CHANGE POINT DETECTION IN END-TO-END MEASUREMENTS TIME SERIES

Diego Ximenes Mendes

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Sistemas e Computação.

Orientador: Edmundo Albuquerque de Souza e Silva

Rio de Janeiro
Janeiro de 2017

CHANGE POINT DETECTION IN END-TO-END MEASUREMENTS TIME SERIES

Diego Ximenes Mendes

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Examinada por:

RIO DE JANEIRO, RJ – BRASIL
JANEIRO DE 2017

Ximenes Mendes, Diego

Change Point Detection in End-to-End Measurements Time Series/Diego Ximenes Mendes. – Rio de Janeiro: UFRJ/COPPE, 2017.

VIII, 19 p.: il.; 29, 7cm.

Orientador: Edmundo Albuquerque de Souza e Silva

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2017.

Bibliography: p. 17 – 19.

1. Change Point Detection. 2. Time Series. 3. Machine Learning. 4. Network Measurements. I. Albuquerque de Souza e Silva, Edmundo. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

CHANGE POINT DETECTION IN END-TO-END MEASUREMENTS TIME SERIES

Diego Ximenes Mendes

Janeiro/2017

Orientador: Edmundo Albuquerque de Souza e Silva

Programa: Engenharia de Sistemas e Computação

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

CHANGE POINT DETECTION IN END-TO-END MEASUREMENTS TIME SERIES

Diego Ximenes Mendes

January/2017

Advisor: Edmundo Albuquerque de Souza e Silva

Department: Systems Engineering and Computer Science

Contents

List of Figures **vii**

List of Tables **viii**

1 Introduction **1**

1.1 Contributions 1

1.2 Dissertation Outline 1

2 Change Point Detection **2**

2.1 Problem Definition 2

2.2 Notation 3

2.3 Sliding Windows 4

2.4 Optimization Model 5

2.5 HMM (Hidden Markov Model) 6

2.6 Bayesian Inference 7

3 Dataset **8**

3.1 End-to-End Packet Loss Fraction Time Series Dataset 8

3.1.1 Methodology 8

3.1.2 Descriptive Analysis 9

3.2 Change Points Dataset 12

3.2.1 Methodology 12

3.2.2 Descriptive Analysis 14

4 Conclusions **16**

Bibliography **17**

List of Figures

2.1 Toy example of a sliding windows method. 5

3.1 Packet Loss Fraction 9
3.2 Client 1 10
3.3 Client 2 10
3.4 Client 3 11
3.5 Client 4 12
3.6 Survey system snapshot 14
3.7 Number of change points per time series distribution 14
3.8 Segment length distribution. Bins of size 10. 15
3.9 Consecutive segments comparison. Bins of size 0.02. 15

List of Tables

Chapter 1

Introduction

1.1 Contributions

1.2 Dissertation Outline

Chapter 2

Change Point Detection

A change point detection algorithm is concerned to identify points in time where the statistical properties of a time series have changed. This problem have a broad application in different knowledge fields, and in general, an algorithm's performance is closely related with the input characteristics. Further, if the latent information of the procedures that generated a time series is missing, the target statistical properties can be considered subjective, bringing difficulties not only in the detection phase but also in the problem formalization.

In this context, this chapter specifies the problem and briefly discusses several change point detection algorithms. The literature of this area is extensive, and it is common to find methods that presents a poor performance due to a variety of reasons, such as they are too specific, or because the mechanisms were only analyzed through theoretical aspects. Therefore, it was chosen a set of techniques that can provide a good practical and theoretical perspectives, and also flexibility to insert adaptations that can better handle some input peculiarities. Furthermore, through this chapter is exposed several obstacles when dealing with real data, and some adopted solutions which are not described in literature.

2.1 Problem Definition

The problem can be categorized in offline or online. In the offline version, to decide if a specific point at time t is a change point, the solver has available the whole time series, including past and future information on t. In the other hand, in the online version, the information is available up to time t. The choice between these options is defined by the application domain. In some cases data are processed in real time, and change points should be detected as soon as possible. But in some applications changes are identified by historical purposes, and offline algorithms can be used.

It is intuitive that the offline case is more robust, since there are more information to make a classification. In practice, to increase the statistical confidence of a

decision, the online definition is relaxed, and to decide if a point at time t is a change point it is possible to use data up to a small window in the future of t, which in real time processing means that the application should wait until more data are available. This trick plays a trade-off between minimizing the time to detect a change and correctly classify a point. Therefore, in some cases, the online version can be transformed in offline by only modifying the input availability.

In this work it is considered the following input and change points attributes, which were defined considering the final application scenario:

- Univariate time series. However, it is possible to extend several methods presented here to deal with multivariate data.
- Unevenly spaced time series, that is, data is not regularly sampled in time.
- Time series with different lengths.
- Unknown number of change points.
- Different number of points between change points.
- Focus on changes in the underlying mean and distribution, disregarding other kinds of changes, such as in periodicity.
- Outliers are not considered statistical changes.
- There is no latent information of the time series.
- It is considered the online and offline options.

2.2 Notation

An univariate time series composed of n points is defined by two vectors, $\mathbf{x} = (x_1, ..., x_n)$ and $\mathbf{y} = (y_1, ..., y_n)$. The value y_i indicates the i-th sampled value, and x_i indicates the associated sample time. It is assumed that the points are sorted by time, that is, $x_{i-1} < x_i$ for $i = 2, ..., n$. Since unevenly spaced time series is considered, $x_i - x_{i-1}$ can be different for different i values. For $s \leq t$ the following convention is adopted: $\mathbf{y}_{s:t} = (y_s, ..., y_t)$.

