2013

With One Breath

Jessica Solfest
College of Saint Benedict/Saint John's University

WITH ONE BREATH

AN HONORS THESIS

College of St. Benedict/St. John's University

In Partial Fulfillment of the Requirements for

All College Honors and Distinction

in the Department of Mathematics

by

Jessica Solfest

April, 2013

With One Breath

Approved by:

Dr. Robert Hesse

Associate professor of Mathematics

Dr. Jennifer Schaefer

Assistant professor of Biology

Dr. Kris Nairn

Associate professor of Mathematics

Dr. Robert Hesse

Chair, Department of Mathematics

Dr. Tony Cunningham

Director, Honors Thesis Program

Acknowledgements

I would like to thank my summer research advisor, Dr. Leonid Rubchinsky, and his colleague, Dr. Sungwoo Ahn, for their guidance and collaboration in developing a new research topic. I am also grateful for my advisor Dr. Jen Schaefer's knowledge and constant encouragement throughout this process. Finally, I thank my advisor Dr. Bob Hesse, whose support, patience, and dedication made this thesis possible.

Introduction

In the summer of 2012 I researched at Indiana University-Purdue University at Indianapolis (IUPUI) through the Mathematical Biosciences Institute undergraduate research program. There, I developed a new research topic: cardiorespiratory synchrony.

To begin analysis, my summer advisor, his colleague, and I noted that cardiac and respiratory relations are akin to neural patterns. Brainwaves exhibit short desychronization periods. We applied my advisor's methods of neural synchrony research to cardiorespiratory synchronization. This topic became my thesis when I returned to the College of St. Benedict.

The indispensable cardiovascular and respiratory systems are both automatically regulated to ensure constant function, and their respective rhythms interact through mediums such as baroreceptors and chemoreceptors. Baroreceptors monitor pressure and send signals to the medulla, the part of the brain that interprets signals from the heart and respiratory system and coordinates appropriate structures to maintain functionality. Baroreceptors are located at specific points throughout the body, including the blood vessels and lungs. Baroreceptors in the lungs monitor stretch during inhalation and exhalation, and conduct information to respiratory rhythmicity centers in the medulla, which control respiratory rate (Martini, Timmons, & Tallitsch 475). Similarly, baroreceptors in blood vessels alert the cardiovascular center in the medulla when blood pressure increases (Sherwood 379). This initiates a complex pathway through which the body returns blood pressure to a normal level.[1]

Chemoreceptors perceive slight changes in dissolved gas concentrations, such as oxygen and carbon dioxide. Chemoreceptors located in the respiratory center of the brain can adjust respiration depth and rate, and those in blood vessels adjust both respiratory and cardiovascular activity[2] (Martini et al.). Thus, heart rate and respiration are interrelated. An example of this occurs during exercise, when both heart and respiration rates quicken to bring sufficient oxygen and nutrients to the working muscles. Muscles require more oxygen when they work harder. The rate of blood flow must then increase to meet muscular demand. In order to pump the blood faster, the heart rate accelerates; likewise, the respiratory rate increases to oxygenate the blood. This is how baroreceptors and chemoreceptors work together to synchronize the cardiovascular and respiratory systems.

[1] High blood pressure sends signals via baroreceptors to the medulla. The medulla then influences the heart to reduce its rate and contraction force (Martini et al. 465).

[2] Chemoreceptors alert the medulla if blood oxygen levels are low or carbon dioxide levels are too high. The medulla then increases heart rate and contraction force, and contraction of muscles that control breathing (Martini et al.). The increased activity works to return blood oxygen levels to normal.

We attempt to mathematically characterize the cardiovascular and respiratory systems in order to understand their behavior and interaction.

Synchronization patterns demonstrate the interaction between heart and respiration rates. Synchrony generally indicates communication between two systems. In this case, both systems are regulated by corresponding control centers in the medulla oblongata (Martini et al.). While these systems are controlled by the same area, they are not completely interdependent and desynchronization periods occur frequently (Ahn, Solfest, Rubchinsky).

In my summer research, we considered heart rate to respiration in an n:1 ratio because their periods differ (see Figure 1) . Heart rate is measured by an ECG and breathing is measured by respiratory impedance. An ECG records the heart's naturally occurring electrical pulse as a continuous wave; each impulse changes the sign of the wave. Respiratory impedance is resistance to airflow, and measures inhalation and exhalation. Figure 1 below compares respiratory impedance and ECG signals. The interaction of two or more systems is referred to as *coupling.*

Figure 1. Respiratory rate compared to heart rate. This graph illustrates the time series differences for heart and respiratory frequencies. Here, the ratio is approximately 3:1, or three heart beats per one breath. (Iyengar, N., et al.).

Unlike previously studied signal comparisons with 1:1 ratios, where either signal can be established as the check point off of which to base the other, cardiac and respiratory signals have different periods. For this reason, we chose the slower wave, respiration, as the checkpoint and noted the value of the heart rate signal each time the respiratory signal changed from negative to positive (see Figure 2). A *phase*, or angle, is part of a complete cycle measured by a specified reference point ("Phase"), and *phase-locking* is when two signals are locked in a synchronous pattern. If heart rate was used as the foundation of phase-locking, the corresponding respiration time would be more inconsistent and possibly obscure synchronous behavior.

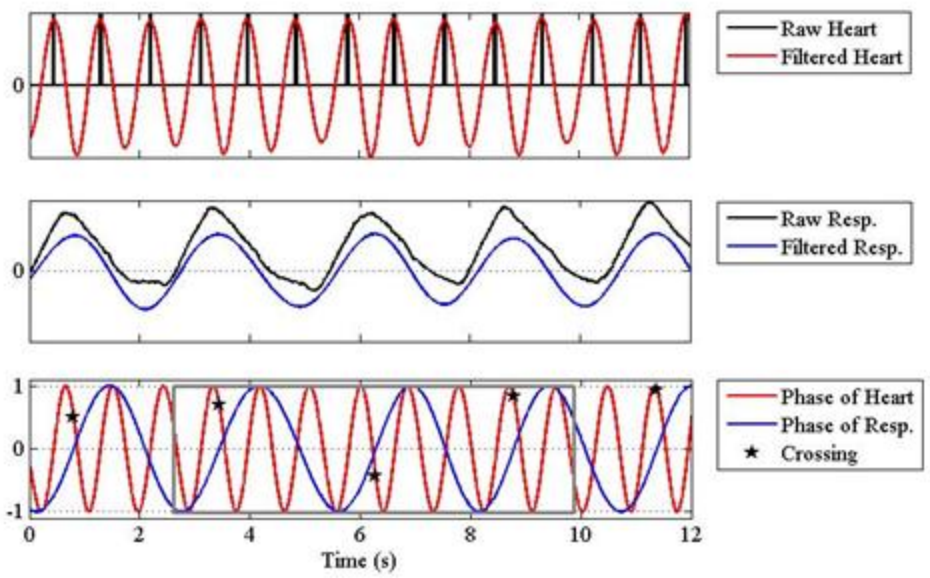

Figure 2. Heart and Respiratory Phases. Ahn, Sungwoo. Indiana University Purdue University, 2012. JPEG.

Figure 2 depicts a filtered ECG signal, filtered respiratory impedance signal, and a graph of the two signals together. It does not matter that filtering changed the signal amplitudes because we look at the time each heart beat or breath occurred, not the force of contraction.

In Figure 2, stars mark heart rate each time respiration changes from negative to positive. We will call this a *crossing.* The amount of signal between each star is a phase, and we represent each phase length as ϕ. We use this direct relationship between the two signals to find phase-locking. Phase-locking is the difference between phases, or $|\varphi_{n,m}| = |n\phi_1 - m\phi_2|$, where $\varphi_{n,m}$ is the generalized phase difference or relative phase, integers n and m describe the locking ratio, and ϕ_1, ϕ_2 are the phases of the oscillators.

