Final Project:

Change points detection in multi-channel time series

Reichman University, VIA Transportation

Or Katz

katzor20@gmail.com

Nadav Elyakim

27nadav@gmail.com

VIA supervisors: Yaron Taub and Nati Aharon

Abstract

In this paper, we present an algorithm for detecting multiple changes in high-dimensional time series data that outperforms previous algorithms. The problem of change point detection in high-dimensional time series has received a lot of attention in the past two decades, from basic statistical methods to the use of advanced deep neural networks architectures. The proposed method is an extension of MULLR [37], an algorithm to detect multiple change points based on changes in mean parameters only in high-dimensional time series data, while relying primarily on assumptions concerning the time series, such as assuming a constant variance parameter to the data distribution. In addition to reducing the MULLR assumptions, our extended approach allows us to detect change points caused by changes in very novel parameters of the data distribution. These parameters include trend (slope) change, variance change or a combination of all parameters. Thus, our algorithm achieved significantly higher performance than other SOTA algorithms currently available. Our method is based on the log-likelihood ratio (LLR) test [38]. This test is a well-established statistical technique, which is used in this study to detect changes in the distribution of a time series. Due to the algorithm design which integrates the

LLR test with linear regression modeling and the ability to process high-dimensional, time series data, our algorithm achieved SOTA results in terms of robustness, sensitivity and confidence. A performance evaluation of the algorithm was conducted on both synthetic and real-world data sets provided by VIA Transportation, a company that develops optimization algorithms for public transportation. The results suggest that the proposed algorithm is a promising tool for the analysis of high-dimensional time series data, and has potential applications in a range of fields, including finance, biology, and engineering.

1 Introduction

1.1 Background

Change point detection problems are commonly seen in many statistical and scientific sectors including functional data analysis [3, 2] and time series inspection [34, 17, 39]. These problems have applications to fields of biomedical engineering [37, 39], genomics [35], financial [9, 5] among many others. Statistical testing and estimation of change points have a long history with extensive literature. However, traditional methods for change point detection are often limited in their ability to detect multiple change points in a high-dimensional data set[16, 27, 6, 16, 26, 34]. This paper studies the problem of detecting multiple change points in high-dimensional time series. The ability to identify a change point from a single series is one of the major challenges in the detection of change points. A time series is a sequence of observations measured at successive times. The use of time series data provides an additional source of information. In this variable, the moment at which each data point was monitored is described, and as a result, the order of dependency between the data points is indicated. Combining the data with the time variable illustrates how the data adjusts over time. A one-dimensional time series is typically presented with the time variable on the x-axis and the data value on the y-axis. Change points divide each time series (channel) into segments, where the values within each segment have similar statistical properties, like slope, mean or standard deviation. Change points are defined at the first time step in each new segment. As a result, the number of change points is always one less than the number of segments. Today, most of the acceptable methods of decision-making are based on time series data monitoring and analysis uni-channel, one source of information at a time, or only one parameter change of the distribution. Our algorithm utilizes a combination of model selection and hypothesis testing techniques to identify multiple change points in high-dimensional data. We demonstrate the effectiveness of our method through a series of simulations and real-world examples. Our results show that the proposed algorithm outperforms existing methods in terms of both detection accuracy and computational efficiency. **By using multiple series together, our proposed algorithm takes advantage of the multidimensional nature of the data and locates the common change point**. Using this approach is intuitively more powerful and allows for enhanced accuracy and sensitivity in detecting change points. The work we present here engages multiple sources of information to predict multiple change points of the distribution parameters while reducing the errors of prediction, enabling a sensitive and higher confidence decision-making model. Our proposed algorithm is a valuable tool for researchers and practitioners who work with high-dimensional data sets and can be applied to a wide range of fields.

1.2 VIA company

This work was produced as a result of a collaboration with the VIA R&D department. VIA is a pioneer in the Transit-Tech sector, using new technologies to power public mobility systems and optimize networks of dynamic shuttles, buses, wheelchair-accessible vehicles, school buses, and au-

tonomous vehicles around the globe. The company's platform uses algorithms to adapt ride-sharing routes to passenger needs. VIA's technology enables minibuses or other large shared taxis to collect passengers from prearranged stops and to adjust routes in real-time according to passenger demand, traffic congestion, and other factors. VIA licenses its technology to cities, transportation authorities, school districts, universities, and private organizations to help them build their own technology-driven transportation networks. The company operates in partnership with over 400 local governments across more than 20 countries worldwide. VIA's city success team uses change point detection algorithms as an automated method for offline performance monitoring. To measure the company's performance and impact, it uses multiple metrics across all cities. In the current system, the data collected and aggregated on a weekly basis is fed into the change point detection algorithm on a periodic basis. By working on a weekly basis, they can ensure that sufficient data is gathered to feed the algorithm. VIA's methods permit them to detect change points in trends, variance or mean parameters. The company uses this algorithm as "Active" monitoring which generates alerts (compared to "passive" monitoring, e.g., dashboards). As a result, shorter response times and accurate analysis are achieved, and leading to a competitive advantage.

1.3 Defining the problem

In addition to Via Company, there are many others that have attempted to identify multiple change points in datasets. First, we define the problem by choosing high-dimensional, tabular, numerical, time series data as input to the algorithm. The output will be a list of indexes associated with the point (the time bin) where changes in the data were identified, i.e. a change point. As part of the problem setup, we consider it as an offline change point detection problem, which considers the entire data set at once, and look back in time to recognize where the change occurred. A wide range of PDFs containing various parameters could be used to represent the data.

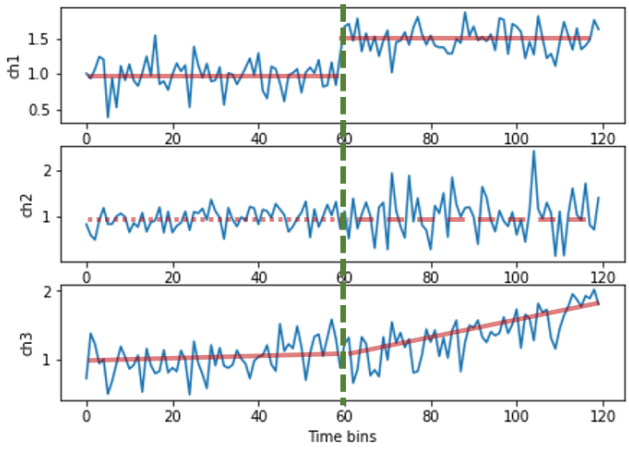

Figure 1: An example dataset contains three channels that can be concatenated together as input to the algorithm. The first channel illustrates a Gaussian distribution with a change in the mean parameter at time bin 60. The second channel depicts a Gaussian distribution with a change in variance parameter at time bin 60. The last channel indicates a change in his trend at time bin 60. Based on this dataset as an input, the algorithm will result in a change point occurring in time bin 60 (the vertical dashed green line).

This work required our method to identify change points corresponding to changes in the mean, variance, slope parameters, or any combination of these. Due to the fact that a change point divides a time series (channel) into two separate segments, we search for a method that optimally slices the multi-dimensional data into segments while ensuring that the data within each segment have similar statistical properties, PDFs, and parameters. This is challenging since the number and location of change points are unknown. Theoretically, a perfect mathematical solution would be to segment each data point into private segments. Another challenge will be to reduce the number of errors. It's possible that a false positive error will occur when the noise in the data is mistakenly interpreted as a change in different parameters that describe a segment's PDF. Similarly, low signal-to-noise ratios that hide origin signal characteristics may lead to a miss of true change points. Working with multi-dimensional datasets can address the challenge of a low signal-to-noise ratio. By examining other channels in the dataset, we can aggregate several insignificant change points at the same location to achieve sufficient significance. This will cancel the rejection of candidate change points. Moreover, identifying a change point in multiple channels at the same location enhances segment partition confidence and significance. To meet our objectives, our proposed solution should be adaptable to different datasets, high-performing, provide stable results, and scalable in both the number of data points and the number of dimensions. Additionally, the solution should increase confidence while dividing the dataset into the optimal number of segments. Currently, VIA monitors enormous amounts of data from its users, operation surroundings, customers and vehicles to make wise business decisions. VIA, then, aggregates the collected data in a 1-dimensional manner. However, this approach leads to two significant drawbacks. First, there are sensitivity issues. While it's impossible to calibrate the algorithm parameters in advance to fit all sorts of datasets, it's very likely to result in false positive (too sensitive) or false negative (insensitive) mistakes. For instance, a too-low tuning of the 'Penalty' hyper-parameter, which is similar to a threshold and influences whether to accept a change point, will result in a high proportion of false positive errors. One reason for this situation may be due to different numerical scales and ranges between the datasets. The second disadvantage of one-dimensional aggregation is that the decision relies on one source of information, which may undermine the confidence of the decision-makers. Throughout this study, we assume that each channel in each of the datasets is Gaussian noise distributed, the channels are independent and identically distributed (i.i.d.). Means change points can occur in the following statistical parameters: intercept (mean-shift), variance and slope (trend) of the channel 1. The objectives of our work are to develop an algorithm that provides better results than the current SOTA algorithms and specifically the algorithm used currently by VIA. Furthermore, we expect the algorithm to provide the user with a high level of confidence in its insights. Finally, our solution should be explainable, i.e. we require our solution tool to enable users to track, understand and reexamine the algorithm determinations in a transparent manner.