The presence of k change points implies that data is split into $k + 1$ segments, also called windows. Let τ_i indicates the i-th change point for $i = 1, ..., k$. Also let $\tau_0 = 0$, $\tau_{k+1} = n$ and $\boldsymbol{\tau} = (\tau_0, ..., \tau_{k+1})$. Then, the i-th segment is defined by $\mathbf{y}_{\tau_{i-1}+1:\tau_i}$, assuming that $\tau_{i-1} < \tau_i$ for $i = 1, ..., k + 1$.

Through the previous definitions, change point detection algorithms mainly aim to find both k and $\boldsymbol{\tau}$.

2.3 Sliding Windows

Sliding windows techniques use two sliding windows over the time series, and reduce the problem of detecting change points to the problem of testing whether data from the segments were generated by different distributions. One approach is to consider a distance metric between two empirical distributions as the base to infer the change points. Letting $d(\mathbf{a}, \mathbf{b})$ be the distance between two empirical distributions defined by the windows $\mathbf{a}$ and $\mathbf{b}$, and considering windows of length m, the Algorithm 1 presents a simple sliding windows method.

Algorithm 1 Sliding Windows

```
1: i ← 1
2: while i + 2m − 1 ≤ n do
3:     if d(y_{i:i+m−1}, y_{i+m:i+2m−1}) > α then
4:         Report i + m − 1 as a change point
5:         i ← i + m
6:     else
7:         i ← i + 1
8:     end if
9: end while
```

In this mechanism, when the distance between the distributions is above some threshold α a change point is reported. This is a common approach for an online application, however, it is possible to increase the classification accuracy in offline cases. As an example, the top plot of figure 2.1 presents a simulated time series. The segment $\mathbf{y}_{1:1000}$ was generated sampling a $N(1, 0.2)$ distribution, and $\mathbf{y}_{1001:2000}$ was sampled through $N(5, 0.2)$. The distribution of a window was constructed binning the data with bins of size 0.02. The bottom plot of the same figure presents the associated Hellinger distance [1] between two sliding windows, where the point (i, H_i) represents the distance between the windows $\mathbf{y}_{i-100:i-1}$ and $\mathbf{y}_{i:i+99}$.

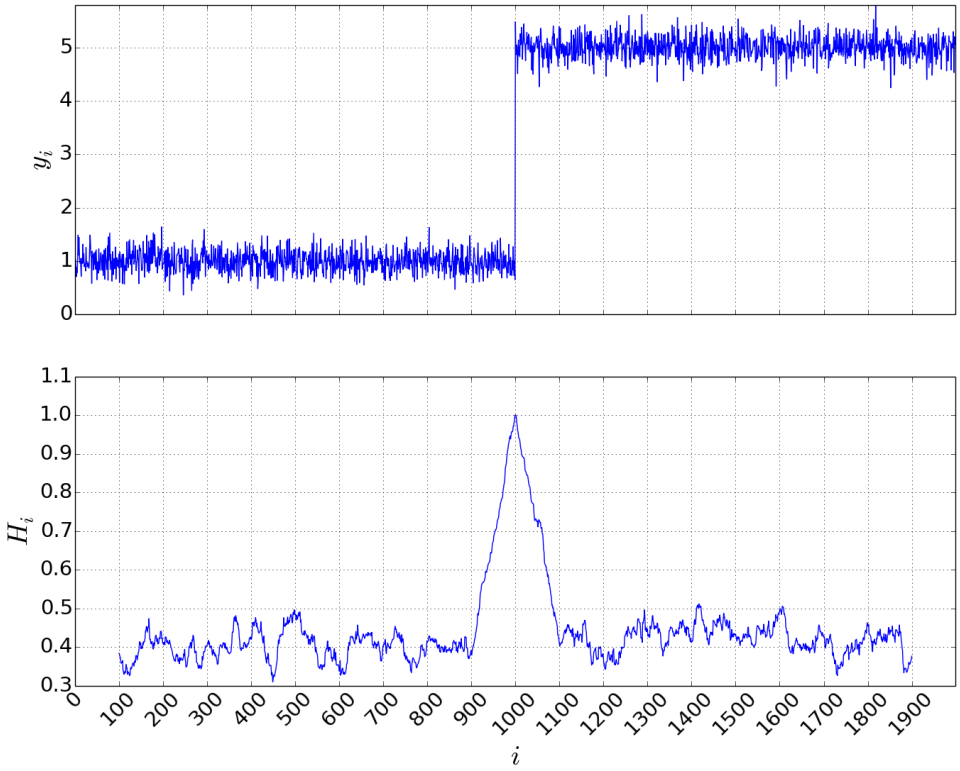


Figure 2.1: Toy example of a sliding windows method.

It can be observed that there is a peak on the distance in the exact location where the distribution changed. However, using only the threshold method it is possible to prematurely infer the position of the change point. Therefore, an alternative is to also use a peak detection algorithm. Besides, the distance function choice has a direct impact on the classification accuracy.

As stated in [2], a performance improvement can be achieved concurrently executing the same sliding windows algorithm, however, with different windows lengths, which facilitates the detection of segments with distinct number of points.

2.4 Optimization Model

Given a fixed value of k, one approach is to define a cost function that measures the homogeneity of a window, and therefore, choose the change points that globally optimize this homogeneity. Let the cost of the i-th segment be defined as $C(\mathbf{y}_{\tau_{i-1}+1:\tau_i})$, then the cost of a segmentation is the sum of all segments costs.

