depolarization is the basic cycle length (BCL). In a healthy heart, the action potential is a phase-locked 1:1 response, i.e. the contraction strength and BCL are consistent across each period and at a consistent frequency. Cardiac alternans occur when the action potential becomes a 2:2 phase-locked response and the BCL shortens.

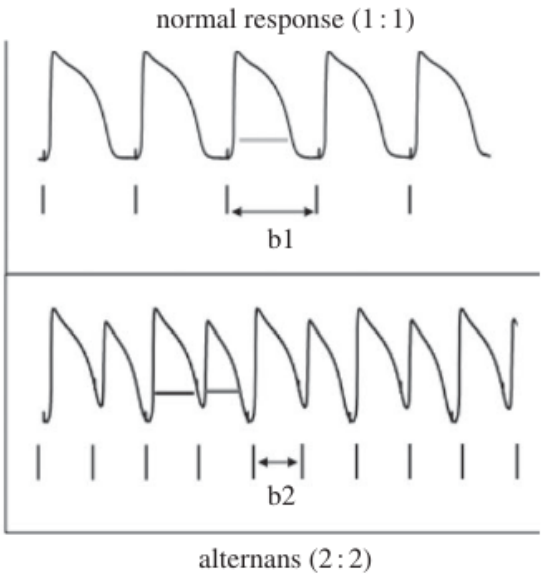

Figure 2: The vertical axis is transmembrane voltage, as in Figure 1. Alternans occur when the BCL becomes short enough that the period bifurcates. Normal responses to stimulus are long BCL (b1). Once the heart enters alternans, the BCL decreases (b2). During alternans, there are two APD lengths, one short and one long. Predicting the BCL that this bifurcation occurs at determines the onset of alternans. (Petrie and Zhao. 2012.)

The transmembrane voltage depolarizations occur with alternating amplitudes during each cycle, resulting in alternating long and short action potential durations of a specific percent of polarization, as shown in Figure 2. This period bifurcation happens as a result of destabilization of the dynamic system representing the heart.

The loss of stability of this system is indicated by when the most negative eigenvalue of the system approaches -1. Due to the complexity of the human heart, any mathematical model describing the system using equations may need potentially infinite degrees of freedom. As such, finding the eigenvalues analytically is not an option. To determine the eigenvalues of this undetermined system, an empirical method was developed that estimates the dominant eigenvalues of the system to predict the system's stability. This method avoids the use of an unspecifiable model and the need to specify all relevant state variables. This technique is based on the principle that when a steady system is mildly perturbed, the resulting dynamics can be predicted by only a small number of the most dominant eigenvectors.

The technique above can be used to predict cardiac alternans from action potentials (APs). These APs are measured as action potential durations (APDs). The measurable state variables are the time intervals corresponding to n% repolarization of the transmembrane voltage (APD_n). Once the eigenvalues have been found, they can be tracked with the basic cycle length (BCL) to determine when alternans will occur.

Mathematical formulation

Since the purpose of this exploration is to delve deeply into linear algebra techniques and methodologies, the error estimation of the eigenvalue was not surveyed, although it is a significant innovation of this paper. It involves topics more relevant to statistics and probability, rather than linear algebra and matrices in particular. Instead the estimation of the eigenvalue of the cardiac action potential system was explored and used to predict cardiac alternans. To begin describing the technique, a number of assumptions must be made regarding the nature of the systems and the data collected. First, the system is assumed to behave continuously, despite the data being collected as a stroboscopic time series. Second, since cardiac alternans do not drastically alter the behavior of the heart, the onset of symptoms does not take the system far from equilibrium. This means that the system can be described relatively accurately by only the dominant eigenvectors. Because the state vectors of the cardiac electrophysical dynamical system are potentially infinite, the state space of the system must be projected onto a pseudo-state space of the APDs at different percentages of repolarization. Due to the eigenvectors of the abbreviated system being approximately equal to the original system, the eigenvalues are also nearly equal to those of the dominant eigenvalues of the state space. The last two assumptions made are that the stable system (healthy heart) has constant pacing intervals and the number of collected variables is greater than the number of dominant eigenvalues of the system.