Synchronization depends on coupling strength and frequency detuning. Coupling strength describes how strongly two systems interact. If there is no interaction, coupling strength is zero (Pikovsky et al. 11). Frequency detuning measures the difference between signals that are out of phase (Pikovsky et al. 5).

Dynamical oscillations may alter their patterns, affecting their coupling. Pattern alterations can be pathological or physiological, such as the need to increase heart rate and breathing rhythm during exercise or Respiratory Sinus Arrhythmia, a condition in which the cardiorespiratory system synchronizes exactly by shortening and lengthening heart beat in relation to inhalation and exhalation (Berntson, 1993). As with any physiological oscillatory system, perfect synchronization is disadvantageous; the systems must be adaptable to their environments and perform separate functions.

Preliminary Work

The data for this research was from the online archive PhysioNet, which contains multiple databases of physiological recordings, including multi-wave signals. Thus equipped, we analyzed cardiac and respiratory rhythms from two separate populations. The first population was from Massachusetts General Hospital/Marquette Foundation Waveform database (MGH/MF). This source chronicled electronic recordings of patients in critical care units, operating rooms, and cardiac catheterization laboratories for an average of one hour (Welch, 1991). The second source, Fantasia, provided two hour ECG recordings and respiratory rates of healthy, resting individuals. These individuals were separated into two groups of twenty, each with ten males and ten females: group one consisted individuals ages 21-34, and group two was comprised of individuals ages 68-85 (Iyengar, 1996). Both sources measured heart rate with an ECG and respiration by respiratory impedance, or resistance to air flow. Each data set included signals from one patient.

$$(1)\quad \gamma_N(t_N) = \left\| \frac{1}{N} \sum_{j=1}^{N} e^{j\theta(t_j)} \right\|^2$$

In equation (1), γ is the strength of phase-locking, or the amount of synchrony, t represents the time at which each crossing occurs, N is the number of crossings, and $\theta = \phi_1(t_j) - \phi_2(t_j)$, the phase difference. As mentioned in reference to figure 2, we recorded the phase value each time the respiratory impedance signal changed from negative to positive to obtain a set of phase values, $\{\phi_{x,i}\}$, where i=1, 2,…N (Ahn, 2011).

Tables 1 and 2 provide examples of weak and strong phase locking for hypothetical data.

Table 1. Weak Phase Locking

t (seconds)	φ_1 $\lvert t_2-t_1 \rvert$	φ_2	θ $\varphi_1-\varphi_2$
0			
5.4	5.4	5.8	-0.4
11.2	5.8	4.4	1.4
15.6	4.4	5.4	-1
21	5.4	3.3	2.1
24.3	3.3	5.2	-1.9
29.5	5.2	6.2	-1
35.7	6.2	4.4	1.8
40.1	4.4	6.1	-1.7
46.2	6.1	4.9	1.2
51.1	4.9	5.2	-0.3
56.3	5.2	3.6	1.6
59.9	3.6	4.9	-1.3
64.8	4.9		

Table 2. Strong Phase Locking

t (seconds)	φ_1 $\lvert t_2-t_1 \rvert$	φ_2	θ $\varphi_1-\varphi_2$
0			
5	5	4.8	0.2
9.8	4.8	4.9	-0.1
14.7	4.9	4.7	0.2
19.4	4.7	5	-0.3
24.4	5	5.4	-0.4
29.8	5.4	4.9	0.5
34.7	4.9	5	-0.1
39.7	5	4.8	0.2
44.5	4.8	5.1	-0.3
49.6	5.1	4.7	0.4
54.3	4.7	5	-0.3
59.3	5	4.9	0.1
64.2	4.9		

Evaluating equation (1) using values from Table 1 yields:

$$(2)\quad \gamma_{12}(t_{12}) = \left\| \frac{1}{12} \sum_{j=1}^{12} e^{j\theta(t_j)} \right\|^2$$

$\gamma_{12}(t_{12}) = 0.0468$. This low number corresponds to weak phase locking, or the low amount of synchrony between heart and respiratory rates.

From the values in Table 2,

$$(3)\quad \gamma_{12}(t_{12}) = \left\| \frac{1}{12} \sum_{j=1}^{12} e^{j\theta(t_j)} \right\|^2$$

and $\gamma_{12}(t_{12}) = 0.935$, which represents strong phase locking, or a high amount of synchrony.

We first used this method of analysis to calculate the phase synchronization index, γ, with equation 1 to establish that the signals contained synchrony. We created a randomized sequence of the data and used it to compute γ_1. Then we computed γ_2 for the actual data sequence and compared γ_1 and γ_2 to confirm that the original data contained significant periods of synchrony (Ahn, 2011). In this equation, γ is the amount of phase-locking and varies from 0 to 1, where $\gamma_1=1$ is exact phase-locking and $\gamma_2=0$. 0 is the absence of synchrony because when the phase angles are inconsistent, they negate each other and their sum (in equation 1) is small. In this situation, any phase shift is equally probable, indicating no synchrony. Likewise, if the phase angles are more uniform (more synchronous) their sum is closer to 1.

Mapping the phases:

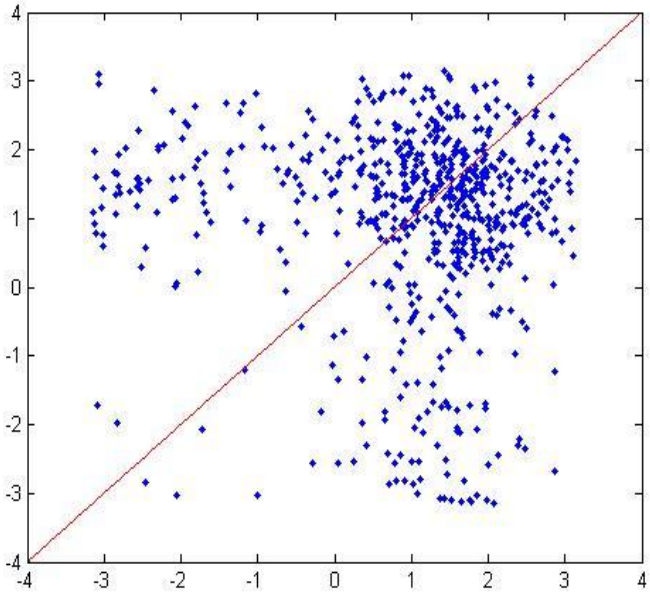

Figure 3. Mapped Data Set. Ahn, Sungwoo. Indiana University Purdue University, 2012. JPEG.

We plotted $\phi_{x,i}$ (present) versus $\phi_{x,i+1}$ (future) for i=1, … N-1 (see Figure 3). Two phases are in perfect synchrony if the corresponding point falls on the line x=y, meaning that the phase angles (θ) are the same; points near x=y are synchronous. If the oscillations are synchronous (determined previously by calculating γ), the data points cluster around the diagonal $\phi_{x,i} = \phi_{x,i+1}$, and we can find the center of this cluster.

After finding the center of a mapped data set, we shifted all points to quadrant I between -π and π and divided this quadrant into four subsections, naming them regions I-IV (see Figure 4). Each section represents ¼ of the phase period, and this simplifies calculations. We shifted the center of the cluster to region I so that this region contained the points that have the preferred phase lag and are, therefore, in synchrony. Regions II-IV contain phases that are out of synchrony (Ahn, 2011).

Once the points were plotted, we analyzed regions I-IV separately and calculated the trajectory of each point. Because each point is based upon $(\phi_{x,i}, \phi_{x,i+1})$, a trajectory that begins at the point $(\phi_{x,i}, \phi_{x,i+1})$ would end at $\left(\phi_{x,i+1}, \phi_{x,i+2}\right)$. All trajectories follow those in Figure 4A, which is why we labeled the regions in this direction. For example, A displays that points originating in region I can move either within the region or to region II. Diagrams B, C, D, and E in Figure 5 illustrate the different circuits a point could make, beginning and ending in region I. This mapping is called a *return map*, or *Poincaré map*. This method "takes a picture" each

time respiration changes from negative to positive by creating a map of the points' positions. In this way, we make the continuous signal discrete.