A common choice for the function C is the MSE (Mean Squared Error), which can capture changes in the mean. Another usual approach is to consider distribution changes through negative maximum log-likelihood functions, considering that data within a window is iid.

Therefore, given a fixed k, the optimal segmentation is obtained through the following optimization problem, which is called the constrained case [3]:

$$\min_{\boldsymbol{\tau}_{1:k}} \sum_{i=1}^{k+1} C(\mathbf{y}_{\tau_{i-1}+1:\tau_i}) \tag{2.1}$$

This problem can be solved using dynamic programming with $O(kn^2 f(n))$ time complexity, where $f(n)$ is related with C evaluation. Several segment cost functions can be evaluated in $O(1)$ after a $O(n)$ preprocessing phase, implying in an overall $O(kn^2)$ complexity. It is possible to proof that MSE, negative maximum log-likelihood functions of normal, exponential, poisson and binomial distributions have this characteristic. Also, the formulation can consider a minimum value of a window length.

Modeling segments with distributions can lead to practical difficulties. One of them is the fact that segments can form degenerate distributions, that is, the data of a window can have zero variance, which is always the case of unitary length windows. In these scenarios the negative maximum log-likelihood can be undefined. Two approaches can be used to overcome this situation. The first one tries to avoid degenerate segments adding a white noise with small variance to the data stream. The second one considers that the cost of any degenerate distribution is equal to a constant.

When the number of change points is unknown, an usual way is to introduce a non decreasing penalty function $g(k)$. Then, the new optimization problem, called penalized case [3], is:

$$\min_{k,\boldsymbol{\tau}_{1:k}} \sum_{i=1}^{k+1} C(\mathbf{y}_{\tau_{i-1}+1:\tau_i}) + g(k) \tag{2.2}$$

This problem can be solved in $O(Kn^2 f(n))$. However, if the penalty function is linear in k, the problem can be formulated more efficiently and solved in $O(n^2 f(n))$.

Also, there are several pruning algorithms to speedup the computation [3–5], in general trying to reduce the $\boldsymbol{\tau}$ search space but maintaining optimality.

2.5 HMM (Hidden Markov Model)

The idea that each segment is associated with a specific latent configuration has a direct interpretation to a HMM model [6–8]. In this context, each window is related to a hidden state of a HMM, and the observation distribution of this state represents the distribution of that segment. Therefore, the mechanism models the time series using a HMM, and through the hidden state path, assesses the times

when a transition between different hidden states occur.

There are several approaches in the detection and training phases. For example, given a trained HMM, the most probable hidden state path can be checked through the viterbi algorithm. Also, it is possible to evaluate the probability of a transition between different hidden states at time t, and then apply a threshold and peak detection methods, as well as in sliding windows techniques. For the training step, it is possible to use several time series to train a single HMM, and then use this model to detect change points in all time series. Another way is to, for each data stream, train a single model using only the target time series.

It is important to note that the structure of the hidden state graph has a large impact on the performance. Using a fully connected graph, the number of states defines the maximum number of distribution configurations. Employing a left to right structure, the number of hidden states will induce the maximum number of segments.

In [8] is stated that when using a fully connected structure, the time interval that a time series stays in the same hidden state is low, which can not reflect real data. To overcome this problem, [8] suggests to increase the time that a time series stands in the same hidden state using a dirichlet prior regularization.

2.6 Bayesian Inference

There are several Bayesian methods which aims to assess the probability that a point is a change point. Following an offline fashion, the work of [9] recursively calculates, for each i, the probability of $\mathbf{y}_{i:n}$ given a change point at i. With these probabilities is possible to simulate the time of the first change point, and then, compute the conditional distribution of the time of the second change given the first, and so on. To achieve this, the mechanism assumes that observations are independents, and that each segment is modeled by conjugate priors. Also, the procedure considers priors to model the number of changes and the time between two consecutive change points. The overall complexity of this method is $O(n^2)$, considering that the likelihood of a segment can be evaluated in $O(1)$.

In [10] it is also considered that parameters of different segments are independents, and that data within a window is iid. However, through an online mode, the procedure is concerned with the estimation of the distribution of the length of the current time since the last change point, called run length, given the data so far observed. To achieve this, the method assumes the probability of current run length given the last run length as a prior. Assuming exponential-family likelihoods to model a segment, the time complexity to process a point is linear in the number of points already observed.

Chapter 3

Dataset

This chapter describes the used datasets, presenting their construction methodology and a brief descriptive analysis.

3.1 End-to-End Packet Loss Fraction Time Series Dataset

3.1.1 Methodology

The exposed time series of this work represent network end-to-end measures of a cable-television infrastructure, which runs DOCSIS with asymmetric download and upload bandwidths. Home routers connected to the cable modem communicate with one or more servers strategically located by the ISP. Measurements results from each home router are consolidated every half hour and, by the end of every day, are transferred to a database. The software responsible for these procedures was developed by TGR in partnership with UFRJ, and is spread over a customers subset of a major Brazilian ISP.