To determine the eigenvalues of the system we will use the principal component approach (PCA). This approach uses a least squares estimation of the eigenvalues based on the number of dominant eigenvalues, which is determined using the PCA along with the values of those eigenvalues. First to estimate the dominant eigenvalues we need to understand how the system interacts. We are assuming that the system can be estimated from a few dominant eigenvalues. Using the definitions of eigenvalues $Av = v\boldsymbol{\lambda}$, we can represent the system as a linear combination of each of the eigenvectors. $b = A + Av_1 + Av_2 + Av_3$. Where b is the nth APD and A represents the steady state APD. Each eigenvector, v, represents a disturbance in the system from a variable. This allows us to use a linear combination of the system to estimate the nth APD values. Where the nth APDs are a sum of the steady state (APD_*) plus each of the disturbances which is represented by the products of the various eigenvectors and eigenvalues.

$$APD_n = APD_* + v_{APD,1}\lambda_1^n + v_{APD,2}\lambda_2^n + v_{APD,3}\lambda_3^n \tag{4.1}$$

Equation 4.1 defines the influence of the eigenvalues on the stability of the system. However, in this system all of the values mentioned above are unknown except the steady state. This equation is built upon the assumption that there are only three dominant eigenvalues and requires

knowledge of these eigenvalues. Therefore, we need to do further mathematical manipulations, so that we can estimate the eigenvalues of the system.

For prediction of various APD values we can then rewrite equation 4.1 as:

$$\mathbf{y}_n = \mathbf{y}_* + C \cdot [\lambda_1, \lambda_2, \lambda_3]^T \tag{4.2}$$

To determine the APD for 90, 70, and 50% we then have

$$y_* = [APD_*^{90}, APD_*^{70}, APD_*^{50}]^T \tag{4.3}$$

and

$$C = \begin{bmatrix} v_{APD_{90},1} & v_{APD_{90},2} & v_{APD_{90},3} \\ v_{APD_{70},1} & v_{APD_{70},2} & v_{APD_{70},3} \\ v_{APD_{50},1} & v_{APD_{50},2} & v_{APD_{50},3} \end{bmatrix} \tag{4.4}$$

Further development of the equation leads us to the form

$$\mathbf{y}_n - \mathbf{y}_* = D \cdot (\mathbf{y}_{n-1} - \mathbf{y}_*) \tag{4.5}$$

Where

$$D = C \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} C^{-1} \tag{4.6}$$

However, using the definition of similar matrices we can estimate the matrix D using measurements $\{y_n\}_{n=1}^N$. From the following matrices:

$$Y_1 = [\mathbf{y}_1 - \mathbf{y}_* \cdots \mathbf{y}_{n-1} - \mathbf{y}_*] \quad \text{and} \quad Y_2 = [\mathbf{y}_2 - \mathbf{y}_* \cdots \mathbf{y}_n - \mathbf{y}_*] \tag{4.7,4.8}$$

Y1 and Y2 can be substituted into equation 4.4.

$$Y_2 = D \cdot Y_1 \Rightarrow D = (Y_2 \cdot Y_1) \cdot (Y_1 \cdot Y_1^T)^{-1} \tag{4.9}$$

Therefore if we know *n* APD values we can construct a matrix Z such that

$$\text{Z}= \begin{bmatrix} \mathbf{y}_1^{(1)} - \mathbf{y}_*^{(1)} & \mathbf{y}_1^{(2)} - \mathbf{y}_*^{(2)} & \cdots & \mathbf{y}_1^{(r)} - \mathbf{y}_*^{(r)} \\ \mathbf{y}_2^{(1)} - \mathbf{y}_*^{(1)} & \mathbf{y}_2^{(2)} - \mathbf{y}_*^{(2)} & \cdots & \mathbf{y}_2^{(r)} - \mathbf{y}_*^{(r)} \\ \vdots & \vdots & & \vdots \\ \mathbf{y}_n^{(1)} - \mathbf{y}_*^{(1)} & \mathbf{y}_n^{(2)} - \mathbf{y}_*^{(2)} & \cdots & \mathbf{y}_n^{(r)} - \mathbf{y}_*^{(r)} \end{bmatrix} \tag{4.10}$$

This matrix is the mathematical model of the system, but in reality, this is hard to accomplish in experimentation to do these calculations. Because this model is designed for the sake of experiments with limited data, *Z* can be rewritten based on the values of $\mathbf{y}_i^m$, which can be measured by an electrocardiogram. This is what allows all of the math in this section to be performed. Ultimately this model eliminates extraneous variables and simplifies the system to physical quantities that can be measured and analyzed. *Z* can be written using this knowledge as

$$Z = \begin{bmatrix} APD_1^{90} - APD_*^{90} & APD_1^{80} - APD_*^{80} & \cdots & APD_1^{40} - APD_*^{40} \\ APD_2^{90} - APD_*^{90} & APD_2^{80} - APD_*^{80} & \cdots & APD_2^{40} - APD_*^{40} \\ \vdots & \vdots & & \vdots \\ APD_n^{90} - APD_*^{90} & APD_n^{80} - APD_*^{80} & \cdots & APD_n^{40} - APD_*^{40} \end{bmatrix} \tag{4.11}$$

