AUTOMATED STATE CHANGE DETECTION OF A GAUSSIAN DISTRIBUTED SIGNAL

A Thesis
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
Electrical Engineering

by
Anirudh Singh
August 2008

Accepted by:
Dr. Adam W. Hoover, Committee Chair
Dr. Eric R. Muth
Dr. Richard R. Brooks

ABSTRACT

This work presents a novel approach to detect a change in the state of a signal. The hypothesis is that in a given state, the values of a signal vary in a sub-part of a Gaussian distribution. By modeling a signal using a sub-part of the Gaussian distribution, a quantitative value representing the state of the signal is obtained. This value is monitored with time to detect a change in the state of the signal. The proposed algorithm is implemented and tested on data obtained from a related research.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Hoover, for all of the helpful guidance he gave me throughout the Masters' program. I'd also like to thank my committee members, Dr. Muth and Dr. Brooks for the time and effort they spent reviewing my thesis. I would also like to thank my parents, my sister and my fiancée. Without their time, effort and persuasion, I would never have been able to stand where I am today. Thanks also belong to all of my friends who were a constant source of inspiration.

TABLE OF CONTENTS

LIST OF FIGURES

List of Figures (Continued)

Figure Page

List of Figures (Continued)

Figure Page

LIST OF TABLES

CHAPTER 1
INTRODUCTION

This research focuses on the problem of detecting a change in the state of a signal. A signal can be defined as a one dimensional entity that is being measured repetitively at fixed intervals of time. The range of values that a signal can take are indicative of its state. Thus detecting a change in the state of a signal implies locating a significant change in the values of the measured entity.

The concept about the states between which the value of a signal changes and the point in time when it changes is somewhat fuzzy and quantifying this idea statistically is an important part of this research. We are interested in signals having a Gaussian probability distribution. We wish to detect a change in such a signal in real-time while new measurements are coming in and to detect this change with the least possible lag. Most real world signals are very noisy. Thus detecting a change in the state of such signals is difficult. So an important objective of this work is to discard instantaneous fluctuations in the value of the signal and consider its overall trend to detect a change in its state. Once a change is detected, based on the application of this work, the information about the change can then be used to trigger a number of processes dealing with bringing the signal back into the desired state or modifying the system according to the change.

This work has potential applications in medicine, power lines, computer network traffic flow and weather monitoring. For example, a physiological signal by itself or in combination with other signals, can be used to monitor the physical or mental state of a person. Monitoring refers to the ability to detect changes in the signals indicating a change in the state of the person. This is of particular interest for doctors especially during surgeries when they need to monitor physiological signals such as heart rate, blood pressure and respiration of a patient to make sure that his/her condition is stable. The method proposed by us could help in the detection of such changes automatically. Our proposed method can be used to monitor traffic on a computer network. Detection of a rise in the traffic flow would trigger counter-measures which re-route the traffic to avoid congestion. This work can be utilized in power systems management to monitor power outages. It can also be used to observe and predict changes in weather (temperature, rainfall, humidity).

Our problem does not lie in detecting a change but in deciding when that change has occurred. Changes keep occurring continuously, but only those that occur gradually over a period of time are of interest. For example, consider that the temperature in a city is being monitored. Let a set of readings of the temperature be measured every half hour from morning until afternoon. Every temperature reading would be different from previous one. But as the morning transitions into afternoon, the temperature would gradually increase to a certain point and then stabilize. It the afternoon, it can be said that the temperature has risen or changed since morning. The question that we need to answer is that over this period, at what point can it be said that the temperature has 'changed'?

Detecting a change in the state of a signal off-line is easier as the complete signal data is available for analysis. The challenge lies in detecting this change in the signal in real-time and as soon as possible after the change has occurred. Detecting a state change in real-time brings forth the dual faceted issue of either detecting a change as close to the actual change in the signal or detecting it as fast as possible.

Another difficulty lies in modeling a signal when nothing is known *a priori* about its characteristics. It thus becomes difficult to formulate an algorithm to detect a change in the signal. But if the probability distribution of the signal or some other statistic is known, it is possible to model the signal based on its distribution and thus monitor its progression in real-time.

Some research groups have considered a signal to be divided into fixed pre-determined states. This makes the problem one of classifying the state of the signal. Picard, Vyzas and Healey [10] came up with the idea of observing the physiological state of a person to predict the person's emotional state. They gather four physiological signals - Electromyogram, Blood Volume Pressure, Skin Conductivity and Respiration. From these signals, they calculate features which include some statistical features, for all the signals. Based on these features, they classify the person's current state into eight emotional categories. Though this work is similar to our research, it is different because it involves detecting the state of a signal and classifying it into predetermined categories. While our work requires detection of a change in the state of a signal. We are not concerned with the classification of a signal's state.

Another group of researchers consider the problem of detecting an arbitrary change in the state of the signal. Raifel & Ron [11] suggested that the signal under consideration may be made up of two or more components. They proposed that any physiological signal is composed of a slowly changing underlying signal (that is assumed to be a low-order polynomial function), the actual physiological signal and some noise. They isolate the underlying signal by using cubic spline interpolation between selected points in the data-set. The noise and the fast changing signal are estimated using least squares estimation (LSE). Although the algorithm is tolerant to noise, the main drawback of this technique is that the points for underlying signal isolation are to be selected carefully to obtain the desired results. Also their algorithm requires knowing the structure of the fast component of the signal a priori, which is not the case with our signal.

Another group discussed the detection of both fast and slowly varying components within a signal. Agharebparast and Leung [1] proposed an algorithm for monitoring network traffic rate continuously using two time sliding window (TSW) filters to estimate the traffic rate and a controller to switch between the filters. One filter has a small window size making it more responsive to changes and the other has a bigger window size to make it more stable. The controller switches to the responsive filter in case of an instantaneous change and to the stable filter in all other cases. Unlike our objective of detecting a change in a signal, they emphasize more on estimating the rate of change.

Charbonnier, Becq and Boit [4] suggested matching the input signal to a predicted model to detect a change. They proposed representing a physiological signal as a set of continuous or discontinuous line segments. They fit the data to a linear model and use the cumulative sum (CUSUM) of the difference between the actual data and the linear approximation to check if it is correct, based on preset thresholds. The algorithm can be run in real-time to determine the trend of the signal but the paper does not discuss the detection of change in trends.

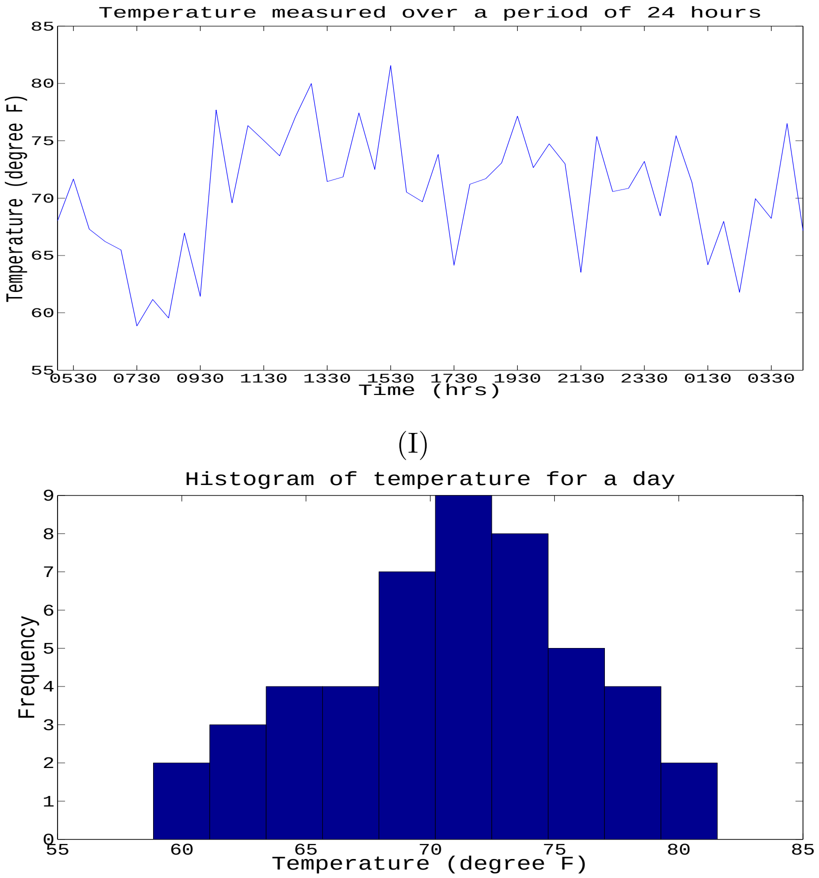
Another prediction technique is suggested by Yang, Dumont and Ansermino [12] where they propose a three step approach to detect changes in the heart rate trend. First, they fit the heart rate signal with a dynamic linear model with independent white noise and observation noise. Then an adaptive Kalman filter is used to determine the noise covariances Q & R. The filtered signal is then used to estimate future values and limits for the observed signal. Adaptive CUSUM testing is then used to determine if the new data follows the estimate by keeping track of a cumulative sum of difference between the estimate and the actual data. The result of CUSUM testing decides the new forgetting coefficients for the estimator and also if the data has changed trend. Two advantages of their algorithm over the one developed by Charbonnier [4] are that their model is not linear, so it gives a better approximation of the signal, and their CUSUM thresholds are dynamic, changing according to the forgetting factor of the estimator.

In this research we are primarily concerned with signals whose frequency histograms can be characterized by a Gaussian distribution. Many real-world signals, including physiological signals such as heart rate, skin conductivity, blood pressure and respiration and climatic signals such as temperature, rainfall and humidity follow this distribution. As an example, consider that the temperature is being monitored throughout the day, as shown in Fig 1.0.1(I). The histogram of the signal (Fig 1.0.1(II)) shows that the measurements follow a Gaussian distribution.

Now consider a part of this signal in which we wish to detect a change (Fig 1.0.2(I)). We hypothesize that the variance of the signal under consideration at a given time lies in only a sub-part of the Gaussian. When a change in the state of the signal occurs, its variance moves to another sub-part of the Gaussian. This can be visualized from the Fig 1.0.2(II). Based on the identification of these sub-parts of the Gaussian, it is possible to locate the point at which the state of the signal changes.

1.1 Motivation

This work was motivated by the problem of detecting a change in the heart rate variability (HRV) of a person. The problem required a signal that represents the HRV of a person to be monitored in real-time. A change in the state of this signal was supposed to be detected as soon as possible. In this section we describe the background of this



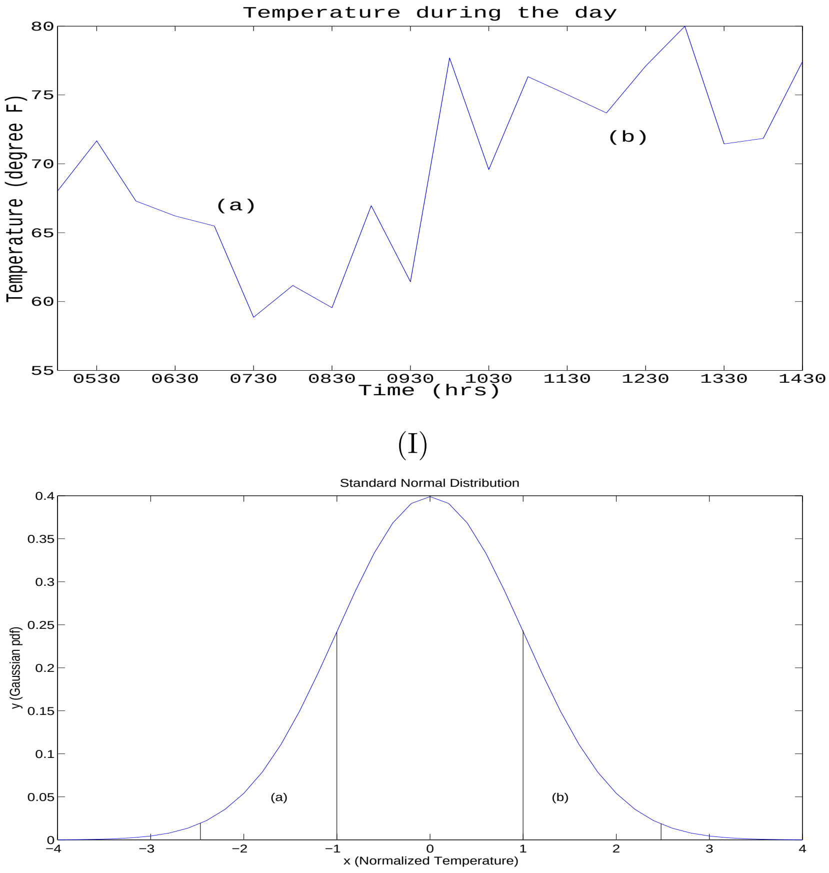
(I)

(II)

Figure 1.0.1 I: Temperature measurements for a period of 24 hours. II: The Histogram of the temperature measurements follows a Gaussian distribution.

problem and how the work previously done related to this research motivates us.

We begin with a description of the various parts of the nervous system of a human body (shown in Figure 1.1.1). The central nervous system (CNS) is the largest section of a human nervous system and consists of the brain and the spinal cord. An extension of the CNS is the peripheral nervous system which is present to support the limbs and organs in the body. The peripheral nervous system can be further divided into the somatic nervous system, that is responsible for body movement and receiving external stimuli and the autonomic nervous system (ANS). The ANS consists of three parts as shown in fig 1.1.1. The enteric nervous system deals with digestion of food in the body. The sympathetic nervous system (SNS) is responds to situations of stress and danger by increasing the body's heart rate (RR) and respiration, inducing a sense of 'arousal'.



(I)

(II)

Figure 1.0.2 I: Sample temperature measurements taken form morning 5 am to 2:30 pm II: The variance of two parts of the signal in two sub-parts of the Gaussian distribution

The parasympathetic nervous system (PNS) on the other hand, works in opposition to the SNS by bringing the body to rest and maintaining homeostasis. The activation of sympathetic and parasympathetic activity can occur independently, together or as a result of the other [7, 2, 3].