2 Related work

2.1 Related work

Several works have been proposed for change-point detection in time series data, with various focuses and approaches. These techniques include both supervised and unsupervised methods. In traditional supervised change point detection techniques, support vector machines, naive Bayes, and logistic regression [12, 13] are used as an example of how change point detection can be treated as a binary classification problem. Unsupervised methods to detect change points include methods that utilize likelihood ratio tests based on the calculation of probability density functions like Change Finder and Cumulative Sum [21, 9, 18, 4] or the calculation of the ratio directly to model the density ratio

between two consequent time intervals, SPLL, for example, [24, 1]. Other unsupervised methods are Bayesian approaches [28, 29], such as the Gaussian Process [7, 31], a probabilistic method that computes the p-value for the actual observation y_t under the reference distribution, $N(y_t, \sigma_t^2)$. In the event that the actual observation does not follow the predictive distribution, a threshold of $\alpha \in (0, 1)$ is used as an indicator of a possible point of change. In addition, subspace modeling has been proposed as a solution [15, 25, 30, 19]. By mapping observations onto a different dimensional feature space and comparing the homogeneity of each sub-sequence, these methods detect change points in the dataset. There are common disadvantages associated with these methods. They rely heavily on mapping functions and have difficulty scaling as data dimensionality increases. The graph-based technique is a method for representing time series observations as a graph and applying statistical tests to detect change points based on the graph structure [41, 8]. Among the other approaches reported are clustering methods that are based on the assumption that observations within clusters are identically distributed and observations between adjacent clusters are not. For example, SWAB and Model fitting [22, 32] consider the problem as a clustering problem with a known or unknown number of clusters. The initial work exploiting deep learning technology was limited to uni-variate time series data input. Initial projects used fully convolution networks to perform the task of end-to-end time series classification and reduce the need to pre-process the data (feature selection), allowing the user to work with raw data. These works were limited to processing single channel time series data [10, 36]. The development of advanced deep learning architectures and their sharing within scientific and research communities contributed to the deployment of those architectures for detecting change points, convolutional networks, long short-term memory, and recurrent neural network models significantly enhancing the performance of fully convolutional networks [20]. Research related to multivariate time series was used in related tasks such as anomaly detection and diagnosis in multivariate time series [40, 11]. It is critical to note that deep-learning and machine-learning solutions share some common drawbacks. In order to achieve a high level of model performance, a training phase is essential. In turn, a significant amount of computational resources and sufficient quality and quantity of training data are required for such a training phase, which is not always accessible in the real world. In most cases, labeled data is required (supervised learning). As a result, the results might be biased in the direction of the training data distribution, which affects the robustness of the solutions. Moreover, in real-world situations, these methods are less flexible since they are based on pre-designed parametric models. Aside from this, these solutions are difficult to analyze and evaluate and are often referred to as 'black boxes' due to the lack of transparency.

2.2 PELT

As a benchmark for this project we chose the change point detection algorithm that was used by VIA's city success team. VIA based its algorithm on PELT algorithm [23], for "Pruned Exact Linear Time", a uni-channel change point detection algorithm. In terms of computational cost, PELT algorithm provides a computationally efficient method for detecting change points in time series data. The algorithm returns a vector with the optimal change points of the data it receives. Because the enumeration of all possible partitions is impossible, the algorithm relies on a pruning rule. Many indexes are discarded, greatly reducing the computational cost while retaining the ability to find the algorithm's optimal segmentation solution in linear time complexity. Due to its dynamic computing architecture, this is possible. The time series is first divided into segments of varying lengths. In the case of a time series with n data points, it can be divided into segments of $1, 2, ..., n-1$ lengths. Then, for each segment length, the algorithm calculates the distance between the data points in the segment and their mean value using a cost function, such as sum of squared errors (SSE). Dynamic programming is then used to determine the optimal set of change points

that minimizes the sum of SSE. For each segment length, the cumulative sum of SSE is calculated and compared with the cumulative sum of SSE for the previous segment length. By pruning change points that do not improve the SSE, the algorithm retains only the optimal change points. Finally, the algorithm outputs the set of optimal change points.

Various cost functions are supported by PELT, which allows it to detect changes caused by changes in mean, variance, and trend. To detect changes in the mean and median of a signal, L1 and L2 norms are used, Normal is used to detect changes in the mean and variance of a Gaussian series, and kernel-based functions (RBF) are used to detect changes in the mean of a transformed signal.

The hyperparameters of PELT are: $min - size$, which controls the minimum distance between change points; for instance, if $min - size = 10$, all change points will be at least 10 samples apart. $cost - func$, which determines the cost function that PELT tries to minimize in order to return the optimal segmentation of the data. Via used PELT with a custom cost function developed by Via. This algorithm allows Via's city success team to detect trend, variance and mean changes in the uni-channel (1-dimensional) data. VIA's custom cost function first calculates Kendall's tau statistical test, to test for a trend in the data. Using the test score and its significance, determine the appropriate cost function. While there is a significant trend in the data, the cost function will be the Normal cost function on the residuals with respect to least squares linear fit i.e., $log(RSS/n)$. Otherwise, the cost function is the Normal cost, $log(variance)$. The two alternative cost functions were chosen taking into account the possible range of results. In the course of analyzing, testing, and evaluating the described algorithm, set as a benchmark for this work, we have identified three main deficiencies that we intend to address and overcome in this paper. First, we noticed a significant amount of false positive and false negative errors caused by general algorithm issues. Second, we found that in some common use cases the algorithm will declare a change point in trend while the true case is a change in mean (step in data), leading to an incorrect cost function usage and errors. This is a result of Kendall's tau statistical test determining a significant trend while there is no actual trend in the data, but a clear change in the mean parameter 2. Third, this algorithm requires a predefined constant penalty value, which serves as a threshold for making changes to points (a change point is valid if it reduces the segmentation cost values by more than the penalty value that is added). Since we are developing an algorithm that interprets and aggregates diverse data sources, redefining constant penalty values will lead to an inappropriate threshold, which will result in errors.

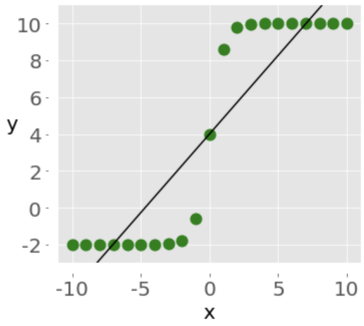

Figure 2: The presented figure highlights a constraint of the Kendall Tau correlation test as it may produce misleading results by inferring a significant positive trend in the data, where only a change in the mean parameter exists. This observation raises a concern regarding the reliability of the Kendall Tau correlation test and emphasizes the need for caution in interpreting its results.

2.3 MULLR

In conducting a comprehensive search, examining and analyzing various approaches to address the problem at hand, we have selected the MULLR algorithm [35, 37] as the most appropriate framework for our research. The reason for this is the algorithm's significant advantages and the potential to remove its major limitations. In order to meet the requirements of the problem we have presented, we

have developed an algorithm based on this method that offers a more comprehensive and extensive solution. MULLR, Multi-channel Log-Likelihood Ratio test, is an algorithm developed to jointly identify multiple mean change points in multi-channel data sets. Using simulated data, this work validates the algorithm and characterizes its detection capability. Moreover, the author presents simulation results demonstrating the improved detection performance of the proposed method compared to existing single-channel methods. The MULLR approach is different from deep learning approaches as it provides a straightforward and flexible method for analyzing multi-channel data sets which requires no training sets to operate and is easy to explain.

MULLR algorithm builds on the binary segmentation and log-likelihood ratio test framework. To identify change points, each time bin is evaluated for the possibility of a change point. For the $k-$th time bin, two hypotheses are considered:

1. H_1- The alternative hypothesis: there is a change point at the time bin k
2. H_0 The null hypothesis: there is no change point at the time bin

The likelihood of the observed data is evaluated under each hypothesis ($L_k^1 A$ and L_k^0), and the log-likelihood ratio value ($LLR_k = \log \frac{L_k^1}{L_k^0}$) is computed and assigned to the $k-$th time bin. This LLR quantifies the degree to which the alternative hypothesis is favored over the null hypothesis. After the LLR value is computed for all bins ($k = 1, ..., N-1$), the maximum LLR time index (time bin) defines the location of a candidate change point. This change point is accepted if the matching LLR value exceeds a significance threshold (r_N, included to robust control the false positive rate, determined by a user-supplied false positive rate (α) and the number of free parameters across all measurement dimensions, degree of freedom). Once a change point is identified, the above multi-channel log-likelihood ratio test is applied to the trace segments before and after the change point and this process continues recursively on binary segmented portions of the trace until no more change points are found. Importantly, the MULLR algorithm assumes that the measurement channels are independent and have a constant noise value. Therefore, the total likelihood function is the sum of the likelihoods of each individual channel. Moreover, this work predetermined the variance parameter for the channels instead of extracting it from the data.