Then taking the dominant components of Z we get a matrix $\overline{Z}$. The dominant components of *Z* are defined by the user, but can be set within a tolerance. By taking the Singular Value Decomposition of *Z*, we were able to reduce Z in dimension and zero out terms depending on which singular values get eliminated. This matrix can be used to reformulate Y_1 and Y_2, which can be used in equation 4.8 to solve for *D*.

$$Y_1 = [\overline{\mathbf{z}}_1', \cdots, \overline{\mathbf{z}}_{n-1}'], \quad Y_2 = [\overline{\mathbf{z}}_2', \cdots, \overline{\mathbf{z}}_n'] \tag{4.12}$$

$$D = (Y_2 \cdot Y_1) \cdot (Y_1 \cdot Y_1^T)^{-1} \tag{4.13}$$

The eigenvalues of D are thus approximately the dominant eigenvalues of the system. The method of conditioning out the values that are not dominant eigenvalues eliminates the eigenvalues that have little to no effect on the system and allows for simple analysis of the stability of the heart as a dynamical system.

Examples and numerical results

Due to limited access to transmembrane voltage data sets we created a randomized data set based upon common transmembrane voltages and BCLs. While this won't be accurate data to represent real heartbeat data it will be realistic enough for us to analyze the usefulness of our model. To implement our model we want to analyze how the system changes across varying BCLs. Because of this, many results may be inaccurate due to assumptions made during the data creating process. A fit was created to an image from the Petrie paper, and this data was then shaped and conditioned to be within reasonable ranges. This created transmembrane voltage data for one heartbeat, as seen in Figure 3. Figure 4 maps each percent of repolarization to its corresponding action potential duration. We assumed that this

heartbeat was the steady state heartbeat, and a constant pacing heartbeat that repeats this beat is the steady state system. From this assumption we calculated the steady state values $\{APD_*^{90}, APD_*^{80} \cdots, APD_*^{40}\}$ for use in the calculation of $\mathbf{y}_*$.

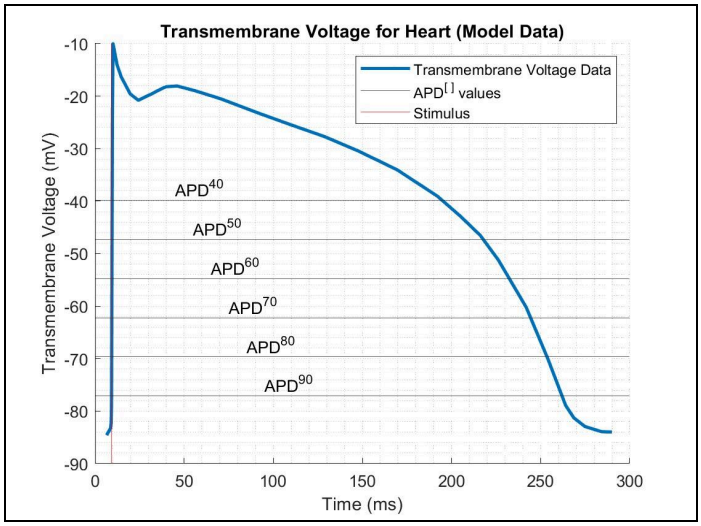

Figure 3: The steady state standard beat, one beat is shown

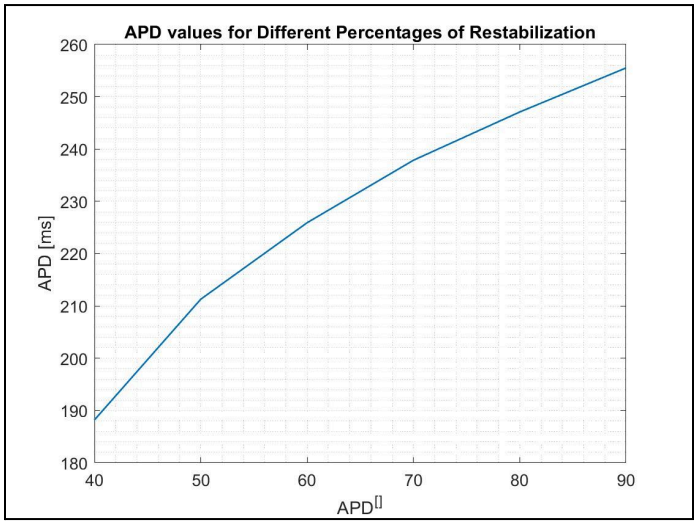

Figure 4: Values of percent repolarization of the original single heartbeat, which are assumed to be the values of the steady state of the system