We then calculated the transition rate as the number of trajectories leaving a region divided by the number of points in the region, and created a histogram to describe each of the four return maps (see Figures 5-7).

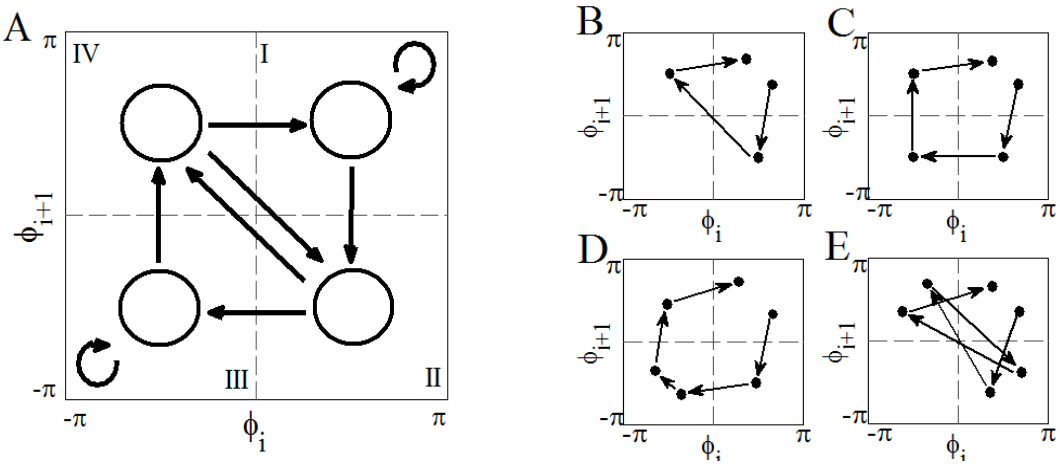

Figure 4. Return Map of Optional Paths; (Ahn, Park, Rubchinsky).

This summer, we found that cardiorespiratory signals exhibit short desynchronization periods. The ratio of heart rate to respiration was usually 2:1 or 3:1, and occasionally 4:1 (see Figures 5, 6, and 7, respectively). Though n: m synchrony may also be present, we represented synchrony as n: 1 because we were most familiar with first-return maps of this ratio.

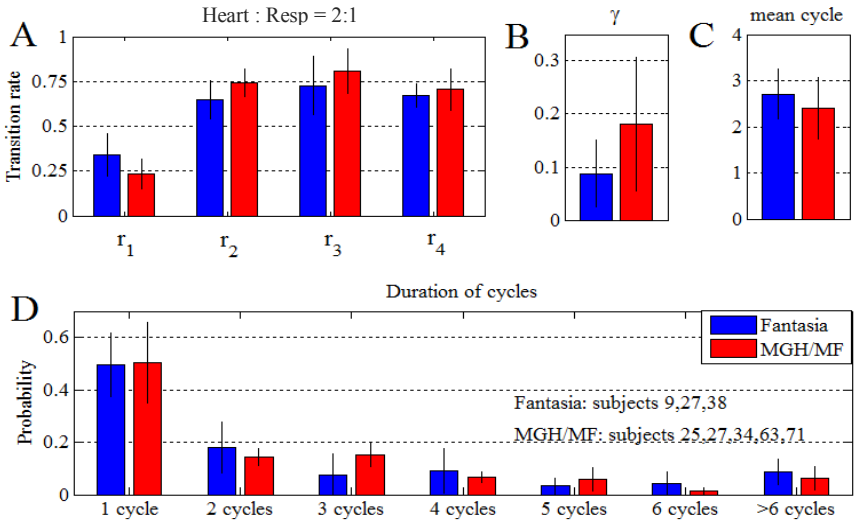

Figure 5. 2:1 Ratio of Heart and Respiratory Synchrony. Ahn, Sungwoo. Indiana University Purdue University, 2012. JPEG. Here, the mode number of trajectories to reach region I is 1 cycle, though the mean is higher because it includes outliers. This histogram indicates that most oscillations are out of phase for only 1 cycle before returning to synchrony.

A: $transition\ rate = \frac{number\ of\ trajectories\ leaving\ a\ region}{total\ points\ in\ the\ region}$. Here, $r_{1,2,3,4}$ represent regions I-IV on the return map (see Figure 2). Note that most points originating in r_1 do not leave; this indicates synchrony.
B: γ is the amount of phase locking ($0 \leq \gamma \leq 1$)
C: average number of cycles needed for a point to return to region I
D: Number of cycles needed for a point to return to region I

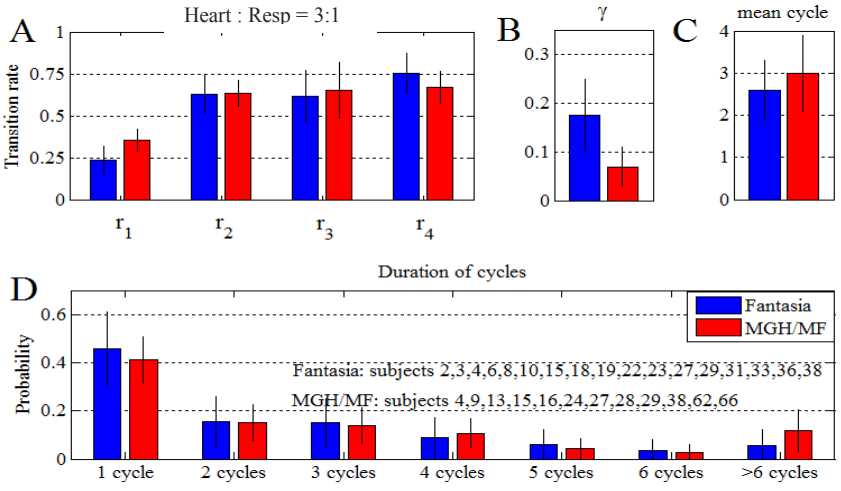

Figure 6. 3:1 Ratio of Heart and Respiratory Synchrony. Ahn, Sungwoo. Indiana University Purdue University, 2012. JPEG.

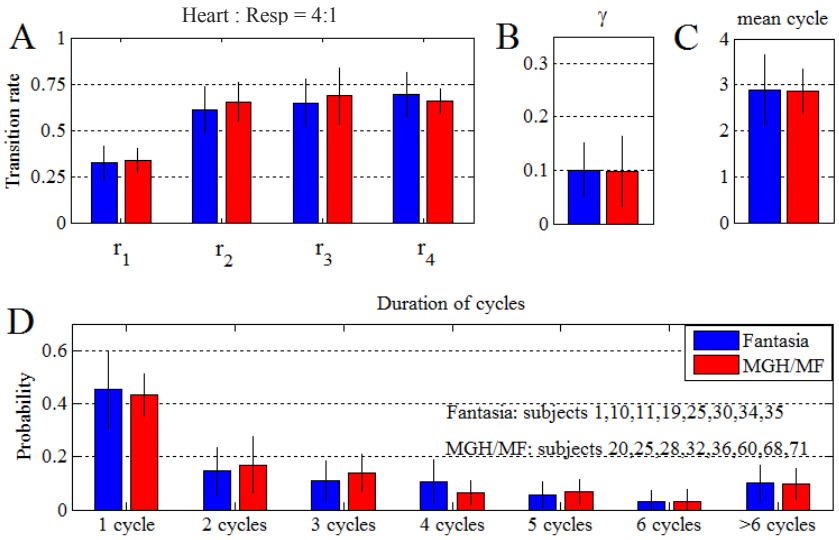

Figure 7. 4:1 Ratio of Heart and Respiratory Synchrony. Ahn, Sungwoo. Indiana University Purdue University, 2012. JPEG.

