2017

Blind Change Point Detection And Regime Segmentation Using Gaussian Process Regression

Sourav Das
University of South Carolina

BLIND CHANGE POINT DETECTION AND REGIME SEGMENTATION USING GAUSSIAN PROCESS REGRESSION

by

SOURAV DAS

Bachelor of Technology

West Bengal University of Technology, 2014

Submitted in Partial Fulfillment of the Requirements

For the Degree of Master of Science in

Computer Science and Engineering

College of Engineering and Computing

University of South Carolina

2017

Accepted by:

Gabriel Terejanu, Director of Thesis

John Rose, Reader

Yan Tong, Reader

Cheryl L. Addy, Vice Provost and Dean of the Graduate School

ACKNOWLEDGEMENTS

I would first like to thank my thesis advisor Dr. Gabriel Terejanu of College of Engineering and Computing at University of South Carolina. He always came up with suggestions whenever I ran into a trouble spot during my research and always helped me to improve both my work and writing. He steered me in the right direction whenever I needed it, not only in my research but also morally. This thesis has been the quintessence of intense learning for me not only in the scientific arena but also on a personal level.

I would also like to thank Dr. John Rose of College of Engineering and Computing at University of South Carolina, head of this thesis committee, and Dr. Yan Tong of College of Engineering and Computing at University of South Carolina, committee member, for their valuable comments during the thesis proposal.

Finally, I express my profound gratitude to my parents and to my girlfriend for providing me with unfaltering support and encouragement throughout my educational journey as well as throughout the process of researching and writing this thesis. This achievement would not have been possible without them.

Sourav Das

Nov 2016

ABSTRACT

Time-series analysis is used heavily in modeling and forecasting weather, economics, medical data as well as in various other fields. Change point detection (CPD) means finding abrupt changes in the time-series when the statistical property of a certain part of it starts to differ. CPD has attracted a lot of attention in the artificial intelligence, machine learning and data mining communities. In this thesis, a novel CPD algorithm is introduced for segmenting multivariate time-series data. The proposed algorithm is a general pipeline to process any high dimensional multivariate time-series data using non-linear non-parametric dynamic system. It consists of manifold learning technique for dimensionality reduction, Gaussian process regression to model the non-linear dynamics of the data and predict the next possible time-step, as well as outlier detection based on Mahalanobis distance to determine the change points. The performance of the new CPD algorithm is assessed on synthetic as well as real-world data for validation. The pipeline is used on federal reserve economic data (FRED) to detect recession. Finally, functional magnetic resonance imaging (fMRI) data of larval zebrafish is used to segment regions of homogeneous brain activity.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS

GP	Gaussian process
ε	Normally distributed noise
W	Weights or parameters
$E[.]$	Expectation
$k(.,.)$	Covariance or kernel function
$p(.)$	Probability
$N(.,.)$	Normal distribution
K	Covariance matrix
$\sum$	Sum or covariance matrix depending on the equation

LIST OF ABBREVIATIONS

CPD……. Change Point Detection

fMRI……. Functional Magnetic Resonance Imaging

FRED…….……. Federal Reserve Economic Data

GP……. Gaussian Process

GPR…….…… Gaussian Process Regression

LLE……. Locally Linear Embedding

PCA……...…….. Principal Component Analysis

CHAPTER 1

INTRODUCTION

Everything in this universe is a dynamic system and changes with time. Starting from expansion of galaxies to our brain activity, from evolution to finance – everything is a time-series and only scale of the time step varies. Time-series are sequences of data over time describing the behavior of systems. This behavior changes over time due to external factors or change in internal dynamics of the systems. Thus time-series analysis has become increasing important in the fields of healthcare, economy, energy, finance and meteorology. As every system is dynamic with respect to time, we need to detect when an abrupt change is occurring that deviates the property of the time-series. Detecting this abrupt change is known as change point detection. Segmentation is a way of implementing change point detection, i.e. the process of segmenting a data-series into different segments or regimes by identifying the points where statistical properties start to differ. These points are known as change points. Thus, the data can be divided into separate homogeneous segments but heterogeneous with respect to each other. Change point detection or segmentation can also be applied prior to using machine learning algorithms in order to increase the prediction accuracy as the data sequence would be segmented into homogenous regions. It can be used as clustering through time.

Change point detection (CPD) or segmentation of time-series has been studied for decades in the fields of statistics, computer science, data mining and bioinformatics. It covers a wide range of real world problems. Here are some examples of its applications:

- Health and disease monitoring: In medical condition monitoring of patients, CPD algorithms have been used to detect abrupt changes in patients' electrocardiogram (ECG) [1] and electroencephalogram (EEG) [2] signals in real-time. CPD is also used in detection and monitoring of disease outbreaks such as influenza-like illness [3].
- Climate change monitoring: Climate change detection and prediction has been of increasing importance in the recent years due to the effect of global warming and increasing emission of greenhouse gases [4], [5].
- Economic and financial monitoring: It is important to both government and industry to have systems that predict the future state of economy. CPD is used in economic analysis for prediction of expansion and recession as well as capturing business cycle dynamics [6], [7], [8].
- Human activity monitoring: With the increase of internet-of -things devices like sensors from smart homes and smartphones, human activity is monitored in order to reduce human interaction as well as to monitor health. CPD is used to analyze the transitions of activities from the sensor data [9].

Segmentation of time-series can be broadly classified into categories [10] as discussed in the following paragraphs:

Parametric versus non-parametric: Parametric models assume that the data is sampled from a certain distribution, which generally is a normal distribution and thus the estimated parameters also have a mean and standard deviation. These models tend to be accurate if and only if the underlying assumption about the data is true, otherwise leads to inaccurate results. Thus, these models are not considered robust even though they are simple and computationally fast. On the contrary, non-parametric models do not have any underlying assumption about the data. They are robust but are computationally expensive and they need more data samples to perform the same task than their parametric counterpart.

Univariate versus multivariate: Univariate models use only one dimension of data in each time step, whereas multivariate models use multiple dimensions of data in the same time step. A lot of research has been conducted on univariate models as it is easier to generate univariate synthetic time-series, though multivariate models tend to be more accurate as they have more information to segment upon.

Offline versus online: Offline segmentation refers to handling the data in batch mode. The entire data is used to partition it into homogenous regimes. While online segmentation refers to handling the data in incremental mode. Streaming data is used to partition it when it arrives in real time. The major drawbacks of offline method are – large computation and storage complexity as well as inability to scale linearly.

Aided versus blind: Aided segmentation methods use prior information about the data sequence, mostly related to the target quantity on which the sequence is segmented upon. While blind segmentation methods have no prior information about the data.

1.1 LITERATURE REVIEW

Change point detection or segmentation has been researched for decades and several methods exist in the literature. Some of them are specially designed for detecting change points, while some of them are designed to obtain a compact representation of the data. In the latter case, individual model is used to describe individual segment of the time-series. Some of these algorithms take user input to find the number of segments to be formed, thus reducing the reconstruction error in the lower dimension space.

In the following paragraph, we will assume the length of time-series as n and the number of segments as k. Whether these algorithms are used for finding change points or for dimensionality reduction, segmentation algorithms are based on these high level methods as discussed in the following paragraphs:

Dynamic programming based: This category of segmentation algorithms is used to compress each segment into a lower dimension space. Given the maximum number to segments to be found and the maximum allowable cost permitted for segmentation (or with respect to some loss function), these algorithms try to find the optimum segments such that the overall cost is minimum. A pruned dynamic programming based segmentation algorithm has been proposed [11], where pruning leads to efficient results even if there are no change points in the data but its worst case complexity is $O(kn^2)$. The drawbacks of this type of algorithms are – the maximum number of segments has to be provided and they are computationally expensive. This means that they cannot be used in real world applications as the number of segments cannot be predicted in prior and they cannot be applied to large datasets as they are computationally very demanding.

Heuristics based: The expensive computational cost of running dynamic programming based segmentation algorithms lead to the development of heuristics based algorithms [12]. These algorithms cannot find optimum segments as their dynamic programming counterparts but they are computationally efficient and produce good results on most cases. Heuristics based algorithms can again be categorized into three methods:

I. Sliding window based: A window of fixed length slides over the data and each segment is grown until a user specified threshold or error criteria is met. It works by anchoring the left most data point of a segment and sequentially extending the right most data point until the error criteria is within a certain permissible threshold. This process repeats until the entire time-series is segmented. The error criteria for compression is a specified threshold of reconstruction error, while in the case of change detection is if an outlier is found i.e. the current data point is statistically different from the past other data points in the segment. Sliding window is appealing as it is quite simple, intuitive, online and has little computational requirements. It only requires a restricted history of past data points to work. The stride in the sliding window algorithm is a hyperparameter, which is not limited to 1 only and if optimized can greatly speed up the algorithm. The drawbacks of this algorithm is that it over segments the time-series that has noisy data.

II. Top-down based: Top-down based segmentation works by considering the entire time-series as one segment and tries to recursively partition each segment such that their statistical difference is maximum. The stopping criteria for compression is when the reconstruction error goes above a certain specified threshold, while in the case of change detection it is the specified number of maximum change points. The

major drawbacks of this algorithm are – the whole dataset has to be used and the maximum number of segments has to be provided in prior.

III. Bottom-up based: Bottom-up based segmentation is the compliment of top-down. It segments the entire time-series in *n/2* segments. Next it iteratively combines adjacent segment pairs as long as the cost of merging or error criteria is within a user-specified threshold in the case of compression. In the case of change detection, it seeks out the smallest possible segments and tries to combine it with the adjacent segments on the basis of a similarity score. This algorithm scales linearly with data and can produce optimum segments but cannot work in real-time.