From this data, we could repeat this heartbeat as many times as desired, to simulate measurements of the transmembrane voltage over n beats. In the MATLAB script, the sample heart data was randomized and repeated to create realistic heart voltage data. The magnitude of randomization depends on the magnitude of the voltage to incorporate measurement inaccuracies of physical systems. The times associated with the

transmembrane voltage were also randomized slightly to allow for control of BCL. These modifications yield the graph in Figure 5.

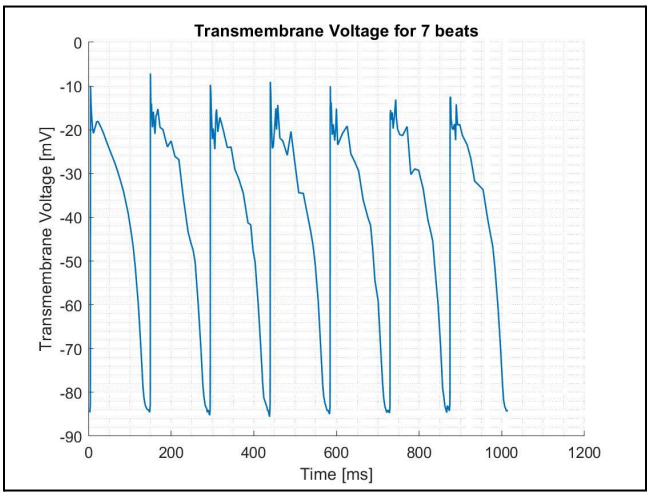

Figure 5: Model with randomized Voltages across a standard BCL: 7 Beats of randomized heart rate data to analyze for alternans.

With this data, we were able to start formulating the matrix Z to analyze for alternans. Using similar methods, we were also able to artificially shorten the BCL value and compare the eigenvalues of the system, as seen in Figure 6. This figure shows all of the different heart rate data superimposed to demonstrate the difference. The BCL decreases from around 292 milliseconds to around 141 milliseconds. This artificial bifurcation allows us to analyze the model and see the impact of the eigenvalues of the system on the stability. This also confirms that the MATLAB script accurately recreates the experiments performed in the Petrie paper.

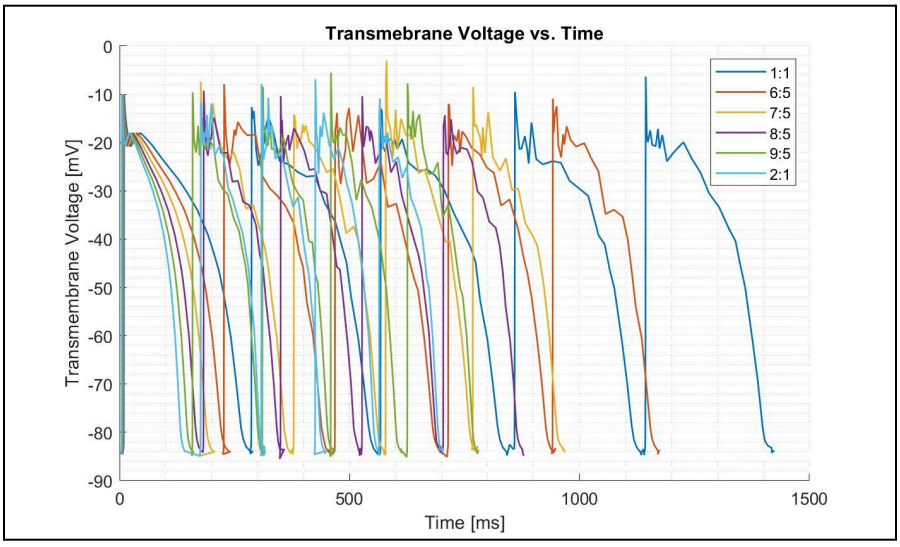

Figure 6: This graph shows our randomized Action Potentials. The length of the BCLs are shown decreasing from a steady state to a bifurcation BCL. The 1:1 data represents the standard, healthy system with constant pacing. The 2:1 data represents a bifurcated, unstable system that may be exhibiting alternan behavior.