In these figures, A shows the transition rate: $\frac{number\ of\ trajectories\ leaving\ a\ region}{total\ points\ in\ the\ region}$. We see from the histograms that points leave region I least, and that this is consistent across the three ratios (2:1, 3:1, and 4:1). B shows γ, or the strength of synchrony. We show γ to establish the existence of synchrony, compare the synchronization strength between ratios, and note that heart rate and respiratory rate are not usually in synch. Our research simply needs synchrony to exist in order to analyze its patterns.

C illustrates the mean number of cycles for a point to return to synchrony (region I). A cycle is the number of steps that a point is away from the first region, minus one. One cycle corresponds to the shortest path back to region I, and two cycles is the next shortest path. In other words, one cycle of desynchronization events indicates the length of going from region II to IV to I, and two cycles relates to the path II→III→IV→I. The length of the third duration becomes more complicated because the paths could be either II→IV→II→IV→I or II→III→III→IV→I (Ahn, Solfest, and Rubchinsky 2013). All ratios had a mean between two and three cycles. D shows the distribution of cycles and the probability for a point to take each cycle.

We can conclude from these histograms that the cardiorespiratory system exhibits primarily short desynchronization periods. These outcomes were notably consistent between two separate databases, one of hospitalized patients and the other of healthy study participants. Also, all MGH/MF patients represented in these histograms have cardiorespiratory system malfunctions, further distinguishing them from the healthy Fantasia individuals.

These results lead us to hypothesize that there is an advantage to the cardiorespiratory system's preferred short desynchronization periods. If respiration or heart rate briefly changes rhythm in

response to environmental stimuli, the integrated desynchronization periods could help them resume their normal synchronization pattern more quickly.

Intermittent desynchronization is not common to all oscillators, particularly in nonliving systems. Two such examples are the Lorenz attractor, which models atmospheric convection, and the Rossler attractor, which generates chaotic oscillations. Physiological oscillators, however, need to adapt in variable environments to maintain homeostasis.

Manipulating Data

At the College of St. Benedict and St. John's University, my advisor and I continued my summer research. We analyzed the frequency of cardiorespiratory synchronization and time series patterns of the phase synchrony, noting when the oscillations were in or out of synch.

We used a Matlab program created by my summer colleague to analyze the data. This program reads the MGH/MF and Fantasia data files from PhysioNet, smooths and filters the signals, and identifies synchrony between them. We successfully used it to compute transition rates, the amount of synchrony (γ), and duration in numerical and plot form. Duration is the length of a cycle, or how long a point spends out of synchrony.

First, we altered the Matlab program to evaluate the MGH/MF and Fantasia data, respectively. We initially encountered difficulty running the program because the College of St. Benedict-St. John's University Matlab program is not equipped with the entire programming package that Matlab offers. We tried to accommodate by changing the filtering window location on the signal and the duration of the signal we analyzed. For the MGH/MF data, we changed the beginning of the original code from

```
fs=360; % fs=sampling frequency.
   % change this according to the file name.
   load mgh071m.mat
   disp('mgh071m.mat')
[s1, len] = size(val);
   t=0:1/fs:60;% this is a one minute.
   % depending on the signals, change the length of the time you use.
   %val = val([1,7], tstart+1:2*length(t));     % initial plot for the
   %first few minutes to see the signals.
   val = val([1,7], length(t)+1: end-15*length(t));  % use the whole signals.
   %val = val([1,7], 2*length(t)+1:round(end/2));
```

to

```
fs=360; % fs=sampling frequency.
   % change this according to the file name.
   load mgh036m.mat
   disp('   mgh036m.mat')
```

```
[s1, len] = size(val);
z=10;
t=0:1/fs:60;% this is a one minute.
tstart=0:1/fs:60*z;
% depending on the signals, change the length of the time you use.
val = val([1,7], tstart)+1:2*length(t)); % initial plot for the
%first few minutes to see the signals.
%val = val([1,7], length(t)+1: end-15*length(t)); % use the whole signals.
%val = val([1,7], 2*length(t)+1:round(end/2)).
```

Here, we included *z* to establish a running window length of 10 minutes, and used "val = val([1,7], tstart)+1:2*length(t)" to focus on the first few minutes of the appropriate signals. A running window looks at overlapping periods of a specific length, and filters data as the window moves along. We ultimately found that the code

```
   fs=360; % fs=sampling frequency.
   % change this according to the file name.
   load mgh036m.mat
   disp('   mgh036m.mat')
[s1, len] = size(val);
t=0:1/fs:60;% this is a one minute.
% depending on the signals, change the length of the time you use.
val = val([1,7], tstart)+1:2*length(t)); % initial plot for the
%first few minutes to see the signals.
val = val([1,7], length(t)+1: end-15*length(t)); % use the whole signals.
%val = val([1,7], 2*length(t)+1:round(end/2))
```

ran most smoothly. We set the sampling frequency (fs) to 360 because the signals were at a rate of 360 samples per second, per signal (Welch).

Similarly, Fantasia became

```
   fs=250; % fs=sampling frequency.
   % change this according to the file name.
   load f1y0905m.mat
[s1, len] = size(val);
   z=10;
   t=0:1/fs:(60);    % this is a one minute.
   tstart=0:1/fs:60;
   % depending on the signals, change the length of the time you use.

   %val = val([2,1], length(t)+1: end-15*length(t));     % initial plot for the
   %first few minutes to see the signals.
   %val = val([2,1], length(t)+1: end-15*length(t));  % use the whole signals.
   val = val([2,1], 2*length(t)+1:round(end/2)).
```

with a sampling frequency of 250 because the signals were digitized at 250 Hz when recorded (Iyengar).

Fantasia differs from MGH/MF in signal duration and number of signals recorded. Because Fantasia was a controlled study, all participants' vitals were recorded for two hours, but the MGH/MF signal time varies depending on the hospital procedure. Also, Fantasia has only two signals, respiratory impedance and heart rate, while MGH/MF has eight different signals: three ECG leads, arterial pressure, pulmonary arterial pressure, central venous pressure, respiratory impedance, and carbon dioxide (Iyengar). Respiratory impedance was a better measure of breathing than carbon dioxide output because the amount of carbon dioxide does not necessarily indicate breath amount, and CO_2 rarely produced good signals when recorded. Therefore, we changed "values" to correspond to appropriate ECG and respiratory impedance signals, depending on how the data was organized. In Fantasia, ECG signals are input value 2 and respiratory impedance signals are value 1, whereas in MGH/MF, these signals are 1 and 7, respectively.

We settled on "length(t)+1: end-15*length(t)" for the length of time in order to evaluate each data set entirely. This uses the data in the middle of the signal because the beginning and end of many signals, particularly from the MGH/MF, are noisy. The bandpass filter has a 10 minute window because this provides enough time to measure multiple breaths, and Matlab filtered the data to identify peaks and synchrony patterns. The filter cut the time series into sections of 5-10 minutes with 75% overlap. This isolated peaks for us to compute local maxima and compare phases between heartbeat and respiratory oscillations. Isolating peaks was particularly important to assess respiratory oscillations because inhalation and exhalation periods are not graphically distinct.

After altering the programs that we renamed *Fantasia* and *Hospitalnew* to remove computing errors, we determined what plots the program produced: the signals, transition rates, synchrony, number of cycles, duration, and frequency ratio for each data set. Then we discovered how to display data written in the code but not produced.

We used these codes to analyze 22 MGH/MF patients and 23 Fantasia participants preselected by my summer colleagues for their good signals. We ran each patient data set through the proper program and computed γ, transition rate, and duration, or the probability of being in a particular state. We noticed in this calculation that MGH/MF patients 015 and 024 had a much shorter cycle and therefore contained some durations of zero.

Constructing Matrices

Figure 11 exhibits a continuous series of jumps, more formally known as a *Markov process* (Meerschaert, 284). In this case, points are able to jump between regions without being stuck, or absorbed, in one state. This type of Markov process is called *ergodic* (Meerschaert, 289).