Hybrid based: Because of the noted shortcomings of the various segmentation algorithms, a hybrid algorithm has been proposed [13], which combines online sliding window and offline bottom-up approaches known as SWAB (Sliding Window And Bottom up). It combines the pros and cons of both techniques to obtain optimum results. The intuition behind this algorithm is that the sliding window approach over-segments the data series, the bottom up approach combines the possible similar adjacent segments to create a larger segment. SWAB algorithm keeps a buffer of chosen initial size. This buffer behaves like a queue. Bottom up is applied in the buffer and the leftmost segment is reported and the corresponding data points are removed from the buffer. More data points are read in into the buffer depending on the results of the sliding window algorithm. This process continues as long as there is incoming data. SWAB can be applied in both data compression as well as in change detection. It scales linearly with data, but the initial size of the buffer is a hyperparameter which needs to be optimized. The trade-off of this

algorithm is that small size of buffer will convert it to a sliding window approach and a large size of buffer will convert it to a bottom up approach.

Probabilistic based: Hidden Markov Models and Bayesian algorithms generally fall in this category of segmentation algorithms. They are discussed in the following.

I. Hidden Markov Model (HMM): HMM is used in diverse research areas like bioinformatics, brain imaging, financial time-series, climate monitoring and network security. It is a commonly used tool for inference in change point analysis. In the context of change point detection, HMM can be applied where the data are the observations and the unknown segments are the hidden states. Change points occur when there is a switch in the hidden states of the system. Change detection problem using HMM can be defined as finding posterior probabilities of the hidden states of the model [14]. This is generally done using Expectation-Maximization algorithm to find the posterior estimates of the model parameters and the state probabilities, given the data and prior distribution of state changes. The major drawback of this algorithm is that it is expensive both in terms of memory and computation.

II. Bayesian: Bayesian based segmentation assume that the entire time-series is divided into non-overlapping regions. The data points within each of these regions are independent and identically distributed from some probability distribution as well as the parameters of the model are independent and identically distributed [15]. Given the data observed and a change point has occurred, the Bayesian approach estimates the posterior distribution over the current run length i.e. the time since the occurrence of last change point. To generate the posterior distribution, a

conditional prior on the change point is generated which gives this algorithm its efficiency. The prior is based on a hazard function, which describes how likely a change point will occur given the current run length.

Clustering based: Segmentation of time-series can be interpreted as clustering of homogeneous data points in time. Principal component analysis (PCA) is widely used in various applications like dimensionality reduction, feature extraction and signal estimation. PCA is an analytic method for estimating generalized eigenvalues and eigenvectors. Generalized eigenvectors work as filters in the joint space of two signals, minimizing the energy of one signal while maximizing the energy of the other. Due to this property they act like filters that can perform signal separation. This property can be used to quantify change points in time-series using a two-step PCA as proposed in [16].

1.2 MOTIVATION

The focus of this thesis is change point detection and regime segmentation on high dimensional multivariate time-series data. I intend to create a non-linear non-parametric dynamic system that can segment the entire time-series into regions with different dynamic activities. Here, a homogeneous region will refer to a sequence of data points that follow the same dynamics. This dynamic system should be flexible enough to work on different types of datasets in various fields. I have used multiple synthetic data as well as multiple real-world data to demonstrate this flexibility.

The curse of dimensionality: Some time-series, such as videos can be very high dimensional depending on their resolution. A high number of features can deteriorate the running time of a computational model, thus making it computationally expensive for

online purposes. A dimensional reduction algorithm must be used to reduce this curse of dimensionality. The dimension of the reduced space should be low enough that the model can perform online segmentation as well as high enough not to lose important information from the data sequence. From the plethora of work on dimensionality reduction, I must choose such an algorithm that generalizes from image data to economic time-series, without inducing too much of a bias depending on the method it uses.

Non-parametric model: In the previous discussion on the pros and cons of parametric versus non-parametric models, I showed that parametric models can be inaccurate in case the assumptions on the data do not hold true. Thus, I choose to use non-parametric models with no initial assumption on the data, or only with the assumption that the noise associated with the data follows a normal distribution. This helps us to create a non-linear dynamic system that models the time-series without too much of a discrepancy. A dynamic system is a function that describes the evolution of data points or their states over time.

Example of a linear parametric regression model:

$$x_{t+1} = Wx_t + \varepsilon_t \,, \tag{1}$$

Example of non-linear non-parametric regression model:

$$x_{t+1} = f(x_t) + \varepsilon_t \,, \tag{2}$$

$$f(.) \sim GP\big(0, k(.,.)\big) \,, \tag{3}$$

Here, W is the weight matrix for the parametric model, f is the non-linear function of the non-parametric model and ε is the normal noise associated with the data. In

the proposed pipeline a prior distribution is assumed over f, i.e. a Gaussian process prior with a zero mean and some covariance function k.

Uncertainty of prediction: While classical regression algorithms attempt to identify the best fit model of the data, Bayesian regression algorithms compute a posterior distribution over models or over new test data. This posterior distribution helps to quantify the uncertainty in model estimates. This uncertainty can be thus used to make more robust predictions on new test data [17]. Thus I am interested not only on the mean prediction from the non-linear non-parametric model, but also on the uncertainty associated with that prediction. This is obtained by also predicting the covariance for each mean prediction from the training data.

We work on functional magnetic resonance imaging (fMRI) data of larval zebrafish. The details on how this data is obtained is proposed in [18]. We use this data to perform change point detection and segment different regions of similar brain activity. As there are no definite change point labels in this data, I validate the working of the proposed pipeline by using synthetic as well as real-world labeled data. This also shows that the pipeline can generalize on different types of datasets in various fields, when performing blind segmentation.

The following Chapter 2 describes the methodology used in this thesis. Chapter 3 presents the experimental results obtained from the proposed pipeline. The conclusion is proposed in Chapter 4.

CHAPTER 2

METHODOLOGY

2.1 PROPOSED PIPELINE

The proposed pipeline is divided into four major stages – data preprocessing, dimensionality reduction, prediction of the next time-step and time-series segmentation. In this thesis, I am working with different types of data – univariate time-series data and multivariate time-series data which includes images as well as economic data. Each stage of the pipeline works as a plug and play mechanism. Thus, components of this pipeline can be replaced with other algorithms if so desired. The pipeline works as a general framework to segment high dimensional multivariate time-series data using non-linear non-parametric dynamic system. The following figure sheds some information on the stages of the proposed pipeline. Finally, the pipeline is described in details in the remaining of this section.

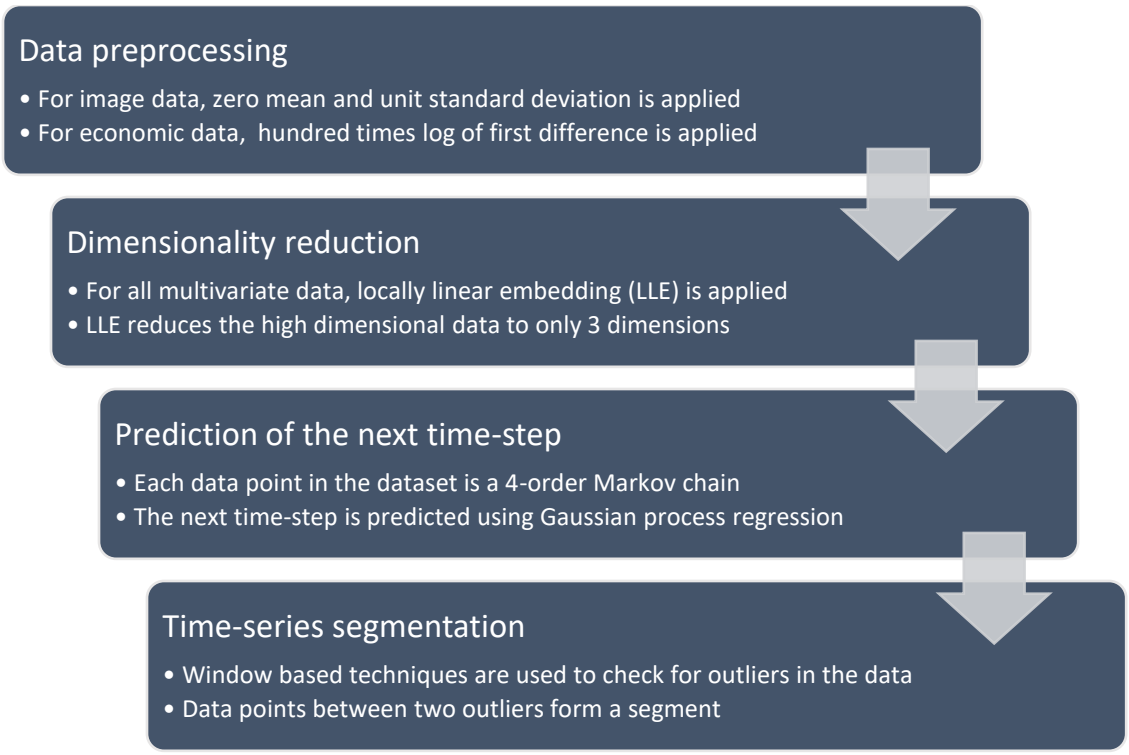

Figure 2.1: The proposed pipeline

2.1.1 Data preprocessing

For synthetic data, I use zero mean and unit standard deviation normalization technique for feature scaling to standardize the range of independent variables or features of the data.

For image data, I use bilateral filter to de-noise the frames obtained from the video used in this work as it is a non-linear, edge-preserving smoothing filter. These frames are noisy and quite limited in number. I use static background subtraction using thresholding and masking to focus only on the dynamic activity within these images. Using histogram equalization after this step results in more range of intensities, but also introduces some noise in the images. Thus, I finally apply Gaussian filter to reduce the effects of the

introduced noise. These images are resized and stored in disk for the next stage of processing. Finally, I use zero mean and unit standard deviation normalization technique for feature scaling to standardize the range of independent variables or features of the data.

For economic data, I take the first difference of all the data i.e. the next time-step is subtracted from the current time-step. I apply natural logarithm on the first difference and scale the results one hundred times. Natural logarithm is used as it has important properties with respect to business analysis. Its greatest advantage is that small changes in natural log of a variable can be directly approximated as percentage changes. Another great advantage is that it linearizes exponential growth and inflation. Since, the logarithm of a product is equal to the sum of the logarithms, it converts multiplicative relationships to additive relationships. By the same principle, it converts exponential trends with proportional variance into linear trends with constant variance.

