POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Optimal Change Point Detection by Error Maximization

AURÉLIEN SERRE
Département de mathématiques et de génie industriel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Mathématiques appliquées

Août 2019

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

Optimal Change Point Detection by Error Maximization

présenté par **Aurélien SERRE**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

Jonathan JALBERT, président
Andrea LODI, membre et directeur de recherche
Yossiri ADULYASAK, membre

ACKNOWLEDGEMENTS

I want to thank my supervisor Andrea Lodi for his good advice and for always pushing me to concentrate on the important questions and to keep the big picture in my mind. Despite his busy schedule he gave me his attention whenever necessary, and was very patient with me who took a while to complete this master. He also assembled a very great team within his chair.

I also want to thank PREDICT and especially Flavien Peysson for giving me a lot of freedom in how to approach to problem and for providing useful datasets.

Je remercie beaucoup Didier Chételat, mon superviseur officieux, qui n'a jamais compté son temps passé à répondre à mes questions, discuter et me donner de bonnes idées. Il a été d'une grande aide pour la relecture de ce mémoire, et ce jusqu'au dernier instant.

I thank all the members of the chair and other labs around, that make up a great team of people with who I had interesting conversations and shared climbing sessions, bike rides, concerts, tennis matches, beers among many others.

Je remercie Mehdi Taobane pour m'avoir aidé à naviguer entre les écueils administratifs, Khalid pour toujours avoir résolu mes problèmes de réseau, Koladé Nourou pour son humour, son sourir et son soutient.

I thank the members of the Montreal Rowing Club, for the great training sessions and competitions. They helped me change my mind, get out of my student life, and helped me get tanned.

RÉSUMÉ

L'entreprise PREDICT qui est notre partenaire sur ce projet est entre autres spécialisé dans la surveillance d'installations industrielles. Cela consiste à surveiller certains paramètres mesurés sur un équipement (comme par exemple des mesures de température, pression, vibrations etc...) au cours du temps de manière à détecter d'éventuels signes précurseurs d'une panne. Cette tâche nécessite très souvent de détecter au préalable les instants auxquels des opérations de maintenance ont été effectuées, ou encore où l'usage fait de l'équipement change. De plus, la détection doit être faite simplement à l'aide des mesures surveillées, sans accès à de l'information supplémentaires à propos des événements à détecter. Ce problème peut être formulé comme un problème de détection de ruptures, qui consiste à estimer les instants où les propriétés statistiques d'une série temporelle changent de manière abrupte.

La détection de ruptures a énormément été étudié en traitement du signal, et a des applications dans de nombreux domaines tels qu'en bioinformatique, en analyse du climat, en finance, en traitement de la parole, ainsi qu'en maintenance conditionnelle. Dans la littérature, de nombreuse méthodes fréquentistes existent qui consistent à associer un coût à toute configuration possible des positions des ruptures à l'aide d'un modèle statistique de la série temporelle. Le nombre de ruptures et leur positions sont ensuite estimés en maximisant ce coût sur toutes les configurations possibles. En général, le coût est conçu pour représenter l'ajustement du modèle constant par morceaux associé à la configuration.

Dans ce mémoire, nous proposons une nouvelle approche du problème de détection de ruptures qui consiste à maximiser la différence des propriétés statistiques entre segments consécutifs séparés par les points de ruptures. Pour cela, nous développons un nouveau type de fonction objectif basé sur la différence de propriétés statistiques, par opposition aux fonctions objectif basées sur l'ajustement utilisées dans la littérature. Étant donné que ce nouveau type de fonction objectif n'est pas compatible avec les algorithmes existants, nous introduisons également deux algorithmes permettant la résolution du problème d'optimisation correspondant à cette nouvelle fonction objectif.

Nous comparons les performances de cette nouvelle approche avec trois méthodes issues de la littérature. Deux d'entre elles, appelées Pruned Exact Linear Time and Segment Neighbourhood sont exacte, tandis que la troisième, Sliding Adjacent Windows est une méthode approximative basée sur une fonction objectif similaire à celle que nous proposons. Nous effectuons cette comparaison à l'aide de deux jeux de données empiriques, dont l'un nous a été fourni par PREDICT et correspond à un cas d'application qui les intéresse. Nous montrons que sur ces deux jeux de données montrent que notre méthode est capable d'estimer la position des ruptures de manière plus précise que les trois méthodes concurrentes. Notre approche peut être appliquée à de nombreux types de séries temporelles différentes, grâce au fait qu'elle peut être combinée avec de nombreux modèles différents pour décrire la série temporelle. De plus, cette approche se révèle efficaces dans des cas d'application concrets de notre partenaire PREDICT.

ABSTRACT

Our partner PREDICT is specialized in condition monitoring of industrial systems, which consists in monitoring certain parameters measured on an equipment (such as temperature, pressure, vibration) through time in order to detect signs indicative of a developing fault. For performing this task, they often need to detect events such as the occurrence of maintenance operations or changes of the conditions in which the equipment is being operated. This detection task needs to be performed using only the monitored measurements, with no additional external information available about the events. This problem can be formulated as a change point detection problem, which consists in detecting and finding the positions of abrupt changes of the statistical properties of a time-series.

Change point detection has been extensively studied in signal processing, and has applications in a wide range of fields such as bioinformatics, climate analysis, finance, speech processing and condition based maintenance. In the literature, many frequentist methods have been developed, where a statistical model of the time-series is used to assign a cost to any possible configuration of the change points. The estimated number and positions of the change points is then obtained by minimizing this cost over the set of all possible configurations. The cost is typically a measure of the goodness of fit of a piecewise constant model that changes at each change point.

In this work, we propose a new approach to change point detection that consists in maximizing the discrepancy of the statistical properties between consecutive segments delimited by the change points. We do this by developing a new type of discrepancy-based objective function different from the goodness of fit-based cost functions from the literature. We also propose an appropriate algorithm for solving the associated optimization problem, since our new type of objective function is not compatible with the existing algorithms.