Among MULLR's limitations are that it requires all channels to have identical noise (variance) properties or that it cannot be used with data that contains useful information due to noise. Furthermore, MULLR is designed specifically for detecting change points in only one parameter, the mean. MULLR requires careful selection of the prior distributions to model the data. This may require expertise in Bayesian statistics and prior knowledge of the data. Finally, MULLR measures the timing and location of change points, but may not reveal the underlying mechanisms that cause those changes. Experimental validation may be needed to interpret the results. MULLR's most significant advantages are the ability to analyze multi-channel data, making it suitable for analyzing complex systems and identifying changes in multiple variables simultaneously. The capability to accurately detect change points in noisy conditions as well as when the signal-to-noise ratio is low. Finlay, a wide range of data measurements can be processed by MULLR, making it a versatile tool for studying a wide range of systems.

Based on the MULLR approach, we have expanded the algorithm to detect change points based not only on one parameter but on three parameters (variance, mean and slope) or a combination of them. In contrast to MULLR, our extension does not require that all channels in the dataset share identical parameter values. Along with the prediction of the location of change points, we provide additional figures that enable the user to analyze and validate the data and algorithm result.

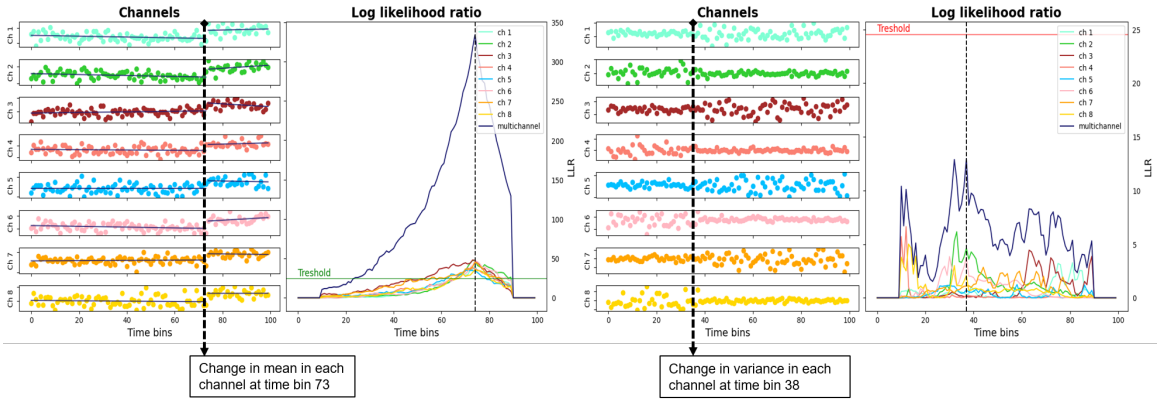

Figure 3: An example of MULLR implementation. On the left figure, there is an eight-Dimensional dataset (contains eight channels) for which we want to locate change points. In the dataset, each channel contains a change point in the mean parameter at time bin 73. The common Log-Likelihood Ratio (LLR, The dark blue vertical line in the left figure) maximum value is located at time point 73, and this value crosses the threshold (The green horizontal line). That means that the algorithm predicts correctly the change point at time 73. On the other hand, on the right figure, there is an eight-dimensional dataset also, but this time in each channel there is a change in the variance parameter (at time point 38). In this case, the common LLR is noisy, its value is low, and it does not cross the threshold. A change point cannot be identified by the algorithm if it occurs in a parameter other than the mean. Such cases will also be handled by our extension to the algorithm.

3 Method EMULLR

Using MULLR as a starting point, we expanded it to fit VIA's problem. In our work, we developed the EMULLR algorithm. This is an Extension Log-Likelihood Ratio test (EMULLR)[1] for Gaussian noise distributions (linear regression). By extending the test, we are able to apply it not only to identify change points caused by a change in the mean parameter but also to each of the three parameters that define the channel's distribution:

1. β^0 - y-intercept (mean-shift) of the channel.
2. β - the slope of the channel (Assuming the channel has some trend).
3. σ^2 - channel variance.

Our method is presented in this section, which contains an overview of the process, illustrating the differences between the original algorithm and our extension, algorithm Psuedu code, detailed mathematical development of the LLR test that takes into account the three parameters mentioned above, and an explanation of the threshold used throughout.//

[1] `https://github.com/ok123123123/Multi_Dim_CP_Detection`

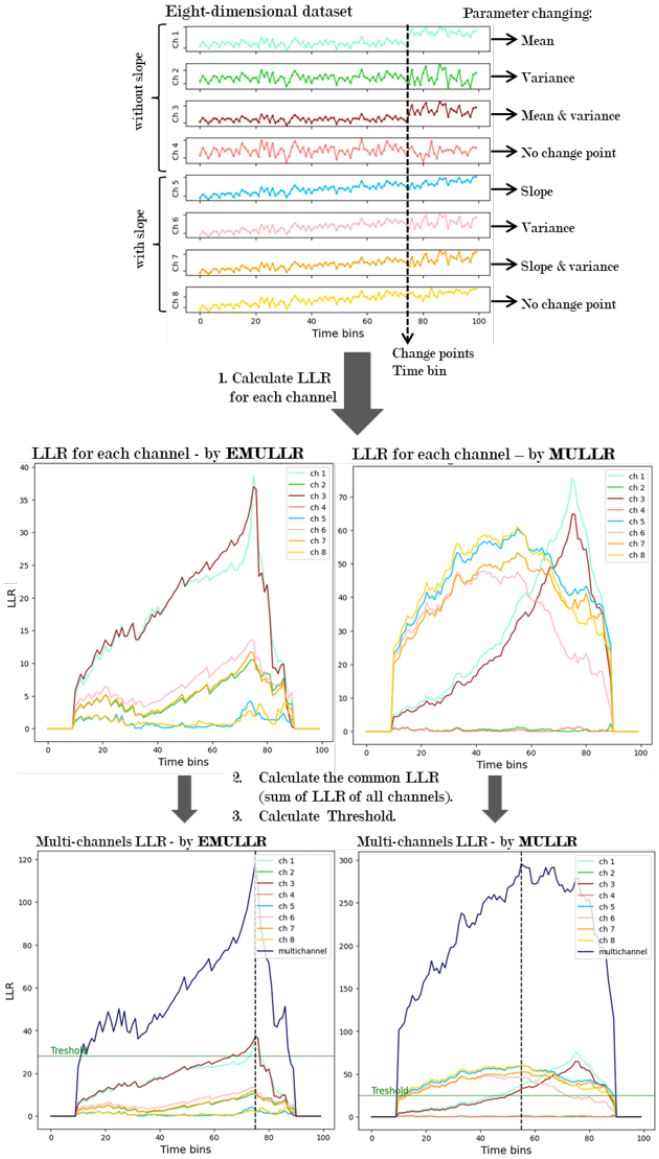

Figure 4: The upper graph shows an example of an eight-dimensional dataset with a change point at time bin 75. This dataset contains all types of change points and their possible combinations. Channels 1-4 are Gaussian distributed (without a trend) and channels 5-8 are based on linear regression (with trend). In order to predict change points, the first step of the algorithm is to calculate the LLR of each channel for all time bins. The calculation results can be seen in the two plots in the middle, on the left for EMULLR and on the right for MULLR (The colors match the channels' raw data above). In the next two steps (the plots at the bottom), we sum up the LLR for all channels (dark blue top line) and calculate the threshold (green horizontal line). If the maximum value of the common LLR (dashed vertical line) is higher than the threshold, the algorithm predicts it as a valid change point. In this case, the EMULLR correctly predicts a change point at time bin 75, while the MULLR returns an error. As can be seen, in the case of EMULLR, for each channel with a change point, the time bin with the maximum LLR value is the time when a change occurs in one or more of the parameters. While MULLR detects a change in the mean parameter only (channels 1 and 3), when a change occurs in other parameters (channels 2, 4-8), the channels' LLR plot shape looks more like a shallow and noisy hill without a peak (on the middle right plot). Therefore, when considering all channels, the MULLR prediction is wrong.

3.1 Overview of the entire EMULLR algorithm

Algorithm 1: A binary search for change points detection using EMULLR

Input: Array of the channels, shape - (number of channels, number of time bins), α (False positive rate), and $*edge$ (A more detailed explanation will follow).