2.1.2 Dimensionality reduction

Dimensionality reduction is a crucial step whether it is an object recognition problem or a statistical analysis of a multivariate population. It not only gives computation and storage benefits but also can be viewed as a feature extraction technique or in general to provide a compact representation in a different coordinate system. For example, if the given data is high dimensional, fitting a model on it generally would result in overfitting the data. This would require a lot of training data to overcome this problem. Thus whenever the intrinsic dimensionality of a dataset is smaller than the original one, dimensionality reduction can give a better understanding of the data.

For all multivariate data, I use dimensionality reduction to reduce the number of features. Initially the popular principal component analysis (PCA) [19] is used for experimentation. It is an eigenvector method designed to capture linear variabilities in high dimensional data. PCA computes the linear projections of highest variance from the top eigenvectors of the data covariance matrix. Since it only computes linear projections of highest variance, this method is not enough to capture the dynamics of the data in low dimensions. In this work, the local neighborhood structure of the data is very crucial as I want to capture the dynamics present within the data, in order to build a non-linear dynamic system. Thus, I use locally linear embedding (LLE) [19] to reduce the number of features. This manifold learning algorithm finds the lower dimension representation of the data which preserves distances within local neighborhoods. It can be interpreted as a series of localized PCAs which are globally combined in a non-linear fashion to get the least reconstruction error in the lower dimension space.

2.1.3 Prediction of the next time-step

In this stage, I am assuming that the next data point or time-step to predict is dependent on some of the previous time-steps. The number of dependencies is a hyperparameter which is kept constant at 4 throughout this work. Thus, I am creating a dataset where each row of the data is a 4-order Markov chain [20]. I am incorporating the data at the beginning of the time-series into a starting window of fixed length, which depends on the data and the application of the pipeline, to predict the next time-step. If the predicted data point is consistent with the current segment, the data point is incorporated into the segment. Otherwise, an entirely new segment is formed. The formation of new segments is based on two approaches – jumping window and sliding window. These

window based approaches help us to model the data in a fast online way and is explained in details in the ‘time-series segmentation’ section.

The data points present in the initial window are modeled using Gaussian process regression (GPR) [21]. GPR does not make any assumptions about the underlying distribution of the data, i.e. it can be either linear, quadratic or even non-polynomial. GPR is a form of non-parametric supervised learning that models data by letting data speak for themselves. In this work, the input is a window or matrix of 4-order Markov chains and the target is a vector of next time-steps with respect to those Markov chains. The relationship among data points is given by a covariance function, also known as the kernel function. Here, I am using the product of squared exponential kernel and periodic kernel. This product is also known as locally periodic kernel; it helps us to generalize the pipeline on various datasets. I optimize the hyperparameters of the resulting GPR model using truncated Newton methods. It is a Hessian-free optimization technique designed for optimizing non-linear functions and converges to global minima. This model is used to predict the next time-step. Finally, Mahalanobis distance is calculated between the predicted data point and the true data point in low dimensional space. This is done to determine outliers with respect to the current segment. Ultimately this results in determining the change-points in the time-series.

2.1.4 Time-series segmentation

In this stage, Mahalanobis distance plays the most important role in determining segments of homogeneous data points as well as creating new segments when heterogeneous data points are encountered. This metric is used to determine outliers in the data with respect to the current segment. Outliers are identified by computing Mahalanobis

distance between the predicted next time-step and the true next time-step in low dimensional space. If this metric is less than a given threshold, then that data point is incorporated sequentially within the current segment. Otherwise, I identify that data point as an outlier. If I encounter an outlier, I partition the data and a new segment is created. The question of interest is how quickly I should partition the data and create a new segment or in general determine it as a change point. The more quickly I declare this as a change point, the more is the possibility of occurrence of a false positive. In order to reduce false positives, I perform Mahalanobis distance on three data points sequentially. If at least two of the them are outliers, I determine that location as a change point.

The creation of new segments also depends on how the data sequence is traversed by the moving window. Here, I use two different traversal mechanisms – jumping window and sliding window. The jumping window approach initially starts with a window of fixed length, and when a change point is detected the window jumps from its current location to the next sequence of new data with its beginning placed at one more than the end of the previous segment. The trade-off here is that, not only the length of the jumping window should be less than the distance between any two change points in the entire time-series, but also it should not be too small so that it ends up over segmenting the data. On the contrary, the sliding window approach starts with a window of fixed length and slides over the data with a stride of one, irrespective of the presence of change points. In both the cases, GPR models are trained only when a new change point is detected or a new segment is formed and not when new data points are incorporated into the current segment.

The usefulness of Mahalanobis distance is that its squared value for multi-variate normal data is intimately related to chi-squared distribution. Thus I can use the chi-squared

distribution table to find out the threshold for detecting outliers, where the degrees of freedom of the distribution is equal to the dimensionality of the latent space. In this work, the level of significance of the chi-squared distribution is kept constant at 0.05.

2.2 BACKGROUND

2.2.1 Locally Linear Embedding (LLE)

The LLE algorithm [22] retains the local geometry of a region in the data by computing linear coefficients that reconstruct each data point from its neighbors. The algorithm uses K nearest neighbors for each data point to find out the coefficients or contributions of the neighbors to that data point. These local coefficients of a particular region are used to reconstruct each data point in the low dimension space. The goal of this algorithm is to minimize the reconstruction error of the low dimension space compared to the high dimension space. The reconstruction errors are measured by the following cost function:

$$\varepsilon(W) = \sum_i \left| \vec{X}_i - \sum_j W_{ij} \vec{X}_j \right|^2, \quad (4)$$

The weights W_{ij} are computed by minimizing the above cost function, subject to two constraints. Firstly, each data point X_i in a local region is reconstructed from its fixed number of neighbors and thus setting $W_{ij} = 0$ if X_j does not belong in this neighborhood. Secondly, sum of the elements in each row of the weight matrix is equal to one, $\sum_j W_{ij} = 1$. This second constraint results in invariance to translation for any particular data point and due to this symmetry, the reconstruction weights preserve the intrinsic geometric characteristics of each neighborhood. The above equation (4) determines invariance to

rotation, rescaling and translation for any given data point. Since the local geometry of a neighborhood is preserved both in the original data space and in the manifold space, the same weights W_{ij} that reconstruct ith data point in original data space should also reconstruct its embedded manifold coordinates in the latent space.

Finally, LLE maps each high dimensional observation $\vec{X}_i$ into a low dimensional vector $\vec{Y}_i$ which represents global embedding coordinates in the manifold. These internal coordinates are *d*-dimensional where *d* is an input to the algorithm, and the coordinates are calculated by minimizing the embedding cost function:

$$\varphi(Y) = \sum_i \left| \vec{Y}_i - \sum_j W_{ij} \vec{Y}_j \right|^2, \quad (5)$$

This cost function, like equation (4), is based on locally linear reconstruction of errors but here the weight matrix W_{ij} is fixed while the low dimensional coordinates Y_i are optimized. These coordinated are obtained by solving a sparse $N \times N$ eigenvector problem subjected to constraints.

LLE Algorithm [23]:

1. Compute a fixed set of neighbors for each data point $\vec{X}_i$ in the original space.
2. Compute the weight matrix W_{ij} that optimally reconstructs each data point $\vec{X}_i$ from its set of neighbors, while minimizing the cost function in equation (4).
3. Compute the low dimensional vector $\vec{Y}_i$ which is optimally reconstructed by the weight matrix W_{ij}, while minimizing the cost function in equation (5).

2.2.2 Gaussian Process (GP)

A Gaussian process [24] by definition is a collection of random variables, any finite number of which have a joint Gaussian distribution. It is a generalization of multivariate Gaussian probability distribution. While a probability distribution describes random variables, GP is a stochastic process that describes probability distributions over functions. The basic assumption is that for any finite subset of data, the marginal distribution of that set follows a multivariate Gaussian distribution and thus data can be sampled from that distribution. GPs are characterized by a mean function $\mu(x)$ and a covariance function $k(x, x')$ of a real process $f(x)$, where *x* and *x'* are random variables in the real space.

$$\mu(x) = E[f(x)], \tag{6}$$

$$k(x, x') = E\left[\left(f(x) - \mu(x)\right)\left(f(x') - \mu(x')\right)\right], \tag{7}$$

The covariance function of a multivariate GP [25] leads to a covariance matrix which must be positive semi-definite. This is a requirement for GP as the covariance matrices computed based on any arbitrary set of input data is identical to kernels computed using Mercer's theorem. Thus in GP any valid kernel function can be used as a covariance function. This is a very powerful feature in Gaussian processes as they give a lot of flexibility depending on the type of data to be modeled.

2.2.2.1 Kernel function

The very popular choice of kernel is the squared exponential, which is also known as the Gaussian kernel or the radial basis function (RBF) kernel.

$$k(x, x') = \sigma_f^2 \, exp\left[-\frac{(x - x\prime)^2}{2l^2}\right], \tag{8}$$

When $x \approx x'$, the kernel function $k(x, x')$ reaches the maximum amplitude of covariance σ_f^2 meaning that $f(x)$ and $f(x')$ are highly correlated. When *x* and *x'* are far apart, the kernel function $k(x, x') \approx 0$ meaning that $f(x)$ and $f(x')$ are highly uncorrelated. The measure of distance between points is given by the parameter called lengthscale *l*. This means that distant observations have negligible effect while estimating the current prediction. The observation at time t will have higher covariance with the observations in recent history and this effect diminishes exponentially as the observations are more in the past. This is a stationary kernel which is invariant to translations of data points.

Another popular choice of kernel when dealing with time-series data is the periodic kernel.

$$k(x, x') = \sigma_f^2 \, exp\left[-\frac{2\,sin^2\left(\frac{\pi|x-x'|}{p}\right)}{l^2}\right], \tag{9}$$

When $x \approx x'$ or $x \approx \alpha x'$, where α depends on the periodic nature of the *sin* function, the kernel function $k(x, x')$ reaches the maximum amplitude of covariance σ_f^2 meaning that $f(x)$ and $f(x')$ are highly correlated. The period *p* determines the distance between repetitions of the function. Here, the observation at time t is highly correlated with the observation at time $t - 1$ as well as with a certain observation in the past. This periodic nature cannot be modeled using the squared exponential kernel. This is a non-stationary kernel.