For example, Nicole likes rides. When she visits Valley Fair, she either rides Steel Venom or the Power Tower. After riding Steel Venom, she can choose to either ride again, or go to the Power Tower. Similarly, each time she rides the Power Tower, she can choose to ride again or go to Steel Venom (see Figure 8). She can make this choice after each ride; this is an example of an ergodic Markov process. If Nicole has only one entry ticket to Valley Fair, she has a third option: to leave the park. Now after each ride, she can choose to either change rides or leave the park (see Figure 9). If she chooses either the Power tower or Steel Venom, her choices remain the same, but she cannot come back if she chooses to leave the park. Leaving the park is an absorbing state.

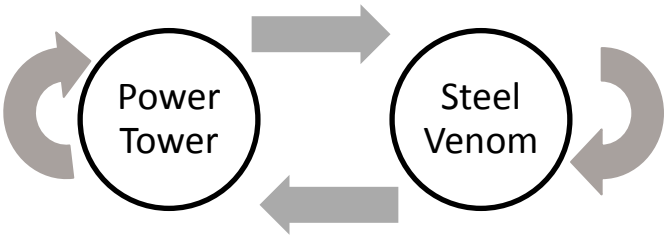

Figure 8. An Ergodic Markov Process.

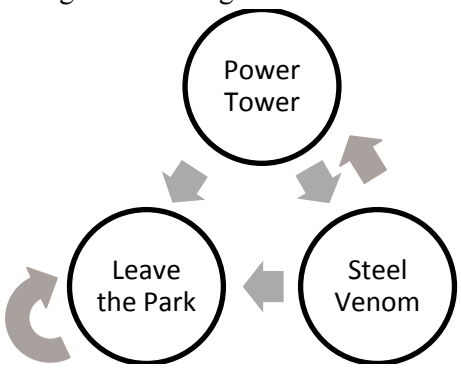

Figure 9. Markov Process with an Absorbing State.

Now we can incorporate probabilities. Nicole prefers Steel Venom slightly more than the Power Tower, and would rather not leave the park. The probabilities of her selecting each state are shown below.

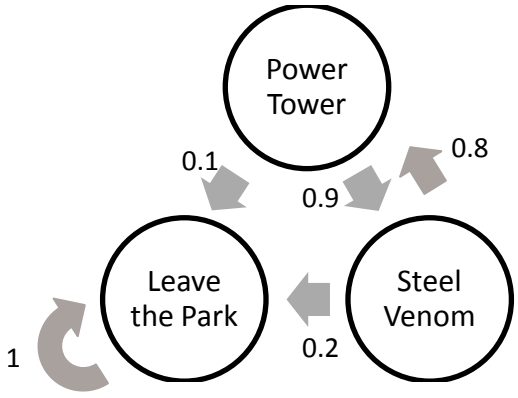

Figure 10. Probabilities with an Absorbing State.

Notice that each option has a total probability of 1 because Nicole must choose a course each time. If she rides Steel Venom, she is 80% likely to choose the Power Tower next and 20% likely to leave the park. Nicole cannot re-enter if she leaves the park, and so she must stay in this state and her probability of remaining outside the park is 1.

Figure 11 displays the ergodic Markov process modeling heart and respiratory rate patterns. Here, we see the different states a point can "choose," with their corresponding transition rates (r).

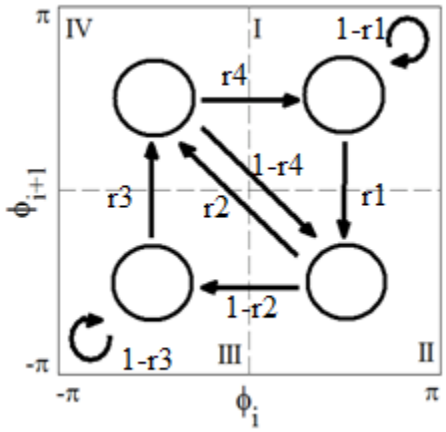

Figure 11. Return Map with Transition Rates Displayed.
Ahn, Sungwoo. Indiana University Purdue University, 2012. JPEG.

A transition matrix is composed of the transition rates to and from each state. In a transition matrix, rows correspond to the state in which a point begins, and the columns represent where the point ends. For example, the return map in Figure 11 displays the options of each transition rate vector. To put this in matrix form, we assign rows as the "start" and columns as the "end" of each transition. We begin in region I and see that $1\text{-}r_1$ begins in region I and returns to region I. Accordingly, it is placed in row 1, column 1 (see Figure 12). Since r_1 begins in region I and ends in region II, it is placed in row 1, column 2. The remaining rates are assigned similarly.

$$\begin{bmatrix} 1-r_1 & r_1 & 0 & 0 \\ 0 & 0 & 1-r_2 & r_2 \\ 0 & 0 & 1-r_3 & r_3 \\ r_4 & 1-r_4 & 0 & 0 \end{bmatrix}$$

Figure 12. Transition Matrix

We computed the average transition rates for each MGH/MF and Fantasia and used this method to create separate transition matrices:

MGH/MF

$$A=\begin{bmatrix} 0.6883 & 0.3117 & 0 & 0 \\ 0 & 0 & 0.2833 & 0.7167 \\ 0 & 0 & 0.3424 & 0.6576 \\ 0.6197 & 0.3803 & 0 & 0 \end{bmatrix}$$

Fantasia

$$B=\begin{bmatrix} 0.6039 & 0.3961 & 0 & 0 \\ 0 & 0 & 0.2697 & 0.7303 \\ 0 & 0 & 0.262 & 0.738 \\ 0.6194 & 0.3806 & 0 & 0 \end{bmatrix}$$

In a transition matrix, no entry is negative and each row sums to one (Isaacson and Madsen, 16). We can understand this with an ergodic Markov process by visualizing a given amount of fluid moving from one state to the next. Though the rate of fluid flow and the amount of fluid in each state are variable, the total amount of fluid always remains equal to one (Meerschaert, 287). We use this concept to find the probability of being in a certain state. Eventually, these probabilities reach an equilibrium, or ***steady state***, where all forces influencing the system are in balance and the rates of change equal zero (Meerschaert, 129). This may occur when the heart and lungs are at rest.

We found the eigenvector, v, associated with the eigenvalue 1 for each group, and obtained the following probabilities:

MGH/MF

$$vA = v1$$

$$v\begin{bmatrix} 0.6883 & 0.3117 & 0 & 0 \\ 0 & 0 & 0.2833 & 0.7167 \\ 0 & 0 & 0.3424 & 0.6576 \\ 0.6197 & 0.3803 & 0 & 0 \end{bmatrix} = v$$

$$v = [0.4499\ 0.2263\ 0.0975\ 0.2263]$$

Fantasia

$$vB = v1$$

$$v\begin{bmatrix} 0.6039 & 0.3961 & 0 & 0 \\ 0 & 0 & 0.2697 & 0.7303 \\ 0 & 0 & 0.262 & 0.738 \\ 0.6194 & 0.3806 & 0 & 0 \end{bmatrix} = v$$

$$v = [0.3980\ 0.2545\ 0.0930\ 0.2545]$$

These matrices show the probabilities of being in regions I, II, III, and IV per row.

Algebraic Analysis

We then calculated the percentage of time that a point spends in each region by transposing matrices A and B, and solving the following systems of equations:

$$[x \quad y \quad z \quad w]A^T = [x \quad y \quad z \quad w]$$

$$[x \quad y \quad z \quad w]B^T = [x \quad y \quad z \quad w]$$

In these equations, x, y, z, and w correspond to the amount of time spent in regions I, II, III, and IV. We discovered that w always equals y. This means that points always spend an equal amount of time in regions II and IV. Solving the first equation with MGH/MF data yields x=0.4499%, y= 0.2263%, z=0.0975%, and w=0.2263%. Consequently, hospitalized patients spend about 45% of time in region I; this is more than the time they spend in any other region. Similarly, we solved the second equation with Fantasia data to ascertain that x= 0.3980%, y=0.2545%, z= 0.0930%, and w=0.2545%. We note, then, that though both MGH/MF and Fantasia participants prefer synchrony, MGH/MF patients spent less time in regions II and IV, and more time in region I. This implies that heart and respiration rates of healthy people prefer the desynchronous transition states (regions II and IV) more and the rates of unhealthy people are more synchronous.