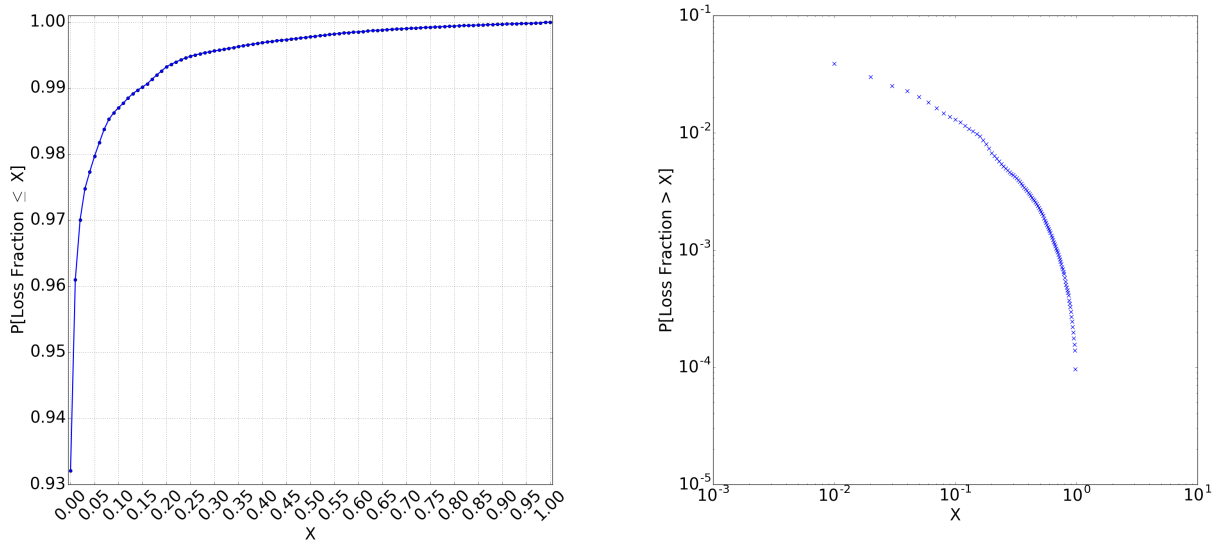
The focus of this project is to analyze the loss fraction, however, [11] presents a further investigation on other metrics, such as link throughput, round trip latency and loss bursts. To measure the round trip packet loss fraction between the home router and the associated server, the home router sends a train of short UDP packets, and then the server bounces back them. The data here presented considers a train of 100 UDP packets of 32 bytes, separated by 1 millisecond.

The resulted time series are unevenly spaced due to a range of reasons. First, measurements are initiated only if the residential link is not under use by the ISP customer. Also, the client may have no Internet connection to start a measurement, or even be without electrical energy.

3.1.2 Descriptive Analysis

This section analysis considers the period from 01/may/2016 to 20/may/2016. It was selected every client in which all measures of this period occurred against the same server. It was also imposed that the home router and the server must belong to the same Brazilian state. These filters resulted in 1870 clients time series and 1537272 measures.

In figure 3.1 is presented the packet loss fraction CDF and CCDF of all clients measures. It is possible to note that 93.2% of the measures have zero losses.

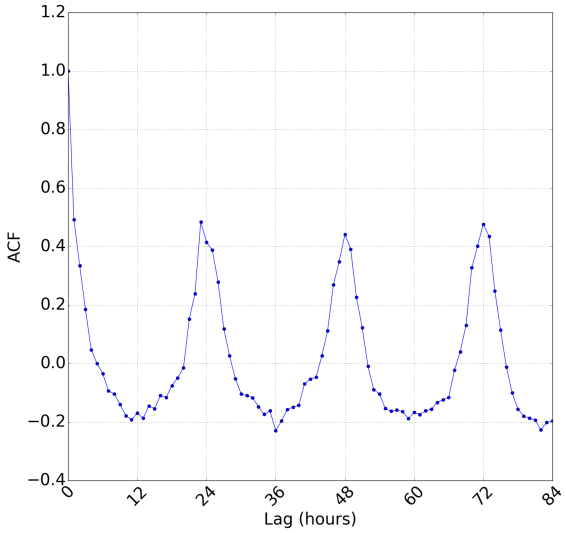


(a) CDF

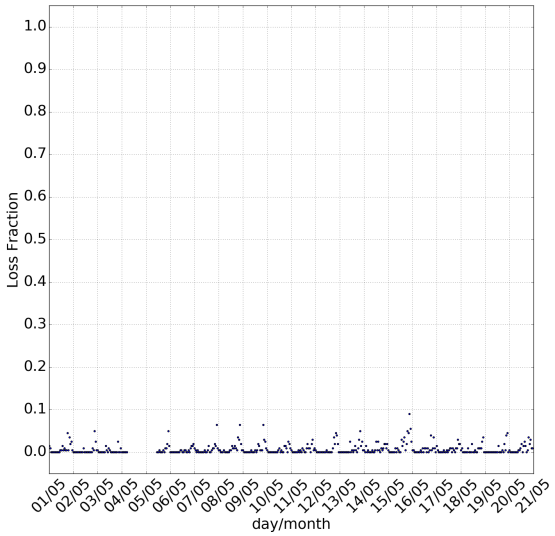
(b) CCDF in log-log scale

Figure 3.1: Packet Loss Fraction

Figures 3.2, 3.3, 3.4 are examples of common autocorrelation patterns in this dataset. Since the data streams are unevenly spaced, it was considered only the date and hour components to compute the autocorrelation function. Therefore, if more than one measure occurred at the same hour and date, it was taken the average of these samples.