The dynamic relationship between sympathetic and parasympathetic nervous systems causes variations in the heart rate known as heart rate variability (HRV). If HRV is respiration induced, it is known as respiratory sinus arrhythmia (RSA) [2]. RSA can be defined as a phenomenon of slowing down of the heart rate when exhaling and speeding up during inhalation [9]. The metrics used to measure HRV and RSA can either be based in the time-domain or the frequency-domain. One frequency-domain based technique utilizes the interbeat interval (IBI, the time interval between two con-

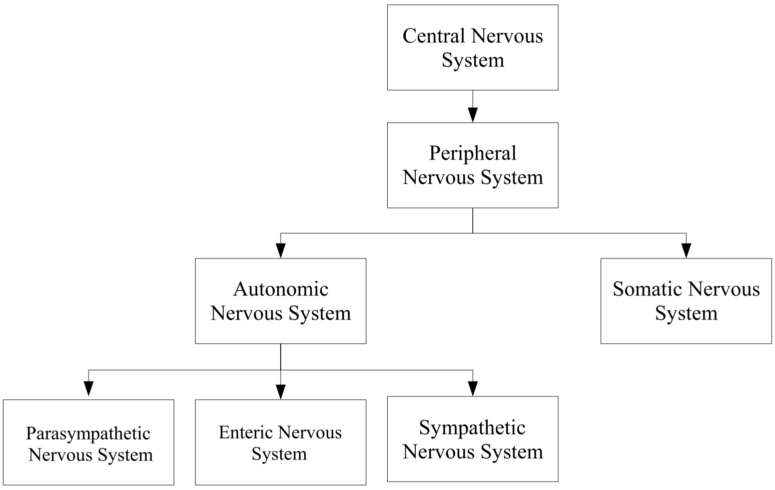
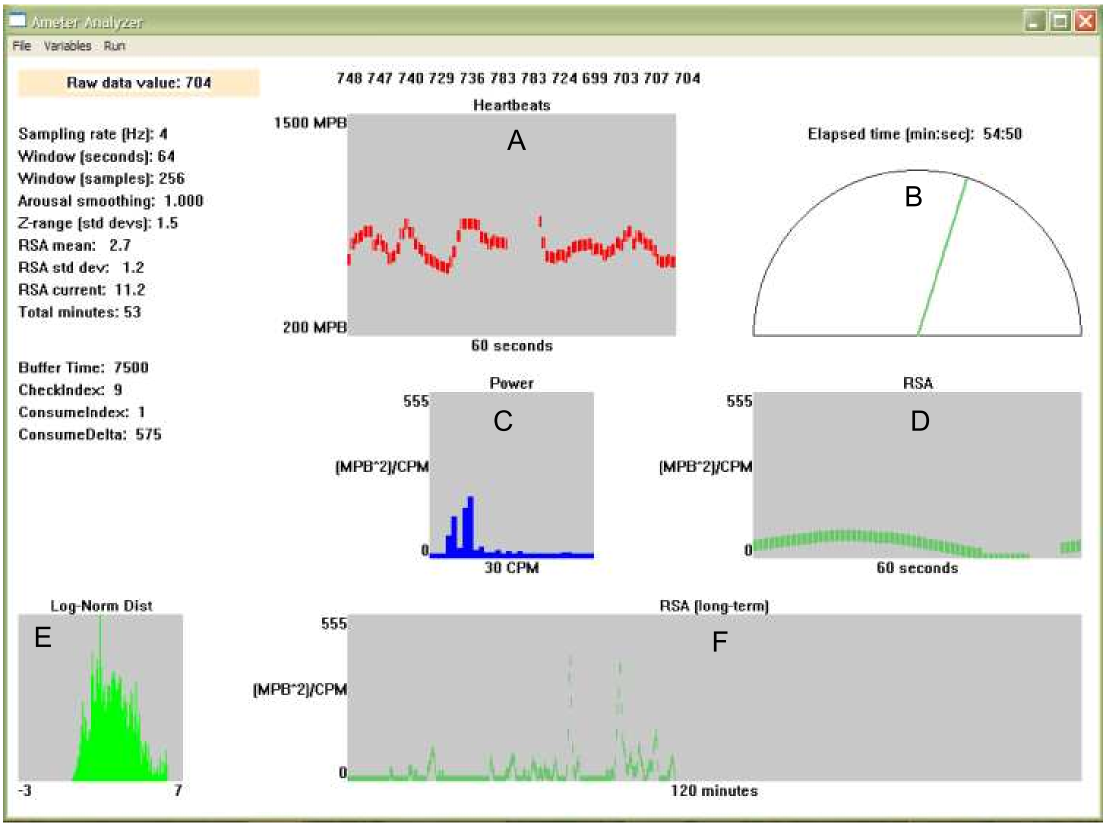


Figure 1.1.1 A chart depicting the interaction between the sub-parts of the human nervous system

secutive heart beats) to construct a power spectrum in the band of 0.0-1.0 Hz. The high frequency component (0.15-.50 Hz) in this band is an index of RSA and is primarily affected by parasympathetic responses [7].

Hoover and Muth [7] developed a non-invasive system, called the *arousal meter* (AM), for real-time measurement of vagal activity using RSA. Their system consists of a heart rate sensor to measure the IBIs and a computational platform to analyze the measured IBIs. The IBIs form a continuously varying signal and typically have a period of 500-2000 ms. But in order to discretize this signal to facilitate frequency analysis, it has to be re-sampled in accordance to the Nyquist sampling theorem. Thus a sampling period of 250 ms is selected. The data is buffered for 64 seconds before performing power analysis using Fast Fourier Transform (FFT). The maximum average power reading is then calculated from the high frequency range and a z score is computed based on it to normalize the reading. Most of the z score readings lie in the range of [-1.5 1.5] and are scaled to a range of [0 1] for the AM output. A screen shot of the arousal meter is shown in Fig 1.1.2. The AM parameters are on the left. Section A in the screen shot shows a plot of the most recent IBIs over time. The corresponding IBIs are shown above section A. Section C shows the power reading for the current iteration. Section D shows the plot of the maximum average power over the last minute and F shows the same statistic over the last two hours. Section E shows a log-normal distribution of the data collected until the current time. Section B shows the output of the AM plotted as angle (0 to π degrees) along a semi-circle with 0 representing the highest arousal.



Raw data value: 704

Sampling rate (Hz): 4
Window (seconds): 64
Window (samples): 256
Arousal smoothing: 1.000
Z-range (std devs): 1.5
RSA mean: 2.7
RSA std dev: 1.2
RSA current: 11.2
Total minutes: 53

Buffer Time: 7500
CheckIndex: 9
ConsumeIndex: 1
ConsumeDelta: 575

748 747 740 729 736 783 783 724 699 703 707 704

Elapsed time (min:sec): 54:50

Figure 1.1.2 The Arousal Meter Interface developed by Hoover and Muth [7]

Hoover, et al. [8] also proposed an algorithm for real-time error detection and correction of the measured IBIs. This method was used prior to sending the IBIs to the AM for frequency analysis described above. The motivation for this work was the high error rate (approximately 10%) in the detection of IBIs amongst actively moving subjects. They buffer the asynchronously measured IBIs for 6 seconds and use an adaptive threshold to test the current IBI against the last corrected IBI. An errant IBI was corrected on the basis of a set of pre-defined rules. If more than 3 consicutive error detections occur, then error detection is temporarily stopped to prevent furhter false detections.

Fishel [6] conducted a study to determine the proficiency of the AM in differentiating between state changes in arousal that occur due to changes in work-loads. A state change is defined as a significant change in the arousal due to a change in the work-load of a person. The state of a person is generally measured from a reference state called a baseline. There are a lot of different ways in which this can be done as mentioned in Fishel [5]. They considered the method of comparing the current state to a resting baseline. As the name suggests, a resting baseline represents the state of a person during the time they were resting with their eyes closed but not sleeping. The subjects in her study were made to undergo a series of high and low workload tasks continuously over a period of 32 minutes. The experiment required the subjects to switch between high and low work-load tasks at time intervals of 0.5,1,2,4 and 8 minutes. A pair of consecutive high and low work-load tasks is called a *task-pair.* The experiment required the subjects to sit on a computer and perform the tasks which consisted of either a shooting (high work-load) task or a surveillance (low work-load)task. They claimed that on an average, they could successfully detect a task change by comparing the means of the states of the two tasks in a signal.

Our approach takes its motivation from the research mentioned above. We hypothesize that because the state of a signal is supposed to be representative of the tasks being performed, it is possible to detect a task change in a signal by monitoring the state of the signal.

CHAPTER 2
METHODS

This chapter describes the process detecting a state change in a signal. The hypothesis is that a signal that follows a Gaussian distribution effectively varies in a small range of the Gaussian distribution's spread. If a state change occurs, the range of values in which the signal varies changes. We propose to split the given signal into two segments and find the range of the Gaussian distribution in which they are varying. By comparing this range for the two segments, it can be decided if a state change occurred. This is the novelty of our approach.

This chapter describes the implementation of the proposed approach. We begin by giving a description of the assumptions necessary to be made for proceeding with this work. The algorithm developed for implementing this approach is then explained. The details about fitting Gaussian and sub-Gaussian models to given data are then described. We then describe the method used to find the optimal split of a given sequence of data. An explanation about the statistic calculated from the parameters of the sub-Gaussian fitting is then provided. This statistic is used for detecting a state change in the data.

2.1 Assumptions

Before proceeding with the description of the proposed approach, we list the set of assumptions here. The first assumption to be made about the input signal is that it is characterized by a Gaussian distribution. Another assumption about the signal is that it does not need to be measured synchronously. In other words, the signal does not need to have data points at fixed intervals of time. But it instead needs to be sampled regularly. For example, let a signal be represented as:

$$x_i, x_{i+1}, x_{i+2}, x_{i+3}, x_{i+4}, \ldots$$

where *i,i+1,i+2,...* are the indices of the data points of the signal.

A signal being analyzed is already assumed to follow a Gaussian distribution and thus is characterized by two parameters: its mean (μ) and its standard deviation (σ). Fitting the Gaussian curve to the signal would require us to solve for both μ and σ.

Knowledge of both these parameters *a priori* significantly reduces the computation time. Thus it is also assumed that the input signal is either already normalized or its mean $\bar{x}$ and standard deviation σ are known *a priori.* In the case of processing non-normalized data, it can be normalized in two simple steps. First, the data points (x_i) in the signal are mean shifted as shown in Equation 2.1.1. This changes the mean value of the data to 0. Next, the data points are auto-scaled to make their standard deviation equal to 1. This is done by dividing all data points by their standard deviation as shown in Equation 2.1.2.

$$x_i' = x_i - \bar{x} \tag{2.1.1}$$

$$x_i' = \frac{x_i}{\sigma} \tag{2.1.2}$$

One way of finding the mean and standard deviation of the signal *a priori* is to somehow calibrate the data based on a sample or segment of the signal varying in its dormant state. This gives an appropriate measure of the signal's true mean and standard deviation. Once an initial estimate of these statistics is obtained, the algorithm proposed in the next section can be implemented and the statistics can be updated as more data is acquired with time. The initial estimate of the μ and σ can also be chosen arbitrarily and updated as more data is read. Thus the estimates of the mean and standard deviation converge to their respective true values with time and data normalization becomes more accurate.

We assume that the range of values that the signal varies inside, reflects the state of the signal. For example, considering computer network traffic, the volume of the traffic is an indicator of the state of the network (blocked or unblocked). A state change would thus be characterized by a change in the range of values in which a signal varies. This concept is slightly fuzzy because there are no defined limits for the range of values for a state of a signal. Our approach helps objectify this fuzzy concept and relate the state of a signal to a range / set of values.

2.2 State Change Detection

The state change detector proposed here is developed so that it can be used to analyze signals in real-time. Let the time line of a simulated sample signal passing through the state change detector be as shown in Figure 2.2.1. The figure helps highlight the different variables and indexes used in the algorithm. The index i represents the point of time succeeding the last detected state change. The index t represents the most recently read data point. The variable w is the minimum window size and illustrates the minimum amount of data required to be buffered before beginning to process it. Thus at least *2*w* amount of data needs to be buffered be proceeding ahead with the algorithm. The index j is varied from i+w to t-w to search for the ideal split of the data sequence. s is the point that is the most optimal split of the data sequence. It divides the data into two sequences both with the best fits of sub-Gaussian curves.

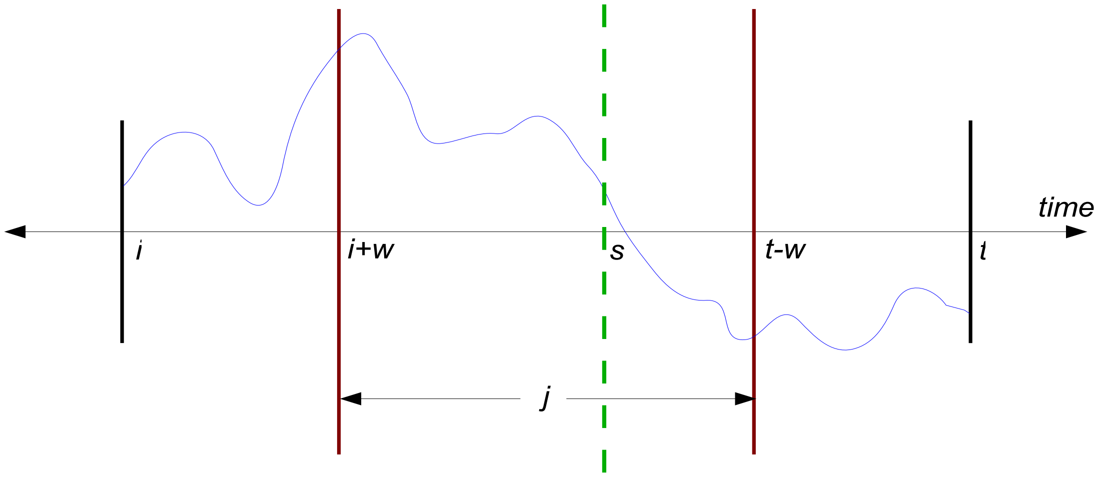


Figure 2.2.1 A time-line of a simulated sample signal to show the different indexes and variables used

The state change detector looks for the most likely split in all the data seen so far. It does this by computing a split score for all possible split points. The split that yields the best score becomes the best split. Each possible split point breaks the data into two sequences, one prior to the split point and one after the split point. Each sequence is modeled by fitting a sub-Gaussian function to its data. These sub-Gaussian fits are then monitored and compared with each other over time for detecting a state change. A state change occurs when the sub-Gaussian fits of the two sequences of the task pair become distinctly separate. Once a state change is detected, the sub-Gaussian fitting starts again from the last detected state change. This process continues till the state change detector stops receiving a signal.