Since the bifurcation is associated with the period length, the eigenvalues of the system can be determined for each period length to determine when the period bifurcation and in turn the alternans will occur. Repeating the process of calculating the eigenvalues of the system for all of the different BCL values allows us to determine the relation of the eigenvalue of the system to the BCL. Figures 7 and 8 show the trend of the eigenvalues across different cycle lengths.

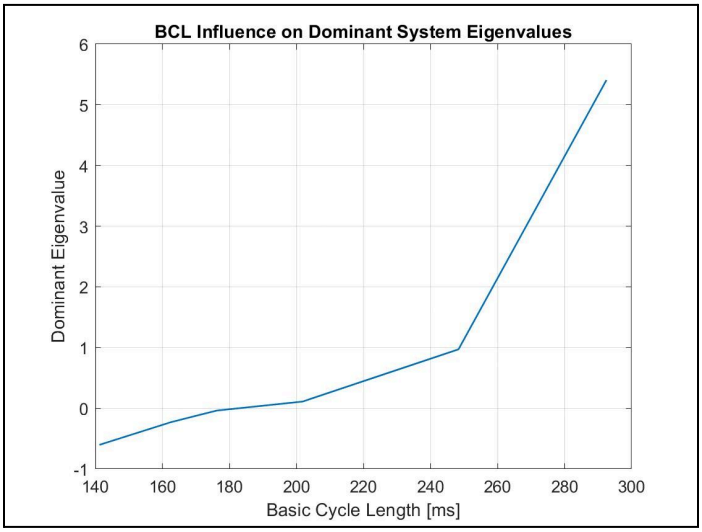

Figure 7: The graph shows the eigenvalues of the system based on the BCL value.

BCL	Eigenvalues
141.2379	-0.6033
162.67	-0.228
176.29	-0.038
201.85	0.1098
248.3908	0.9711
292.52	5.407

Figure 8: This table has the basic cycle lengths and eigenvalues from the graph. As the BCL decreases, the eigenvalue decreases to approach -1.

The system is determined to be unstable based on the behavior of the eigenvalues as the BCL decreases. When the system decreases to half of its original BCL, the eigenvalue approaches -1. For the tests performed, the eigenvalue reached -0.6033. This trend indicates a lack of stability in the system and suggests that alternans will become prevalent as the BCLs decrease. This correlates with literature of ventricular fibrillation analyses and the analysis of Pietre and Zhao.

Discussion and Conclusions

In likelihood, this program and practice would be more useful for actual heart data from an ECG. The main indicator of instability is a decreasing BCL over hundreds of heartbeats. The BCL value would be decreasing over time, not just jumping to different discrete ratios. This program could still be useful to analyze the stability of a heart, because any ECG data can be inputted and analyzed for a stable or unstable heart beat. This utility extends beyond the detection of cardiac alternans. Not only can it be used to analyze and diagnose other forms of cardiac arrhythmia, but this technique can be applied to any application where the only data available is a time series of variables and the system is dynamical.

Our model has not been used to test real patient cardiac data, so the actual validity and precision of this model has not been completely validated. In order to completely validate this model, we would like to analyze real data from a patient without any alternans, and real data from a patient with alternans. This could help elucidate the mechanics of the technique. Further interrogation of non-alternans systems would help to provide understanding of the necessary components of the system and what might need to be changed to utilize the system in a variety of applications.

This principle component approach can also be modified to aid in analysis of other dynamical systems, such as a spring pendulum system. Many systems have dramatic impacts on users, such as the heart, where the behavior and stability can indicate the existence of life threatening disorders. Ultimately, this method proves that helpful stability analysis and predictions of these complex systems is possible through accurate data collection and a computer program, like the one we generated.