2.2.2.2 Combination of kernels

The choice of kernels depends on the type of the dataset and the type of the application. There are many other types of kernels like linear kernel, polynomial kernel, matern kernel, Brownian kernel and neural network kernel. There are several ways to combine kernels – product, sum, convolution. Product is the standard way to combine kernels while sum is also often used. As proposed in [26], various univariate time-series can be approximated using different combinations of the basic kernels.

Product: Intuitively, multiplying two kernels can be interpreted as an AND operation, meaning the resulting kernel will have a high value only if both the two base kernels have a high value.

$$k(x, x') = k_1(x, x') \, . \, k_2(x, x') \, , \tag{10}$$

Sum: Intuitively, adding two kernels can be interpreted as an OR operation, meaning the resulting kernel will have a high value if either of the two base kernels have a high value.

$$k(x, x') = k_1(x, x') + k_2(x, x') \, , \tag{11}$$

2.2.2.3 Gaussian process regression (GPR)

Let (X, y) be a training set of independent and identically distributed random variables following a certain distribution. Using GPR [27, 28] the data is modelled that follows a true distribution function f and has additive Gaussian white noise ϵ, $\epsilon \sim N(0, \sigma^2)$, around the function. A zero-mean Gaussian process prior is assumed as the prior distribution over the function f.

$$y = f(X) + \epsilon\,, \tag{12}$$

$$f(.) \sim GP\big(0, k(.,.)\big)\,, \tag{13}$$

Since the prior is a GP, the posterior is also a GP as the marginal likelihood is a normalizing constant. This can be proved by using the Bayes' theorem.

$$posterior = \frac{likelihood\ x\ prior}{marginal\ likelihood}\,, \tag{14}$$

$$p(f|X,y) = \frac{p(y|X,f)\,p(f|X)}{p(y|X)}\,, \tag{15}$$

The likelihood is given by:

$$p(y|X,f) \sim N(f, \sigma^2 I)\,, \tag{16}$$

The marginal likelihood the integral of the likelihood times the prior:

$$p(y|X) = \int p(y|X,f)p(f|X)\,df \sim N(0, K + \sigma^2 I)\,, \tag{17}$$

For prediction, assuming *n* training input and output pairs (X, y) and *m* test inputs X_m, the joint training and test marginal likelihood is:

$$p(y, y_m) = N(0, K_{n+m} + \sigma^2 I)\,, \tag{18}$$

where, $K_{n+m} = \begin{bmatrix} K_n & K_{nm} \\ K_{mn} & K_m \end{bmatrix}$, (19)

Conditioning on training outputs:

$$p(y_m|y) = N(\mu_m, \sum_m)\,, \tag{20}$$

The mean and covariance of correlated predictions are given by the following equations:

$$\mu_m = K_{mn}\,[K_n + \sigma^2 I]^{-1}\,y\,, \tag{21}$$

$$\sum_m = K_m - K_{mn}\,[K_n + \sigma^2 I]^{-1}\,K_{nm} + \sigma^2 I\,, \tag{22}$$

The cost of computation of $[K_n + \sigma^2 I]^{-1}$ is $O(n^3)$. So often during implementation marginal variances, i.e. diagonal of $\sum_m$, are used. This is sufficient enough to compute prediction for a single test input. This reduces the cost of prediction for each test input to $O(n)$ for the mean and $O(n^2)$ for the variance.

The hyperparameters of GPR i.e. lengthscale and noise variance, are optimized by minimizing the negative logarithm of marginal likelihood with respect to the hyperparameters. Thus GPR is a simple yet powerful non-parametric model, where the hyperparameters are obtained directly from the training data without requiring to use techniques like cross validation.

2.2.3 Mahalanobis Distance

The Mahalanobis distance [28] is a measure of the number of standard deviations away a data point is from the mean of a distribution. This metric of distance grows as the data point moves away from the mean along the principal axis. This is a scale invariant metric that takes into account the correlations of the data set. If a bivariate normal distribution is taken for consideration and concentric probability density ellipses are drawn, then the probability density is high for ellipses near the origin and the density continues to decrease as we go farther from the origin. The level of correlation between the data set, i.e.

the mean of the distribution, and a single test point can be found out by calculating the position of this test point with respect to the probability density ellipses. These ellipses are a generalization of the units of standard deviation away from the mean.

The Mahalanobis distance of a data point $\vec{x} = (x_1, x_2, \dots, x_n)^T$ from a data set with mean $\vec{\mu} = (\mu_1, \mu_2, \dots, \mu_n)^T$ and covariance matrix $\sum$ is defined as:

$$D_M(\vec{x}) = \sqrt{(\vec{x} - \vec{\mu})\sum^{-1}(\vec{x} - \vec{\mu})}\,, \tag{23}$$

Mahalanobis distance has the following properties:

- It takes into account that the variances in each dimension are different.
- It takes into account the covariance among variables.
- It corresponds to Euclidian distance for uncorrelated variables with unit variance, or if all the dimensions are rescaled to have unit variance.
- Its square corresponds to the canonical form of the equation of an ellipse.
- Its square, for multivariate normal data, is intimately related to the chi-squared distribution.

Thus, this metric can be used for outlier detection by keeping a threshold that corresponds to a certain level of significance and certain degrees of freedom of the chi-squared distribution.

CHAPTER 3

RESULTS

3.1 THE DATA

Multiple types of data are used in this thesis to prove the generalization capability of the proposed pipeline. The following sheds information on the types of data used.

Synthetic data:

- Univariate: Four different functions of two different types, exponential and linearly periodic, are concatenated together to get three major change points.

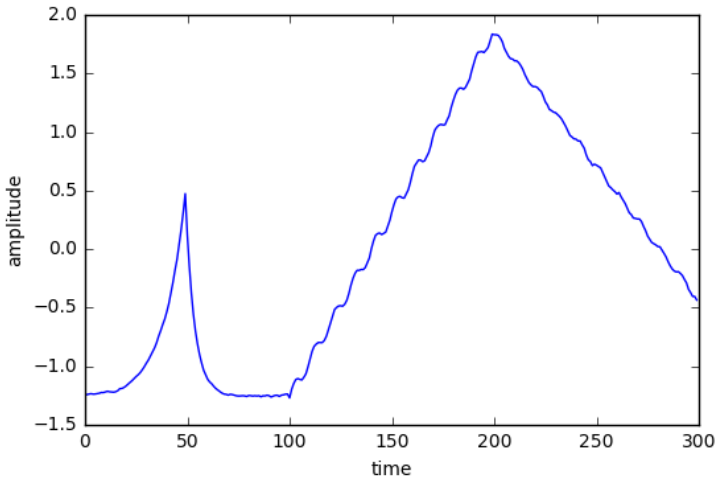

Figure 3.1: Synthetic univariate data

- Multivariate: Five concentric circles with increasing radius and exponentially weighted intensity are drawn on an image with zero intensity background. This whole circle is translated horizontally from left to right and again right to left along

the center of the image. The circle is also translated diagonally from left top to right bottom and again from right bottom to left top. Thus, these images form a time-series reflecting the dynamics of the circle.

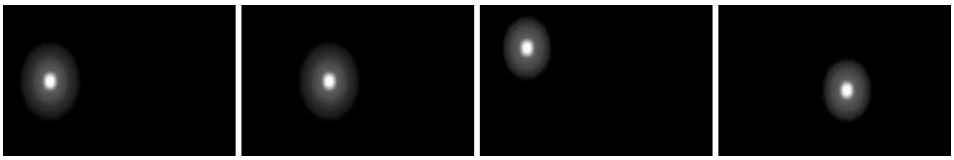

Figure 3.2: Synthetic multivariate image data showing circle dynamics

Real-world data:

- Retail trade economic data: This data is acquired from Federal Reserve Economic Data (FRED). It consists of all monthly and seasonally adjusted retail trade data from January 1992 to September 2016. This data is used to predict recession as a change point in economics. The recession indicator data is also obtained from FRED.

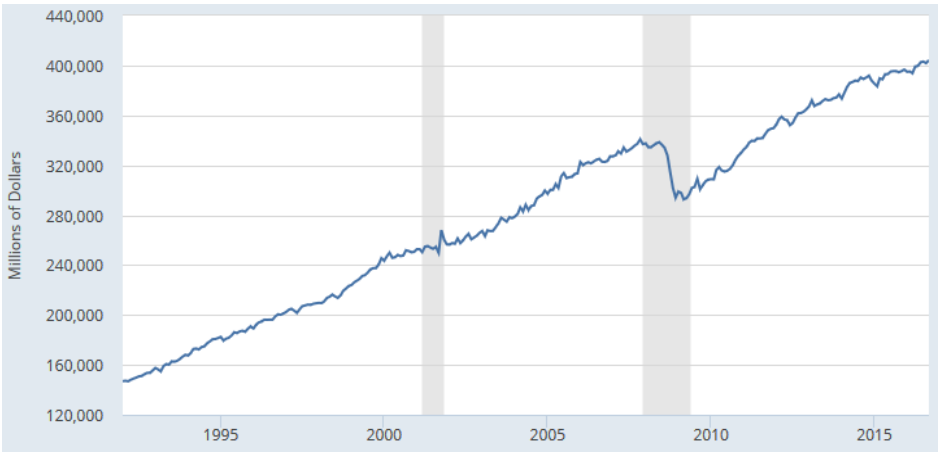

Figure 3.3: Graph of total retail trade, the gray bars indicate recession

- fMRI images of larval zebrafish: This fMRI video of larval zebrafish is based on the procedure proposed in [17] and is obtained from YouTube. These sequential

fMRI images reflect the brain activity dynamics of larval zebrafish. They are used to segment similar brain activity patterns in time.