Output: List of the change points (CP) which founded.

```
BinarySearch(channels, α)
llr ← CalculateLlrRatio (channels) ;        // sum of Channels LLR at any time bin
threshold ← CalculateThreshold (numberOfChannel, numOfTimeBins, α)
maxLlr ← max(lrr)
predTimeBin ← arg max(lrr)
if maxLlr > threshold // A CP was found.  Perform a recursive step
 then
    SubChannelsLeft ← channels[<predTimeBin]
    SubChannelsRight ← channels[>predTimeBin]
    ListLeftCP ← BinarySearch(SubChannelsLeft, α)
    ListRightCP ← BinarySearch(SubChannelsRight, α)
ListCP ← Concatenate(ListLeftCP, pred_time_bins, ListRightCP)
return ListCP
```

$*edge$ - The interval of time bins from the beginning and end of the channel from which we will not search for change points. In order to ensure the correctness of the algorithm that is based on the estimation of distribution parameters, we are required to make sure that there are "enough" data points in each distribution. The *edge* parameter basically defines what this minimum number of points is.

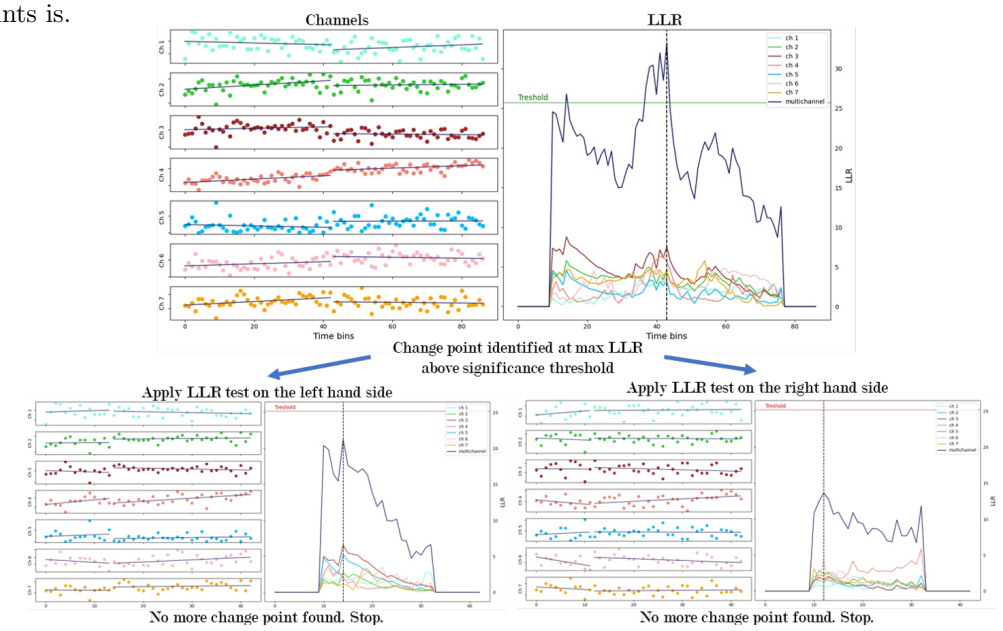

Figure 5: Identifying a change point in a data set that contains 7 channels. In the first stage of the binary search, the "root", a change point was located in time bin 43 using the LLR test. Following that, the channels were split into 2 segments, left and right of the time bin that was found in the previous stage. The LLR was applied to both segments, but no new change points were found.

3.2 Derivation of the log-likelihood ratio test

The following section provides an overview of the mathematical development for calculating the log-likelihood ratio so that it takes into account a possible change in each of the parameters which define the distribution. As mentioned before, we assume that each channel is Gaussian-noise distributed (linear regression model) and that the channels are independent. Each channel has N data points (time bins), defined as follows:

1. The distribution of X (channel) is arbitrary.
2. If $X = x$, then $Y = \beta^0 + \beta x + \epsilon$. ϵ is random noise variable and β^0, β are constants.
3. $\epsilon \sim N(0, \sigma^2)$ and is independent of X.

The probability density function of the above:

$$(1) \quad \prod_{i=1}^{N} p(y_i|x_i;\beta^0,\beta,\sigma^2) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_i-(\beta^0+\beta x_i))^2}{2\sigma^2}}$$

The log-likelihood function:

$$(2) \quad LL(\beta^0,\beta,\sigma^2) = \sum_{i=1}^{N} p(y_i|x_i;\beta^0,\beta,\sigma^2)$$

We test the null hypothesis of no change in the PDF parameters $(\beta^0, \beta, \sigma^2)$:

$$H_0 : \beta_1^0 = \beta_2^0 = ... = \beta_N^0 = \beta^0 \; and \; \beta_1 = \beta_2 = ... = \beta_N = \beta \; and \; \sigma_1^2 = \sigma_2^2 = ... = \sigma_N^2 = \sigma^2$$

Against the alternative hypothesis of a change in one or more of the parameters at time k $(\beta^0, \beta, \sigma^2)$:

$$H_1 : \beta_1^0 = ... = \beta_k^0 \neq \beta_{k+1}^0 = ... = \beta_N^0 \; or \; \beta_1 = ... = \beta_k \neq \beta_{k+1} = ... = \beta_N \; or \; \sigma_1^2 = ... = \sigma_k^2 \; \sigma_{k+1}^2 = ... = \sigma_N^2$$

We notate the left segment of a channel until the $k-$th time bin as $_{:k}$ and after the $k-$th time bin as $_{k:}$.

To calculate the log-likelihood ratio of a single channel we plug the alternative hypothesis over the null hypothesis for a change after the K'th time bin:

$$(3) \quad LLR = \frac{H_1}{H_0} = \frac{LL(\beta_{:k}^0, \beta_{:k}, \sigma_{:k}^2) \cdot LL(\beta_{k:}^0, \beta_{k:}, \sigma_{k:}^2)}{LL(\beta^0, \beta, \sigma^2)} =$$

$$\sum_{i=1}^{k} p(X_i = x_i; \beta_{:k}^0, \beta_{:k}, \sigma_{:k}^2) + \sum_{i=k+1}^{N} p(X_i = x_i; \beta_{k:}^0, \beta_{:k}, \sigma_{:k}^2) - \sum_{i=1}^{k} p(X_i = x_i; \beta^0, \beta, \sigma^2) =$$

$$-\frac{k}{2}\ln\sigma_{:k}^2 - \frac{1}{2\sigma_{:k}^2}\sum_{i=1}^{k}(y_i - \beta_{:k} - \beta_{:k}x_i)^2 - \frac{(N-k)}{2}\ln\sigma_{k:}^2 - \frac{1}{2\sigma_{k:}^2}\sum_{i=k+1}^{N}(y_i - \beta_{k:} - \beta_{k:}x_i)^2 +$$

$$\frac{N}{2}\ln\sigma^2 + \frac{1}{2\sigma^2}\sum_{i=1}^{N}(y_i - \beta - \beta x_i)^2$$

Next, we want to find the values of the parameters which ideally suit and describe our data distribution (channel distribution) - in order to calculate the maximum log-likelihood we used MLE, a maximum likelihood estimate. We will find the parameters which maximize the log-likelihood ratio (3) by partially deriving each parameter and equating it to 0:

1. First, for $\beta_{:k}$, $\beta_{k:}$ and β:

   (a) $\frac{dLLR}{d\beta_{:k}} = 0 \rightarrow \beta_{:k_{max}} = \frac{\sum_{i=1}^{k}(y_i - \overline{y}_{:k})(x_i - \overline{x}_{:k})}{\sum_{i=1}^{k}(x_i - \overline{x}_{:k})^2}$

   (b) $\frac{dLLR}{d\beta_{k:}} = 0 \rightarrow \beta_{k:_{max}} = \frac{\sum_{i=k+1}^{N}(y_i - \overline{y}_{k:})(x_i - \overline{x}_{k:})}{\sum_{i=k+1}^{N}(x_i - \overline{x}_{k:})^2}$

   (c) $\frac{dLLR}{d\beta} = 0 \rightarrow \beta_{max} = \frac{\sum_{i=1}^{N}(y_i - \overline{y})(x_i - \overline{x})}{\sum_{i=1}^{N}(x_i - \overline{x})^2}$

2. Second, for $\beta^0_{:k}$, $\beta^0_{k:}$ and β:

   (a) $\frac{dLLR}{d\beta^0_{:k}} = 0 \rightarrow \beta^0_{:k_{max}} = \overline{y}_{:k} - \beta_{:k_{max}}\overline{x}_{:k}$

   (b) $\frac{dLLR}{d\beta^0_{k:}} = 0 \rightarrow \beta^0_{k:_{max}} = \overline{y}_{k:} - \beta_{k:_{max}}\overline{x}_{k:}$

   (c) $\frac{dLLR}{d\beta^0} = 0 \rightarrow \beta^0_{max} = \overline{y} - \beta_{max}\overline{x}$

3. Last, for $\sigma^2_{:k}$, $\sigma^2_{k:}$ and σ^2:

   (a) $\frac{dLLR}{d\sigma^2_{:k}} = 0 \rightarrow \sigma^2_{:k} = \frac{1}{k}\sum_{i=1}^{k}(y_i - (\beta^0_{:k_{max}} - \beta_{:k_{max}}x_i))^2$

   (b) $\frac{dLLR}{d\sigma^2_{k:}} = 0 \rightarrow \sigma^2_{k:} = \frac{1}{N-k}\sum_{i=k+1}^{N}(y_i - (\beta^0_{k:_{max}} - \beta_{k:_{max}}x_i))^2$

   (c) $\frac{dLLR}{d\sigma^2} = 0 \rightarrow \sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(y_i - (\beta^0_{max} - \beta_{max}x_i))^2$

Once we have found the parameters that maximize the expression, we will plug them into the LLR expression (3) above. This will enable us to get the LLR value for time bin k. Assuming there are C independent Gaussian noise channels, the combined log-likelihood ratio for all channels in each time bin (k) equals the sum of the log-likelihood ratios of each channel in time bin (k). That is:

$$(4) \quad Total\_LLR_k = \sum_{1}^{C} LLR_k$$

3.3 Threshold for change point significance

In order to decrease the number of false positives we will accept the candidate change point in time k (assuming that the max of the channel's LLR is in time k), LLR_k if it is greater than the calculated threshold value. MULLR original article [35, 37] details the threshold and how it is calculated based on the works of Vostrikova [33] and Gombay [14]. The threshold depends on three parameters:

1. d - the number of degrees of freedom. In our case, each channel has 3 parameters $(\beta^0, \beta, \sigma^2)$, meaning 3 degrees of freedom. Therefore $d = 3 \cdot C$ (C is the number of channels).
2. α - allowed false positive rate. Level of confidence.
3. N - the number of data points (time bins).