2.2.1 Algorithm

The algorithm proposed here primarily consists of three loops. The outermost loop is used to read a new data point. The next loop identifies the best split of the data into two data sequences. This is done by computing and comparing parameters of a sub-Gaussian fit to the data for all possible split points. Once the data has been split, the inner-most loops are used to compute the parameters of the sub-Gaussian fits to the two data sequences obtained.

1. Let the input signal be represented by:

   $$x_i, x_{i+1}, x_{i+2}, x_{i+3}, ..., x_t$$

   where each x_i is a discrete measurement and the current time is represented by t. x_{i-1} is the last detected state change.

2. Let a state change time j vary between $(i+w)$,$(i+w+1)$,...,$(t$-$w)$ where w is the minimum window size. The minimum window size refers to the minimum amount of data that can be aggregated into a state.

3. Compute the frequency distribution of the two parts that the signal is divided into and denote it as $y_{i,j}$ and $y_{j+1,t}$.

4. Compute r $(y_{i,j})$ and r$(y_{j+1,t})$ as the residuals of the sub-Gaussian fits to the two window sequences.

5. The index j that provides the smallest residual r(j) is chosen as the best split point.

6. An overlap statistic O (a_1, b_1),(a_2, b_2) is computed from the sub-Gaussian fits of the two window sequences. (a_1, b_1), (a_2, b_2) are the respective parameters of the sub-Gaussian fit to the first and second window sequence.

7. Compare the calculated overlap statistic to a threshold P to decide if a state change has occurred.

8. If state change is detected, update i with *j+1*. Identify and output the time of the state change and the time of detection of the state change. Adjust buffer so as to have *i+(2*w)* samples worth of data.

9. Read new data point, increment t and continue from step 1.

2.2.2 Gaussian Fitting

The probability distribution function of the Gaussian distribution is represented by Equation 2.2.3 and is shown in Figure 2.2.2.

$$y = ke^{\frac{-(x-\mu)^2}{2\sigma^2}} \tag{2.2.3}$$

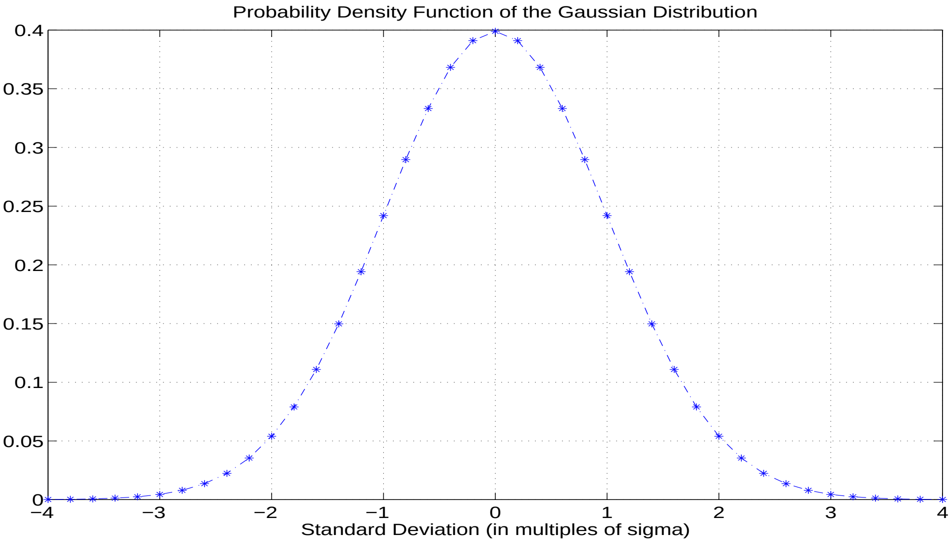


Figure 2.2.2 The probability distribution of the Gaussian distribution

Fitting a Gaussian curve to a signal generally involves solving for the parameters: μ, σ and k in the above equation. The parameter μ signifies the amount of shift required to be made in the mean of the Gaussian curve to make it equal to the mean of the signal and the parameter σ signifies the spread of both tails of the Gaussian curve. The third parameter k is a scaling factor. The best fit is found when a set of parameters yields the least residual. Before fitting the Gaussian curve to the data, the frequency of the data ($y_{i,t}$)is computed as shown in Equation 2.2.4. The range of $y_{i,t}$ is limited to [-4,4]. The bin size when computing the frequency is considered as 0.2.

$$y_{i,t} = freq(x_i, x_{i+1}, x_{i+2}, ..., x_t) \tag{2.2.4}$$

The residuals (e_i) are computed by finding the difference between the frequency of the data points (y_i) and the Gaussian curve as shown in Equation 2.2.5. The sum of the square of residuals is denoted by E (Equation 2.2.6). The residuals are given as:

$$e_i = y_{i,t} - ke^{\frac{-x_i^2}{2}} \qquad i = 1...t \tag{2.2.5}$$

Thus, the sum of squares of residuals would be:

$$E = \sum(e_i^2)$$

In other words,

$$E = \sum(y_i - ke^{\frac{-x_i^2}{2}})^2 \tag{2.2.6}$$

Now, in order to obtain the best fit, the sum of square of residuals has to be minimized by varying the value of k. This can be done by computing the partial derivative of E with respect to k as shown in Equation 2.2.7. On solving for the partial derivative (Equation 2.2.8), the value of k obtained gives the minimum residual between the data and the sub-Gaussian.

From 2.2.6, the partial derivative of E with respect to k is given as:

$$\frac{\partial E}{\partial k} = 2\sum(y_i - ke^{\frac{-x_i^2}{2}})(-e^{\frac{-x_i^2}{2}})$$

Equating this to 0 to solve for point of minima,

$$\frac{\partial E}{\partial k} = 0$$

Thus we get:

$$2\sum(y_i - ke^{\frac{-x_i^2}{2}})(-e^{\frac{-x_i^2}{2}}) = 0 \tag{2.2.7}$$

On expanding,

$$-\sum((y_i e^{\frac{-x_i^2}{2}}) + k\sigma(e^{\frac{-x_i^2}{2}})^2) = 0$$

On solving for k, we get:

$$k = \frac{\sum(y_i e^{\frac{-x_i^2}{2}})}{\sum(e^{\frac{-x_i^2}{2}})^2} \tag{2.2.8}$$

Examples of fitting the Gaussian curve to sample simulated data along with the respective parameters are shown in Figure 2.2.3. The reason that we chose to analyze a normalized signal is that its mean is 0 and standard deviation is 1. Thus, instead of solving for three parameters, only parameter k is to be found. At a given instant of time, the parameters need to be estimated for all possible window combinations. This needs to be done again at the each time step. Thus reducing the number of parameters from three to one helps drastically reduce the computation time. Since this whole approach is proposed keeping in mind that the processing needs to be done in real-time, saving computation time becomes very important.

2.2.3 Sub-Gaussian Fitting

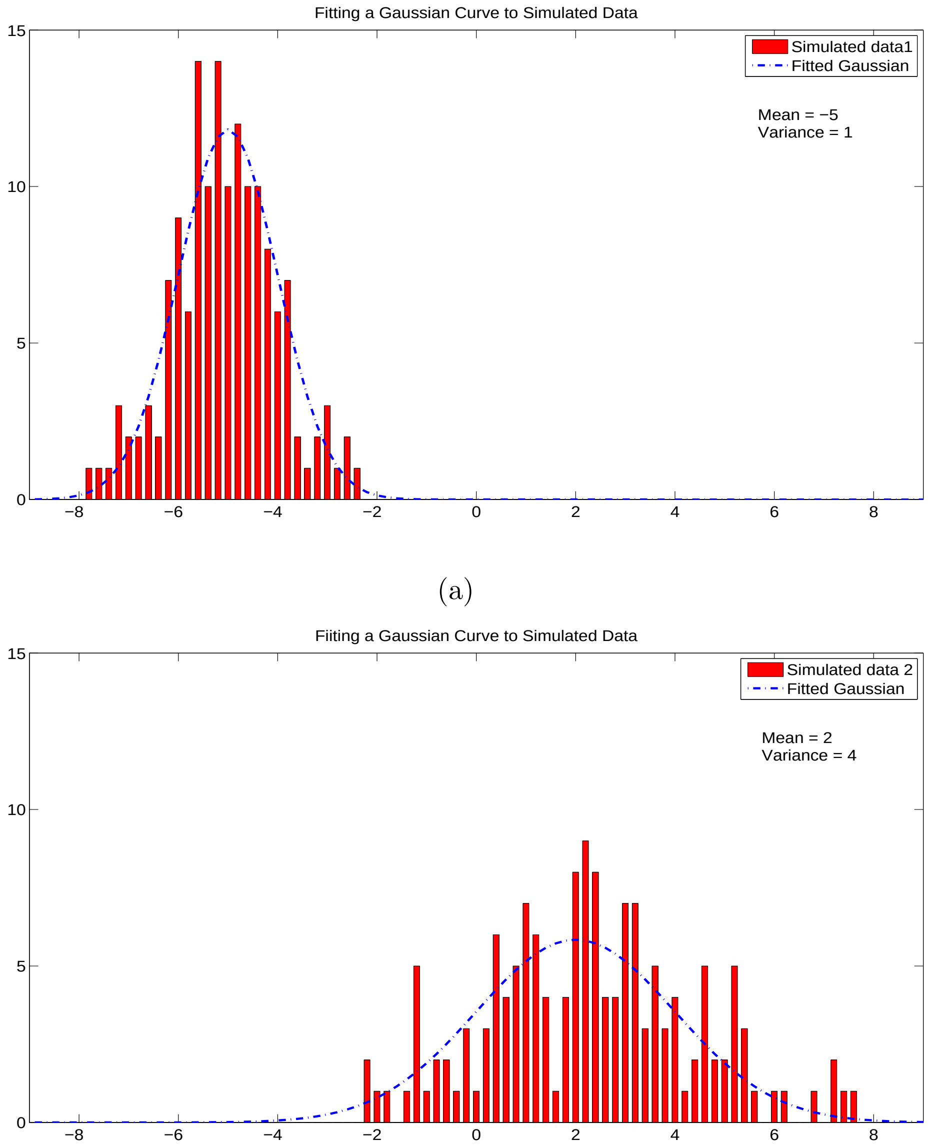
A sub-Gaussian curve can be defined as a curve that follows the Gaussian distribution between a range of values and is *zero* otherwise. Thus it can be represented in the form of a piecewise function as shown in Equation 2.2.9. Examples of the sub-Gaussian curve with its parameters are shown in Figure 2.2.4.

The function defining a sub-Gaussian is given as:

$$\begin{aligned} Y &= 0 && x < a \\ Y &= ke^{\frac{-x^2}{2}} && a \leq x \leq b \\ Y &= 0 && x > b \end{aligned} \tag{2.2.9}$$

Fitting a sub-Gaussian to the data means solving for the parameters a, b and k in the above equation. This implies that the sub-Gaussian curve needs to be scaled appropriately to the data, by varying the scaling factor k, within the range a-b. To do this, we first find the residuals and the sum of square of residuals as shown in Equations 2.2.5 and 2.2.6. The partial derivatives are then computed to find the optimal k.

The residuals will be given by the equation:


Fitting a Gaussian Curve to Simulated Data

(a)

Fiiting a Gaussian Curve to Simulated Data

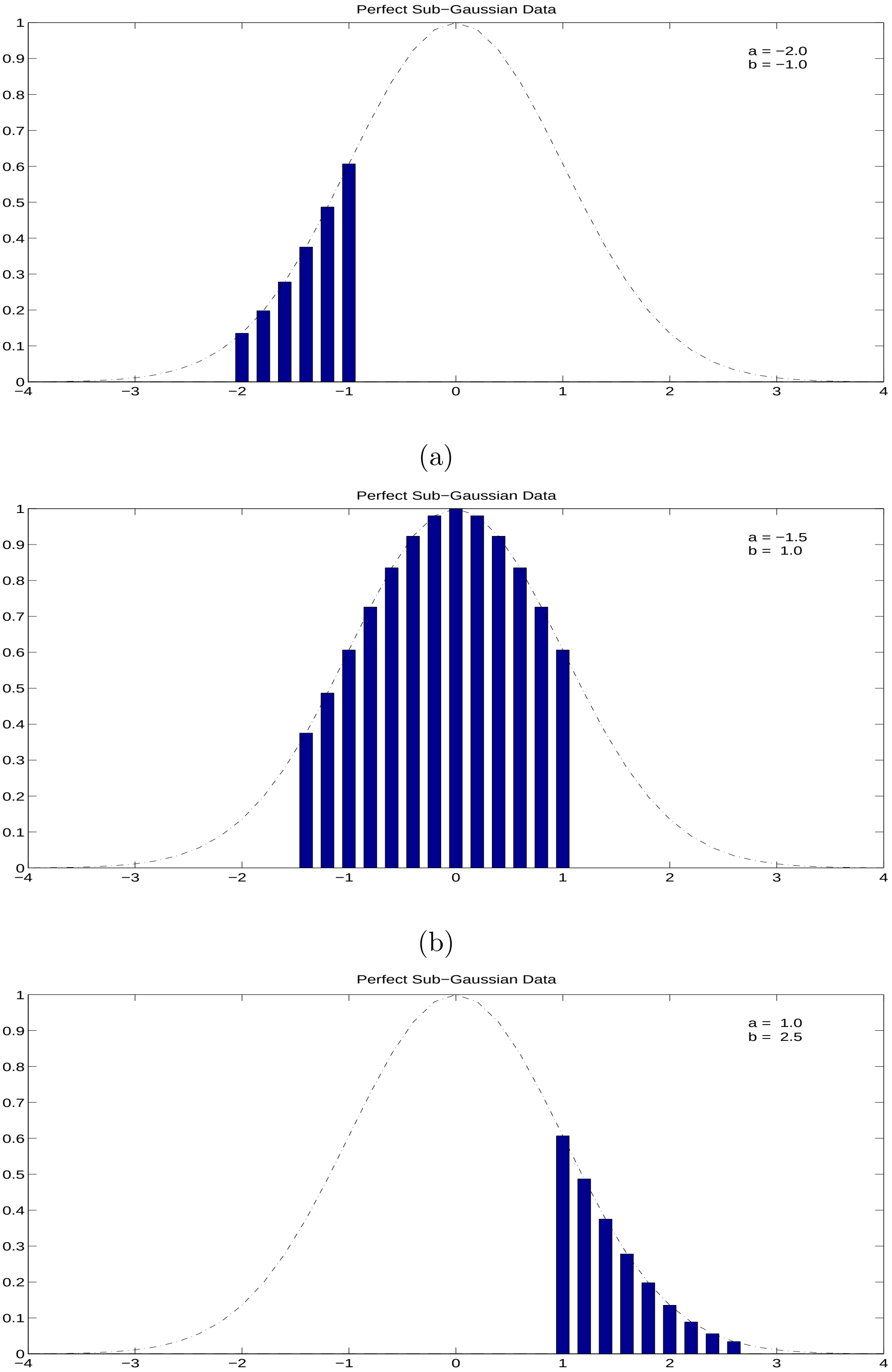
(b)