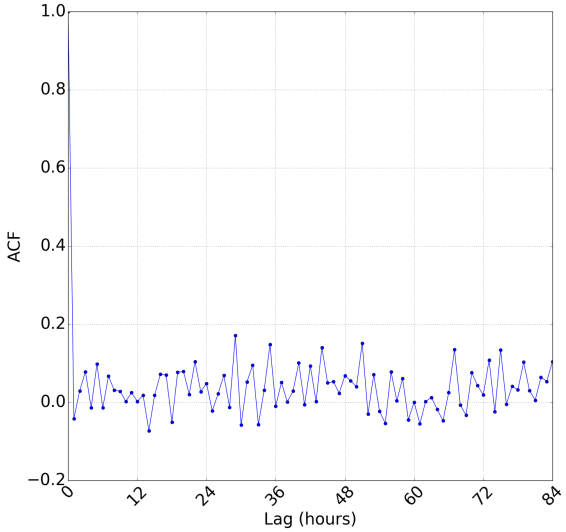


(a) Autocorrelation

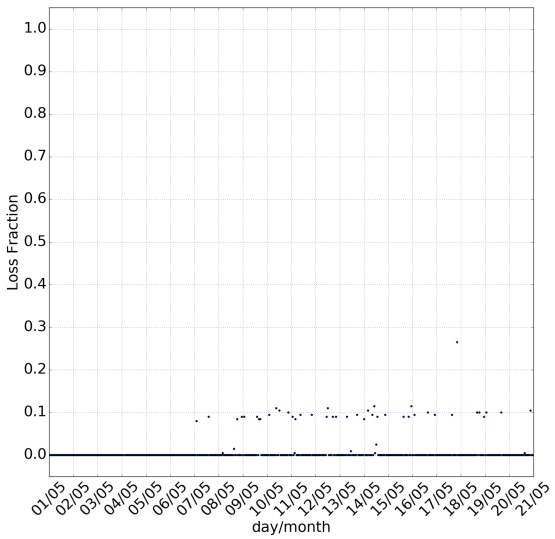


(b) Time series

Figure 3.2: Client 1

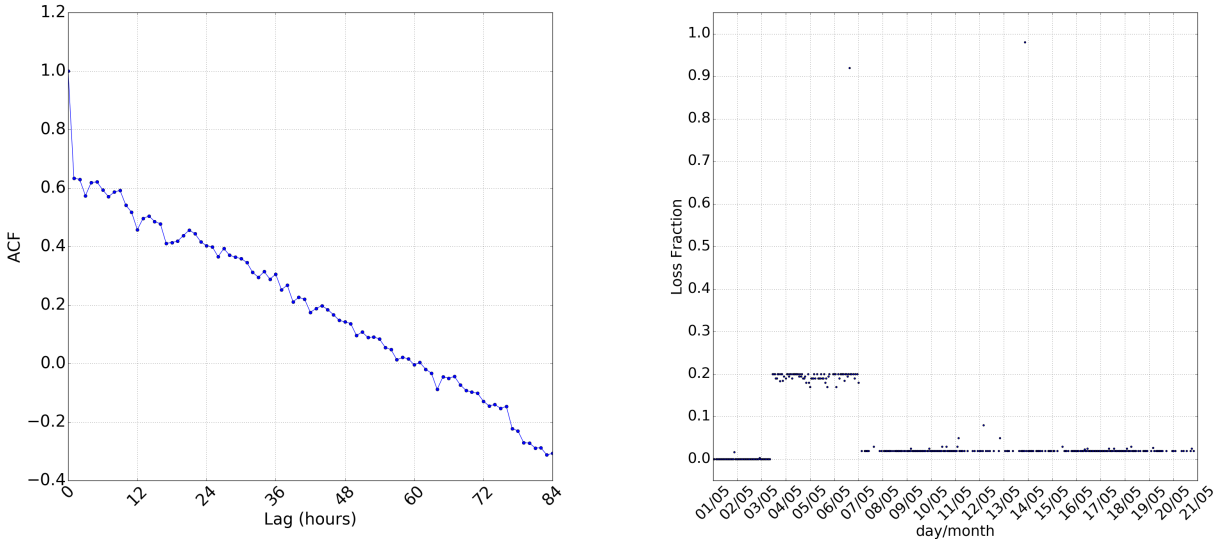


(a) Autocorrelation



(b) Time series

Figure 3.3: Client 2



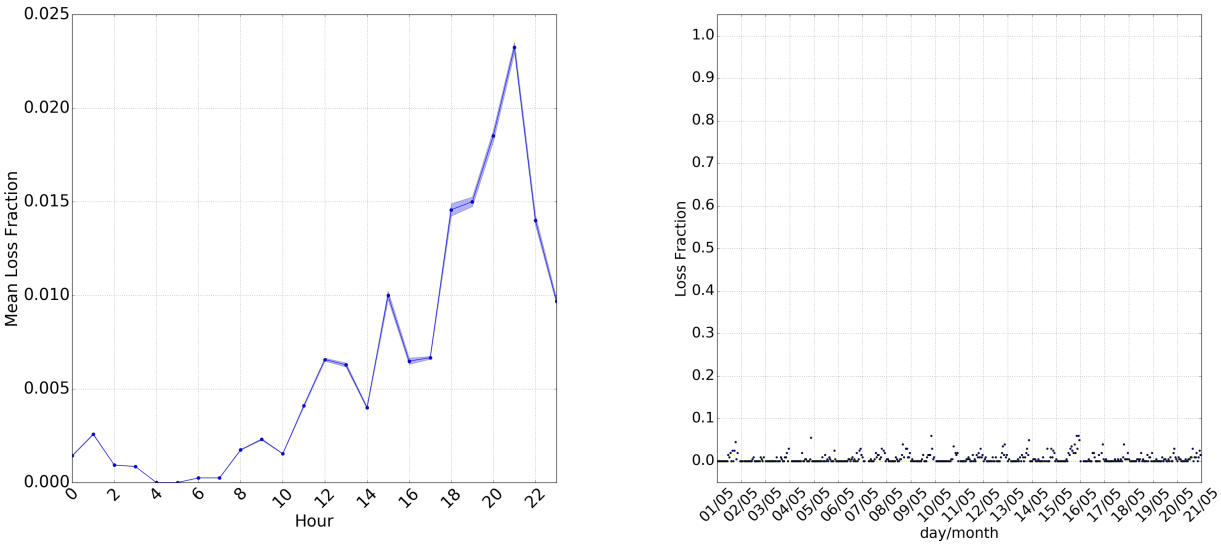
(a) Autocorrelation

(b) Time series

Figure 3.4: Client 3

The autocorrelation of figure 3.2 have a periodic pattern, with peaks around multiples of 24 hours. In this client, it is possible to observe that losses are more usual at night. However, in figure 3.3, the autocorrelation quickly decreases and fluctuates around zero. The corresponding time series has a common characteristic, from 07/may on, measures with losses alternated with zero loss measures. Figure 3.4 shows a linear decreasing autocorrelation, and the associated time series presents abrupt changes in the mean.

As in figure 3.2, some clients have a daily pattern in which losses occur more frequently at night. This fact can be linked with congestion, since the Internet usage increases at the end of a day. Figure 3.5 corroborates that observation, illustrating the mean and variance of measures that occurred in a specific hour through all days.



(a) Mean and variance per hour

(b) Time series

Figure 3.5: Client 4

With a visual analysis, as in the previous plots, it is possible to observe different time series and change points patterns.

3.2 Change Points Dataset

3.2.1 Methodology

There are several approaches to construct a change points dataset. Some works create simulated time series, in which distinct segments are sampled by the same generative model with different parameters [12]. In general, this type of data is more easily handled by change point detection algorithms, since some methods assume the same models used in the dataset build process. Also, real data can have complex characteristics that are difficult to be reproduced by generative models. Another strategy is to join segments from different real time series [8]. However, this can introduce unreal change points scenarios.

When the latent information of the time series are available, and if there is a complete knowledge of what configurations changes in the latent state impact data, it is possible to check the change points only analyzing this underlying information. Considering the application domain of the present work, this approach would be impractical. First, this would need the expertise of how the configurations of network topology, routers congestion, physical equipment problems, among other features, affect the loss fraction. Second, this kind of information is absent in the dataset, and would be too complex to collect it.