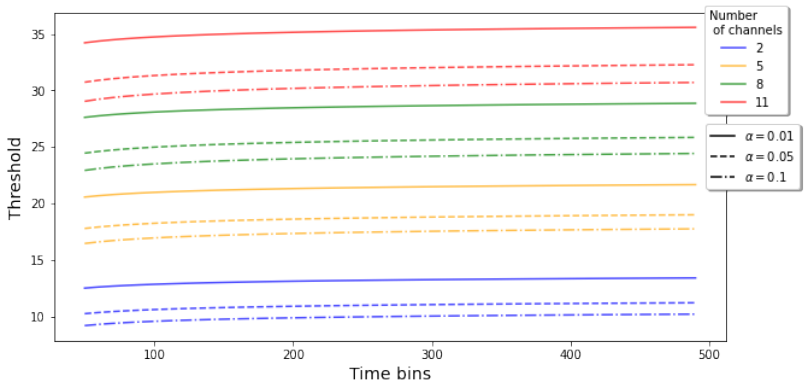

Figure 6: Threshold for change point significance and its dependence on α and N.

Apart from α, which is determined by the user as a business decision, the dataset determines the value of the other two threshold parameters automatically. Therefore, the algorithm depends only on one hyperparameter - α. As a result, it has a high degree of robustness and does not require calibration against different datasets. This advantage, which allows the solution to be applied to a wide variety of additional use cases without requiring adjustments, is very important in this problem of finding change points in different datasets.

4 Evaluation

We faced several challenges when evaluating our algorithm's performance. In order to examine the performance of our algorithm, we tested it using two data types, synthetic data and real-world data. Using synthetic data allowed us to utilize the ability of full control over the data attributes and parameters. The synthetic datasets were specifically chosen to challenge the proposed algorithm and test its performance in a realistic setting and in extreme cases. Furthermore, the data was not normalized to preserve the original characteristics of the data and to ensure that any changes in mean, variance and slope (trend) would be accurately detected by the algorithm. In terms of real-world data, we received data sets from VIA that in most cases did not contain ground truth labeling, since they did not have the algorithm or business knowledge to identify the multi-channel change points

we were looking for. We tackled this challenge through two different approaches. First, two out of all datasets contain ground truth labels. The labels were tagged by VIA's experts and correspond to major changes in VIA's service such as the launching of new services or impact updates. When ground truth labels were available we used them to evaluate our algorithm performance. In datasets that are not labeled we compared the results of our multi-channel algorithm, EMULLR, with the results of the single-channel algorithm (PELT) currently used by VIA and on the MULRR algorithm. This way, it was possible to estimate the performance of our algorithm.

5 Experiments

5.1 Experiments using synthetic data

In this section, we will present experiments performed on synthetic data to test the performance of the algorithm. We will also analyze the effect of different experimental setups and data characteristics on algorithm performance. The following is a list of some of the experiments available in the project repository:

1. Experiments changing the dataset characteristics:
   (a) **Number of channels** in each sample dataset.
   (b) Number of **time bins** in each channel in dataset.
   (c) Number of **change points** in each channel.
   (d) **Signal to noise ratio** - The intensity of the change in the mean or the slope parameter after the change point.
   (e) **Var increased by** - How much the variance increase or decrease after the change point.
2. Experimental parameters of the algorithm itself:
   (a) **Allowed deviation** - The allowed deviation time bins between the true label and the predication. For example, a change point prediction at time 59 of a true label at time 60 is considered correct if $allowed\_deviation = 1$.
   (b) **Alpha** - False positive rate.
   (c) **Edges** - As explained earlier.

To produce a wide and completely random variety of dataset samples for each experiment, for each channel, the distribution's parameter values (mean, slope and variance) are chosen arbitrarily. The parameter or parameter set of the distribution in which a change will occur is also randomly selected for every channel in a specific dataset. The time bins at which changes occur are also chosen randomly (unless otherwise specified), but are common to all channels of a given dataset. Lastly, after generating each channel, the seed is changed.

We evaluated the algorithm based on the average results of 1000 random generated repetitions for each experiment. We used precision, recall and the F1-score as performance evaluation measures. Accuracy is not an appropriate score for the evaluation of this task since even when the algorithm returns poor results with high error rates, the True-negative will always be high since this type of problem suffers from data imbalance as there are typically many more within-segment data points than change points.
Detailed results of all experiments can be found in the project repository[2].

[2] `https://github.com/ok123123123/Multi_Dim_CP_Detection`

5.1.1 Experiment 1 - Reference use-case datasets

In the first experiment, we will examine very basic cases. The purpose of the experiment is to evaluate the performance of the algorithm on 1000 different samples of random datasets. Each dataset represents a dataset that corresponds to the reference use-case that the algorithm proposes to solve. No experimental parameter was changed between datasets.

In the first and most basic experiment, the number of channels in each dataset is 8 channels with 200 time bins each. *var_increased_by* set to 1.5 and *signal_to_noise_ratio* set to 2. All channels of each data set have 2 change points in the same time bin. We Set $\alpha = 0.01$ and $edegs = 10$ To reduce False Positive and avoid "noise". The additional parameter *allowed_deviation* is set to 1. We ran the experiment on 1000 random dataset samples as defined above and achieved **Precision 89%, Recall 90% and F1 90% with EMULLR** and Precision 14%, Recall 69% and F1 24% with MULLR. From this experiment, it is already possible to begin to conclude that the algorithm achieves its goals and certainly performs much better than the original MULLR. In light of this determination, we will not compare the other experiments with the original MULLR algorithm.

Next, we will examine the performance of each experiment for different values of one of the experimental parameters while the other remains the same as their values in the first experiment.

5.1.2 Experiment 2 - Number of channels

It was necessary to ensure that the algorithm fulfilled its purpose, which is that the more channels in the dataset, the better the performance. The amount of channels in VIA use-cases is 6-12 channels in each dataset. In the left plot in Figure 7 except of the variable number of channels, all parameters remained unchanged from the first basic experiment. On the middle plot *signal_noise_ratio* changes to 6 and on the right plot *var_increased_by* changes to 3. Based on the results of the experiment shown in Figure 7, as we predicted, the performances of the algorithms are low when the number of channels is small (1-3) and stabilizes around 90% F1 score when the number of channels is greater than 6. Despite the change in intensity of the change, this principle is still reflected in both the middle plot and the right plot.

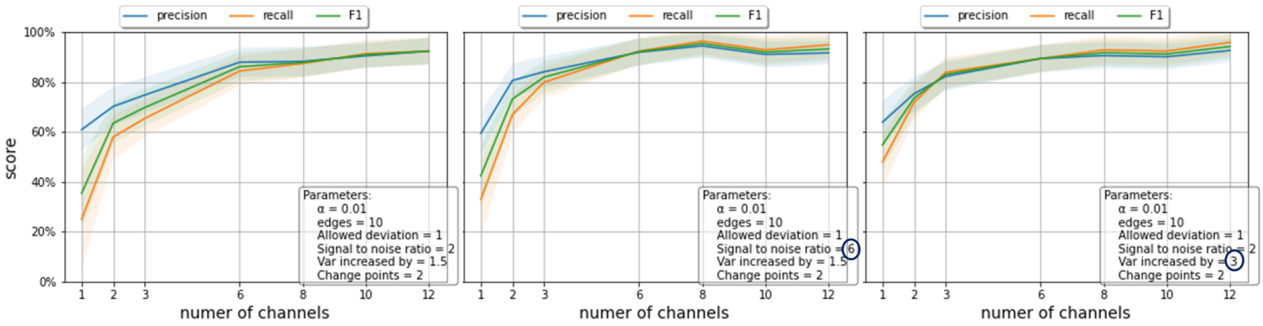

Figure 7: Experiment 2: Number of channels - results.