Figure 2.2.3 Plots of fitted Gaussian over simulated data with different means and variances

$$E = y_{i,t} - Y$$

The partial derivative of E with respect to k is given as:

$$\frac{\partial E}{\partial k} = 2\sum(y_{i,t} - ke^{\frac{-x_i^2}{2}})(-e^{\frac{-x_i^2}{2}})\Bigg|_{x=b}^{x=a}$$



(a)

(b)

(c)

Figure 2.2.4 Plot of simulated perfect sub-Gaussian data with different range

Equating this to 0 to solve for point of minima,

$$\frac{\partial E}{\partial k} = 0$$

Thus we get:

$$2\sum(y_{i,t} - ke^{\frac{-x_i^2}{2}})(-e^{\frac{-x_i^2}{2}})\Bigg|_{x=b}^{x=a} = 0 \tag{2.2.10}$$

On solving for k, we get:

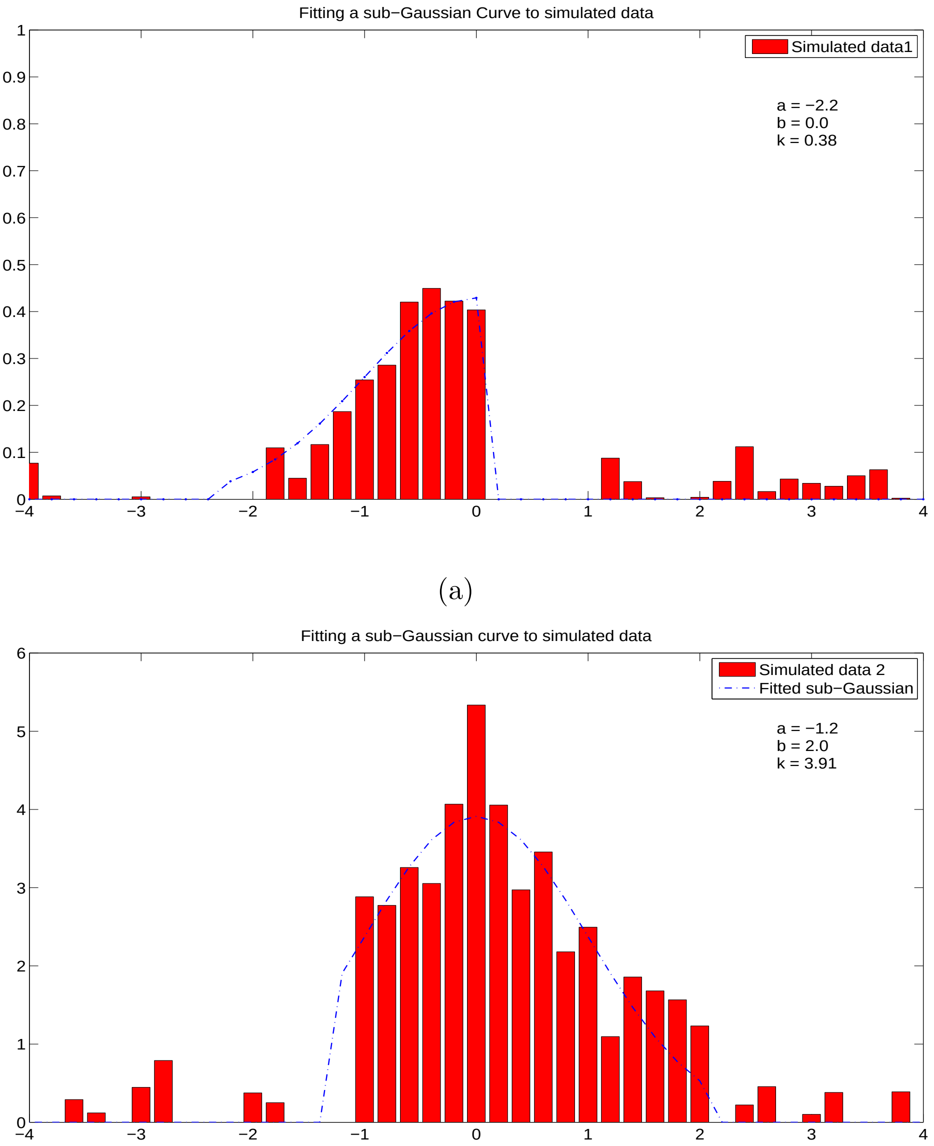
$$k = \frac{\sum(y_{i,t}e^{\frac{-x_i^2}{2}})}{\sum(e^{\frac{-x_i^2}{2}})^2}\Bigg|_{x=b}^{x=a} \tag{2.2.11}$$

Since 99.6% of the Gaussian data lies within the range -3σ to $+3\sigma$, a and b are allowed to vary within this range with a certain minimum gap between them. If the distance between a and b is too small, the detector will trigger more often. It will detect changes which are actually minor fluctuations of the signal. Thus a minimum gap of 1σ is maintained betweeen a and$b$. The range of a thus becomes -3σ to $+2\sigma$ and that of b becomes $a + 1\sigma$ to $+3\sigma$. In this way, for each combination of the value of a and b, an optimal k is computed and the respective sum of residuals (E) is found. For a given sequence of data, the best sub-Gaussian fit is chosen as the one that results in the least E. This fit is recognized by the parameters a, b and k. Examples of fitting a sub-Gaussian curve to sample simulated data along with the respective parameters are shown in Figure 2.2.5.

2.2.4 Splitting the Data

The underlying purpose of this work is to be able to do the signal analysis in real-time. To this effect, a technique called the sliding time windows is used to analyze the data. The data is split into two windows of time with a some minimum amount of data in each window. The minimum window size, w can be decided based on either the minimum amount of data points that should be present in the window or as a function of time (minimum seconds / minutes worth of data). This concept further explained by Figure 2.2.6.

At any given time, we start by encapsulating the pre-decided minimum amount of data from the beginning of the signal into the first window and the rest in the second



(a)

(b)

Figure 2.2.5 Plots of fitted sub-Gaussian over simulated data with different parameters: a, b and k

window. We hypothesize that each window has enough data to reflect the range of the sub-Gaussian that it varies in. we then perform sub-Gaussian fitting both windows and compute the respective parameters. The average of the sum of residuals for each window gives an indication of the goodness of the sub-Gaussian fit. This value for both windows is noted down as we proceed to the next combination of windows at the same

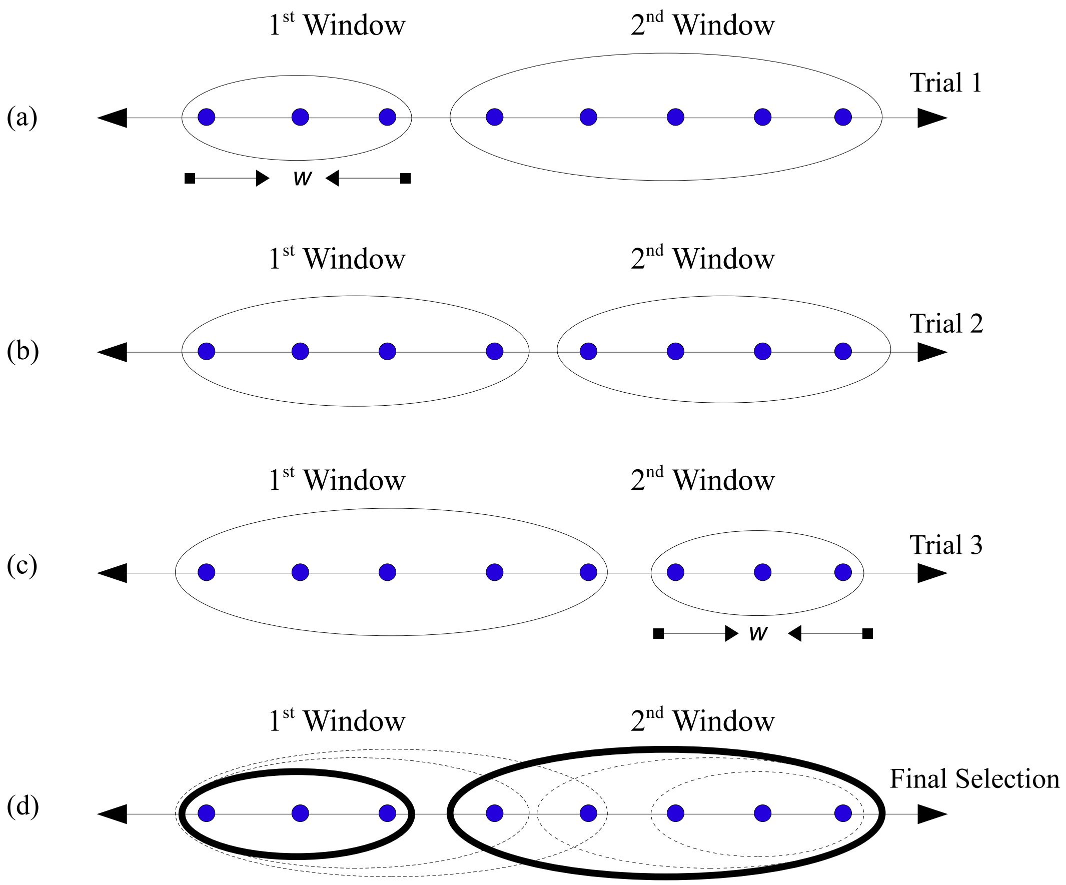


Figure 2.2.6 An example of the Sliding Time Windows at one instant of time. The horizontal axis denotes time and it has data points marked on it in the form of blue dots. The ovals are used to display the data points present in a window. w denotes the minimum window size. Figures (a), (b) and (c) show the different window combinations possible at the given instant of time and Figure (d) highlights the combination of windows that is chosen for analysis at the given instant of time.

time instant and compute the sum of residuals again. Once all possible combinations of windows have been tried, we chose the best split of the data as the one that gives the least combined average of the sum of residuals for its respective windows. In this way, at every point in time, the data sequence is split into two parts in such a way that together, they both have the least residual when a sub-Gaussian is fit to the data.

At the next instant of time, a new data point gets added to the sequence. The whole data stream is again processed as described above to yield the best split of the sequence. The process is called 'Sliding Time Windows' because at each instant of time, we slide the split point to process different window combinations.

2.2.5 Overlap Statistic

Given the sub-Gaussian fits to both data sequences and the respective parameters of the fits (a_1,b_1 for the first window of data and a_2,b_2 for the second window of data), we compute a statistic called the overlap statistic as shown in Equation 2.2.12 - 2.2.14.

$$P_1 = \frac{b' - a'}{b_1 - a_1} \tag{2.2.12}$$

where P_1 signifies amount of overlap with respect to first window

$$P_2 = \frac{b' - a'}{b_2 - a_2} \tag{2.2.13}$$

where P_2 signifies amount of overlap with respect to second window. Taking average of both of the above values, we get the overlap statistic as:

$$Overlap = \frac{P_1 + P_2}{2} \tag{2.2.14}$$

Here, the value of $b'$and a' depends on the relative position of the two fits and are calculated as a piecewise function as shown in Equations 2.2.15 and 2.2.16

a' is calculated as:

$$\begin{gathered} a' = a_1 \quad a_2 \leq a_1 \ \& \ a_1 \leq b_2 \\ a' = a_2 \quad a_1 < a_2 \ \& \ a_2 \leq b_1 \\ a' = 0 \ \ otherwise \end{gathered} \tag{2.2.15}$$

And b' is calculated as:

$$\begin{gathered} b' = b_1 \quad a_2 \leq b_1 \ \& \ b_1 \leq b_2 \\ b' = b_2 \quad a_1 \leq b_2 \ \& \ b_2 < b_1 \\ b' = 0 \ \ otherwise \end{gathered} \tag{2.2.16}$$

The variable P_1 signifies the amount of overlap between the two fits as a ratio of the overlapping region to the region of the first fit. And the variable P_2 signifies the amount of overlap between the two fits as a ratio of the overlapping region to the region of the second fit. So the actual overlap becomes the average of these two ratios. The range of values that the overlap statistic can take is $[0, 1)$.

2.2.6 Pseudo-Code

The entire algorithm described above can be recapitulated in the following pseudo code:

```
Read data point $x_t$

Add to buffer: $x_i$,$x_{i+1}$,$x_{i+2}$,...,$x_t$

    while $(x_t)$

        for $j = i + w$ to $t - w$

            $r(j) = subGaussianFit(i, j, t)$

        $s = min(r(j))$

        $O = Overlap(a_1, b_1, a_2, b_2)$

        if $O < P$: state change detected, i=j+1
```

2.2.7 Examples of Actual Data

Some examples of the actual data during various stages of the algorithm are provided here. Figure 2.2.7 shows a plot of the actual data used in this research. As can be seen in the figure, the signal appears to change its range of values when the task changes. This is the state change we seek to detect.

These states represent the varying intensity tasks that the subject was performing during the recording of the signal. Figure 2.2.8 shows the histogram of the complete data at the end of both tasks. It can be seen from this figure that the distribution of the signal almost matches a Gaussian distribution. The signal thus satisfies our first assumption of being characterized by a Gaussian distribution.