Reference

Petrie, A., & Zhao, X. (2012). Estimating eigenvalues of dynamical systems from time series with applications to predicting cardiac alternans. *Proceedings of the Royal Society A, 468*, 3649–3666. https://doi.org/10.1098/rspa.2012.0098

Appendix

MATLAB Code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% matrix_project.m is designed to take in a shape of a tmv graph and
% condition it to reasonable values. Then it calculates APD vales and
% outputs the conditioned data and APD values to be used in later
% functions.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
close all; clear; clc;
%Reading in data
data = readmatrix("APD_graph.xlsx");
time = data(:,1);
v = data(:,2);
%Scaling and conditioning to match image (Approximate)
time = time*290/(max(time));
v = v*75/abs(max(v));
v = v-85;
%% Calculating APD values
first = 4;
last = 9;
% APD^[] values that you wish to measure
values = first:1:last;
%The change in voltage across a heartbeat
v_diff = abs(max(v)-min(v));
%Splitting data into rise and fall for ease
[peakV,peaki] = max(v);
riseV = v(1:peaki);
riseT = time(1:peaki);
fallV = v(peaki+9:end);
fallT = time(peaki+9:end);
t_stim = time(1);
% Calculating different APD values
for i = 1:length(values)
   v_apd{values(i)} = max(v)-values(i)/10*v_diff;
   t1{values(i)} = interp1(riseV,riseT,v_apd{values(i)});
   t2{values(i)} = interp1(fallV,fallT,v_apd{values(i)});
   APDv{values(i)} = t2{values(i)}-t_stim;
end
```

```
for i = first:last
   APD(i-(first-1)) = APDv{i};
end
csvwrite("tmv.csv",v)
csvwrite("time.csv",time)
csvwrite("APD_ss",APD);

% Plots
figure()
hold on;
grid minor;
plot(time,v,"Linewidth",2);
xlabel("Time (ms)");
ylabel("Transmembrane Voltage (mV)");
for i = first:last
   yline(v_apd{i});

end
xline(t1{i},"Color",'r');
text(45,v_apd{4}(1,1)+2.5,"APD^{40}");
text(55,v_apd{5}(1,1)+2.5,"APD^{50}");
text(65,v_apd{6}(1,1)+2.5,"APD^{60}");
text(75,v_apd{7}(1,1)+2.5,"APD^{70}");
text(85,v_apd{8}(1,1)+2.5,"APD^{80}");
text(95,v_apd{9}(1,1)+2.5,"APD^{90}");
title("Transmembrane Voltage for Heart (Model Data)")
legend('Transmembrane Voltage Data','APD^{[ ]}
values','','','','','','Stimulus')
figure()
plot(10*values,APD,'linewidth',1);
grid minor;
xlabel('APD^{[]}');
ylabel('APD [ms]')
title("APD values for Different Percentages of Restabilization")


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This script focuses on recreating multiple pulses for different
% heartbeats and analyzing the eigenvalues of that system. The heartbeats
% are synthesized by random numbers and are inherently biased but should be
% releatively useful.
% THIS ONE IS DIFFERENT BECAUSE IT VARIES CYCLE LENGTH (BCL)
% LIST OF ASSUMPTIONS THAT MAY BE (PROBABLY ARE) FALSE
% - Steady state value can be approximated by the mean
% - Every beat is the same length
% - The data we have is reasonably accurate
% Change first and last to make sure that the APD values are matching- also
% divide by ten. Change the number of heartbeats to make sure you are
% matching their assumptions. Currently at n=7 beats (>6) and 6 APD values
```

```
% from 40 to 90 (first = 4, last = 9).
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
close all; clc; clear;
n = 6; % Number of heartbeats
% APD^[] values that you wish to measure (x/10)
first = 4;
last = 9;
% Number of Dominant Compnents
m = 5;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Read in data from last script which conditioned the data to be within
%reasonable values. From "matrix_project.m"
tmvm = readmatrix("tmv.csv");
time = readmatrix("time.csv");
APD_ss = readmatrix("APD_ss");
time = time/2; %Change this to artificially vary BCL (2=bifurcation)
values = first:1:last;
tmv_tot = tmvm;
t_tot = time;
duration_beat = time(end);
endIndex = 1;
magnitude = abs(max(tmvm)-min(tmvm));
num_vals = last-first;
%Prealocating
Z = zeros(n,num_vals);
peakV = zeros(n,1);
peaki = zeros(n,1);
riseV = cell(n,1);%
riseT = cell(n,1);
fallT = cell(n,1);
fallV = cell(n,1);
t2 = cell(last,1);
tmv = cell(1,n);
v_apd = cell(1,last);
v_diff = zeros(n,1);
APDsc = cell(1,last);
APDs = zeros(num_vals,1);
t_stim = zeros(1,n);
APDv = cell(1,last);
Y1 = zeros(n,m);
Y2 = zeros(n,m);
%% Creating the random heartbeats
for i = 1:n
   for j = 1:size(tmvm)
       tmv{i}(j,1) = tmvm(j)+0.05*abs(magnitude+tmvm(j))*randn(1,1);
       t{i}(j,1) = time(j) + 0.01*time(j)*randn(1,1);