We compare the performance of this new approach against two exact methods called Pruned Exact Linear Time and Segment Neighbourhood, as well as an approximate method based on a similar objective function called Sliding Adjacent Windows. This comparison is performed on two real-world datasets, one of them being supplied by our partner PREDICT, and corresponding to a use case they would be interested in. On both of these datasets, we show that our approach is able to estimate the positions of the change points more accurately than the three competitors. Our approach can be applied on a wide range of different types time-series, since it can accommodate many different models for the data. Moreover, it proves to be useful to our partner PREDICT on concrete use cases.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ACRONYMS

CBM	Condition-based maintenance
CPD	change point detection
i.i.d.	independent and identically distributed
SN	Segment Neighbourhood
OP	Optimal Partitioning
PELT	Pruned Exact Linear Time
BS	Binary Segmentation
SAW	Sliding Adjacent Windows
CROPS	Changepoints for a Range Of PenaltieS
CUSUM	Cumulative Sum
ARL	avergage run length
BIC	Bayesian information criterion
AIC	Akaike information criterion
HQC	Hannan–Quinn information criterion
DAG	directed acyclic graph
VAR	vector autoregressive
OTAWA	Optimal Two Adjacent Windows Algorithm
STFT	Short-time Fourier transform
MSE	mean square error
OLS	ordinary least squares
GLS	generalized least squares

LIST OF APPENDICES

CHAPTER 1 INTRODUCTION

1.1 Maintenance for industrial equipments

Maintenance is an important component of running industrial equipments. Indeed, any industrial equipment is subject to degradations over time, and it is essential to perform maintenance on it, in order to keep it in a state where it is able to fulfill its function. The most simple scheme consists in scheduling the maintenance actions a priori, without any knowledge of the present state of the equipment. This is called planned maintenance.

1.1.1 Maintenance scheduling problem

However the degradation process is stochastic, making the time before failure vary from one maintenance cycle to the next. The planned maintenance will thus necessarily not be able to happen right on time. If a failure happens before the scheduled maintenance, urgent corrective maintenance is required, increasing costs. Some costs are also associated to the unexpected failure of the equipment and the additional downtime induced. On the other hand, if a maintenance action is scheduled before the occurrence of a failure, some parts and workforce could have been spared by performing the maintenance later.

When scheduling the maintenance at regular time intervals, the duration of the interval has to be set according to the mean time before failure, as well as the trade off between failure rate and maintenance cost the practitioner is willing to make. For example in aviation, maintenance and checks are scheduled at intervals much shorter than the expected time before failure, because failures are considered highly unacceptable and their rate must be kept low despite higher maintenance costs.

1.1.2 Condition based maintenance (CBM)

The problem with planned maintenance is that it is not able to reach optimality, in the sense that maintenance can never be performed right when need arise. Moreover, it's not very efficient in applications where the failure rate needs to be extremely low such as nuclear power plants or aviation, because parts are changed before all their useful life has been consumed. These are the reasons that sparked the development of Condition-Based Maintenance (CBM). CBM aims at reducing those inefficiencies by using sensors to monitor the condition of an equipment, and perform maintenance right when needed. This information acquired about the condition of equipments can also help optimizing maintenance scheduling, by knowing in advance where and when maintenance is required. Moreover, it helps the operator to locate more accurately which part of the system needs maintenance. At the heart of CBM is **condition monitoring**, which consist in monitoring certain parameters measured on an industrial equipment (such as temperature, pressure, vibration) through time in order to detect signs indicative of a fault arising.

1.2 Our industrial partner

Our partner for this research project is the French company PREDICT, who develops technologies for optimizing the operating performance of factories, vehicles, aircrafts and ships. They are involved in many industrial sectors such as machine tools, industrial vehicles, the steel industry, energy production (hydroelectricity, nuclear, marine current, wind, gas turbine), naval (defense, offshore, maritime), aeronautics, and space. PREDICT designs, develops and implements two lines of software products :

- CASIP : a real-time and embedded solution for proactive maintenance,
- KASEM : a collaborative platform for massive data analysis for early detection, anticipation, predictive diagnostics, real-time health check-ups, prognosis, investigation and proactive therapy.

1.2.1 Condition monitoring

Condition monitoring is an expertise of our partner PREDICT. Depending on the application, they either perform this task in an online setting, or offline in a periodical manner. They have custom monitoring algorithms already developed for identifying signs of a developing fault. However, these algorithms need to be calibrated for the specific operating mode the monitored equipment is in. Different operating modes can correspond to different usage of the equipment (whether a machining tool is used for cutting metal or plastic for instance), or to different conditions of the equipment (whether some significant amount of wear has already been experienced by the equipment, or it has just undergone maintenance). Unfortunately, external information is often missing about the operating mode an equipment is in at any given time. This is often due to the maintenance or usage information of the equipment not being logged. It can also happen that it is considered sensitive information by the operator, who is thus not willing to share it. This sparks the need for automatically detecting changes of the operating mode of an equipment, from the data being monitored only.

1.3 Change point detection

Change point detection is the task of finding the times at which the underlying model of a time-series changes abruptly. It has been extensively studied starting as early as 1954, and is still a very active subfield of signal processing, with many new methods being developed, spanning both Bayesian and frequentist approaches. It has applications in a wide range of fields such as bioinformatics (gait analysis, physiological data, genomics, ECG data), climate analysis and prediction, oceanography, finance, maintenance, human activities analysis, image analysis (security with CCTV images analysis, remote sensing), and speech processing.

1.3.1 Link with the detection of changes of operating mode

We can reasonably assume that changes in operating mode translate into changes of the statistical properties of the time-series representing the measurement being monitored. Moreover, we assume that the changes between operating modes are abrupt. Indeed they correspond either to maintenance events or to changes in type of usage being made of the equipment. Both of these changes usually occur while the equipment is not in use, which should mean that no measurements are being made during the change, thus leading to an abrupt change in the measurement time-series. For these reasons, we can formulate the problem of detecting changes of operating modes as a change point detection problem.