The approach followed by this project was to use visual classifications, as it was done in [13]. An application domain expert was exposed to a set of time series, and visually indicated his opinion about the change points locations. It is known that visual inspection methods can bring erroneous conclusions [14], and also amplify subjectivity, however, it was the best alternative considering the data availability and the objective of working with real data.

Through a web system the user freely marked the change points with a mouse. The fact that data is not regularly sampled in time could bring an unwanted visual change perception. Therefore, the X axis of the displayed time series represented only the temporal order of the measures.

A single specialist classified all time series. This person has experience with network measurements and statistical modeling, however, without background in change point detection. The user could take any time to make a classification, and it was able to access the system in different days. Additionally, it was provided a set of tips to the specialist:

- In the case of packet loss fraction, mean changes between 0 and 0.1 are more sensible to the end users.
- The time axis only represents the temporal order of the measurements. However, in general, consecutive points in time axis are separated by 30 minutes.
- Outlier is not a statistical change. An outlier is an observation that lies outside the overall pattern of a distribution.

To provide a clear visualization, the change points dataset was composed by time series with 10 days of data. Therefore, each time series of the previous dataset was split in two, one from 01/may/2016 to 10/may/2016, and other from 11/may/2016 to 20/may/2016. Also, it was selected only the ones that have at least 85% of the maximum possible number of points during the specified period, considering that data is sampled at most two times in a hour. Change points can be interpreted as rare events in this dataset, and most data streams have almost all measures with zero losses. Therefore, to increase the dataset entropy, it was only selected time series that have at least one window of length 48 with more than 5 measures with loss fraction larger than 0.01. These filters resulted in 522 time series.

Figure 3.6 presents a system snapshot. The vertical red line means that the user marked a change point in that position.