Figure 2.2.9 shows the distribution of the two parts of the signal at the end of the task pair. It is visible that the distributions of the two parts of the signal are distinct and vary in different sub-regions of the Gaussian curve. This supports our hypothesis about the variation of the different states of a signal belonging to different sub-regions of the Gaussian distribution.

Figure 2.2.10 shows the sub-Gaussian fits to the two tasks of the data along with the histograms of the tasks. These fits are obtained by calculating the optimal values

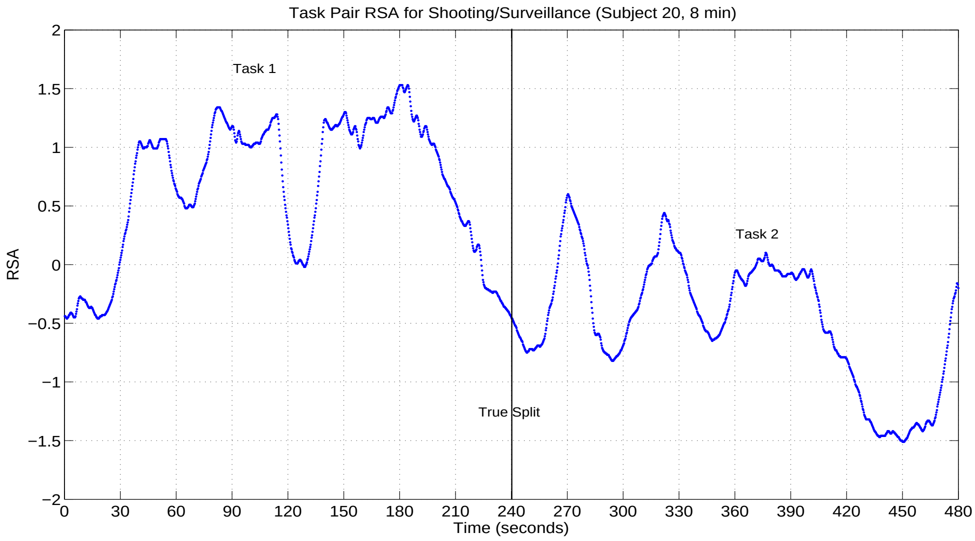


Figure 2.2.7 An example of the Gauss normalized RSA data of a 8 minutes long task-pair. The plot also shows the ground truth - the point in time when the task actually changes.

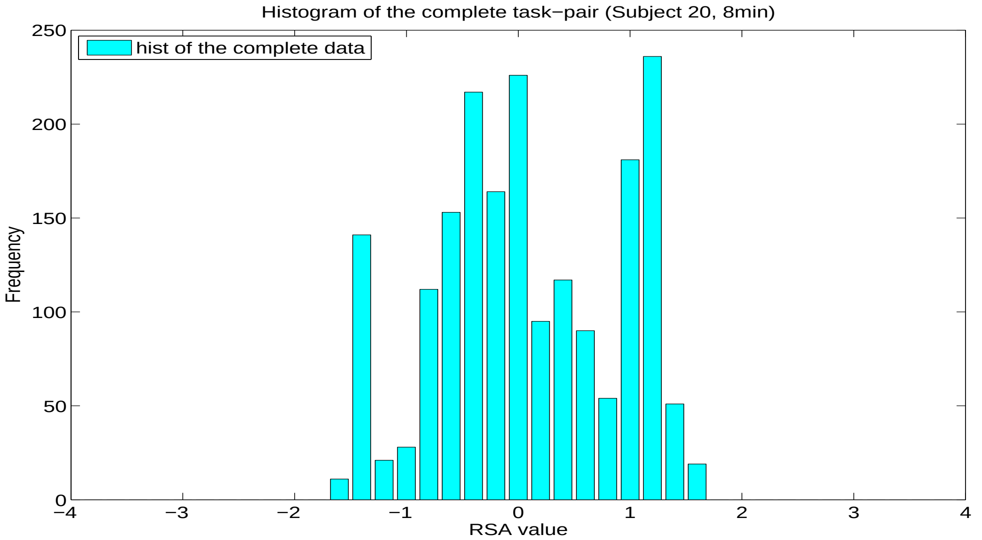


Figure 2.2.8 A plot of the histogram of the complete data (both task-pairs).

of the parameters (k, a and b) so that the sum of residuals is minimum. The values of the parameters (a_1,b_1) and (a_2,b_2) show that there is very little overlap between the regions of the sub-Gaussian fits. Thus a state change can be said to have occurred.

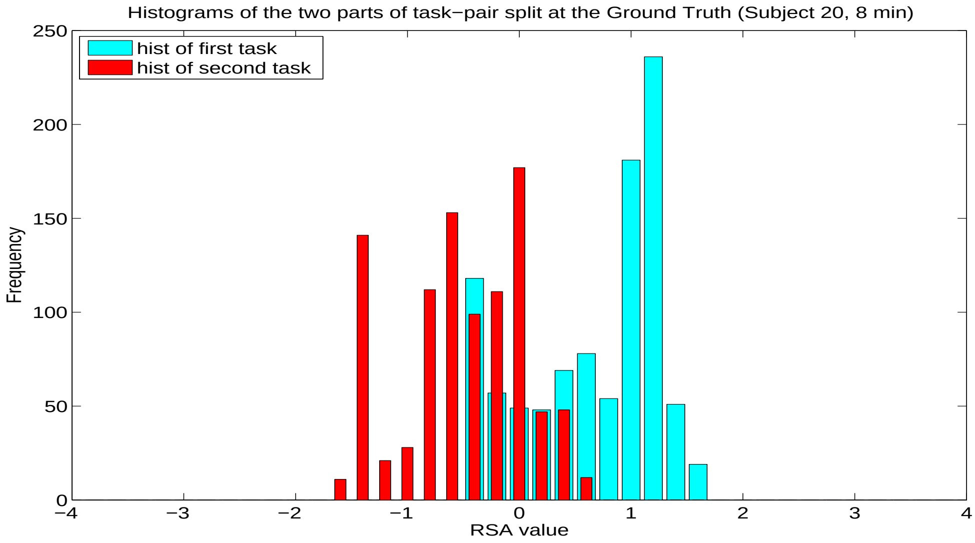


Figure 2.2.9 A plot of the histograms of the RSA of the two tasks in the data shown above plotted separately. It can be seen that the two tasks have distinctly separate distributions

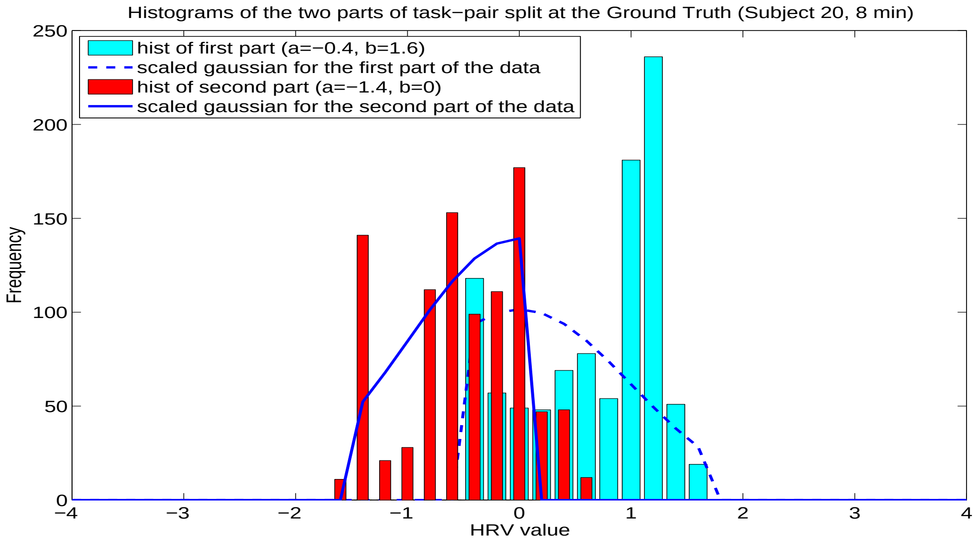


Figure 2.2.10 A plot showing the sub-Gaussian fits of the two tasks shown above superimposed on the histograms of the respective tasks. The plot also shoes the parameters (a_1,b_1,a_2,b_2) of the sub-Gaussian fits.

CHAPTER 3
EXPERIMENTAL RESULTS

This chapter describes the results obtained on running the methods on the data. The data was analyzed off-line but we emulated on-line processing. We ran the entire recording of a sample of data through the state change detector. While processing the data we kept logging the state change detections made. If a state change was detected, the process would start over from the point of detection after buffering enough data. This loop would go on until the end of the recording of the data sample.

3.1 The Data

In this section, the background of the data used for this work is described. We also mention the origin of the data and the way it is organized. In the end, a pre-analysis of the data is presented. This helps select the data samples that will be used for testing our algorithm.

3.1.1 Data Acquisition

The data used for this research was obtained from the experiments conducted by Fishel [6] on 41 subjects. This data consists of a physiological measure of the heart rate variability of a person. It is called the respiratory sinus arrhythmia (RSA) and corresponds to a frequency analysis of the time intervals between consecutive heart beats. A single data file consists of a series of RSA values collected over the period of the experiment for each subject. The RSA values are sampled at a frequency of 4 Hz. In her experiments [6], Fishel made her subjects perform tasks of different intensities. A particular recording of such an experiment required the subject to perform a high intensity task for half the duration of the recording and a low intensity task for the last half. A recording consisting of such two tasks constitutes a *task-pair*. Each experiment consists of task-pairs of variable lengths, specifically 1 minute, 2 minutes, 4 minutes, 8 minutes and 16 minutes long. A certain minimum amount of data needs to be buffered by our algorithm before analyzing it. This is determined by the minimum window size (w). The 1, 2 and 4 minutes long tasks do not have substantial amount of data left to be analyzed after accounting for this buffer. So only the 8 minutes and 16 minutes long task pairs are considered for the experiments. This leaves us with 82 data files: 41 for the 8 minutes long task-pairs and another 41 for the 16 minutes long task-pairs.

3.1.2 Experimental Setup

The program written for implementing the state change detector was codded in C. The specifications of the computer used for implementing the algorithm are shown in Table 3.1.1.

Computer	IBM PC
Processor	Pentium 4, 3.0 GHz
Memory	512 MB
OS	Windows XP
Platform	Microsoft Visual C++ 6.0

Table 3.1.1 System Specification.

The input data for a particular subject consists of individual files for the five task-pairs of varying lengths that the subject was made to perform. All five data files for different subjects are assembled into numbered folders. Thus each folder represents one subject. The folders are placed together in a directory. The program processes all the task-pairs by accessing a file from one subject at a time. Every task-pair data file is run through the state change detector which emulates real-time processing. After processing one task-pair file, the program moves onto the next one.

3.1.3 Some Definitions

Figure 3.1.1 shows a sample result. The figure shows the RSA of a task-pair that has already been processed through the state change detector. The black line in the center of the figure marks the ground truth. The ground truth is the point where the state of the signal is ideally supposed to change. There are two additional types of markings on the figure. The first (in green) shows the state change detections or the point of time where a state change was detected (s). The other marking (in red) shows the time at which the state change was detected (t). The time of a state change detection is at least w data points after the state change.

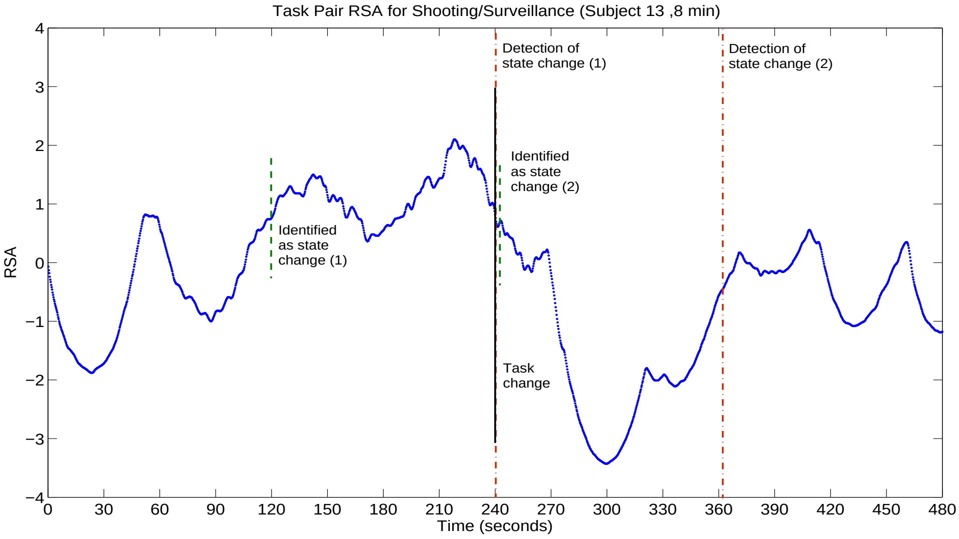


Figure 3.1.1 A sample result that shows the identified state changes along with the actual task change on the RSA of the input data.

In order to evaluate our results, we use the following definitions. At a given time t, consider a task change to have occurred at time s as shown in Figure 3.1.2. A *true positive (TP)* occurs when a state change is detected within a threshold of $\pm T$ seconds of the task change. A *false positive (FP)* occurs when a detected state change lies outside the threshold T of the known state change. A state change for a given sample of data is considered to be *missed* if a true positive is not detected for that data sample. The range of a detection being a TP or a FP is shown in Figure 3.1.2.