1.4 Objective

The objective of our work is to propose a solution for automatically detecting changes of operating mode of an industrial equipment. Since this detection task is used in situations where external information about the changes is missing, our solution must be unsupervised, meaning it must only use the information contained in the time-series of the monitored parameters, without any additional information. Moreover, the solution must be able to adapt to a wide variety of time-series, without making assumptions about its statistical properties. We are interested here in the case where condition monitoring is performed in an offline manner.

We propose to formulate this problem as a change point detection problem. Our solution consist in developing a new approach for offline unsupervised change point detection. This approach will be based on an underlying model for describing the time-series of monitored measurement, in order to accommodate for a wide variety of time-series. Note that while our goal is to detect changes of operating mode, change point detection (CPD) might be able to detect changes associated to the actual faults as well, despite them usually being of lower magnitude. However this is not a problem here, since in the work flow of PREDICT a human will check the results of the detection before applying the monitoring algorithms, and it can even be considered as an added bonus.

1.5 Outline

We will begin this work with a review of the literature on change point detection (CPD) in Chapter 2. Since quite an extensive body of literature is available on the subject, our review will be selective, and mainly focus on frequentist optimization-based approaches. Chapter 3 will be dedicated to the description of the new approach to CPD that we are proposing. We will start by exposing our new objective function and some time-series models it can accommodate. Then we will expose two algorithms for solving the optimization problem used for estimating the positions of the change points, propose methods for estimating the number of change points, and study the computational cost of the global method proposed. In Chapter 4, we will compare our approach to other methods from the literature. For this purpose, we will use two real-world datasets with annotations. One of these datasets is provided by our partner PREDICT, and corresponds to an actual case where they could use CPD for detecting changes of operating mode in the context of condition monitoring. We will conclude with Chapter 5, where we first summarize our work, then expose how PREDICT benefited from the partnership, and finally discuss the limitations of our work as well as further research directions.

CHAPTER 2 LITERATURE REVIEW

2.1 Introduction to change point detection

Change point detection is the task of finding the times at which the underlying model of a time-series changes abruptly [1]. More generally, change point detection can be applied not only to time-series, but to any sequence of data that is ordered based on some covariate information. For example, change point detection can be performed on measurements of temperatures through time, or on data characterizing genes along a chromosome. For simplicity, we will assume time-series data in this work, but everything said can be applied seamlessly on any kind of sequential data.

Many time-series data represent measurements performed on complex systems such as industrial equipments, an economic system, human activities, or natural phenomena. These systems can be in different states throughout time, exhibiting different behaviors. We can reasonably assume that the transitions between states are reflected by changes of some statistical properties in the corresponding time-series. If these transitions are abrupt, the corresponding time-series can be modeled as a piecewise stationary time-series. Change point detection is useful in those cases when the transitions are known to be abrupt, as it is the problem of detecting and finding the positions of abrupt changes of the statistical properties of a time-series [2]. It can thus be used for inferring such piecewise stationary models of time-series, by determining the segments within which specific stationary models describe the data. Some change point detection methods in the literature can even be seen as a mere model selection problems for the considered time-series. They can indeed be formulated as maximizing the fitness of a segmented model over every possible segmentations of the time-series [1, 3].

Change point detection has been used in a wide spectrum of different fields, such as bioinformatics (gait analysis [4], physiological data [5–7], genomics [8], ECG data [9]), climate analysis and prediction [10], oceanography [11], finance [9, 12], maintenance [13], human activities analysis, image analysis (security with CCTV images analysis [14], remote sensing), and speech processing [15]. It is closely related to the problem of change point estimation, where the goal is to characterize and interpret known changes in the time-series, that are not necessarily assumed to be abrupt [2].

Change point detection methods can be further separated into two groups :

- *Online* methods, where samples are received in real time, and the goal is to detect a change as soon as possible (often called event or anomaly detection); or

- *Offline* methods, where the goal is to detect changes retrospectively, once every samples are available (often called segmentation or edge detection).

This work will focus on the *offline* problem. Additionally, supervised and unsupervised methods can be differentiated, where supervised methods require some example time-series with the change points annotated, in order to be trained, while unsupervised methods don't require any example. We will focus on unsupervised methods here, since in most of the applications of our partner PREDICT, no such annotated example time-series are available because of the lack of external information about the changes of operating mode. Finally, in the literature the difference is made between single and multiple change point detection, since single change point detection usually correspond to much simpler algorithms. This work focuses on multiple change point detection, since in general more than one change of operating mode is to be detected.

In most applications of change point detection, the end goal is to find the position of change points, or to model and predict time-series. But it can also be applied as a preprocessing step, to determine homogeneous segments within a time-series that we want to study individually. For example, this is used in gait analysis, where recordings of accelerometer measurements are increasingly used, in which subjects perform different activities throughout the period of recording [4]. It can also be used as a way to compress the information contained in a time-series, by only recording the positions of the segments and the parameters describing the data within them. Provided the number of segments is much smaller than the number of samples, and the model for the data within segments is relatively simple, the compression can be very efficient. This can then support other applications, such as fast similarity measures for search, matching or clustering of time-series [9].

The rest of this chapter will start with some notations and assumptions that will be used throughout this work. We will then expose some existing methods through a framework that enables us to classify a good portion of them. Finally, we will discuss some other methods from the literature, that do not fit in the framework.