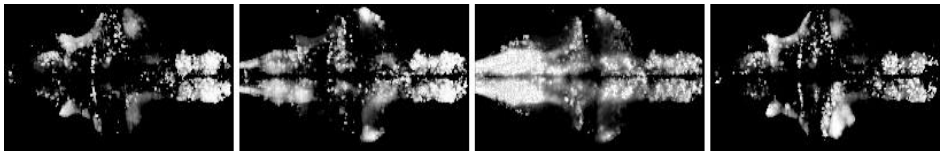

Figure 3.4: Different types of brain activity in larval zebrafish

3.2 PIPELINE SPECIFICATION

For univariate data, Gaussian process regression (GPR) and Mahalanobis distance is applied directly to detect change points. Whereas for multivariate data, locally linear embedding (LLE) is applied before I apply GPR and outlier detection. For image data, I use pixel values of each image which results in 20000 features as each of these images has a resolution of 200x100. For economic data, I use 19 features where all of them are monthly and seasonally adjusted values. In both the cases, I reduce the dimension of the data to only 3 dimensions. This also helps us to visualize the lower dimensional data.

One GPR model is used to capture the dynamics of each feature. The resulting hypermodel consists of the k different GPR models, where k is the dimension of the latent space, which is 3 in this work. I predict the mean and the covariance for each new test data point. These are used to compute Mahalanobis distance for change point detection.

All the values shown in the following sections reflect time-steps. The values corresponding to change points show the particular time-step where a change point occurred. The values corresponding to segments are shown in parenthesis, which indicates the range of time-steps where the data points are homogeneous with respect to each other.

3.3 RESULTS ON SYNTHETIC DATA

Segmentation of univariate data using jumping window approach:

Predicted change points (shown in Fig 3.5): 27, 51, 96, 124, 193

Real change points (shown in Fig 3.5): 50, 100, 200

Predicted segments: (0 - 26), (27 - 50), (51 - 95), (96 - 123), (124 - 192), (193 - 292)

Real segments: (0 - 50), (51 - 100), (101 - 200), (201 - 300)

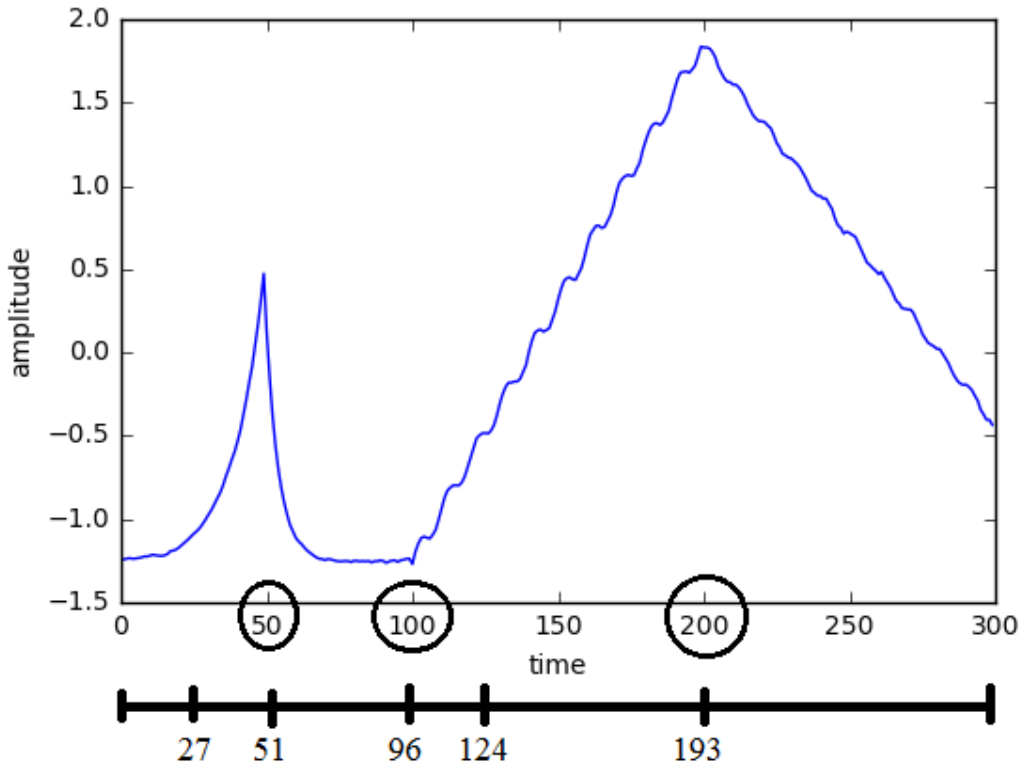

Figure 3.5: Predicted change points in synthetic univariate data using jumping window

Segmentation of univariate data using sliding window approach:

Predicted change points (shown in Figure 3.6): 54, 109, 186, 215

Real change points (shown in Figure 3.6): 50, 100, 200

Predicted segments: (0 - 53), (54 - 108), (109 - 185), (186 - 214), (215 - 292)

Real segments: (0 - 50), (51 - 100), (101 - 200), (201 - 300)

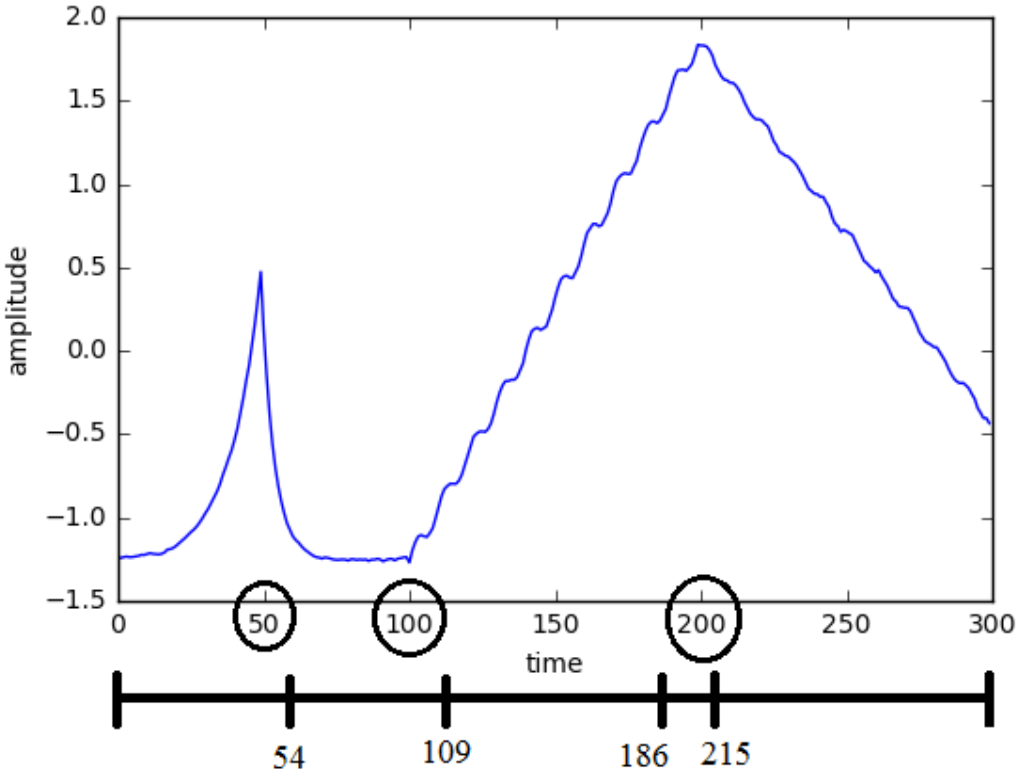

Figure 3.6: Predicted change points in synthetic univariate data using sliding window

Multivariate data:

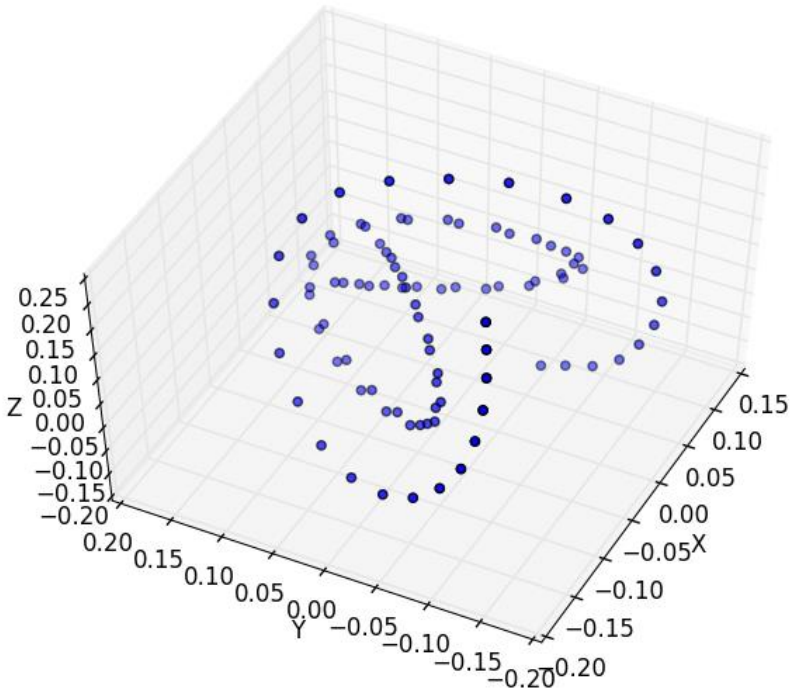

Figure 3.7: LLE on synthetic multivariate data using 16 neighbors and 3 components

Segmentation of multivariate data using jumping window:

Predicted change points: 27, 63, 90

Real change points: 30, 61, 92

Predicted segments (shown in Figure 3.8): (0 - 26), (27 - 62), (63 - 89), (90 - 105)

Real segments: (0 - 30), (31 - 61), (62 - 92), (93 - 123)