   end
   time = time+duration_beat;
```

```
    tmv_tot = cat(1,tmv_tot,tmv{i});
    t_tot = cat(1,t_tot,time);
end
figure()
hold on;
plot(t_tot,tmv_tot,"Linewidth",1);
xlabel("Time [ms]");
ylabel("Transmembrane Voltage [mV]");
title(sprintf("Transmembrane Voltage for %g beats",n+1));
grid minor;
%% Analyzing TMV data for APD values
% Outer loop iterates through each heartbeat
for j = 1: n
    %Splitting the individual beats into rise and fall
    [peakV(j),peaki] = max(tmv{j});
    riseV{j} = tmv{j}(endIndex:peaki);
    riseT = time(endIndex:peaki);
    fallV{j} = tmv{j}(peaki+9:end);
    fallT = time(peaki+9:end);
    endIndex = endIndex+ size(tmvm,1);
    v_diff(j) = abs(max(tmv{j})-min(tmv{j}));
    % Iterating through the different values of APD desired
    for i = 1:length(values)
        v_apd{values(i)} = peakV(j)-values(i)/10*v_diff(j);

        t2{values(i)} = interp1(fallV{j},fallT,v_apd{values(i)});
        t_stim(j) = t_tot(endIndex+1);

        APDv{values(i)} = t2{values(i)}-t_stim;
    end
end
%% Forming the matrix, SVD
y_star = APDs;
for i = 1:n
    for j = first:last
        Z(i,j-(first-1)) = APDv{j}(i)-APD_ss(j-(first-1)); % i is beat number, j
is APD value
    end
end
Z = flip(Z,2);
% Singular Value Decomposition of Z
[U,S,V] = svd(Z,0);  % MAYBE ACTUALLY DO THIS? Z = USV'
% zb should be mxN, N=n=#beats
Zb = Z(:,1:m);
%Calculating Y1 and Y2
for i = 1:n-1
    Y1(i,:) = Zb(i,:)';
end
for i = 2:n
```

```
   Y2(i,:) = Zb(i,:)';
end
Y1 = Y1';
Y2 = Y2';
%% Calculating D
D = (Y2*Y2')*inv(Y1*Y1');
% Calculating the eigenvalues of the system
eigenvalue = eig(D);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This script focuses on recreating multiple pulses for different
% heartbeats and analyzing the eigenvalues of that system. The heartbeats
% are synthesized by random numbers and are inherently biased but should be
% releatively useful.
% THIS ONE IS DIFFERENT BECAUSE IT VARIES CYCLE LENGTH (BCL)
% LIST OF ASSUMPTIONS THAT MAY BE (PROBABLY ARE) FALSE
% - Steady state value can be approximated by the mean
% - Every beat is the same length
% - The data we have is reasonably accurate
% Change first and last to make sure that the APD values are matching- also
% divide by ten. Change the number of heartbeats to make sure you are
% matching their assumptions. Currently at n=7 beats (>6) and 6 APD values
% from 40 to 90 (first = 4, last = 9).
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
close all; clc; clear;
n = 4; % Number of heartbeats (this will graphically add one more no se)
% APD^[] values that you wish to measure (x/10)
first = 4;
last = 9;
% Number of Dominant Compnents
m = 5;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Read in data from last script which conditioned the data to be within
%reasonable values. From "matrix_project.m"
tmvm = readmatrix("tmv.csv");
time = readmatrix("time.csv");
APD_ss = readmatrix("APD_ss");
time = time/1;
values = first:1:last;
tmv_tot = tmvm;
t_tot = time;
duration_beat = time(end);
endIndex = 1;
magnitude = abs(max(tmvm)-min(tmvm));
num_vals = last-first;
```