2.2 Notations and assumptions

In this work, we denote $\boldsymbol{x} = (\boldsymbol{x}_t)_{t=1}^{T}$ a T samples long time-series, where $\boldsymbol{x}_t \in \mathbb{R}^d; \forall 1 \leq t \leq T, d \in \mathbb{N}^*$. If $d = 1$, the time-series is univariate. The $(j - i)-$samples long segment made of the samples of $\boldsymbol{x}$ between indices i and $j - 1$, $(1 \leq i < j \leq T + 1)$ is denoted $\boldsymbol{x}_{i:j} = (\boldsymbol{x}_t)_{t=i}^{j-1}$. Finally, any possible segmentation of the time-series $\boldsymbol{x}$ can be referred to as the set of indices of its m change points, that we will denote $\mathcal{T} = \{t_k\}_{k=1}^{m} \subseteq \{2, \ldots, T - 1\}$, where $1 < t_1 < t_2 < \cdots < t_m < T$. Having a change point at index t_i means that there is an abrupt change of some statistical properties of the time-series between samples $\boldsymbol{x}_{t_i - 1}$ and $\boldsymbol{x}_{t_i}$. The cardinality of the segmentation $\mathcal{T}$ is $|\mathcal{T}| = m$, and we will use in our notations the dummy indices $t_0 = 1$ and $t_{m+1} = T$. We will denote $\mathcal{S}_{m,x} = \{\mathcal{T} \text{ s.t. } |\mathcal{T}| = m\}$ the set of all segmentations of the time-series $\boldsymbol{x}$ with m change points, and $\mathcal{S}_{\boldsymbol{x}} = \bigcup_{m=1}^{T-2} \mathcal{S}_{m,x}$ the set of all possible segmentations of the time-series $\boldsymbol{x}$. Note that all possible segmentations are all the subsets $\mathcal{T} \subseteq \{2, \ldots, T-1\}$. So $\mathcal{S}_{\boldsymbol{x}}$ is the power set of $\{2, \ldots, T-1\}$.

We assume that the time-series $\boldsymbol{x}$ on which we are performing change point detection follows a piecewise stationary model. This ensures that there are abrupt changes of some statistical properties to be detected by the change point detection method. We will denote by $t_1^* < t_2^* < \cdots < t_m^*$ the indices of these abrupt changes, and $\mathcal{T}^* = \{t_1^*, t_2^*, \ldots, t_m^*\}$ the corresponding true segmentation. The change point detection problem consists in estimating this true segmentation $\mathcal{T}^*$.

2.3 Framework encompassing many change point detection methods

In the literature, both Bayesian and frequentist approaches to change point detection have been developed. In this work, we will only focus on frequentist methods. The review [1] proposes a structured classification that encompasses many of the frequentist change point detection methods developed in the recent literature. This classification is based on the idea of breaking down change point detection methods into multiple interchangeable components.

2.3.1 Problem formulation

Some assumptions need to be made about the methods that this classification will encompass. First we assume that a change point detection method can be formulated as the problem of finding among the set of all possible segmentations $\mathcal{S}_{\boldsymbol{x}}$ of the given time-series $\boldsymbol{x}$, one that minimizes a criterion $V(\mathcal{T}, \boldsymbol{x})$. We also assume that this criterion function $V(\mathcal{T}, \boldsymbol{x})$ is additive, meaning that it is a sum of the costs associated to each segments in the global segmentation. More formally, this means that the criterion is written as

$$V(\mathcal{T}, \boldsymbol{x}) = \sum_{k=0}^{m} c(\boldsymbol{x}_{t_k : t_{k+1}}), \tag{2.1}$$

where $c(\cdot)$ is a function associating a cost to a segment. Usually, the cost function measures the goodness of fit of the data contained within a segment to a specific model. The estimated segmentation is one that minimizes the criterion $V(\mathcal{T}, \boldsymbol{x})$. It is found by solving a discrete optimization problem of the form

$$\min_{\mathcal{T} \in \mathcal{S}_{\boldsymbol{x}}} V(\mathcal{T}, \boldsymbol{x}). \tag{2.2}$$

However, an important challenge lies in estimating the number of change points in a given time-series. It has been observed that many cost functions tend to largely overestimate the number of change points [16]. This can be interpreted by the fact that adding a change point can only improve the goodness of fit within segments. To avoid such overfitting, some modifications have to be made to the problem (2.2). Two different variants of the problem can be used, depending on the prior information available about the true segmentation and the strategy used for estimating the number of change points :

- **Constrained problem :** if the true number of change points is known a priori, we can use the constrained version of the optimization problem (2.2), in which the number of change points is constrained

$$\begin{aligned} \min_{\mathcal{T} \in \mathcal{S}_{\boldsymbol{x}}} \ & V(\mathcal{T}, \boldsymbol{x}) \\ s.t. \ & |\mathcal{T}| = m; \end{aligned} \tag{2.3}$$

- **Penalized problem :** if the true number of change points is unknown, we can use the penalized version of the optimization problem (2.2), where a penalty term is introduced in the objective function

$$\min_{\mathcal{T} \in \mathcal{S}_{\boldsymbol{x}}} V(\mathcal{T}, \boldsymbol{x}) + pen(\mathcal{T}). \tag{2.4}$$

  The role of this penalty term is to penalize the segmentations with high numbers of change points, in order to counteract the overfitting effect. So usually, the penalty function $pen(\mathcal{T})$ is designed to be monotonically increasing with respect to the number of change points in $\mathcal{T}$, in the case of a minimization problem. Thanks to the addition of the penalty term and the removal of the constraint on the number of change points, both the number of change points and their positions are jointly estimated when solving the problem (2.4).

2.3.2 Framework

Thanks to the formulation of the change point detection problem as the minimization of a criterion, any method that fall into it can be seen as a combination of the following components [1] :

- An appropriate cost function over the segments of a time-series $\boldsymbol{x}$, used in the criterion;

- An algorithm for solving the optimization problem consisting of minimizing the segmentation cost over the set of possible segmentations $\mathcal{S}_{\boldsymbol{x}}$;
- A strategy for estimating the number of change points.

The cost function is chosen according to prior knowledge we have about the time-series. Indeed, it is generally based on a model representing the time-series, and thus depends on the type of model appropriate for the data considered. The choice of the cost function can also depend on constraints on the computational cost of the final algorithm, as different costs can have different computational complexities. Regarding the solving method to be used, the most important choice is between exact and approximate methods. It thus depends mostly on the constraints on the computational cost of the final algorithms, as well as the requirements in terms of precision of the estimated segmentation. Moreover, the penalty function and the solving method are very dependent, as many solving methods can only be used with restricted types of penalty function.