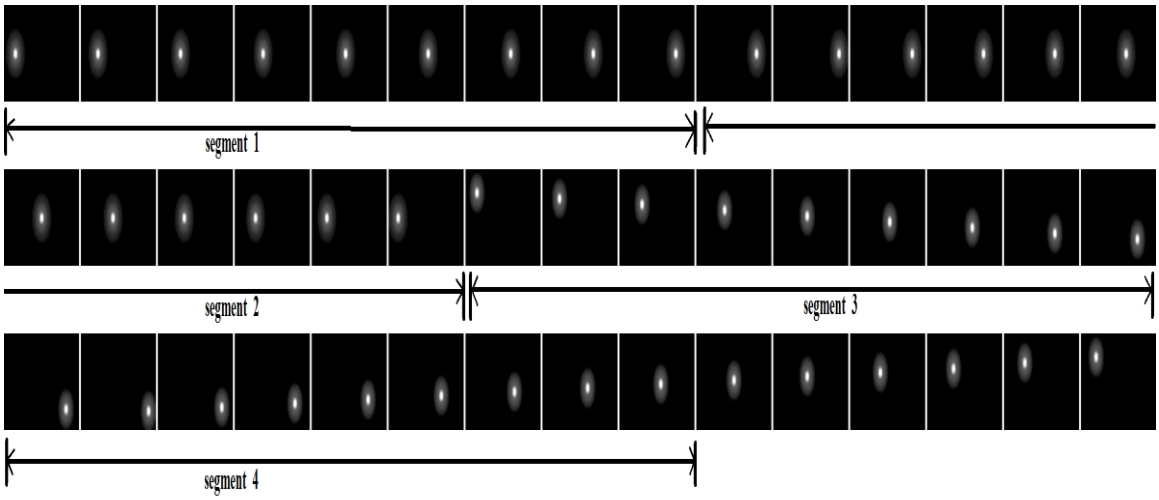

Figure 3.8: Predicted segments on synthetic multivariate data using jumping window

Segmentation of multivariate data using sliding window:

Predicted change points: 26, 61, 91

Real change points: 30, 61, 92

Predicted segments (shown in Figure 3.9): (0 - 25), (39 - 60), (78 - 90), (108 - 115)

Real segments: (0 - 30), (31 - 61), (62 - 92), (93 - 123)

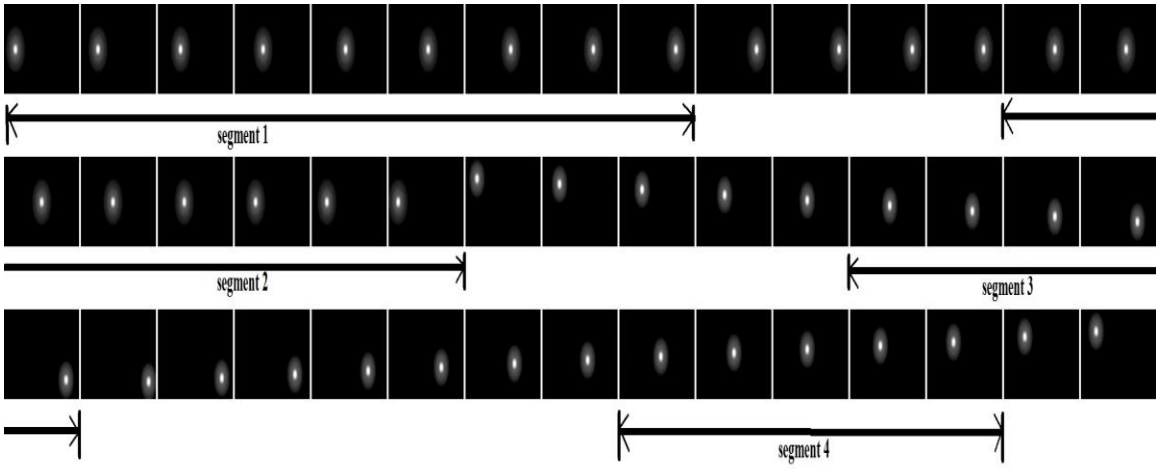

Figure 3.9: Predicted segments on synthetic multivariate data using sliding window

3.4 RESULTS ON REAL-WORLD DATA

Retail trade data:

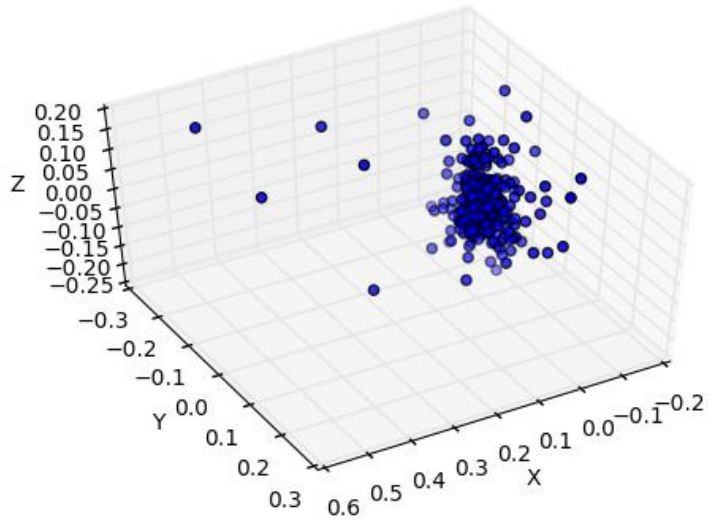

Figure 3.10: LLE on retail trade data using 16 neighbors and 3 components

Segmentation of retail trade data using jumping window:

Predicted change points (shown in Figure 3.11): 109, 195

Real change points: 111, 119, 192, 210

Predicted segments: (0 - 108), (109 - 194), (195 - 288)

Real segments: (0 - 110), (111 - 118), (119 - 191), (192 - 209), (210 - 297)

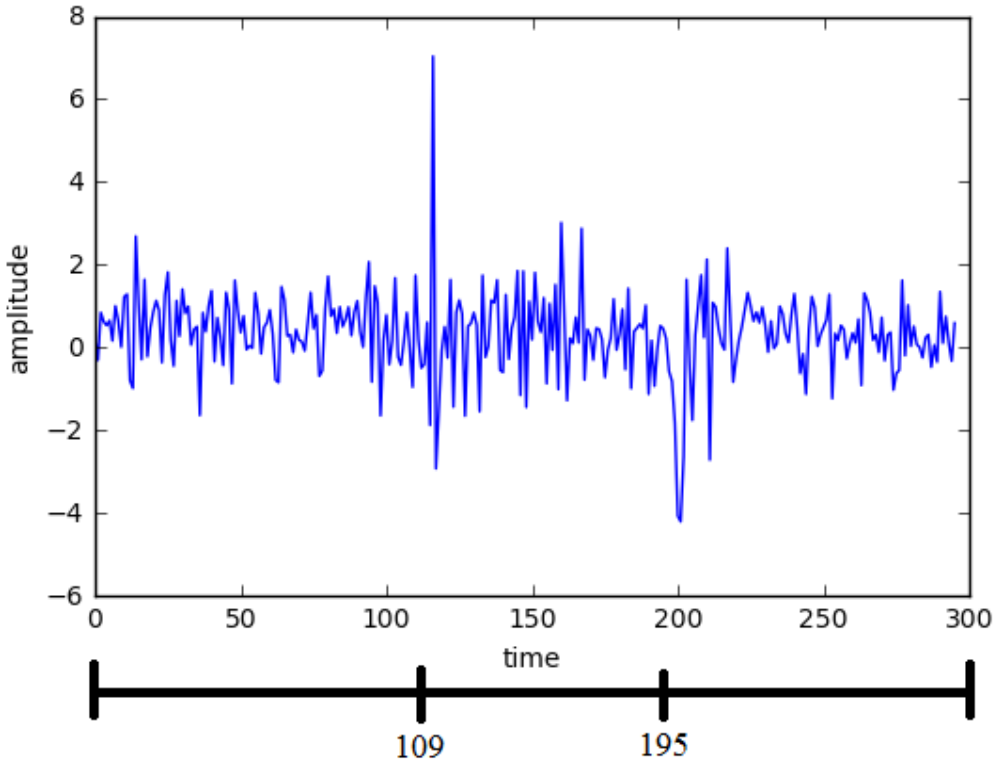

Figure 3.11: Predicted change points on retail trade data using jumping window

Segmentation of retail trade data using sliding window:

Predicted change points (shown in Figure 3.12): 109, 156, 194, 212

Real change points: 111, 119, 192, 210

Predicted segments: (0 - 108), (112 - 155), (156 - 193), (197 - 211), (214 - 288)

Real segments: (0 - 110), (111 - 118), (119 - 191), (192 - 209), (210 - 297)

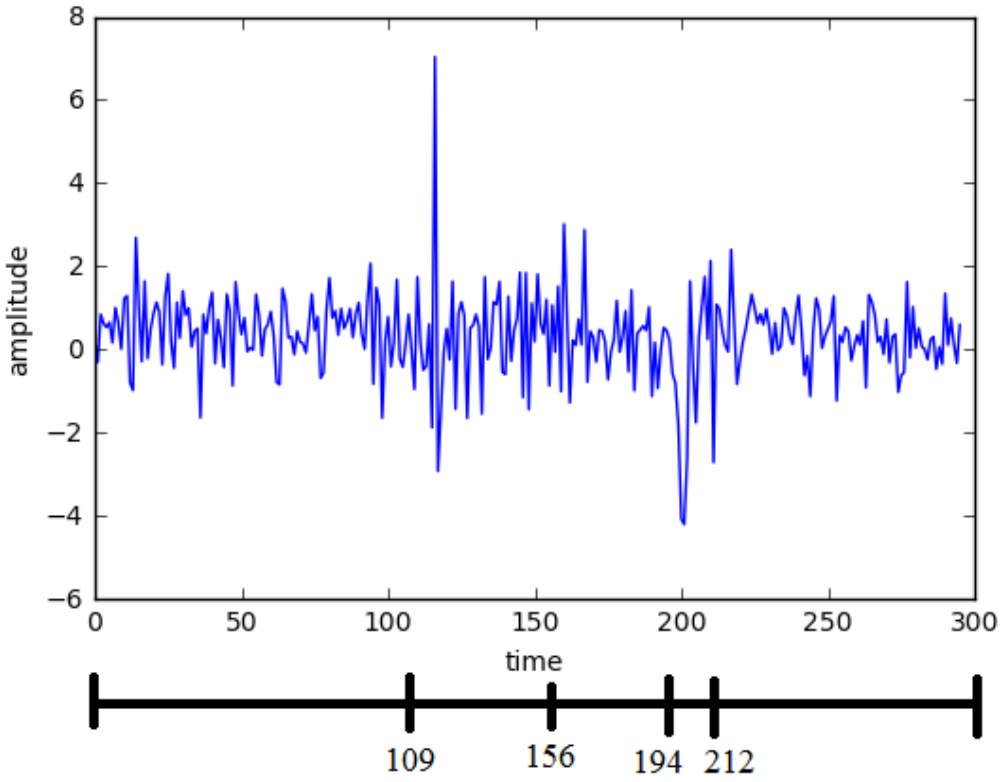

Figure 3.12: Predicted change points on retail trade data using sliding window

fMRI data:

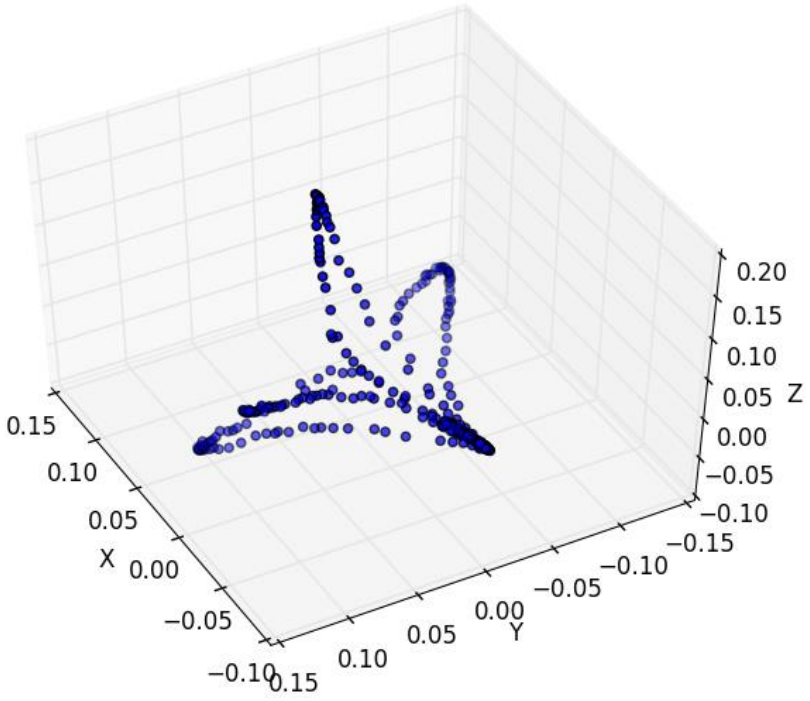

Figure 3.13: LLE on fMRI data using 16 neighbors and 3 components

Segmentation of fMRI data using jumping window approach:

Predicted segments (shown in Figure 3.14): (0 - 21), (22 - 41), (42 - 62), (63 - 84), (85 - 114), (115 - 145), (146 - 177), (178 - 197), (198 - 236), (237 - 275), (276 - 311)

Real segments: Not available as this is unlabeled data. The embedding space provides intuition regarding the segmentation of the fMRI data as shown in Figure 3.15.

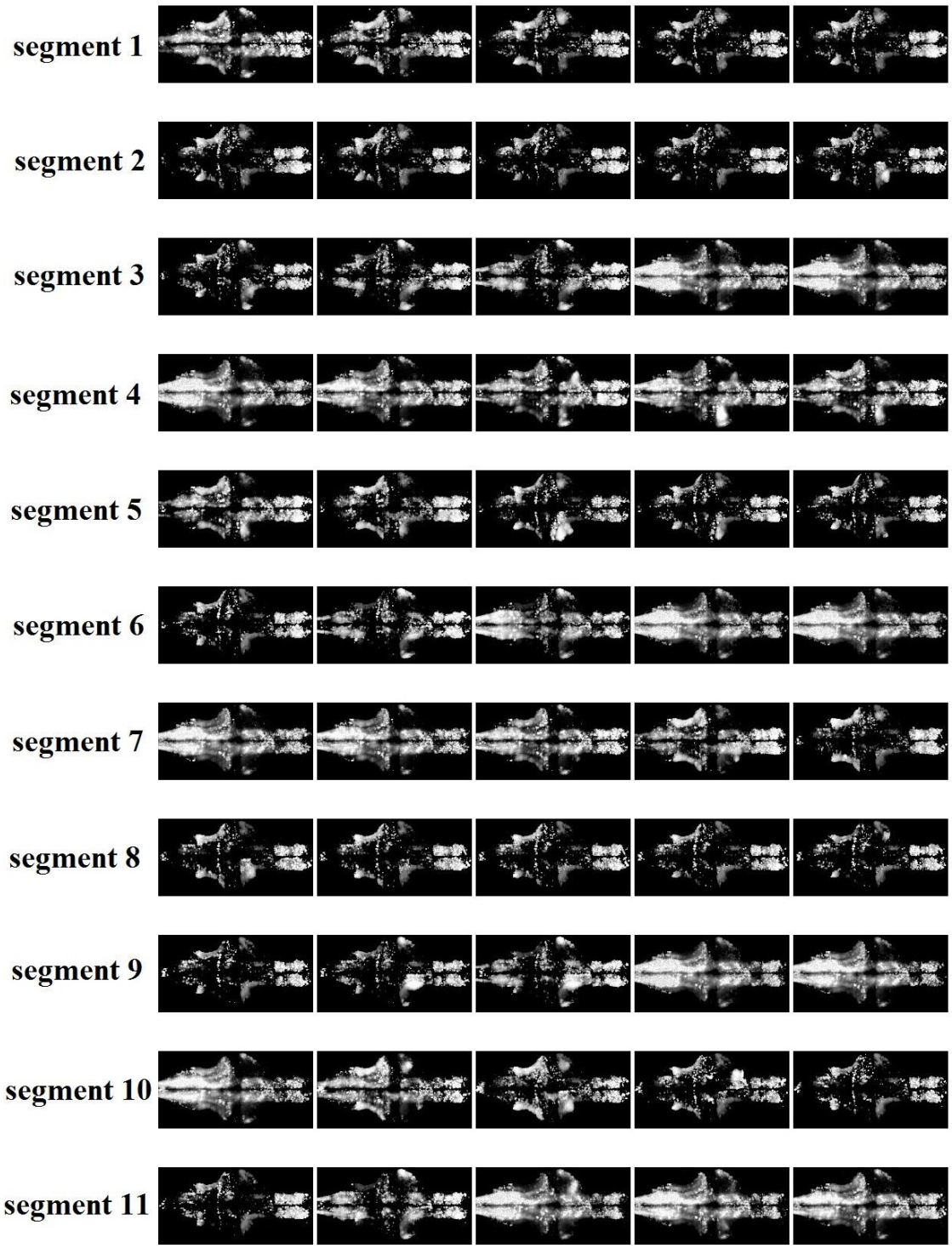

Figure 3.14: All predicted segments of fMRI data using jumping window

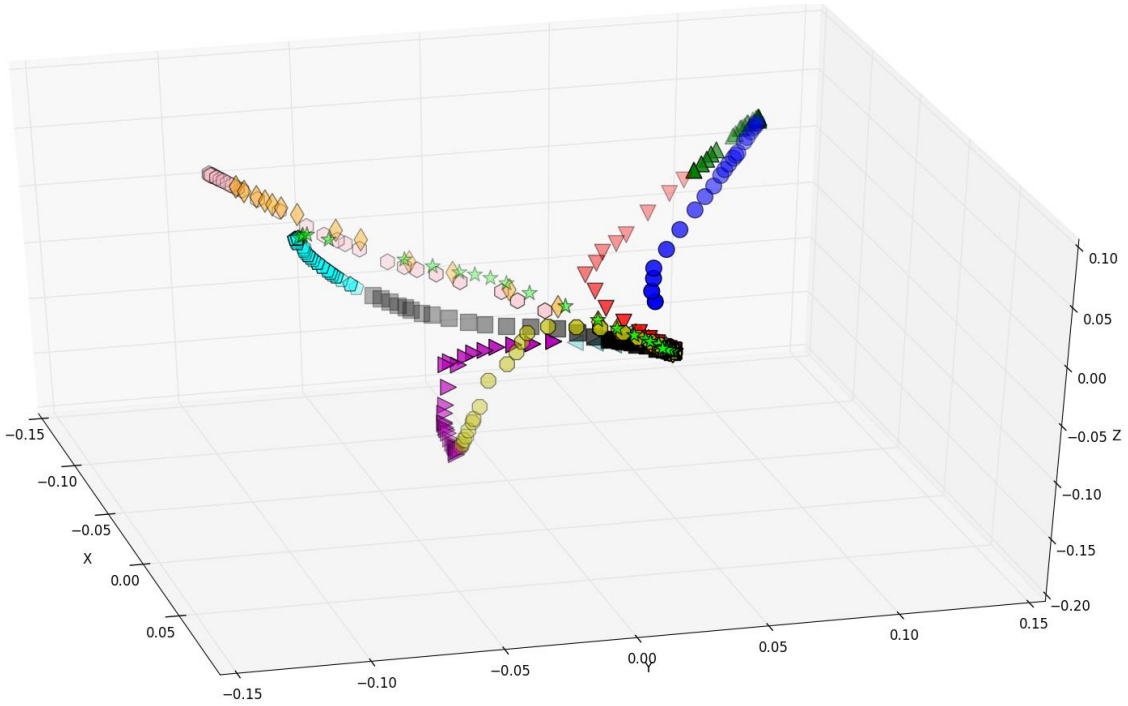

Figure 3.15: Different segments in embedding space using jumping window

Segmentation of fMRI data using sliding window approach:

Predicted segments (shown in Figure 3.16): (0 - 24), (28 - 49), (50 - 81), (92 - 137), (154 - 173), (174 - 184), (185 - 196), (210 - 243), (253 - 274), (275 - 282), (283 - 311)

Real segments: Not available as this is unlabeled data. The embedding space provides intuition regarding the segmentation of the fMRI data as shown in Figure 3.17.