Recall that we hypothesized in my summer research that short desynchronization periods are advantageous to heart and respiratory synchronization patterns. Now we add that healthy patients take longer to regain synchrony. With these, we may conclude that healthy subjects' heart and respiratory rate patterns are more adaptable because they spend more time in the transition states.

We further analyzed the transition rates and determined that the data is normal. We also compared each transition rate r_1-r_4 between the MGH/MF and Fantasia groups with a comparison of means test. This gives us their p-values. A p-value indicates whether or not to reject the null hypothesis that the two groups have the same transition rates. If a p-value is less than .05, we

can reject the null hypothesis and assume that the rate differences are statistically significant; if the p-value is greater than .05, we fail to reject the null hypothesis. The p-values for each rate are $p[r_1]=.056$, $p[r_2]=.587$, $p[r_3]=.021$, $p[r_4]=.994$. Rate 1 is close and worth further study, but it is still above .05 and we assume the groups' transition rates are statistically the same. This suggests that only rate 3 differs between the two groups. Recall that the average r_3 was 0.6576 for MGH/MF and 0.7380 for Fantasia. By definition of transition rate, we realize that more points leave region 3 of Fantasia than MGH/MF, which agrees with our discovery that points spend less time in region 3 of Fantasia. If points spend less time in region III and the same amount of time in regions II and IV, this means that the points spend more time in synchrony.

These results indicate that unhealthy individuals evince more synchronous behavior within the cardiorespiratory system.

Additional research in this area may include developing a formula or mathematical model to predict oscillatory patterns of the cardiorespiratory system. One could also compare results of patients with differing illnesses because the MGH/MF database has detailed descriptions of each patient.

We could apply future mathematical models medically treat dysfunctional systems. The model could give doctors a standard by which to compare ailing patients and properly address their needs.

References

1. Ahn, S., Park, C., & Rubchinsky, L. L. (January 01, 2011). Detecting the temporal structure of intermittent phase locking. *Physical Review. E, Statistical, Nonlinear, and Soft Matter Physics, 84,* 1.)

2. Ahn, S., Solfest, J., Rubchinsky, L.L. *Fine temporal structure of cardiorespiratory synchronization*. (2013). MS. Indiana University Purdue University Indianapolis, IN.

3. Berntson, G. G., Cacioppo, J. T., & Quigley, K. S. (January 01, 1993). Respiratory sinus arrhythmia: autonomic origins, physiological mechanisms, and psychophysiological implications. *Psychophysiology, 30,* 2, 183-96.
4. Goldberger AL, Amaral LAN, Glass L, Hausdorff JM, Ivanov PCh, Mark RG, Mietus JE, Moody GB, Peng C-K, Stanley HE. PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals. *Circulation* 101(23):e215-e220 [Circulation Electronic Pages; <http://circ.ahajournals.org/cgi/content/full/101/23/e215]>; 2000 (June 13).
5. Isaacson, Dean L., and Richard W. Madsen. *Markov Chains, Theory and Applications*. N.p.: John Wiley & Sons, 1976. Print.
6. Iyengar, N., Peng, C., Morin, R., Goldberger, A., & Lipsitz, L. (n.d.). Fantasia Database. *PhysioNet*. 13 Jun. 2000. Web. 31 Jul. 2012. <http://physionet.org/cgi-bin/atm/ATM>.
7. Martini, F., Timmons, M. J., and Tallitsch, R.B. *Human Anatomy*. 7th ed. Boston: Pearson Benjamin Cummings, 2012. Print.
8. Meerschaert, Mark M. *Mathematical Modeling*. 2nd ed. San Diego: Academic, 1999. Print.

9. "Phase." *The American Heritage Dictionary of the English Language: Fourth Edition*. 4th ed. Boston, MA: Houghton Mifflin, 2000. N. pag. Print.

10. Pikovsky, Arkady, Michael Rosenblum, and J. Kurths. *Synchronization: A Universal Concept in Nonlinear Sciences*. Cambridge: Cambridge UP, 2001. Print.

11. Sherwood, Lauralee. *Human Physiology: From Cells to Systems*. 8th ed. Belmont, CA: Brooks/Cole, Cengage Learning, 2013. Print.

12. Welch JP, Ford PJ, Teplick RS, Rubsamen RM. The Massachusetts General Hospital-Marquette Foundation Hemodynamic and Electrocardiographic Database -- Comprehensive collection of critical care waveforms. *J Clinical Monitoring* 7(1):96-97 (1991).

13. Yasuma, F., & Jun-ichiro Hayano. (2004). Respiratory sinus arrhythmia*: Why does the heartbeat synchronize with respiratory rhythm? *Chest, 125*(2), 683-90. Retrieved from http://search.proquest.com/docview/200438564?accountid=14070

Appendix A: Matlab Program

Hospitalnew

```
% written by Sungwoo Ahn
function eeg_synch_freq

   clear
   clc

   % need to change this according to the machine.
   fs=360; % fs=sampling frequency.
   % change this according to the file name.
   load mgh025m.mat
   disp('mgh025m.mat')
   [s1, len] = size(val);
   t=0:1/fs:60;% this is a one minute.


   % depending on the signals, change the length of the time you use.

   %val = val([1,7], tstart+1:2*length(t));     % initial plot for the
   %first few minutes to see the signals.
   val = val([1,7], length(t)+1: end-15*length(t));  % use the whole signals.
   %val = val([1,7], 2*length(t)+1:round(end/2));
    [s1, len] = size(val);


   if mean(val(1, :))<0
      val(1, :) = val(1, :) + abs(mean(val(1, :)));
   end
   thr = mean(val(1, :)) + 2.*std(val(1,:));
   % these values can be changed depending on the signals.
   % try to plot them first: see below - let "fig_flag=1".
   % find local max for ECG
   [Spk, spktime] = SpikeExtraction(val(1,:), thr);
   freq1 = fs./diff(spktime); % frequency of val(1, :)

   % want to look at the signals first ?
   fig_flag=1;
   if fig_flag==1
      figure
      subplot(2,1,1)
      plot(val(1,:))
      subplot(2,1,2)
      plot(Spk,'-r')
      figure
```

```
   hist(freq1)
end

% find local max for Resp.
y2=malowess((val(2, :)/600 + 1)/2,(1:len),'Order',1);
[xmax,imax,xmin,imin] = extrema(y2);
imax = sort(imax);

% if two peaks are close to each other, then consider it as one peak.
for j=1:length(imax)-1
   if (imax(j+1) - imax(j)) < 100  % here 50 is about 0.28 sec.
      imax(j+1) = imax(j);
   end
end
% get the unique element.
imax = unique(imax);
freq2 = fs./diff(imax);

if fig_flag==1
   figure
   plot(1:len, (val(2, :)/600 + 1)/2)
   hold on
   plot(1:len, y2, '-k')
   spk = zeros(1, length(val(2, :)));
   spk(imax) = 1;
   plot(1:len, spk,'-r')
   figure
   hist(freq2)
end

len = length(val(1,:));
len1 = fs*60*1;
len2 = floor(len/len1);
for k=1:len2
   x1=[];x2=[];x3=[];y1=[];y2=[];y3=[];t_spktime=[];t_isi1=[];t_imax=[];t_isi3=[];
   x2 = find((spktime <= len1*k) &(spktime > len1*(k-1)));
   y2 = find( (imax <= len1*k) & (imax > len1*(k-1)) );
   t_spktime = spktime(x2);
   t_freq1 = fs./diff(t_spktime);
   t_imax = imax(y2);
   t_freq2 = fs./diff(t_imax);
   freq_ratio(k) = mean(t_freq1)/mean(t_freq2);
end

% filtering the signals so that the spectral band by using 10-30 Hz band-pass.
```