2.4 Cost function

The criterion (2.1) that is optimized can be interpreted as the overall cost of the segmentation $\mathcal{T}$. Its formulation is based on a cost function $c : P_{\boldsymbol{x}} \mapsto \mathbb{R}$, where $P_{\boldsymbol{x}}$ is the set of all possible segments within the time-series $\boldsymbol{x}$. This cost function associates a segment specific cost $c(\boldsymbol{x}_{i:j})$ to any segment $\boldsymbol{x}_{i:j} \in P_{\boldsymbol{x}}$, and is usually derived from an appropriate model representing the time-series $\boldsymbol{x}$. It measures the goodness of fit of the data within the segment $\boldsymbol{x}_{i:j}$ to this model.

The intuition is that in a good segmentation $\mathcal{T}$, the data within each segment $\boldsymbol{x}_{t_k:t_{k+1}}, k = 0, \dots, m$ should be homogeneous, leaving the abrupt changes at the boundaries between segments. In this case, the segment specific costs $c(\boldsymbol{x}_{t_k:t_{k+1}})$ should be minimal, as the model fits well to homogeneous data, in turn minimizing the overall cost of the segmentation $V(\mathcal{T}, \boldsymbol{x})$.

In the following, we will list different models for describing the time-series, with their associated cost function.

2.4.1 Piecewise i.i.d. signal

In many cases, the samples of the time-series $\boldsymbol{x}$ can be modeled as independent random variables following a piecewise constant distribution. In general, both the family of distribution and its parameters can change between segments. In most cases however, we assume that the family of the distribution doesn't change, and that only the parameters do. This means that the samples $\boldsymbol{X}_t$ of the time-series are independent random variables such that

$$\boldsymbol{X}_t \sim \sum_{k=1}^{m^*+1} f(\cdot|\boldsymbol{\theta}_{k(t)}), \tag{2.5}$$

where $\mathcal{T}^* = \{t_k^*\}_{k=1}^{m^*}$ is the true segmentation, $k(t) = \min\{k \text{ s.t. } 1 \leq k \leq m^* \text{ and } t_k^* > t\}$ is the index of the segment containing the sample $\boldsymbol{X}_t$, and $f(\cdot|\boldsymbol{\theta})$ is the given probability density functions parametrized by $\boldsymbol{\theta}$. In such a case, the parameter $\boldsymbol{\theta}$ is the statistical property that is changing at the times t_k^* that we want to estimate.

For any given segment $\boldsymbol{x}_{i:j}$ of $\boldsymbol{x}$, the segment specific parameters $\boldsymbol{\theta}_k$ of the distribution can be estimated through maximum-likelihood, and the associated cost is minus the maximum log-likelihood,

$$c_{i.i.d.}(\boldsymbol{x}_{i:j}) = -\sup_{\boldsymbol{\theta}} \sum_{t=i}^{j-1} \log f(\boldsymbol{x}_t|\boldsymbol{\theta}) \equiv -\log \widehat{\mathcal{L}}(\boldsymbol{x}_{i:j}), \tag{2.6}$$

where $\widehat{\mathcal{L}}(\boldsymbol{x}_{i:j})$ is the maximum likelihood of the model on data $\boldsymbol{x}_{i:j}$. Note that when using the cost function $c_{i.i.d.}$, change point detection is equivalent to maximum likelihood estimation. Indeed, as the criterion $V(\mathcal{T}, \boldsymbol{x})$ is the sum of the segment specific costs, its value is the negative maximum log-likelihood of the piecewise i.i.d. model,

$$V(\mathcal{T}, \boldsymbol{x}) = \sum_{k=0}^{m} c(\boldsymbol{x}_{t_k:t_{k+1}}) = \sum_{k=0}^{m} -\log \widehat{\mathcal{L}}(\boldsymbol{x}_{t_k:t_{k+1}}) = -\log \widehat{\mathcal{L}}(\boldsymbol{x}_{1:T}, \mathcal{T}), \tag{2.7}$$

where $\widehat{\mathcal{L}}(\boldsymbol{x}_{1:T}, \mathcal{T})$ is the maximum likelihood of the piecewise i.i.d. model associated with the segmentation $\mathcal{T}$ on the whole time-series $\boldsymbol{x}_{1:T}$. Solving the problem (2.3) is then equivalent to maximum likelihood estimation of the piecewise stationary model under constraint on the number of change points. Solving (2.4) on the other hand would be equivalent to maximizing what we could call a *penalized log-likelihood.*

The choice of the probability distribution f is often guided by prior knowledge about the time-series. Here is a list of different distributions often used in the literature.

Normal distribution The Gaussian distribution has been the first one introduced in the change point detection literature, and is one of the most studied. It can be used for detecting different types of changes, depending on the parameters that are considered to be segment specific or shared across all the samples of the time-series.

- Used with the assumption of a fixed variance and a mean changing between segments, the normal distribution is useful for detecting changes in the mean of the time-series. In this case, once removing the terms that sum to a constant for all segmentations, hence not influencing the optimal solution (additive constant in the criterion), the cost function $c_{i.i.d.}$ becomes

$$c_{L_2}(\boldsymbol{x}_{i:j}) = \sum_{t=i}^{j-1} \|\boldsymbol{x}_t - \hat{\boldsymbol{\mu}}_{ij}\|_2^2, \tag{2.8}$$

where $\hat{\boldsymbol{\mu}}_{ij}$ is the sample mean within the segment $\boldsymbol{x}_{i:j}$, and $\|\cdot\|_2$ is the Euclidean norm.

This model is often called mean-shift. It has been introduced in the initial paper from E.S. Page [13] for an application in industrial quality control. It has been further studied in [17–21]. It has also been used with simulated data in [16].