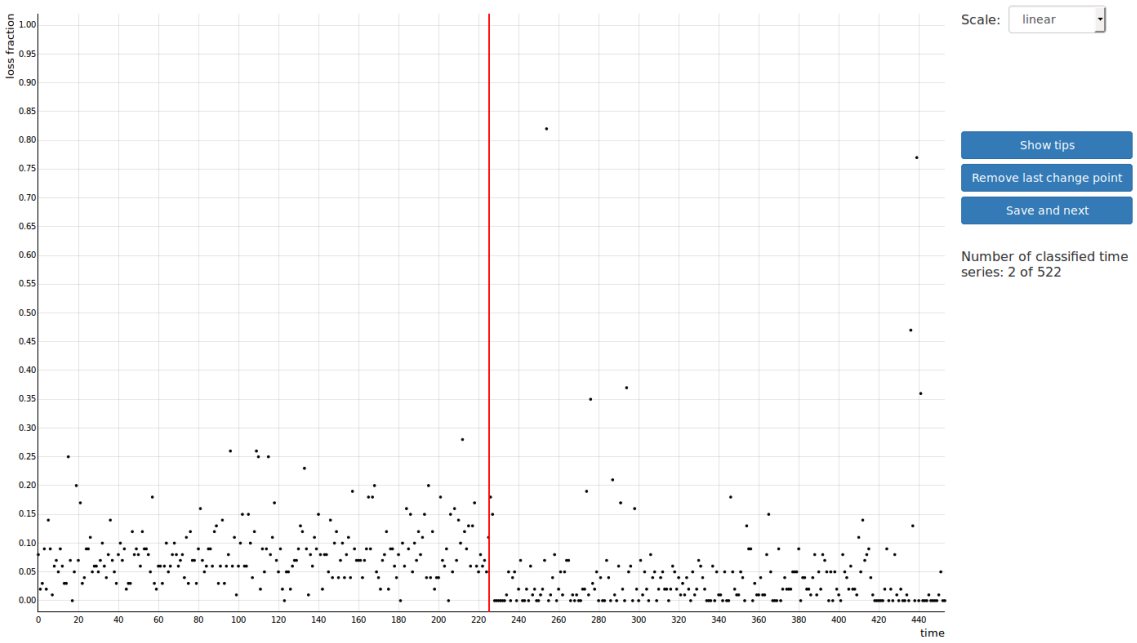


Figure 3.6: Survey system snapshot

3.2.2 Descriptive Analysis

The resulted dataset consists of 522 time series, 241470 measures, and 866 change points. Next, in figure 3.7, is presented the number of change points per time series distribution.

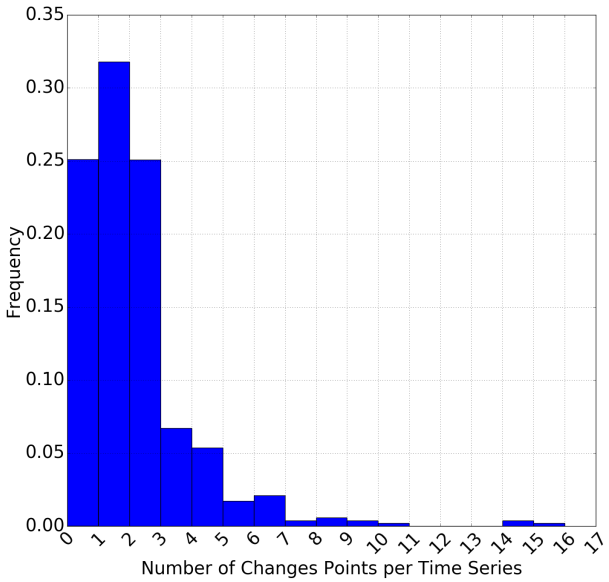


Figure 3.7: Number of change points per time series distribution

Figure 3.8 presents the segment length distribution, conditioning on data streams that have at least one change point. Since the time series have continuations past

the first segment, and in the future of the last, this analysis dismissed the first and last windows. This avoids conclusions misunderstandings about how much time a client stands with the same pattern.

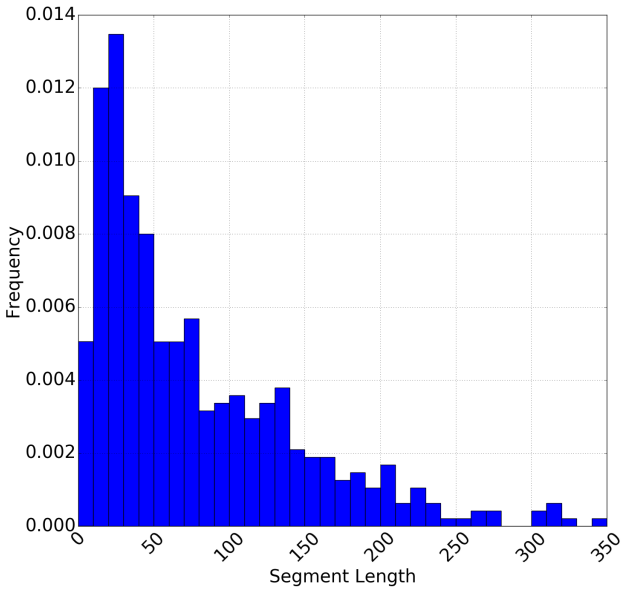
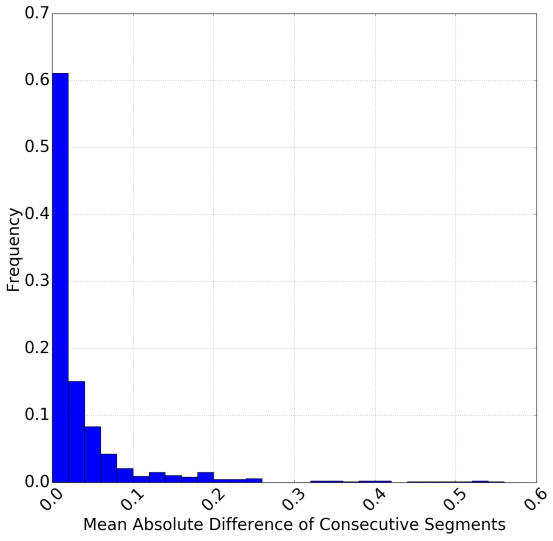
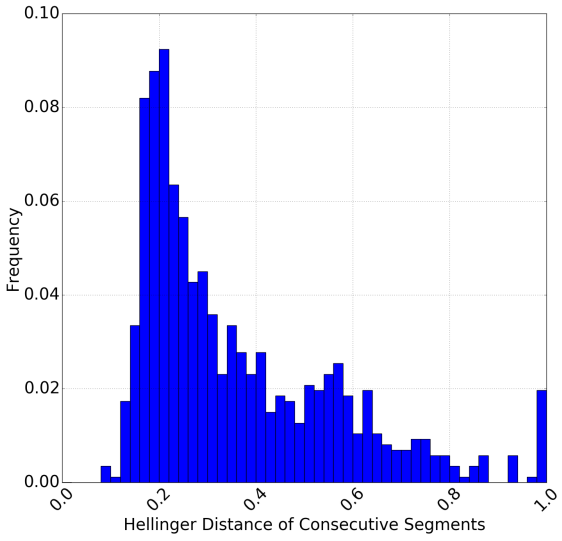


Figure 3.8: Segment length distribution. Bins of size 10.