We tested different values of the threshold T ranging from ± 10 seconds to ± 80 seconds. But for our analysis, $T = \pm 60$ seconds is selected as the the threshold that determines whether a state change detection matches a task change. Frequency analysis was done on the data recorded in Fishels's experiments [6] to obtain the RSA values. This required 64 seconds worth of data to be buffered continuously for the analysis. Thus the RSA value at a given point of time depends on data collected in the previous 64 seconds. So after a task change has occurred, some part of the RSA value will still reflect the state of the previous task for the next 64 seconds. The RSA values after 64 seconds will depend entirely on the state of the new task. Thus we chose the threshold

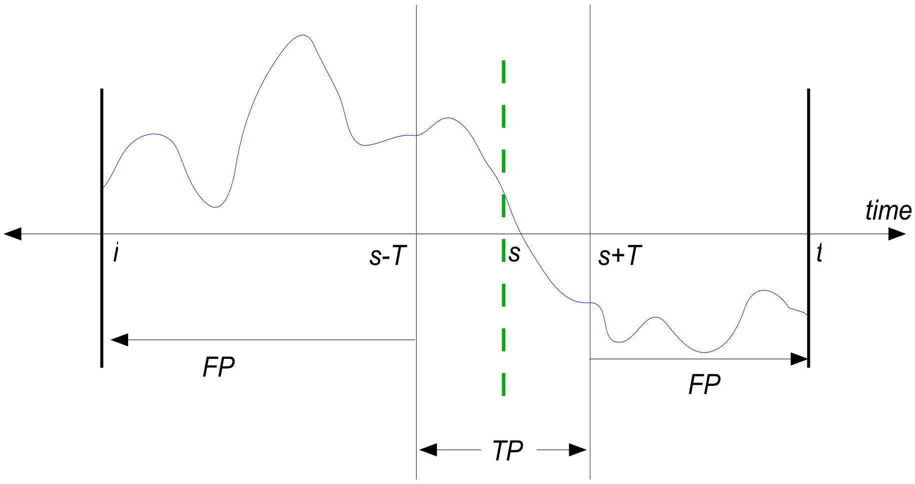


Figure 3.1.2 An example of simulated data that highlights the range of the true positives (TP) and the false positives (FP).

T to be ±60 seconds to approximately account for the 64 second buffer.

3.1.4 Data Analysis

Even though a state change should ideally be present for each task change (once per recording), it is not always the case. In many cases, the state change does not line up with a change in the tasks. This could happen because the state of the signal does not follow the trend of the task. Other than that, the most culpable reasons can be errant equipment, human error while recording data or non-responsiveness of the subject to the tasks. Thus a pre-analysis of the data was done to determine the samples that would be usable for this research.

The data was first analyzed subjectively on the basis of visual cues. This was done by looking at the raw data plots and categorizing the task-pairs into one of the three classes: differentiable, not differentiable or cannot say. Each sample of data was then analyzed statistically by running it through a modification of our algorithm. We processed the data off-line and considered all the data points at the same time. The data sequence was split at the known state change. The parameters of the sub-Gaussian fits were then found and the overlap statistic was computed. If the overlap statistic was less than a threshold (0.35), the data was deemed differentiable, otherwise it was not considered to be differentiable. These statistics are compiled in the table 3.1.2 along with the visual analysis of the data.

Visual Analysis		**Statistical Analysis**	
No. of Task-pairs	**Analysis**	**No. of Task-pairs**	**Analysis**
23	Visually differentiable	20	Differentiable
		3	Not differentiable
32	Visually not differentiable	6	Differentiable
		26	Not differentiable
27	Cannot say	9	Differentiable
		18	Not differentiable

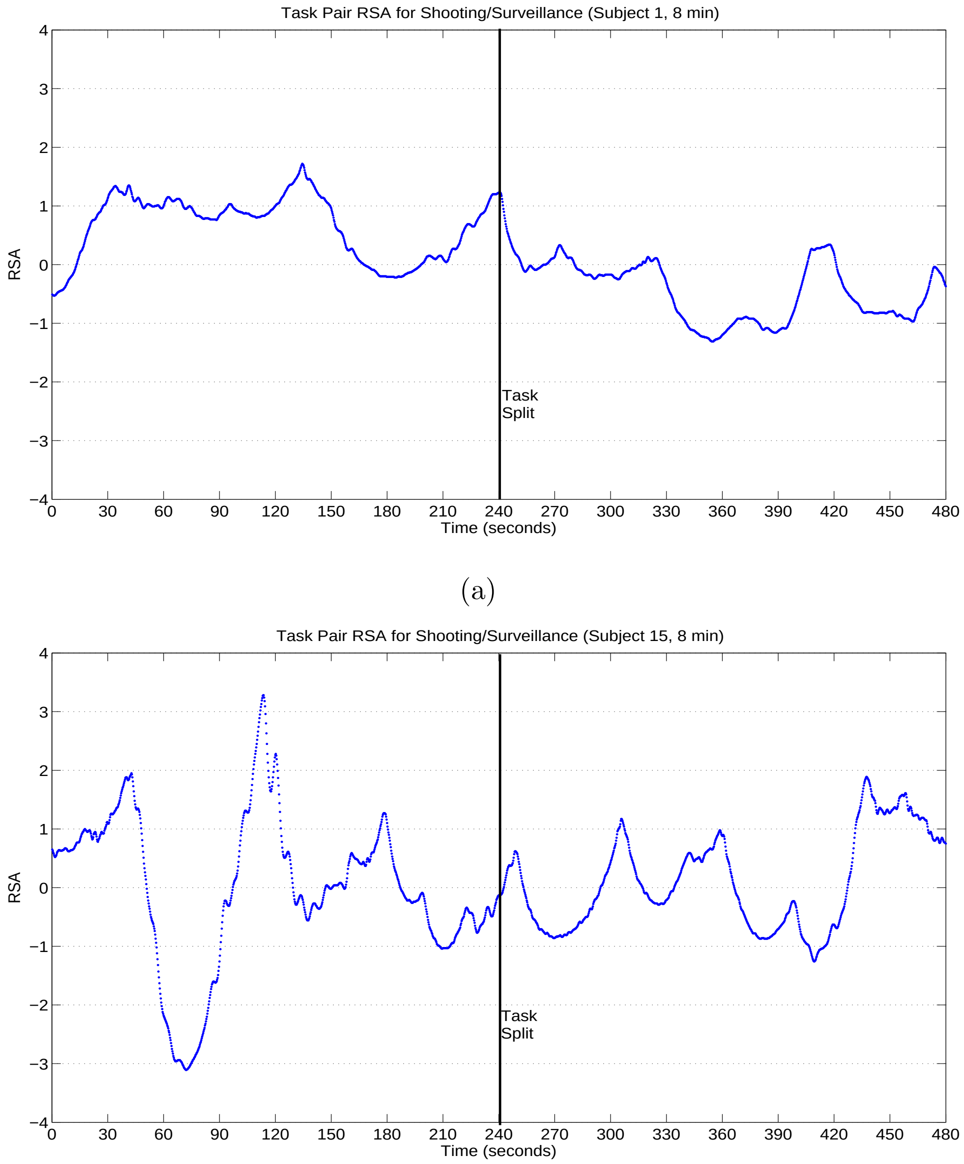
Table 3.1.2 The classification of data both visually and statistically

An example of a visually as well as statistically differentiable task is shown in Figure 3.1.3 (a). In this case, the task change does correspond with a state change in the signal. In contrast, an example of a visually and statistically not differentiable task is shown in Figure 3.1.3 (b). In this example, even though the task changes at 240 seconds, there is no evidence of state change occurring near the task change. It can also be observed from Table 3.1.2 that the statistical analysis of the data is in agreement with the visual categorization of the task-pairs in most of the cases.

This analysis indicates the number of data samples in which the task change could be effectively determined. Doing this analysis helps select and process the 'differentiable' samples of data. The 35 samples of data that were statistically classified as differentiable (Table 3.1.2) were considered for further analysis.

3.2 Parameters

The outcome of our algorithm primarily depends on two parameters: the minimum window size and the overlap statistic threshold. The *minimum window size* helps decide the minimum amount of data that needs to be buffered before processing it. For example, if the minimum window size is two minutes, then we would need to buffer at least four minutes of data. The size of the minimum window is indirectly proportional to the number of state change detections. If the minimum window size is too small, the detector triggers too often. This leads to a higher number of false positives. But on the other hand, because of the frequent triggers, the chances of detecting a state change



(a)

(b)

Figure 3.1.3 An example of data that is both visually and statistically: (a) differentiable, (b) not differentiable.

within the threshold T of the task change increases. Thus a smaller minimum window size also leads to a higher number of true positive detections.

The minimum window sizes that we tested were: 1, 1.5, 2, 2.5 and 3 minutes per task. A minimum window size smaller than 1 minute is not chosen as it would not accumulate enough data in either window to effectively represent the state of the signal.

We also did not test a window size more than 3 minutes for two reasons. The first reason is that the larger the window size, the more the data that the algorithm needs to buffer before processing. Thus there is a longer delay between a state change and the time that it is detected. Figure 3.2.1 shows that the minimum delay after which a state change can be detected is w.

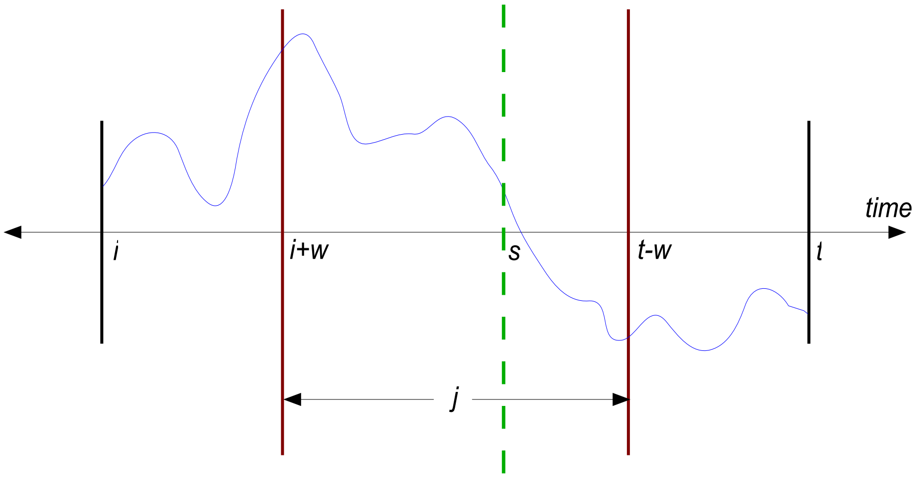


Figure 3.2.1 A time-line of a sample simulated signal shows the minimum delay w after a state change before detection

If a larger window size w' is used (as shown in Figure 3.2.2), then the delay also increases. An increase in the delay is not desired as the algorithm is supposed to be implemented for real-time applications. The second reason for not using a larger window size is that the number of possible ways that the data segment can be split (j) reduces with increase in w. This reduces the total number of possible state change points that can be detected. The reduction in the possible values of j can also be observed by comparing Figure 3.2.1 with Figure 3.2.2. Thus the number of true positives decreases.

The overlap statistic is a measure of the amount of overlap between the regions of the sub-Gaussian fits of the two windows of the data. An overlap statistic value of 0 means that there is no overlap between the regions of the sub-Gaussian fits of the two windows. Thus the data in the two windows varies in completely different sub-sections of a Gaussian distribution. On the other hand, a value of 1.0 means that there is complete overlap between the regions of the sub-Gaussian fits. The overlap statistic is monitored continuously over time. If its value decreases below the *overlap statistic threshold*, a

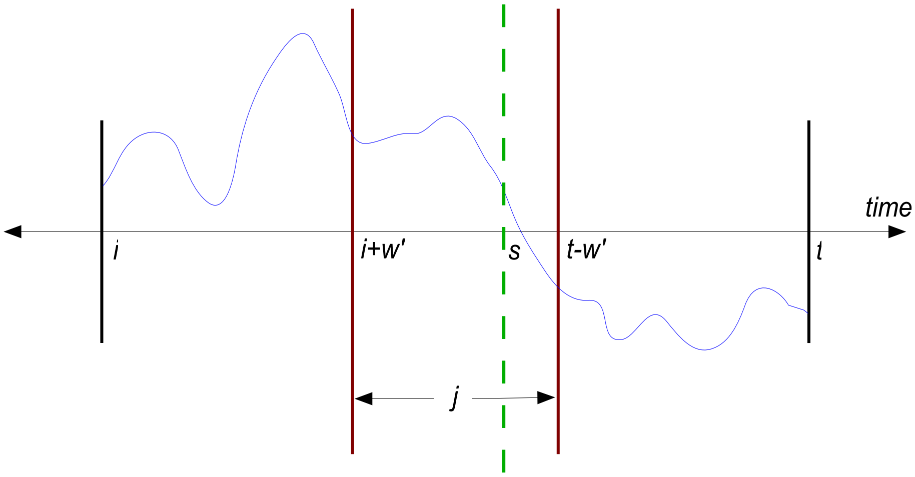


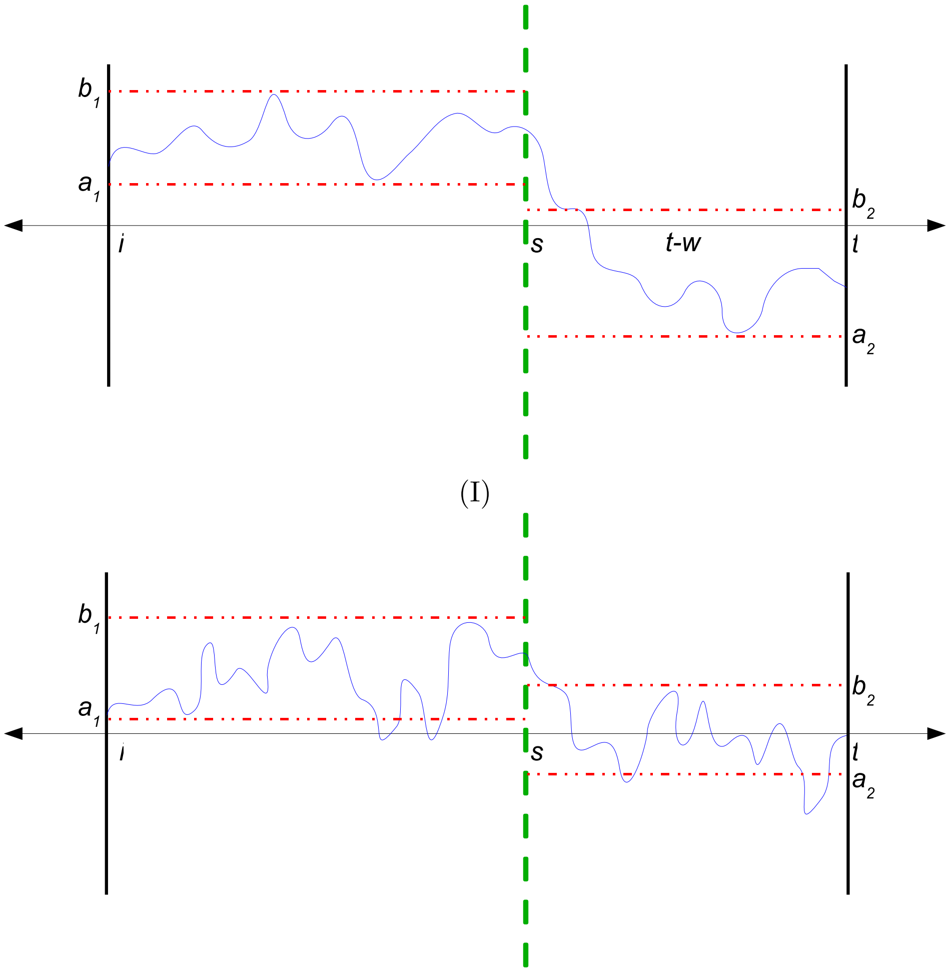
Figure 3.2.2 A time-line of a sample simulated signal shows the minimum delay *w'* after a state change before detection

state change is triggered. The values of thresholds that we tested were: 0.0, 0.05, 0.10, 0.15, 0.20, 0.25, 0.30 and 0.35.