- On the contrary, the mean can be assumed to be shared across all the samples of the time-series, while the variance changes between segments. This model for detecting changes in variance is often used in finance [12]. The cost function $c_{i.i.d.}$ is rewritten as

  $$c_{\Sigma}(\boldsymbol{x}_{i:j}) = (j-i)\log|\hat{\Sigma}_{i:j}| + \sum_{t=i}^{j-1} (\boldsymbol{x}_t - \boldsymbol{\mu})^T \hat{\Sigma}_{i:j}^{-1} (\boldsymbol{x}_t - \boldsymbol{\mu}), \tag{2.9}$$

  where $\boldsymbol{\mu}$ is the constant mean value, and $\hat{\Sigma}_{i:j}$ is the sample covariance matrix of the data within the segment $\boldsymbol{x}_{i:j}$. In cases where the mean value $\boldsymbol{\mu}$ is not known a priori, it can be considered as an additional parameter and estimated via maximum likelihood, i.e. by replacing $\boldsymbol{\mu}$ by the sample mean over the whole time-series $\hat{\boldsymbol{\mu}}_{1T} = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{x}_t$ in the cost (2.9).

  This model is used in [11] to detect changes in variance within univariate time-series with constant unknown mean. It is applied to oceanographic and financial data as well as simulated data.

- We can go further and let both the variance *and* the mean change abruptly, in order to detect changes in both of these parameters. The cost function $c_{i.i.d.}$ then becomes

  $$c_{\mu\Sigma}(\boldsymbol{x}_{i:j}) = (j-i)\log|\hat{\Sigma}_{i:j}| + \sum_{t=i}^{j-1} (\boldsymbol{x}_t - \hat{\boldsymbol{\mu}}_{ij})^T \hat{\Sigma}_{i:j}^{-1} (\boldsymbol{x}_t - \hat{\boldsymbol{\mu}}_{ij}), \tag{2.10}$$

  where $\hat{\boldsymbol{\mu}}_{ij}$ is the sample mean and $\hat{\Sigma}_{i:j}$ is the sample covariance matrix of the data within the segment $\boldsymbol{x}_{i:j}$.

  This model is applied to geological data in [22], to financial data in [23] and to simulated data in [12, 16]. It is also studied more theoretically in [24].

It is explained in [23,25] that this model is able to detect change in the first two moments of random variables not necessarily following a normal distribution, even thought the model is based on the Gaussian distribution.

Poisson distribution Change point detection can be applied to time-series representing count numbers. If the time-series can be modeled as independent Poisson distributed samples with piecewise constant rate parameter, the cost function $c_{i.i.d.}$ can be used with the Poisson distribution in order to detect abrupt changes in the rate parameter. Once removing the terms that sum to a constant for all segmentations, the cost function $c_{i.i.d.}$ becomes

$$c_{Poisson}(\boldsymbol{x}_{i:j}) = -(j-i)\hat{\boldsymbol{\mu}}_{ij} \log \hat{\boldsymbol{\mu}}_{ij}, \tag{2.11}$$

where $\hat{\boldsymbol{\mu}}_{ij}$ is the sample mean within the segment $\boldsymbol{x}_{i:j}$.

Change point detection with the cost function $c_{Poisson}$ has been applied to real data in [21,26]. Both of these papers consider a dataset reporting the number of coal-mining disasters by year in Britain between 1851 and 1962.

2.4.2 Piecewise autoregressive model

The autoregressive model is a popular model for time-series analysis, where each sample is represented as a linear combination of the p previous values. Some time-series can be modeled by a piecewise autoregressive model. Let $\mathcal{T}^* = \{t_k^*\}_{k=0}^{m^*}$ be the true segmentation, with t_k^* being the times at which the underlying autoregressive model changes abruptly. In the case of a univariate time-series, the piecewise autoregressive model of order p models the samples as random variables that verify

$$X_t = c_k + \sum_{i=1}^{p} \varphi_{ik} X_{t-i} + \varepsilon_t, \quad \forall t,\, t_k^* \leq t < t_{k+1}^*, \quad k = 0, \dots, m^*, \tag{2.12}$$

where the vector of regression coefficients $\boldsymbol{\varphi}_k = (\varphi_{1k}, \dots, \varphi_{pk})^T \in \mathbb{R}^p$ and the intercept $c_k \in \mathbb{R}$ are unknown parameters of a model specific to segment $X_{t_k:t_{k+1}}$, and ε_t is an error term. Change point detection can be performed on such time-series in order to detect changes in the autoregressive structure of a time-series. To that end, we can estimate the parameters $\hat{\boldsymbol{\varphi}}_{ij}$ and $\hat{c}_{ij}$ of the autoregressive models specific to each segment $x_{i:j}$ of any given candidate segmentation. The estimation can be performed for instance via ordinary least squares (OLS), generalized least squares (GLS), Lasso or Ridge regression. The segment specific cost can then be defined as the sum of the squared residuals

$$c_{AR}(x_{i:j}) = \sum_{t=i}^{j-1} \|x_t - \hat{c}_{ij} - \sum_{l=1}^{p} \hat{\varphi}_{lk} x_{t-l}\|_2^2, \tag{2.13}$$

where $\|\cdot\|_2$ is the Euclidean norm.

Piecewise autoregressive models are used in [21, 27–30]. They are also applied on fMRI data in [31]. [11] uses it on simulated data, with in addition letting the order parameter p change between segments as well. The autoregressive model of order 1 is studied in [24, 32].

We presented here the case of a univariate time-series x, in order to keep the notations simple. However this model can be extended to the case of multivariate time-series as well, using a vector autoregressive (VAR) model. In this case, each component is described as a linear combination of the lagged values of every components. This type of model is studied in [33, 34].