```
%Prealocating
Z = zeros(n,num_vals);
peakV = zeros(n,1);
peaki = zeros(n,1);
riseV = cell(n,1);%
riseT = cell(n,1);
fallT = cell(n,1);
fallV = cell(n,1);
t2 = cell(last,1);
tmv = cell(1,n);
v_apd = cell(1,last);
v_diff = zeros(n,1);
APDsc = cell(1,last);
APDs = zeros(num_vals,1);
t_stim = zeros(1,n);
APDv = cell(1,last);
Y1 = zeros(n,m);
Y2 = zeros(n,m);
BCL = zeros(n-1,1);
%% Creating the random heartbeats
for i = 1:n
   for j = 1:size(tmvm)
       tmv{i}(j,1) = tmvm(j)+0.05*abs(magnitude+tmvm(j))*randn(1,1);
       t{i}(j,1) = time(j)+ 10*randn(1,1)^2;
       %t{i}(j,1) = time(j) - time(j)/1.5;
   end
   time = time + t{i}(end,1) - t{i}(1,1);
   tmv_tot = cat(1,tmv_tot,tmv{i});
   t_tot = cat(1,t_tot,time);
end
figure()
hold on;
plot(t_tot,tmv_tot,"Linewidth",1);
xlabel("Time [ms]");
ylabel("Transmembrane Voltage [mV]");
title(sprintf("Transmembrane Voltage for %g beats",n+1));
grid minor;
%% Analyzing TMV data for APD values
% Outer loop iterates through each heartbeat
for j = 1: n
   %Splitting the individual beats into rise and fall
   [peakV(j),peaki] = max(tmv{j});
   riseV{j} = tmv{j}(endIndex:peaki);
   riseT = time(endIndex:peaki);
   fallV{j} = tmv{j}(peaki+9:end);
   fallT = time(peaki+9:end);
   endIndex = endIndex+ size(tmvm,1);
   v_diff(j) = abs(max(tmv{j})-min(tmv{j}));
   % Iterating through the different values of APD desired
```

```matlab
   for i = 1:length(values)
       v_apd{values(i)} = peakV(j)-values(i)/10*v_diff(j);

       t2{values(i)} = interp1(fallV{j},fallT,v_apd{values(i)});
       t_stim(j) = t_tot(endIndex+1);

       APDv{values(i)} = t2{values(i)}-t_stim;
   end
end
%% Forming the matrix, SVD
y_star = APDs;
for i = 1:n
   for j = first:last
       Z(i,j-(first-1)) = APDv{j}(i)-APD_ss(j-(first-1)); % i is beat number, j
is APD value
   end
end
Z = flip(Z,2);
% Singular Value Decomposition of Z
[U,S,V] = svd(Z,0);  % MAYBE ACTUALLY DO THIS? Z = USV'
% zb should be mxN, N=n=#beats
Zb = Z(:,1:m);
%Calculating Y1 and Y2
for i = 1:n-1
   Y1(i,:) = Zb(i,:)';
end
for i = 2:n
   Y2(i,:) = Zb(i,:)';
end
Y1 = Y1';
Y2 = Y2';
%% Calculating D
D = (Y2*Y2')*inv(Y1*Y1');
% Calculating the eigenvalues of the system
eigenvalue = eig(D)
for i = 1:size(t_stim,2)-1
   BCL(i) = t_stim(i+1)-t_stim(i);
end
BCL = mean(BCL)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% analysis.m just takes in the data from the other functions and plots the
% data.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
close all; clear; clc;
%% Plot of tmv vs time
```

```
data = readmatrix("APD_analysis.xlsx","Sheet","Sheet1");
tmv1 = data(:,1);
tmv2 = data(:,3);
tmv3 = data(:,5);
tmv4 = data(:,7);
tmv5 = data(:,9);
tmv6 = data(:,11);
time1 = data(:,2);
time2 = data(:,4);
time3 = data(:,6);
time4 = data(:,8);
time5 = data(:,10);
time6 = data(:,12);
%% Graph showing difference in tmv based on BCL
figure()
hold on;
plot(time1, tmv1,"Linewidth", 1);
plot(time2, tmv2,"Linewidth", 1);
plot(time3, tmv3,"Linewidth", 1);
plot(time4, tmv4,"Linewidth", 1);
plot(time5, tmv5,"Linewidth", 1);
plot(time6, tmv6,"Linewidth", 1);
legend('1:1','6:5','7:5','8:5','9:5','2:1');
xlabel("Time [ms]");
ylabel("Transmembrane Voltage [mV]");
grid minor;
title("Transmebrane Voltage vs. Time");
bcl_data = readmatrix("APD_analysis.xlsx","Sheet","Sheet2");
BCL = bcl_data(:,1);
eVals = bcl_data(:,2);
%% Plot of eigenvalues vs BCL
figure()
plot(BCL,eVals,"Linewidth",1);
xlabel("Basic Cycle Length [ms]");
ylabel("Dominant Eigenvalue");
grid on;
title("BCL Influence on Dominant System Eigenvalues")
```