The overlap statistic threshold basically decides the sensitivity of the state change detection. A lower threshold results in detections where the states of the two parts of the signal are distinctly separate as shown in Figure 3.2.3 (I). Thus there are less number of false positive detections. But at the same time, the number of true positives detected also decreases. This happens because in many cases the state of a signal may change its sub-Gaussian region only marginally between two states as shown in Figure 3.2.3 (II). Such changes can be detected if the overlap statistic threshold is increased. As the threshold increases, the number of true positives detected also increases. But on the other hand, the number of false positives also increases.

When choosing the parameters, we keep the following goals in mind. The minimum window size should not be so small that the state change detector triggers too often, resulting in numerous false positives. It should also not be so large that it induces long delays in the detection time. The overlap statistic threshold should not so small that it reduces the number of true positive detections. Neither should it be too large to increase the number of false positives.

The algorithm is tested on the selected data samples for the different values chosen for the two parameters. The results thus obtained yield the state change detections for



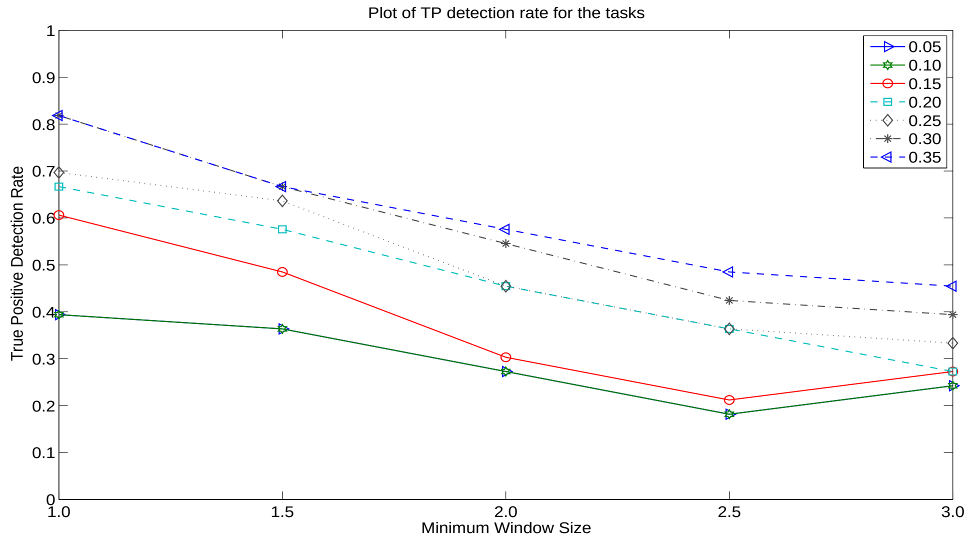
(I)

(II)

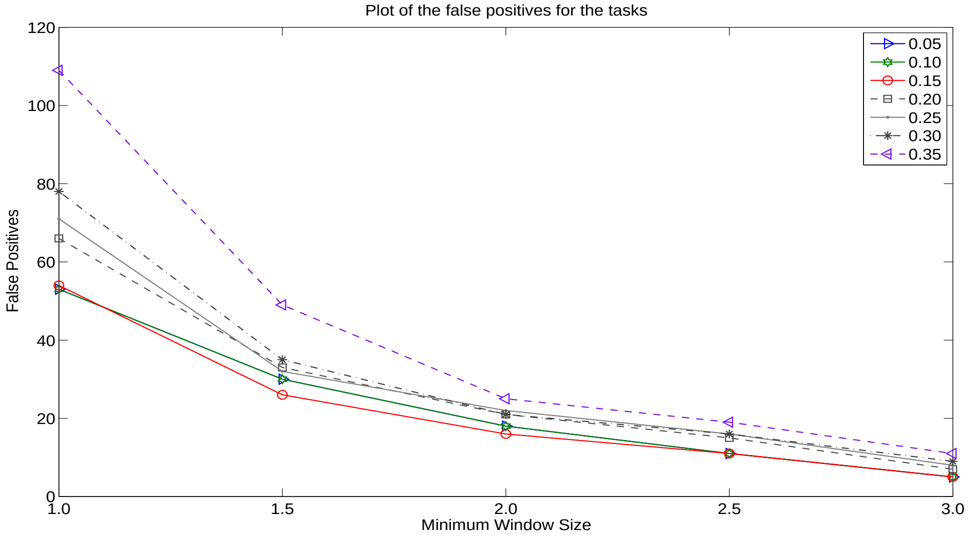
Figure 3.2.3 Examples of simulated data along with the respective parameters of the sub-Gaussian fits of the two parts of the data. (I) In the first example, the regions of the fit are widely separated. Thus the value overlap statistic is equal or close to zero. (II) In this example the regions are overlapping. Thus the value of the overlap statistic is larger.

the various parameter combinations tested. From these results, we generate tables showing the number of false positives and true positive detection rates for different parameter combinations. These tables can also be plotted for a better visual representation.

Figure 3.2.4 (I) shows the plot of the true positive detection rate against the minimum window size for different values of the overlap statistic threshold. The overall trend of the variation of the TP detection rate can be seen in the plot. It can thus be observed that the TP detection rate decreases with an increase in the minimum window size. As explained earlier, this happens because as the minimum window size decreases, the detector has less number chances to trigger.



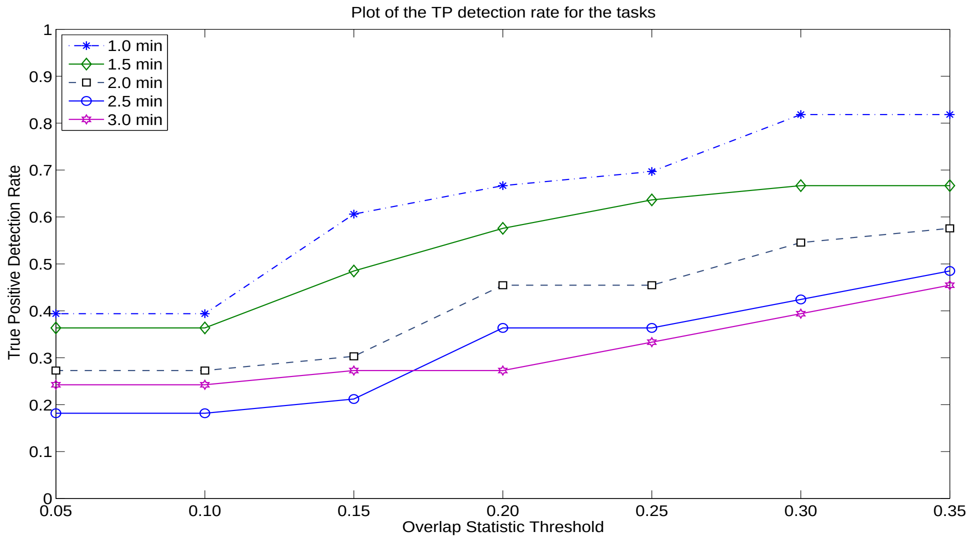
(I)



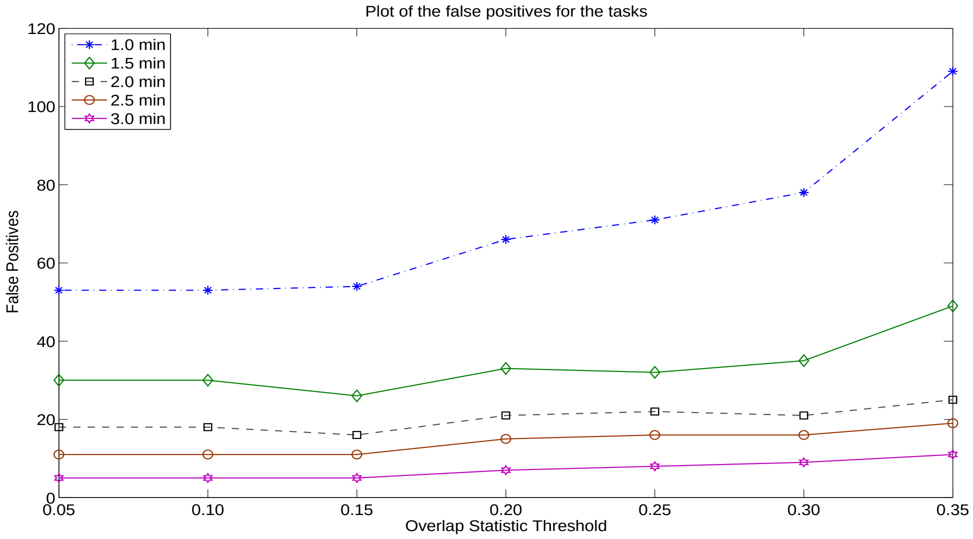
(II)

Figure 3.2.4 Plots showing the effects of varying the minimum window size on the (I) true positive detection rate (II) false positives.

Figure 3.2.4 (II) shows the plot of the number of false positives detected against the minimum window size for different values of the overlap statistic threshold. The trend of the false positives shows a drastic decrease with increase in the minimum window size. The number of false positives detected are high for the 1 minute window because in that case the detector triggers too often. The trend shows that the number of false positives do not decrease by a significant amount if the window size is increased beyond 2 minutes. Figure 3.2.5 (I) shows the variation of the TP detection rate with the overlap



(I)



(II)

Figure 3.2.5 Plots showing the effect of varying the overlap statistic threshold on the (I) true positive detection rate (II) false positives

statistic threshold for different minimum window sizes. The TP detection rate increases with increase in the overlap statistic threshold but the increase is not prominent for overlap statistic threshold greater than 0.20.

The Figure 3.2.5 (II) shows that the number of false positives are relatively stable over the range of the overlap statistic threshold with values of approximately 20 or less for minimum window sizes of 2 minutes or higher.

We also tabulated and plotted the FPs and TP detection rates obtained from the 8 minutes and 16 minutes tasks separately. We found that the trend of the plots remains the same. There is only a change in its values. Thus we base our choice of parameters on the plots shown above.

Based on the above plots, we select values of the parameters that would give us the optimal results. An optimum result would mean obtaining a higher true positive detection rate and a lower number of false positives. The parameters selected for obtaining the results are shown in Table 3.2.1.

Parameter	**Selected value**
Minimum window size	2 minutes
Overlap Statistic threshold	0.20

Table 3.2.1 Selected parameters

3.3 Results

Selecting the values of the parameters as: *minimum window size = 2 minutes* and *overlap statistic threshold = 0.20*, our results can be summarized as in Table 3.3.1.

Visual Analysis		**Statistical Analysis**	
No. of Task-pairs	**Analysis**	**No. of Task-pairs**	**Analysis**
20	Visually differentiable	11	Differentiated
		9	Not differentiated
6	Visually not differentiable	3	Differentiated
		3	Not differentiated
9	Cannot say	2	Differentiated
		7	Not differentiated

Table 3.3.1 The results of running our algorithm on the selected data compared with the visual cues

Figures 3.3.1, 3.3.2 and 3.3.3 show examples of the data where our algorithm was successful in detecting a state change corresponding to a task change. It can be observed from Figure 3.3.1 that a state change was identified at 121 seconds when 241 seconds worth of data had been read. Since the minimum window size is 2 minutes, this means that the state change detector triggered almost immediately after buffering 4 minutes of data. This state change lies outside a threshold of $T = \pm 60$ seconds of the change in tasks and is thus a false positive. It can also be seen that another state change was identified at 243 seconds when 363 seconds worth of data had been read. This state change lies within the threshold of the change in tasks. Thus it is marked as a true positive.

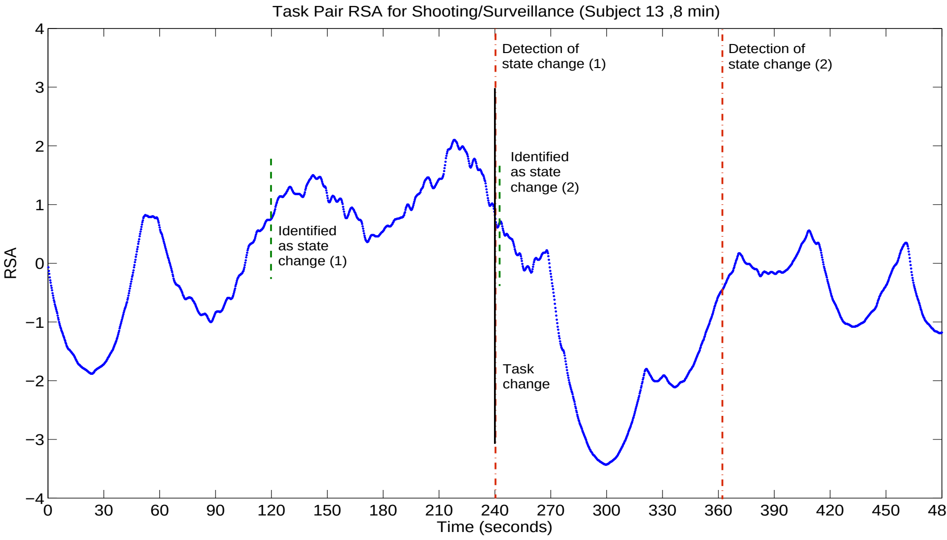


Figure 3.3.1 An example of the RSA data along with the resulting state change detections.