Piecewise linear models Note that the piecewise autoregressive model is a special case of the piecewise linear model, which is used to model the time-series with respect to other covariate time-series. For a univariate time-series, this model is formulated as

$$X_t = \boldsymbol{u}_k^T \boldsymbol{y}_t + \boldsymbol{v}^T \boldsymbol{z}_t + \varepsilon_t, \quad \forall t,\, t_k^* \leq t < t_{k+1}^*, \quad k = 0, \ldots, m^*, \tag{2.14}$$

where $\boldsymbol{u}_k \in \mathbb{R}^p$ and $\boldsymbol{v} \in \mathbb{R}^q$ are unknown regression parameters, ε_t is an error term, and $\boldsymbol{y} = (\boldsymbol{y}_t)_{t=1}^T$ and $\boldsymbol{z} = (\boldsymbol{z}_t)_{t=1}^T$ are covariate time-series with values in $\mathbb{R}^p$ and $\mathbb{R}^q$. The autoregressive model is the special case where the covariate time-series $\boldsymbol{y}$ is such that $\forall t = p+1..T$, $\boldsymbol{y}_t = [x_{t-1}, x_{t-2}, \ldots, x_{t-p}]$. If segment specific models are estimated via OLS, the corresponding segment specific costs are

$$c_{LR}(x_{i:j}) = \min_{\boldsymbol{u}_k \in \mathbb{R}^p, \boldsymbol{v} \in \mathbb{R}^q} \sum_{t=i}^{j-1} (x_t - \boldsymbol{u}_k^T \boldsymbol{y}_t - \boldsymbol{v}^T \boldsymbol{z}_t)^2. \tag{2.15}$$

This model is widely used in the literature, especially in econometrics where it is referred to as partial structural change model. A pure structural change model can be obtained by simply removing the term $\boldsymbol{v}^T \boldsymbol{z}_t$.

The piecewise linear model is studied for example in [33,35]. It is used in [5] with physiological data and in [9] with the time directly as the covariate variable $y_t = t$. It has also been studied from a more theoretical standpoint in [36–38]. The multivariate version is exposed in [33].

In some cases where the noise distribution has a heavy tail, it can be interesting to use the absolute residuals instead in the definition of the cost c_{LR} [25]. This has been used with econometrics data in [39].

2.4.3 Kernel models

Kernel-based cost functions work by mapping a sample $\boldsymbol{x}_t \in \mathbb{R}^d$ of the time-series $\boldsymbol{x}$ into a space of functions called a reproducing kernel Hilbert space, that we will note $\mathcal{H}$. The definition of the mapping is based on a kernel $k\colon \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}$, and the mapping function is $\phi\colon \mathbb{R}^d \mapsto \mathcal{H}$, which maps the sample $\boldsymbol{x}_t$ to $\phi(\boldsymbol{x}_t) = k(\boldsymbol{x}_t, \cdot)$. The inner-product of $\mathbb{H}$ is $\langle \phi(\boldsymbol{x}_s), \phi(\boldsymbol{x}_t) \rangle_{\mathcal{H}} = k(\boldsymbol{x}_s, \boldsymbol{x}_t)$, and the norm is such as $\|\phi(\boldsymbol{x}_t)\|^2_{\mathcal{H}} = k(\boldsymbol{x}_t, \boldsymbol{x}_t)$. The cost function is then defined as

$$c_{kernel}(\boldsymbol{x}_{i:j}) = \sum_{t=i}^{j-1} \|\phi(\boldsymbol{x}_t) - \overline{\boldsymbol{\mu}}_{i:j}\|^2_{\mathcal{H}}, \tag{2.16}$$

where $\overline{\boldsymbol{\mu}}_{i:j} \in \mathcal{H}$ is the empirical mean of $\phi(\boldsymbol{x}_t)$ on segment $\boldsymbol{x}_{i:j}$. In practice, when computing $c_{kernel}(\boldsymbol{x}_{i:j})$, we don't need to compute the embedding. Indeed, thanks to the "kernel trick", the cost function can be rewritten as

$$c_{kernel}(\boldsymbol{x}_{i:j}) = \sum_{t=i}^{j-1} k(\boldsymbol{x}_t, \boldsymbol{x}_t) - \frac{1}{j-i} \sum_{s=i}^{j-1} \sum_{t=i}^{j-1} k(\boldsymbol{x}_s, \boldsymbol{x}_t). \tag{2.17}$$

The intuition behind this cost function is that, with a kernel $k(\cdot, \cdot)$ well suited to the time-series considered, samples that follow the same probability distribution should have similar embeddings. The cost $c_{kernel}(\boldsymbol{x}_{i:j})$ defined at (2.16) is actually similar to the cost $c_{L_2}(\boldsymbol{x}_{i:j})$ for detecting changes in mean, but can be seen as detecting mean shift in the embedded time-series $\{\phi(\boldsymbol{x}_t)\}_{t=1}^{T}$. And detecting mean shift in the embedding space $\mathcal{H}$ can be seen as detecting changes in the underlying probability distribution of the original samples.

The kernel-based cost function is different from the other cost functions described previously, is the sense that it is a non-parametric approach to change point detection. This means that no estimation of segment specific parameters of an underlying model is needed for computing the cost function. This has the advantage of usually being more computationally efficient than parametric model. Also, non-parametric methods have the advantage of being compatible with time-series modeled as random processes following piecewise constant distributions, but for which the distribution is either non parametric, or unknown.

Kernel models have been used with empirical data for the problem of emotion recognition in [40], and on Brain-Computer Interface data in [41]. The use of the kernel model (2.16) is also discussed for the task of video segmentation in [42], though no experiments on real data are performed. It has also been studied for application on DNA sequences through simulated data in [43].

2.5 Optimization

Once the cost function $V(\mathcal{T}, \boldsymbol{x})$ is defined, we need a method for searching the set of admissible segmentations. In the case of the penalized version of the problem (2.4), this set is the power set $\mathcal{S}_{\boldsymbol{x}_{1:T}} = \{\mathcal{T} \subseteq \{2, \ldots, T-1\}\}$, of cardinality $|\mathcal{S}_{\boldsymbol{x}_{1:T}}| = 2^{T-2}$. In the case of the constrained version of the problem (2.3), the set of admissible segmentations $\mathcal{S}_{m,\boldsymbol{x}_{1:T}} = \{\mathcal{T} \subseteq \{2, \ldots, T-1\} \,\text{s.t.}\, |\mathcal{T}| = m\}$ is smaller, but still contains $|\mathcal{S}_{m,\boldsymbol{x}_{1:T}}| = \binom{T-2}{m}$ elements. In any case, exhaustive enumeration is intractable. In this section we describe different solutions for efficiently solving either the penalized or the constrained optimization problem. They are classified in two groups : exact and approximate methods.