Figure 3.16: All predicted segments of fMRI data using sliding window

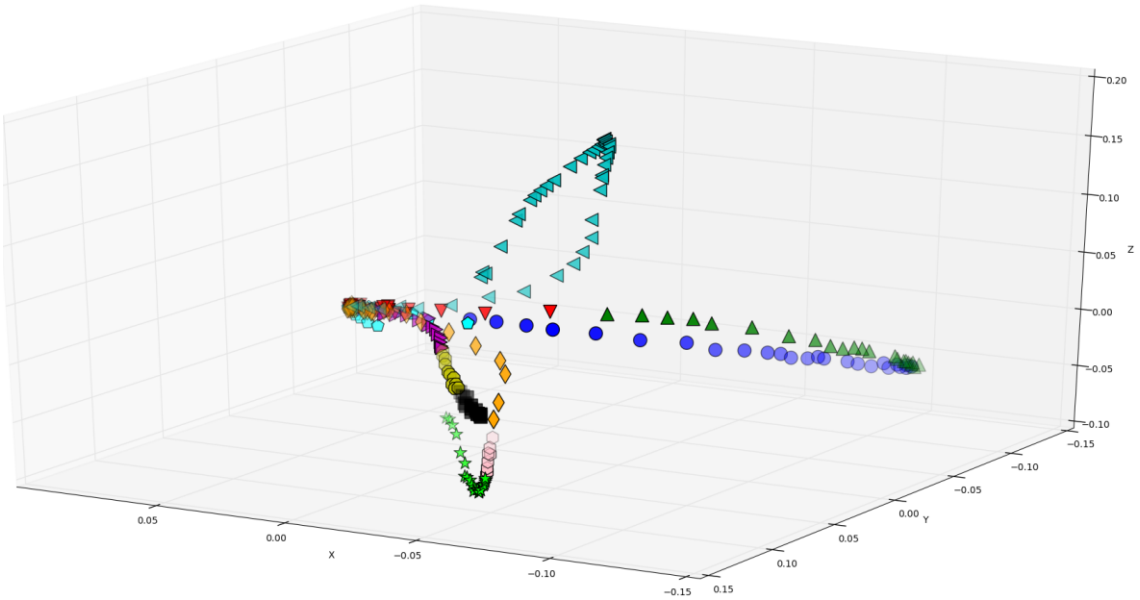

Figure 3.17: Different segments in embedding space using sliding window

CHAPTER 4

CONCLUSION

The proposed pipeline is a blind non-parametric multivariate approach to segment high dimensional time-series data. The pipeline works on various datasets and each stage of the pipeline can be replaced by a different algorithm if so desired. As shown in the results, the jumping window method outperforms the sliding window method. This is because in sliding window approach, data points from the previous segment are used for partitioning the next segment. Thus, the data points within each instantaneous window are sometimes heterogeneous in nature and sometimes homogeneous depending on the location of the sliding window. In case of jumping window approach, the window jumps from existing segment to the next stream of incoming data after a change point has occurred. Thus, the data points within each instance of the window are always homogeneous in nature.

The pipeline has the elements of working as an online method, with the only limitation of locally linear embedding (LLE). Currently, in this work LLE needs the entire batch of data for dimensionality reduction. An incremental LLE has been proposed in [30], which if implemented can turn the pipeline to a truly online method. Online methods are important in the age of big data for scalability purposes.

There are some hyperparameters in the pipeline which are not optimized and are kept constant. They are tweaked manually to get better segmentation. These are:

- The number of dependencies on the previous time-steps or the order of the Markov chain: This is kept constant at 4 throughout the work.
- The dimension of the embedding space: This is kept constant at 3 throughout the work.
- The combination of kernels: Depending on the data, this can be dynamically generated as proposed in [26]. We use the product of squared exponential and periodic kernel throughout the work.
- The length of moving window: We use different user specified lengths of windows for different data.
- The stride of sliding window: This is kept constant at 1 throughout the work.

If the above mentioned concerns are addressed, I believe the proposed pipeline will get significant improvement in segmenting high dimensional multivariate time-series data.

In all the experiments, the proposed pipeline was run on a single CPU using scikit-learn [31] and GPy Gaussian process framework [32].

REFERENCES

[1] Qi, Jin-Peng, et al. "A Novel Method for Fast Change-Point Detection on Simulated Time Series and Electrocardiogram Data." *PloS one* 9.4 (2014): e93365.

[2] Lawhern, Vernon, Scott Kerick, and Kay A. Robbins. "Detecting alpha spindle events in EEG time series using adaptive autoregressive models." *BMC neuroscience* 14.1 (2013): 1.

[3] Kass-Hout, Taha A., et al. "Application of change point analysis to daily influenza-like illness emergency department visits." *Journal of the American Medical Informatics Association* 19.6 (2012): 1075-1081.

[4] Gallagher, Colin, Robert Lund, and Michael Robbins. "Changepoint detection in climate time series with long-term trends." *Journal of Climate* 26.14 (2013): 4994-5006.

[5] Lund, Robert, et al. "Changepoint detection in periodic and autocorrelated time series." *Journal of Climate* 20.20 (2007): 5178-5190.

[6] Chauvet, Marcelle, and Jeremy M. Piger. "Identifying Business Cycle Turning Points in Real Time (Digest Summary)." *Federal Reserve Bank of St. Louis Review* 85.2 (2003): 47-61.

[7] Chauvet, Marcelle. "An econometric characterization of business cycle dynamics with factor structure and regime switching." *International economic review* (1998): 969-996.

[8] Andersson, Eva, David Bock, and Marianne Frisén. "Some statistical aspects of methods for detection of turning points in business cycles." *Journal of Applied Statistics* 33.3 (2006): 257-278.

[9] Yamada, Makoto, et al. "Change-Point Detection with Feature Selection in High-Dimensional Time-Series Data." *IJCAI*. 2013.

[10] Panagiotou, Vana. *Blind segmentation of time-series: A two-level approach*. Diss. TU Delft, Delft University of Technology, 2015.

[11] Rigaill, Guillem. "A pruned dynamic programming algorithm to recover the best segmentations with 1 to K_max change-points." *Journal de la Société Française de Statistique* 156.4 (2015): 180-205.

[12] Keogh, Eamonn, et al. "Segmenting time series: A survey and novel approach." *Data mining in time series databases* 57 (2004): 1-22.

[13] Keogh, Eamonn, et al. "An online algorithm for segmenting time series." *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on*. IEEE, 2001.

[14] Ge, Xianping, and Padhraic Smyth. *Segmental semi-Markov models for change-point detection with applications to semiconductor manufacturing*. Technical report, University of California at Irvine, March, 2000.

[15] Adams, Ryan Prescott, and David JC MacKay. "Bayesian online changepoint detection." *arXiv preprint arXiv:0710.3742* (2007).

[16] Rao, Yadunandana N., and Jose C. Principe. "A fast on-line generalized eigendecomposition algorithm for time series segmentation." *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*. IEEE, 2000.

[17] Do, Chuong B. "Gaussian processes." (2008).

[18] Ahrens, Misha B., et al. "Whole-brain functional imaging at cellular resolution using light-sheet microscopy." *Nature methods* 10.5 (2013): 413-420.

[19] Ghodsi, Ali. "Dimensionality reduction a short tutorial." *Department of Statistics and Actuarial Science, Univ. of Waterloo, Ontario, Canada* 37 (2006): 38.

[20] Raftery, Adrian E. "A model for high-order Markov chains." *Journal of the Royal Statistical Society. Series B (Methodological)* (1985): 528-539.

[21] Ebden, Mark. "Gaussian processes for regression: A quick introduction." *The Website of Robotics Research Group in Department on Engineering Science, University of Oxford* (2008).

[22] Saul, Lawrence K., and Sam T. Roweis. "Think globally, fit locally: unsupervised learning of low dimensional manifolds." *Journal of Machine Learning Research* 4.Jun (2003): 119-155.

[23] Saul, Lawrence K., and Sam T. Roweis. "An introduction to locally linear embedding." *unpublished. Available at: http://www. cs. toronto. edu/~ roweis/lle/publications. html* (2000).

[24] Rasmussen, Carl Edward. "Gaussian processes in machine learning." *Advanced lectures on machine learning*. Springer Berlin Heidelberg, 2004. 63-71.

[25] Chandola, Varun, and Ranga Raju Vatsavai. "A Gaussian Process Based Online Change Detection Algorithm for Monitoring Periodic Time Series." *SDM*. 2011.

[26] Duvenaud, David K., et al. "Structure Discovery in Nonparametric Regression through Compositional Kernel Search." *ICML (3)*. 2013.

[27] Williams, Christopher KI, and Carl Edward Rasmussen. "Gaussian processes for regression." (1996).

[28] Rasmussen, Carl Edward. "Gaussian processes for machine learning." (2006).

[29] Warren, Rik, Robert F. Smith, and Anne K. Cybenko. *Use of Mahalanobis distance for detecting outliers and outlier clusters in markedly non-normal data: a vehicular traffic example*. SRA INTERNATIONAL INC DAYTON OH, 2011.

[30] Schuon, Sebastian, et al. "Truly incremental locally linear embedding." *1st International Workshop on Cognition for Technical Systems*. 2008.

[31] Pedregosa, Fabian, et al. "Scikit-learn: Machine learning in Python." *Journal of Machine Learning Research* 12.Oct (2011): 2825-2830.

[32] Hensman, James, Nicolo Fusi, and Neil D. Lawrence. "Gaussian processes for big data." *arXiv preprint arXiv:1309.6835* (2013).