5.1.3 Experiment 3 - Signal to noise ratio and variance increasing

In the next experiment we tested the algorithm's sensitivity and its ability to identify change points even if the change was very small. *signal_to_noise_ratio* defines the intensity of the change in the mean (when the mean parameter change) and *var_increased_by* defines the intensity of variance (if the variance parameter changed). Thus, two cases were analyzed: The first case examined the parameter *signal_to_noise_ratio* with datasets whose change points must always occur in the mean, and the second case examined the parameter *var_increased_by* on datasets where the change points occur only in the variance.

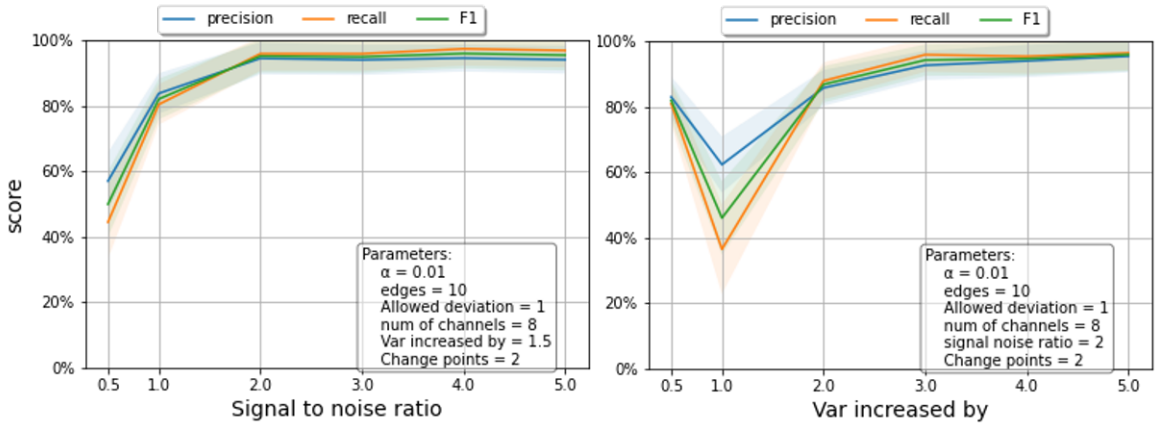

Figure 8: Experiment 3: 'Signal to noise ratio' and 'Variance increasing' - results.

As we expected, in both cases, as can be seen in Figure 8, the greater the intensity of the change, the better the performance of the algorithm. The algorithm is not effective (F1 under 80%) in cases where the *signal_to_noise_ratio* is less than one or in cases where *var_increased_by* is less than 2. In addition, it can be seen that it doesn't matter if the variance is increased or decreased by the same multiplier (say, multiply by 2 or 0.5), the results will be similar. However, it is important to note that the cases examined are on datasets in which the changes that occurred are only in the parameter being tested, and in reference use-cases the changes will be more varied.

5.1.4 Experiment 4 - Alpha

In order to find the right balance between all error types, we tested different α values. We found that the most optimal $\alpha = 0.01$. The results of the experiment are detailed in Figure 9 - The balance point between false positive (precision) and true negative(recall) found to be the point where the best F1 score is achieved marked with a black circle.

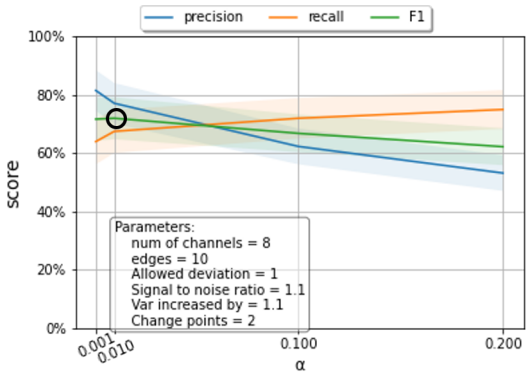

Figure 9: Experiment 4: α - results.

5.1.5 Experiment 5 - Time bins

Next, we examined the length of the channels in the datasets, the time bins of the channels, and how it affects the performance of the algorithm. We were required to verify what is the "minimum" length of channels in datasets which the algorithm is still effective for and also if there is a maximum length for channels.

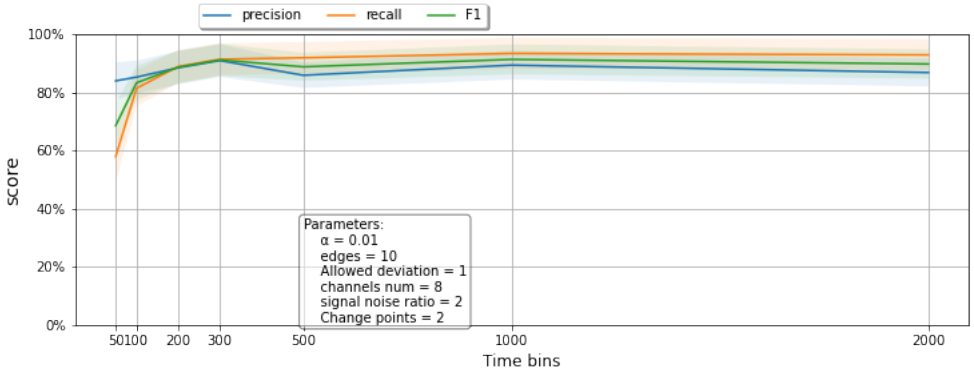

Figure 10: Experiment 5: Time bins - results.

It seems that for datasets in which there are 2 change points, the algorithm will achieve good results (F1 greater than 80%) for datasets with a minimum channel length of 100 time bins. In cases where the length of the channels is too small (below 100 time bins), the basic assumptions of the model are violated - enough data points to estimate the parameters of the distribution and accordingly, the algorithm will achieve poor results.

5.1.6 Experiment 6 - Allowed deviation

Here we examined the algorithm's accuracy in finding the exact time bin in which the change will happen. We also investigated the right margin of error to allow for it. Of course, in the experiment, for each dataset, the change point occurs at exactly the same time bin in all channels. However, in real-world datasets, the change point can occur in each channel at a slightly different time bin.

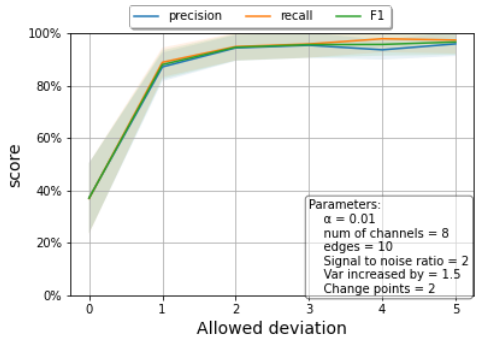

Figure 11: Experiment 6: Allowed deviation - results.

In Figure 11, the experimental results indicate that a deviation of a one-time bin is sufficient for the algorithm to achieve acceptable results. In real-world datasets, it is probably more appropriate to allow more than a one-time bin deviation, since not all change points will appear in the same time bin in all channels.

5.1.7 Experiment 7 - Most (or 50%) of the channels have change point

The previous experiments generated datasets in which the change point location was placed within the same time bin across all channels. Here we release this constraint, as this setup is more likely to happen in real world datasets, and evaluate algorithm performance. The sample data sets in this experiment contain change points in only 50% of the channels (in a dataset with an odd number of channels a small majority of the channels will contain the same change point time bin). Also, we tested whether the number of channels in the dataset affected this issue.

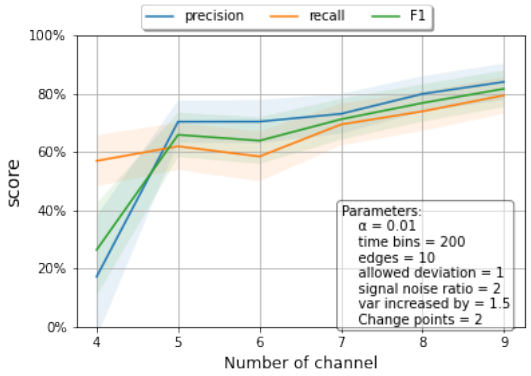

Figure 12: Experiment 7: Partial channels contain the same change point location - results.

The experimental results in Figure 12 show that at least 8 channels or more are required so that if the change point occurs in only half or slightly more of the channels the algorithm will achieve high results (more than 80% F1). In cases where the dataset contains less than 5 channels, as long as the change does not occur in all channels, the algorithm's performance drops significantly. This supports the claim that EMULLR is more suitable for multi-channel datasets.

5.1.8 Experiment 8 - No change points

Our algorithm was also tested on a large number of datasets in which the channels do not contain any change points to ensure that these results are not a coincidence. The multi-channel LLR was very low in all cases and did not exceed the threshold, indicating that the results were absolutely accurate.