Appendix A: Matlab Program

```
 % use FIR and SNR, Hanning window

   % val(1, :): ECG
   % val(2, :): Resp.
   [phi_Resp, phi_ECG]=filtering(fs, val(1,:), val(2, :), mean(freq1), mean(freq2));

   %%% From the phase variables, we construct the return map.
   %%% See the inside for detail.
   [rate, synch, duration, cycles]=NewReturnMap(phi_Resp, phi_ECG)

   % if p-value in "NewReturnMap" is larger than 0.05, then there is no
   % results because there is no preferred phase angle.

   % rate: the transition rates r_{1,2,3,4}
   % synch: the synchrony index \gamma
   % duration: the durations of desynchronization events
   % mean_cycles: the mean length of desynchronization events.
   % cycles: the set of desynchronization events.

   if nnz(rate)>1 % when there is a preferred angle, it compute the rate and durations.
      % otherwise, there is no such thing. That is rat=[0 0 0 0];
      results = [mean(freq_ratio), median(freq_ratio), synch, rate, duration];
      % change this name also
      %save results01.mat results
      plot_results(rate, synch, cycles, duration, mean(freq_ratio));
   end

end


function [Spk, spktime] = SpikeExtraction(stnv, threshold)

   alllen = length(stnv);

   Spk=[];spktime=[];
   x = (stnv>threshold);
   diffx = diff(x);
   a=find(diffx==1);
   b=find(diffx==-1);

   if length(a)<length(b)
      b(1)=[];
   end
   len = min(length(a), length(b));
   for m=1:len
```

```
  tempx = zeros(1, length(x));
      tempx(a(m):b(m))=1;
      tempstnv = tempx.*stnv;
      [maxind1, maxind2] = max(tempstnv);
      spktime(m) = maxind2;
   end
   for j=1:m-1
      if (spktime(j+1) - spktime(j)) < 30
         spktime(j+1) = spktime(j);
      end
   end
   spktime = unique(spktime);

   Spk = zeros(1, alllen);
   Spk(spktime)=1;
end

function plot_results(rate, synch, cycles, duration, freq_ratio)

   figure
   subplot(2,2,1)
   bar(1:4, rate)
   ylabel('Transition rate')

   subplot(2,6,4)
   bar(1, synch)
   title('\gamma')

   subplot(2,6,5)
   bar(1, mean(cycles))
   title('Mean lengths')

   subplot(2,6,6)
   bar(1, freq_ratio)
   title('Freq ratio')

   subplot(2,1,2)
   bar(1:length(duration), duration)
   ylabel('Probability')

end


function [phi_Resp, phi_ECG]=filtering(fs, ECG, Resp, freq_ecg, freq_resp)
```

Appendix A: Matlab Program

```
% pwelch : Power spectral density(PSD) using Welch's method
% [Pxx,w] = pwelch(x,window,noverlap,nfft)
% window : x is divided into segments according to window
% noverlap : # of signal samples that are common to two adjacent segments.
% if noverlap = [], then 50% overlap (default)
% nfft : the length of the FFT
% if nfft = [],  the (default) nfft = max{256,2^{k}(> window)}

 % check the frequency band of LFP
   %LFP_before=figure;
   %figure
   %subplot(2,1,1)
   %pwelch(VLFP,fs*2,fs*0.750,[],fs)

   %subplot(2,1,2)
   %pwelch(Spk,fs*2,fs*0.750,[],fs)


% kaiserord estimates the minimum FIR filter order that will approximately
% meet the given set of specifications.
% [n,Wn,beta,ftype] = kaiserord(f,a,dev, fs)
% beta : parameter.
% fir1 : can use the resulting order n, frequency vector Wn, multiband magnitude type ftype, and the
Kaiser window parameter beta.
% f : a vector of band edges
% a : a vector specifying the desired amplitude on the bands defined by f.
% Note : length(f)=2*length(a)-2
% dev (use percentage) : a vector the same size as a that specifies the maximum allowable
% error or deviation between the frequency response of the output filter
% and its desired amplitude, for each band.
% fs: downsample.

   % for ECG
   wa=freq_ecg/2; wb=freq_ecg*2;
   [n,Wn,bta,filtype] = kaiserord( [wa/2, wa, wb, wb*1.1], [0 1 0], [0.01 0.05 0.01],fs );

% FIR fileter.
% To design an FIR filter b that approximately meets the specifications given by kaiser parameters f, a,
and dev, use the following command.
% b = fir1(n,Wn,kaiser(n+1,beta),ftype,'noscale')
   b = fir1(n,Wn, filtype, kaiser(n+1,bta),  'noscale');
% y = filtfilt(b,a,x) : performs zero-phase digital filtering by processing
% the input data in both the forward and reverse directions.
% filtfilt : minimizes start-up and ending transients by matching initial
```

```
% conditions, and works for both real and complex inputs.
   yECG=filtfilt(b,1,ECG);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%

   % for Resp
   wa=freq_resp/2; wb=freq_resp*2;
   [n,Wn,bta,filtype] = kaiserord( [wa/2, wa, wb, wb*1.1], [0 1 0], [0.01 0.05 0.01],fs );
   b = fir1(n,Wn,filtype,kaiser(n+1,bta), 'noscale');
   yResp=filtfilt(b,1,Resp);

   % Phase construction from Filtered data.
   [phi_Resp, phi_ECG]=phase_const(yResp, yECG);

   %left_end=1;
   %right_end= min(length(phi_Resp), length(phi_ECG));

end


function [phi_Spk, phi_LFP]=phase_const(ySpk,yVLFP)

%%% From the oscillatory signal so obtained, an angle of rotation, or
%%% phase, can be conviently defined in the complex unit circle.

%%%% Gabor Representation
   N=length(ySpk);
   Gabor_ySpk=hilbert(ySpk,N);
   Gabor_yLFP=hilbert(yVLFP,N);

%%% Phase Construction
   modulus_Spk=abs(Gabor_ySpk);
   modulus_LFP=abs(Gabor_yLFP);

%%% to obtain a phase evolution of the oscillation, the analytic signal is projected into the unit circle.
%%% thus, we need to obtain the argument(or angle).
   phi_Spk=angle(Gabor_ySpk./modulus_Spk);
   phi_LFP=angle(Gabor_yLFP./modulus_LFP);

end
```