Figures 3.3.2 and 3.3.2 show examples of 16 minute long task-pair data where only one state change was identified during the entire run of the data sample. In both cases, the state change corresponds with the task change and is thus marked as a true positive. An important point to be noticed here is that in both cases, the state change detector is invariant to the fluctuations in the RSA of the data. This means that the detector is so stable that it would not be triggered by temporary fluctuations but only by an overall change in the trend of the data.

Figures 3.3.4 and 3.3.5 show examples of the RSA data for which the state change could not be detected. It can be seen from Figure 3.3.4 that the first state change is

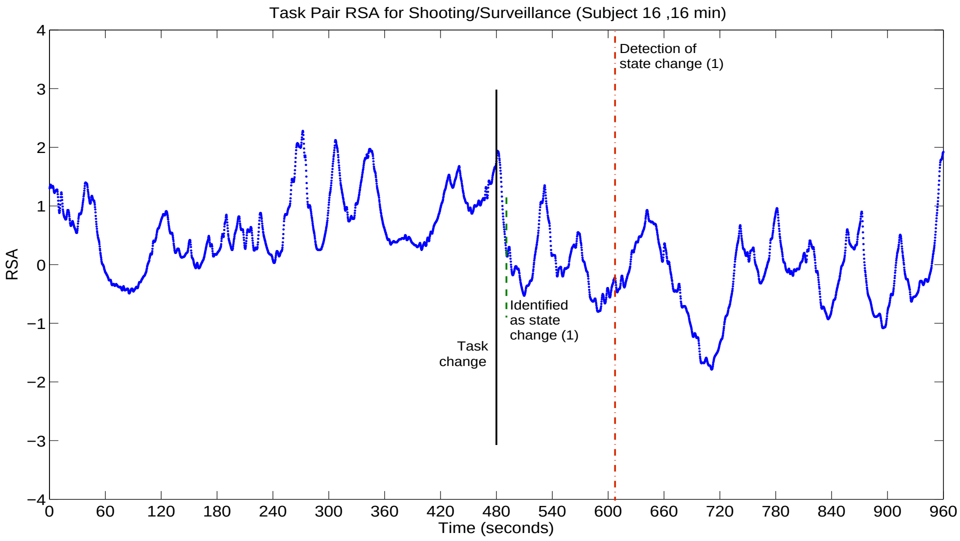


Figure 3.3.2 An example of the RSA data along with the resulting state change detections.

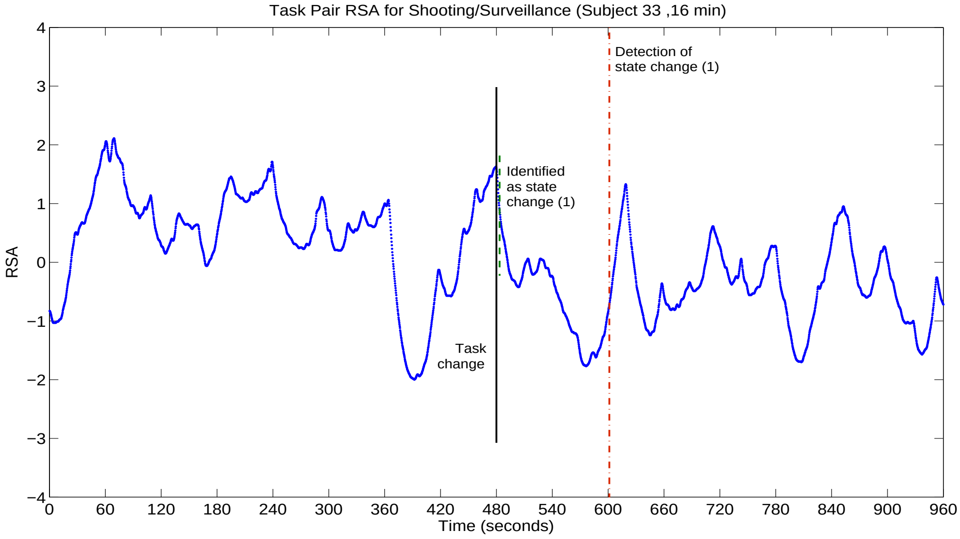


Figure 3.3.3 An example of the RSA data along with the resulting state change detections.

identified at 121 seconds immediately after 4 minutes of data has been buffered. Though there is no significant change in the regions of the sub-Gaussian fits of the two identified states, a state change was triggered. We believe that this is because of the limit imposed on the minimum distance (1σ) to be maintained between the parameters of the fit (a and b). The limit was imposed to make the detector invariant to fluctuations in the signal. But the limit apparently makes the detector sensitive to minor state variations. The figure also shows that another state change was identified at 305 seconds. Though this detection is not a true positive, it is clear from the figure that the state of the signal did change at that point. Figure 3.3.5 shows another example where two false positives

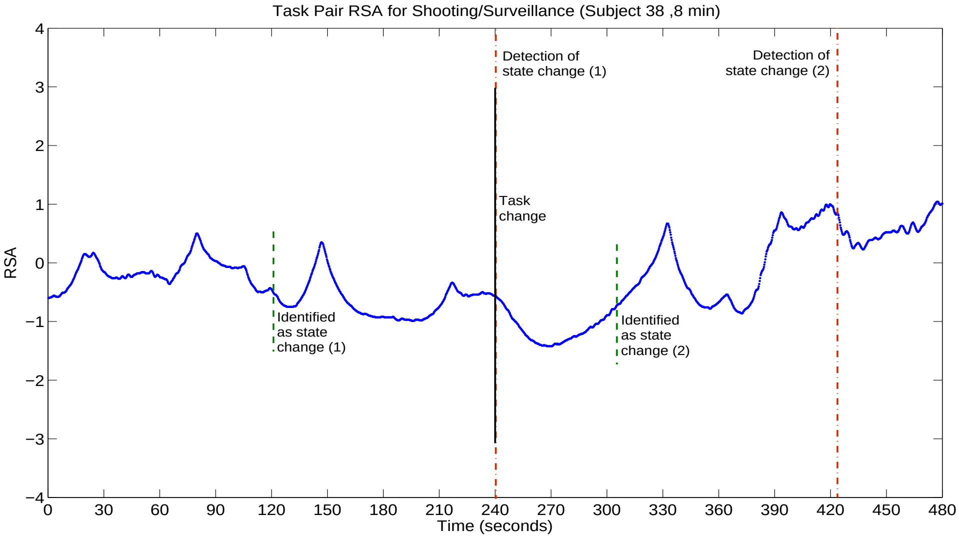


Figure 3.3.4 An example of the RSA data along with the resulting state change detections.

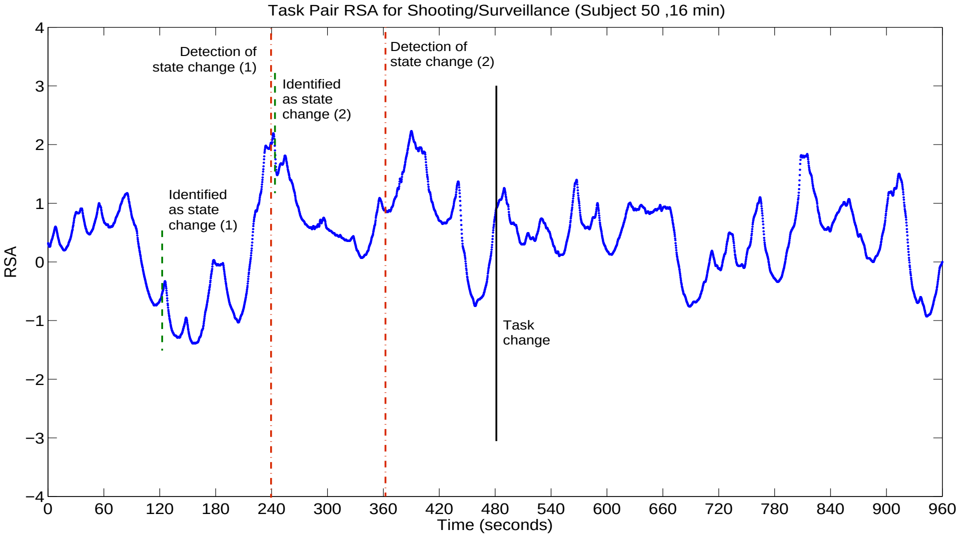


Figure 3.3.5 An example of the RSA data along with the resulting state change detections.

were detected. But again in this case it looks like the state of the signal did change approximately around the time of both the detections.

These results show that our state change detector was able to accurately identify the task change in approximately 45% of the data samples. But the important point is that most of the detections that were marked as false positives were actually probable changes in the state of the signal.

CHAPTER 4
CONCLUSIONS

This work examines the problem of detecting a change in the state of a signal. It requires us to create a method of analyzing the signal in real-time to detect the change. The innovativeness of our approach lies in our proposition that the state of the signal varies in only a part of the Gaussian distribution. By monitoring the region of variation of a signal, we can detect a change in its state. Our methods were tested on 35 samples of data. The data consisted of a signal that represents the heart rate variability (HRV) of a person. The signals were recorded while a person was made to perform tasks of varying intensities. It is presumed that the state of the signal changes with a change in the task being performed by the person. For the chosen parameters, our algorithm yields a true positive detection rate of 45%. The number of false positive detections are approximately limited to one false positive per every two data samples.

The most significant advantage of our approach is that the outcome of the state change detector is not influenced by fluctuations in the input signal. This is important because the signals in most of the applications that this algorithm is intended for are very noisy. Thus the fact that our approach is resistant (to a large extent) to the noise in a signal, suggests that our algorithm would perform well for such applications.

An important observation made from our results was that most of the false positive detections flagged by our detector were legitimate changes in the state of the data. But these detections did not correspond with the task change in the data. This phenomenon can be explained when we consider that the state of the signal is not always representative of the task.

One aspect that limits us from drawing sturdier conclusions from our results is the lack of enough data. Our algorithm was implemented on 35 usable samples of data. A larger set of input data samples could have helped infer more concrete conclusions from our results. The trend of the variation of the TPs due to the changes in the minimum window size and the overlap statistic threshold is not well demarcated. But we believe that an increased number of data samples would help in improving this trend.

One suggestion for further work would be to consider the minimum distance to be maintained between a and b as a parameter and evaluate its effect on the results. If more data is collected, then we could also look at finding the effects of increasing the overlap statistic threshold beyond 0.35.

The state change detector is designed to detect changes in the state of a signal. The performance of the detector regarding this aspect can be seen from the fact that most of the false positives detected were actual changes in the state of the signal. Thus we would like to suggest another method to evaluate our state change detector. This could be done by comparing its outcome with visual cues. The state changes can be decided upon by visually going through the test signals. These visually detected changes can then be compared with the outcome of the state change detector.

The results obtained by us endorse the novelty of our approach. We believe that with the help of further work done based on our approach, it is possible to design a state change detector that could be integrated into real-time applications.

APPENDICES

Appendix A

Abbreviations and Definitions

AM – Arousal Meter

ANS – Autonomic Nervous System

CNS – Central Nervous System

CUSUM – Cumulative Sum

HRV – Heart Rate Variability

FFT – Fast Fourier Transform

FP – False Positive

PNS – Parasympathetic Nervous System

RSA – Respiratory Sinus Arrhythmia

RR – Heart rate

SNS – Sympathetic Nervous System

TP – True Positive

REFERENCES

[1] F. Agharebparast and V. Leung. A new traffic rate estimation and monitoring algorithm for the qos-enabled internet. In *IEEE Global Telecommunications Conference*, pages 3883–3887, Dec 1–5 2003.

[2] J. Allen, A. Chambers, and D. Towers. The many metrics of cardiac chronotropy: A pragmatic primer and a brief comparison of metrics. *Biological Psychology*, 74(2):243–262, Feb 2007.

[3] G. Brenston, J. Cacioppo, and K. Quigly. Automatic determinism:laws of autonomic control, the doctrine of autonomic space, and the laws of autonomic constraint. *Psychological Review*, 98:459–487, 1991.

[4] S. Charbonnier, G. Becq, and L. Biot. On-line segmentation algorithm for continuously monitored data in intensive care units. *IEEE Transactions on Biomedical Engineering*, 51(3):484–496, Mar 2004.

[5] S. Fishel, E. Muth, and A. Hoover. Establishing appropriate physiological baseline procedures for real-time physiological measurement. *Journal of Cognitive Engineering and Decision Making*, 1(3):286–308, Fall 2007.

[6] S. Fishel, E. Muth, A. Hoover, and L. Gugerty. Determining the resolution of a real-time arousal gauge. In *Proceedings of the SPIE - The International Society for Optical Engineering*, pages 621816–1–12, Apr 19–21 2006.

[7] A. Hoover and E. Muth. A real-time index of vagal activity. *International Journal Human-Computer Interaction*, 17(2):197–209, June 2004.

[8] A. Hoover, J. Rand, S. Fishel, J. Moss, J. Pappas, and E. Muth. Real-time correction of heart interbeat intervals. *IEEE Transactions on Biomedical Engineering*, 54(5):946–950, May 2007.

[9] R. Neff, J. Wang, S. Baxi, C. Evans, and D. Mendelowitz. Respiratory sinus arrhythmia: Endogenous activation of nicotinic receptors mediates respiratory modulation of brainstem cardioinhibitory parasympathetic neurons. *Circulation Research*, 93(6):565–572, Sept 2003.

[10] R. Picard, E. Vyzas, and J. Healey. Toward machine emotional intelligence: Analysis of affective physiological state. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(10):1175–1191, Oct 2001.

[11] M. Raifel and S. Ron. Estimation of slowly changing components of physiological signals. *IEEE Transations on Biomedical Engineering*, 44(3):215–220, Mar 1997.

[12] P. Yang, G. Dumont, and J. Ansermino. Adaptive change detection in heart rate in trend monitoring in anesthetized children. *IEEE Transactions on Biomedical Engineering*, 53(11):2211–2219, Nov 2006.