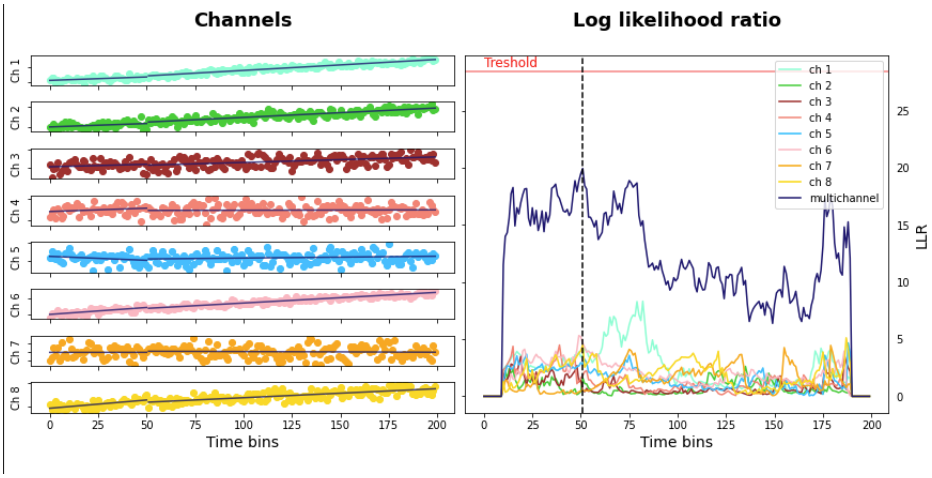

Figure 13: An example of how the EMULLR algorithm run on a data set whose channels do not contain any changes. This figure illustrates that the LLR of the multi-channels does not exceed the threshold.

5.2 Experiments using VIA's real data

EMULLR was tested on six different real world datasets provided by VIA, each dataset contains 6-11 channels. These datasets represent different cities and measure KPIs of VIA's service efficiency and quality. Datasets 1 and 2 are labeled. The ground truth labels correspond to the dates when a major change happened, such as an update service launch or version update. In those cases, it was unclear if and how those changes would affect performance and service quality. As for the other datasets, they were not labeled with ground truth, so we could only compare EMULLR results to MULRR and to the benchmark, the predictions of the single-channel PELT algorithm for each channel.

Data set	True label	EMULLR	MULLR	PELT										
				ch1	ch2	ch3	ch4	ch5	ch6	ch7	ch8	ch9	ch10	ch11
1	43	43	14, 43	-	-	40	44	44	38	-				
2	41	40, 56	15, 29, 55, 71	-	-	57	58	57	40	-				
3	unknow	20, 41, 68	16, 27, 41, 51, 67	43	51	43	47	43	-	47	28, 50	47	20, 42	41
4	unknow	16, 37, 59	16, 38, 50, 60, 75	35,38	37	38	37	38	37	37, 61	19, 73	-	17	51
5	unknow	14, 26, 46	15, 26, 37, 47	40	-	47	47	29	27	26	27, 48	50	40	
6	unknow	34, 45, 56	18, 33, 45, 57, 67	45	47, 68	47	47	45	65	47	48	55	53	46

Figure 14: A summary of the results obtained from the experiments on VIA company real-world datasets as compared to the ground truth labels (in channels 1-2) and to the PELT and MULLR results. $\alpha = 0.01$, $edge = 10$.

As shown in Figure 14, the prediction of the EMULLR algorithm is consistent with the ground truth labels of datasets 1 and 2 (marked in green). Furthermore, in dataset 2, we received an additional prediction compatible with the PELT algorithm prediction in channels 3-5 and MULLR prediction (marked in yellow). In dataset 1, there is one error in the MULLR prediction, while in dataset 2, there are three errors in the MULLR prediction (marked in red). As can be seen from the

results obtained from these two labeled datasets, EMULLR appears to obtain the highest results among the three algorithms.

In datasets 3-6, in the absolute majority of cases, the prediction provided by EMULLR algorithm was also received in one or more of the channels by the PELT algorithm.
Analyzing the raw data plot of the EMULLR algorithm reveals that the algorithm detects change points that the PELT algorithm misses. This is due to EMULLR use of multi-channel observation. By contrast, the PELT algorithm is constrained and may not identify change points due to its single-channel observation. An example of this can be seen in figure 15, where EMULLR predicts a change point that corresponds to the raw data analysis. According to the analysis, most channels show a change in trend at the predicted change point. Due to the multi-channel approach, even though the changes are small, the cumulative change is large and crosses the threshold. By contrast, PELT was not able to detect this change in any of the channels.

When comparing MULLR with EMULLR, most of the predictions are similar, but MULLR has many more predictions, which are most likely false positives.

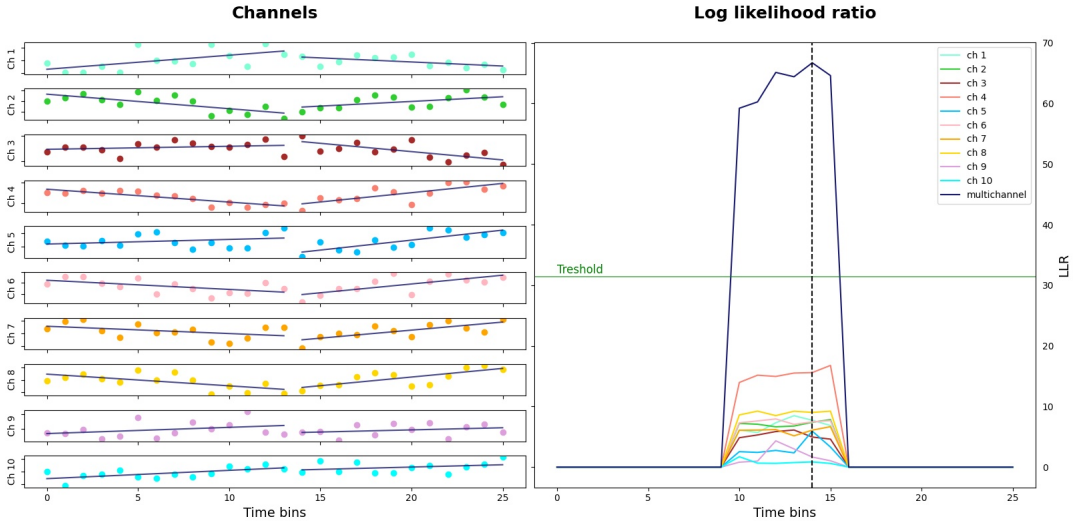

Figure 15: Via's real-world data, dataset number 5, Predicting a change point at time bin 14: Results from a recursive analysis of the left sub-data set.

6 Limitations

Despite the strengths of our algorithm for detecting multiple change points in high-dimensional time series data, several limitations need to be acknowledged. Time series analysis typically requires a large number of data points to achieve consistency and reliability. An extensive data set ensures a representative sample size and that analysis can cut through noisy data. It also ensures that any trends or patterns discovered are not outliers and can account for seasonal variance. When dividing data into segments, we might have difficulties aggregating small numbers of data points. Furthermore, this problem was solved by limiting the minimum time bins (distance) between two candidate change points and preserving the minimum number of data points for each segment of the data. In addition, the algorithm was restricted from predicting CP too close to the beginning or end of the

channel's data. This restriction can be controlled by the 'edge' parameter.
This work was focused on developing a stable, consistent and accurate algorithm to offline predict multiple CP in multidimensional data. We did not devote any time or effort to researching methods of pre-processing the data in order to improve our results. Such methods might include deep learning tools, latent space projection (PCA, encoder-decoder), or data cleaning (outlier detection). Our algorithm uses a binary search algorithm as a search method, this is not optimal ($O(N * log(N))$). Although our algorithm is stable and robust due to its utilization of multi-sources of information, In the event that a change point is located in very few channels of the dataset and it is not significant enough, it might reject in our multi-dimension data analysis. In those cases, a uni-dimension approach, such as PELT will be preferable. Further limitations include the inability to reject CP and track non-linear trends. This is due to the fact that we use linear regression to model the data. We observed in experiment 2 that the performance of our algorithm increases as the dimensions of the data increase. When we feed single-channel data to our algorithm, we are likely to achieve poor results with an average F1 score of 32, here as well uni-dimension approach will be better. Importantly, we recommend that the data fed into the algorithm meet the assumptions we mentioned above. The suggested algorithm, like different other solutions, suffers from generic changes like the exposure to errors caused by outliers. Additionally, even though we expanded the number of parameters for which our algorithm can predict change, it still has difficulty with accurate change predictions for certain parameters and data characteristics. One such example is the change in seasonality. Seasonality is a characteristic of a time series in which the data experiences regular and predictable changes that recur every constant time interval. One example of time series data with significant seasonality is the temperature over the course of the day or during the calendar year.