Appendix A: Matlab Program

```
function [rate, gamma, probintdata, cycles]=NewReturnMap(phi_LFP, phi_Spk)

   left_end=1;
   right_end = min(length(phi_LFP), length(phi_Spk));

   rate=[0 0 0 0]; probintdata=zeros(1, 9); cycles=[]; gamma=0;
   minimal_SNR_length=512;
   LERE_index=find(right_end-left_end>minimal_SNR_length);
   LERE_length=length(LERE_index);
   counter=1; all_cycles=[];

   if LERE_length>0
      for jj=1:LERE_length
         le=left_end(LERE_index(jj));
         re=right_end(LERE_index(jj));

         %gamma(counter) = abs(sum(exp(1i*(phi_Spk(le:re) - phi_LFP(le:re))))/length(phi_Spk(le:re)))^2;
         %counter = counter+1;

         for phi_LFP_threshold=0 %-3:3

          % LFP check point
            cross_index=find(diff(sign(phi_LFP(le:re)-phi_LFP_threshold))>0);
            xn=phi_Spk(cross_index+le-1);

            gamma(counter) = abs(sum(exp(1i*xn))/length(xn))^2;
            counter = counter+1;

            %%% to compute the ratios of the return map, we shift the phases
            % so that most values are in the first quadrant.
            [N,X]=hist(xn,10);

            % test whether there is any preferred phase angle.
            % compare this with the uniform distribution.
            len=length(xn);
            r = -pi + 2*pi.*rand(len,1);
            [h,p] = kstest2(xn, r);
            if p>.05
               p
               return
            end

            [maxVal,maxind]=max(N);
            clust_center=X(maxind);
            xn=xn+pi;
```

```
            shift=clust_center-pi/2;
            xn=xn-shift;
            xn=mod(xn,2*pi);
            xn=xn-pi;
          end
          [rate, probintdata, cycles]=escaperatio2(xn);
          %%% compute the ratios of the return map.
        end
    end

end

function [ratio_R, Probvaldata, All_cycles] = escaperatio2(xn)
%%% compute the ratios of the return map.
    ratio_R=[0 0 0 0]; Probvalrate=[0 0 0 0 0 0];Probvaldata=[0 0 0 0 0 0 0 0 0 0];All_cycles=[];
    %%% change the phase as +1, 0, or -1. Note that 0 is very rare so we
    %%% ignore the case.

    % shifting values so that the most values are in the first quadrant.

    group1=0;
    group1t=0;
    group2=0;
    group2t=0;
    group3=0;
    group3t=0;
    group4=0;
    group4t=0;

    for gg=1:length(xn)-2
        yn=xn(gg);
        zn=xn(gg+1);
        wn=xn(gg+2);
        if yn>0 && zn>0
            group1t=group1t+1;
            if wn <0
                group1=group1+1;
            end
        elseif yn>0 && zn<0
            group2t=group2t+1;
            if wn >0
                group2=group2+1;
            end
        elseif yn<0 && zn<0
            group3t=group3t+1;
```

```
        if wn>0
            group3=group3+1;
        end
    elseif yn<0 && zn>0
        group4t=group4t+1;
        if wn>0
            group4=group4+1;
        end
    end
end
ratio1=group1/group1t;
ratio2=group2/group2t;
ratio3=group3/group3t;
ratio4=group4/group4t;

ratio_R=[ratio1, ratio2, ratio3, ratio4];

[Probvaldata, All_cycles] = DeviationLengthCheck(xn);

end


function [Prob, DD]=DeviationLengthCheck(xn)

N=length(xn);
DD=0;
counter=1;
L=0;
ii=1;
while ii<N
    if xn(ii)>0 && xn(ii+1)>0
        DD(counter)=1;
        counter=counter+1;
        ii=ii+1;
    else
        L=L+1;
        ii=ii+1;
        if ii==N
            DD(counter)=L;
            counter=counter+1;
            L=0;
            break
        end
        while xn(ii)<0 || xn(ii+1)<0
            ii=ii+1;
```

```
            L=L+1;
            if ii==N
               DD(counter)=L;
               counter=counter+1;
               L=0;
               break
            end
         end
         if ii<N
         DD(counter)=L;
         counter=counter+1;
         L=0;
         end
      end
   end
   xx1 = find(DD==1);
   DD(xx1)=[];
   DD = DD-1;

   % compute the probabilities of durations of desynchronization events.
   for mm=1:8
      Prob(mm) = nnz(find(DD==mm))/length(DD);
   end
   Prob(mm+1) = 1- sum(Prob);


end


function [xmax,imax,xmin,imin] = extrema(x)
%EXTREMA   Gets the global extrema points from a time series.
%   [XMAX,IMAX,XMIN,IMIN] = EXTREMA(X) returns the global minima and maxima
%   points of the vector X ignoring NaN's, where
%    XMAX - maxima points in descending order
%    IMAX - indexes of the XMAX
%    XMIN - minima points in descending order
%    IMIN - indexes of the XMIN
%
%   DEFINITION (from http://en.wikipedia.org/wiki/Maxima_and_minima):
%   In mathematics, maxima and minima, also known as extrema, are points in
%   the domain of a function at which the function takes a largest value
%   (maximum) or smallest value (minimum), either within a given
%   neighbourhood (local extrema) or on the function domain in its entirety
%   (global extrema).
%
```

Appendix A: Matlab Program

```
%   Example:
%      x = 2*pi*linspace(-1,1);
%      y = cos(x) - 0.5 + 0.5*rand(size(x)); y(40:45) = 1.85; y(50:53)=NaN;
%      [ymax,imax,ymin,imin] = extrema(y);
%      plot(x,y,x(imax),ymax,'g.',x(imin),ymin,'r.')
%
%   See also EXTREMA2, MAX, MIN

%   Written by
%   Lic. on Physics Carlos Adri? Vargas Aguilera
%   Physical Oceanography MS candidate
%   UNIVERSIDAD DE GUADALAJARA
%   Mexico, 2004
%
%   nubeobscura@hotmail.com

% From      : http://www.mathworks.com/matlabcentral/fileexchange
% File ID    : 12275
% Submited at: 2006-09-14
% 2006-11-11 : English translation from spanish.
% 2006-11-17 : Accept NaN's.
% 2007-04-09 : Change name to MAXIMA, and definition added.


xmax = [];
imax = [];
xmin = [];
imin = [];

% Vector input?
Nt = numel(x);
if Nt ~= length(x)
 error('Entry must be a vector.')
end

% NaN's:
inan = find(isnan(x));
indx = 1:Nt;
if ~isempty(inan)
 indx(inan) = [];
 x(inan) = [];
 Nt = length(x);
end

% Difference between subsequent elements:
```

Appendix A: Matlab Program

```
dx = diff(x);

% Is an horizontal line?
if ~any(dx)
 return
end

% Flat peaks? Put the middle element:
a = find(dx~=0);             % Indexes where x changes
lm = find(diff(a)~=1) + 1;    % Indexes where a do not changes
d = a(lm) - a(lm-1);         % Number of elements in the flat peak
a(lm) = a(lm) - floor(d/2);   % Save middle elements
a(end+1) = Nt;

% Peaks?
xa  = x(a);             % Serie without flat peaks
b = (diff(xa) > 0);     % 1  =>  positive slopes (minima begin)
                        % 0  =>  negative slopes (maxima begin)
xb  = diff(b);          % -1 =>  maxima indexes (but one)
                        % +1 =>  minima indexes (but one)
imax = find(xb == -1) + 1; % maxima indexes
imin = find(xb == +1) + 1; % minima indexes
imax = a(imax);
imin = a(imin);

nmaxi = length(imax);
nmini = length(imin);

% Maximum or minumim on a flat peak at the ends?
if (nmaxi==0) && (nmini==0)
 if x(1) > x(Nt)
  xmax = x(1);
  imax = indx(1);
  xmin = x(Nt);
  imin = indx(Nt);
 elseif x(1) < x(Nt)
  xmax = x(Nt);
  imax = indx(Nt);
  xmin = x(1);
  imin = indx(1);
 end
 return
end

% Maximum or minumim at the ends?
```

Appendix A: Matlab Program

```
if (nmaxi==0)
 imax(1:2) = [1 Nt];
elseif (nmini==0)
 imin(1:2) = [1 Nt];
else
 if imax(1) < imin(1)
  imin(2:nmini+1) = imin;
  imin(1) = 1;
 else
  imax(2:nmaxi+1) = imax;
  imax(1) = 1;
 end
 if imax(end) > imin(end)
  imin(end+1) = Nt;
 else
  imax(end+1) = Nt;
 end
end
xmax = x(imax);
xmin = x(imin);

% NaN's:
if ~isempty(inan)
 imax = indx(imax);
 imin = indx(imin);
end

% Same size as x:
imax = reshape(imax,size(xmax));
imin = reshape(imin,size(xmin));

% Descending order:
[temp,inmax] = sort(-xmax); clear temp
xmax = xmax(inmax);
imax = imax(inmax);
[xmin,inmin] = sort(xmin);
imin = imin(inmin);

end
```

Fantasia is essentially the same code, except for the changes noted on pages 10 and 11.