In figure 3.9 is made a comparison between consecutive segments, presenting the absolute mean difference and the discrete Hellinger distance distributions.



(a) Absolute mean difference



(b) Hellinger distance

Figure 3.9: Consecutive segments comparison. Bins of size 0.02.

Chapter 4

Conclusions

Bibliography

[1] WIKIPEDIA. "Hellinger distance". 2016. https://en.wikipedia.org/wiki/Hellinger_distance/.

[2] KIFER, D., BEN-DAVID, S., GEHRKE, J. "Detecting Change in Data Streams". In: *Proceedings of the Thirtieth International Conference on Very Large Data Bases - Volume 30*, VLDB '04, pp. 180–191. VLDB Endowment, 2004. ISBN: 0-12-088469-0. Disponível em: <http://dl.acm.org/citation.cfm?id=1316689.1316707>.

[3] MAIDSTONE, R., HOCKING, T., RIGAILL, G., et al. "On optimal multiple changepoint algorithms for large data", *Statistics and Computing*, pp. 1–15, 2016.

[4] KILLICK, R., FEARNHEAD, P., ECKLEY, I. "Optimal detection of changepoints with a linear computational cost", *Journal of the American Statistical Association*, v. 107, n. 500, pp. 1590–1598, 2012.

[5] HAYNES, K., ECKLEY, I. A., FEARNHEAD, P. "Computationally Efficient Changepoint Detection for a Range of Penalties", *Journal of Computational and Graphical Statistics*, v. 0, n. ja, pp. 1–28, 0. doi: 10.1080/10618600.2015.1116445. Disponível em: <http://dx.doi.org/10.1080/10618600.2015.1116445>.

[6] KEHAGIAS, A. "A hidden Markov model segmentation procedure for hydrological and environmental time series", *Stochastic Environmental Research and Risk Assessment*, v. 18, n. 2, pp. 117–130, 2004. ISSN: 1436-3259. doi: 10.1007/s00477-003-0145-5. Disponível em: <http://dx.doi.org/10.1007/s00477-003-0145-5>.

[7] LUONG, T. M., ROZENHOLC, Y., NUEL, G. "Fast estimation of posterior probabilities in change-point analysis through a constrained hidden Markov model", *Computational Statistics and Data Analysis*, v. 68, pp. 129 – 140, 2013. ISSN: 0167-9473. doi: http://dx.doi.org/10.1016/

j.csda.2013.06.020. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167947313002326>.

[8] MONTAÑEZ, G. D., AMIZADEH, S., LAPTEV, N. "Inertial Hidden Markov Models: Modeling Change in Multivariate Time Series". In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pp. 1819–1825. AAAI Press, 2015. ISBN: 0-262-51129-0. Disponível em: <http://dl.acm.org/citation.cfm?id=2886521.2886573>.

[9] FEARNHEAD, P. "Exact and efficient Bayesian inference for multiple changepoint problems", *Statistics and Computing*, v. 16, n. 2, pp. 203–213, 2006. doi: 10.1007/s11222-006-8450-8. Disponível em: <http://dx.doi.org/10.1007/s11222-006-8450-8>.

[10] ADAMS, R. P., MACKAY, D. J. "Bayesian online changepoint detection", *arXiv preprint arXiv:0710.3742*, 2007.

[11] MENDES, D. X., SENGES, G. D. S., SANTOS, G. H. A. D., et al. "A Preliminary Performance Measurement Study of Residential Broadband Services in Brazil". In: *Proceedings of the 2016 Workshop on Fostering Latin-American Research in Data Communication Networks*, LANCOMM '16, pp. 16–18, New York, NY, USA, 2016. ACM. ISBN: 978-1-4503-4426-5. doi: 2940116.2940135. Disponível em: <http://doi.acm.org/2940116.2940135>.

[12] LIU, S., YAMADA, M., COLLIER, N., et al. "Change-point detection in time-series data by relative density-ratio estimation", *Neural Networks*, v. 43, pp. 72 – 83, 2013. ISSN: 0893-6080. doi: http://dx.doi.org/10.1016/j.neunet.2013.01.012. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0893608013000270>.

[13] HOCKING, T., RIGAILL, G., PHILIPPE VERT, J., et al. "Learning Sparse Penalties for Change-point Detection using Max Margin Interval Regression". In: Dasgupta, S., Mcallester, D. (Eds.), *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, v. 28, pp. 172–180. JMLR Workshop and Conference Proceedings, maio 2013. Disponível em: <http://jmlr.org/proceedings/papers/v28/hocking13.pdf>.

[14] JAMES, N. A., KEJARIWAL, A., MATTESON, D. S. "Leveraging Cloud Data to Mitigate User Experience from" Breaking Bad"", *arXiv preprint arXiv:1411.7955*, 2014.

[15] AUGER, I. E., LAWRENCE, C. E. "Algorithms for the optimal identification of segment neighborhoods", *Bulletin of Mathematical Biology*, v. 51, n. 1, pp. 39–54, 1989. ISSN: 1522-9602. doi: 10.1007/BF02458835. Disponível em: <http://dx.doi.org/10.1007/BF02458835>.