7 Conclusions

In conclusion, we have presented a novel algorithm for detecting multiple change points in high-dimensional time series data, which offers several advantages over existing methods. Our algorithm demonstrated superior performance through various experiments on synthetic and real-world datasets provided by VIA Transportation. The algorithm's higher sensitivity allows for the detection of higher-resolution change points and improves the level of confidence in the algorithm's predictions even when the signal-to-noise ratio is low. Additionally, the algorithm can detect changes caused by different statistical reasons and offers the ability to detect multiple change points in different combinations of data parameters.
Our algorithm is robust and does not require parameters to train or fine-tune, and only has minimal hyperparameters. It is computationally and time-efficient, requiring only basic CPU power and working in $N * log(N)$ time complexity. The algorithm is user-friendly, taking datasets as input and predicted CP as output, and offers the user control over the level of confidence they expect (alpha). Furthermore, the algorithm is simple to understand and analyze, based on classic mathematics and statistical tools, with no black boxes. It also provides a unique analysis tool and plot methodology to explore the algorithm results in-depth and perform a qualitative analysis of the results, understanding which channel "contributed" more and which less. Based on the output as shown in Figure 5 the business analyst can analyze the trends that existed before the change appeared and after it - in each channel and all of them together.
To summarize, our results suggest that the proposed algorithm is a promising tool for the analysis of high-dimensional time series data and has potential applications in various fields, such as finance, biology, and engineering. Future directions for further research could include CP detection in addition to time series data parameters and characteristics like changes in the seasonality of the

data, to redundant the algorithm limitation or assumptions taken in this work, further refinement and optimization of the algorithm for specific applications and testing its performance on additional real-world datasets.

References

[1] Cesare Alippi, Giacomo Boracchi, Diego Carrera, and Manuel Roveri. Change detection in multivariate datastreams: Likelihood and detectability loss. *arXiv preprint arXiv:1510.04850*, 2015.

[2] John AD Aston and Claudia Kirch. Evaluating stationarity via change-point alternatives with applications to fmri data. *The Annals of Applied Statistics*, 6(4):1906–1948, 2012.

[3] Alexander Aue, Robertas Gabrys, Lajos Horváth, and Piotr Kokoszka. Estimation of a change-point in the mean function of functional data. *Journal of Multivariate Analysis*, 100(10):2254–2269, 2009.

[4] Alexander Aue, Siegfried Hörmann, Lajos Horváth, and Matthew Reimherr. Break detection in the covariance structure of multivariate time series models. 2009.

[5] Jushan Bai. Common breaks in means and variances for panel data. *Journal of Econometrics*, 157(1):78–92, 2010.

[6] Matteo Barigozzi, Haeran Cho, and Piotr Fryzlewicz. Simultaneous multiple change-point and factor analysis for high-dimensional time series. *Journal of Econometrics*, 206(1):187–225, 2018.

[7] Varun Chandola and Ranga Raju Vatsavai. Scalable time series change detection for biomass monitoring using gaussian process. In *CIDU*, pages 69–82, 2010.

[8] Hao Chen and Nancy Zhang. Graph-based change-point detection. 2015.

[9] Haeran Cho and Piotr Fryzlewicz. Multiple-change-point detection for high dimensional time series via sparsified binary segmentation. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 77(2):475–507, 2015.

[10] Zhicheng Cui, Wenlin Chen, and Yixin Chen. Multi-scale convolutional neural networks for time series classification. *arXiv preprint arXiv:1603.06995*, 2016.

[11] Ailin Deng and Bryan Hooi. Graph neural network-based anomaly detection in multivariate time series. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 4027–4035, 2021.

[12] Frédéric Desobry, Manuel Davy, and Christian Doncarli. An online kernel change detection algorithm. *IEEE Transactions on Signal Processing*, 53(8):2961–2974, 2005.

[13] Kyle D Feuz, Diane J Cook, Cody Rosasco, Kayela Robertson, and Maureen Schmitter-Edgecombe. Automated detection of activity transitions for prompting. *IEEE transactions on human-machine systems*, 45(5):575–585, 2014.

[14] Edit Gombay and Lajos Horváth. On the rate of approximations for maximum likelihood tests in change-point models. *Journal of Multivariate Analysis*, 56(1):120–152, 1996.

[15] Zaid Harchaoui, Félicien Vallet, Alexandre Lung-Yut-Fong, and Olivier Cappé. A regularized kernel-based approach to unsupervised audio segmentation. In *2009 IEEE international conference on acoustics, speech and signal processing*, pages 1665–1668. IEEE, 2009.

[16] Mark Holmes, Ivan Kojadinovic, and Jean-François Quessy. Nonparametric tests for change-point detection à la gombay and horváth. *Journal of Multivariate Analysis*, 115:16–32, 2013.

[17] Lajos Horváth, Piotr Kokoszka, and Josef Steinebach. Testing for changes in multivariate dependent observations with an application to temperature changes. *Journal of Multivariate Analysis*, 68(1):96–119, 1999.

[18] Daniel R Jeske, Veronica Montes De Oca, Wolfgang Bischoff, and Mazda Marvasti. Cusum techniques for timeslot sequences with applications to network surveillance. *Computational statistics & data analysis*, 53(12):4332–4344, 2009.

[19] Shuhao Jiao, Tong Shen, Zhaoxia Yu, and Hernando Ombao. Change-point detection using spectral pca for multivariate time series. *arXiv preprint arXiv:2101.04334*, 2021.

[20] Fazle Karim, Somshubra Majumdar, Houshang Darabi, and Shun Chen. Lstm fully convolutional networks for time series classification. *IEEE access*, 6:1662–1669, 2017.

[21] Yoshinobu Kawahara and Masashi Sugiyama. Change-point detection in time-series data by direct density-ratio estimation. In *Proceedings of the 2009 SIAM international conference on data mining*, pages 389–400. SIAM, 2009.

[22] Eamonn Keogh, Selina Chu, David Hart, and Michael Pazzani. An online algorithm for segmenting time series. In *Proceedings 2001 IEEE international conference on data mining*, pages 289–296. IEEE, 2001.

[23] Rebecca Killick, Paul Fearnhead, and Idris A Eckley. Optimal detection of changepoints with a linear computational cost. *Journal of the American Statistical Association*, 107(500):1590–1598, 2012.

[24] Ludmila I Kuncheva. Change detection in streaming multivariate data using likelihood detectors. *IEEE transactions on knowledge and data engineering*, 25(5):1175–1180, 2011.

[25] Ludmila I Kuncheva and William J Faithfull. Pca feature extraction for change detection in multidimensional unlabeled data. *IEEE transactions on neural networks and learning systems*, 25(1):69–80, 2013.

[26] Sokbae Lee, Yuan Liao, Myung Hwan Seo, and Youngki Shin. Oracle estimation of a change point in high-dimensional quantile regression. *Journal of the American Statistical Association*, 113(523):1184–1194, 2018.

[27] Sokbae Lee, Myung Hwan Seo, and Youngki Shin. The lasso for high dimensional regression with a possible change point. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(1):193–210, 2016.

[28] Eyal Nitzan, Topi Halme, and Visa Koivunen. Bayesian methods for multiple change-point detection with reduced communication. *IEEE Transactions on Signal Processing*, 68:4871–4886, 2020.

[29] Stefano Peluso, Siddhartha Chib, and Antonietta Mira. Semiparametric multivariate and multiple change-point modeling. 2019.

[30] Abdulhakim A Qahtan, Basma Alharbi, Suojin Wang, and Xiangliang Zhang. A pca-based change detection framework for multidimensional data streams: Change detection in multidimensional data streams. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 935–944, 2015.

[31] Yunus Saatçi, Ryan D Turner, and Carl E Rasmussen. Gaussian process change point models. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 927–934, 2010.

[32] Dang-Hoan Tran. Automated change detection and reactive clustering in multivariate streaming data. In *2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF)*, pages 1–6. IEEE, 2019.

[33] L. Yu. Vostrikova. Detection of a "disorder" in a wiener process. *Theory of Probability & Its Applications*, 26(2):356–362, 1982.

[34] Runmin Wang, Changbo Zhu, Stanislav Volgushev, and Xiaofeng Shao. Inference for change points in high-dimensional data via selfnormalization. *The Annals of Statistics*, 50(2):781–806, 2022.

[35] Yao Wang, Chunguo Wu, Zhaohua Ji, Binghong Wang, and Yanchun Liang. Non-parametric change-point method for differential gene expression detection. *PloS one*, 6(5):e20060, 2011.

[36] Zhiguang Wang, Weizhong Yan, and Tim Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International joint conference on neural networks (IJCNN)*, pages 1578–1585. IEEE, 2017.

[37] Hugh Wilson and Quan Wang. Joint detection of change points in multichannel single-molecule measurements. *The Journal of Physical Chemistry B*, 125(49):13425–13435, 2021.

[38] Barnet Woolf. The log likelihood ratio test (the g-test). *Annals of human genetics*, 21(4):397–409, 1957.

[39] Chun Yip Yau and Zifeng Zhao. Inference for multiple change points in time series via likelihood ratio scan statistics. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(4):895–916, 2016.

[40] Chuxu Zhang, Dongjin Song, Yuncong Chen, Xinyang Feng, Cristian Lumezanu, Wei Cheng, Jingchao Ni, Bo Zong, Haifeng Chen, and Nitesh V Chawla. A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 1409–1416, 2019.

[41] Yuxuan Zhang and Hao Chen. Graph-based multiple change-point detection. *arXiv preprint arXiv:2110.01170*, 2021.