2.5.1 Exact methods

Let us first introduce a proposition that will be used in the algorithms we are about to introduce.

Proposition 2.1. *Consider an optimization problem* $\min_{\mathcal{T}} V(\mathcal{T}, \boldsymbol{x})$ *with an additive criterion as defined in* (2.1)*, which can be written as* $\min_{\mathcal{T}} \sum_{k=0}^{m} c(\boldsymbol{x}_{t_k:t_{k+1}})$*. Let* $\mathcal{T}^* = \{t_k^*\}_{k=1}^{m} \in \mathcal{S}_{\boldsymbol{x}_{1:T}}$ *be the optimal segmentation of a time-series* $\boldsymbol{x}_{1:T}$*, that is satisfying* $V(\mathcal{T}^*, \boldsymbol{x}_{1:T}) \leq V(\mathcal{T}, \boldsymbol{x}_{1:T})$, $\forall \mathcal{T} \in \mathcal{S}_{\boldsymbol{x}_{1:T}}$*. Then any sub-segmentation* $\mathcal{T}' = \{t_k^*\}_{k=i}^{j} \subseteq \mathcal{T}^*, 1 \leq i < j \leq m$ *of* $\mathcal{T}^*$ *is an optimal segmentation of the segment* $\boldsymbol{x}_{t_{i-1}^*:t_{j+1}^*}$*, i.e.* $V(\mathcal{T}', \boldsymbol{x}_{t_{i-1}^*:t_{j+1}^*}) \leq V(\mathcal{T}, \boldsymbol{x}_{t_{i-1}^*:t_{j+1}^*})$, $\forall \mathcal{T} \in \mathcal{S}_{\boldsymbol{x}_{t_{i-1}^*:t_{j+1}^*}}$*.*

This proposition is demonstrated in an even more general case in [3] as *Proof 1.* Intuitively, it states that if a segmentation is optimal on a given time-series, then any sub-segmentation (subset of its segments which are consecutive) is optimal for the portion of the time-series it covers.

Segment Neighborhood The Segment Neighbourhood (SN) method, introduced in [44], proposes a way of solving the constrained problem (2.3). It is based on dynamic programming [45], and uses Proposition 2.1 to recursively decompose the optimization problem into smaller sub-problems. Let M be the user specified number of change points to be detected, and $\boldsymbol{x}_{1:T}$ the T samples long time-series considered. Thanks to Proposition 2.1, the constrained problem can be rewritten as

$$\min_{\mathcal{T}\in\mathcal{S}_{M,\boldsymbol{x}_{1:T}}} V(\mathcal{T},\boldsymbol{x}_{1:T}) = \min_{0=t_0<t_1<\dots<t_{M+1}=T} \sum_{k=0}^{M} c(\boldsymbol{x}_{t_k:t_{k+1}}) \tag{2.18}$$

$$= \min_{t\le T-M} \Big[c(\boldsymbol{x}_{1:t}) + \min_{t=t_0<t_1<\dots<t_M=T} \sum_{k=0}^{M-1} c(\boldsymbol{x}_{t_k:t_{k+1}})\Big] \tag{2.19}$$

$$= \min_{t\le T-M} \Big[c(\boldsymbol{x}_{1:t}) + \min_{\mathcal{T}\in\mathcal{S}_{M-1,\boldsymbol{x}_{t:T}}} V(\mathcal{T},\boldsymbol{x}_{t:T})\Big]. \tag{2.20}$$

Intuitively, Equation 2.20 shows that the optimal segmentation with M change points of data $\boldsymbol{x}_{1:T}$ can be easily computed if the optimal segmentations with $M-1$ change points of all segments of the form $\boldsymbol{x}_{t:T}, \quad \forall 1 < t \le T-M$ are known. The SN algorithms then consist of recursively applying this observation (with decreasing values of the number of change points), in order to solve the original constrained problem. Thanks to this recursion over the number of change points, the algorithm actually computes all the optimal segmentations with $m = 1, \dots, M$ change points. It can thus be used with the maximum number of change points to be detected $M = M_{max}$, in order to then solve the penalized problem (2.4). Assuming the costs $c(\boldsymbol{x}_{i:j})$ can be computed or accessed in $\mathcal{O}(1)$, the computational complexity of this method is $\mathcal{O}(MT^2)$. For applications where the number of change points is linear in the length of the time-series considered, the complexity is thus $\mathcal{O}(T^3)$.

Optimal Partitioning Optimal Partitioning (OP) is a method for solving the penalized problem (2.4) with a penalty term of the form $pen(\mathcal{T}) = \beta|\mathcal{T}|$, where β is a smoothing parameter to be chosen by the user. This type of penalty function linear in the number of change points is very common in the literature. It has interesting statistical properties, and also allows for fast algorithms such as OP. More details will be given about it in Section 2.6.1.

The OP algorithm has been introduced in [3], and is also based on dynamic programming [45]. First, we transform the penalized criterion such that it is additive. Thanks to property (2.1) of the unpenalized criterion and the linearity of the penalty function, we can conveniently rewrite the penalized problem (2.4) by distributing the penalty term into the segment specific costs

$$\min_{\mathcal{T}\in\mathcal{S}_{\boldsymbol{x}}} V(\mathcal{T},\boldsymbol{x}_{1:T}) + pen(\mathcal{T}) = \min_{\mathcal{T}\in\mathcal{S}_{\boldsymbol{x}}} \sum_{k=0}^{|\mathcal{T}|} \Big[c(\boldsymbol{x}_{t_k:t_{k+1}})\Big] + \beta|\mathcal{T}| \tag{2.21}$$

$$= -\beta + \min_{\mathcal{T}\in\mathcal{S}_{\boldsymbol{x}}} \sum_{k=0}^{|\mathcal{T}|} \Big[c(\boldsymbol{x}_{t_k:t_{k+1}}) + \beta\Big]. \tag{2.22}$$

We can now use Proposition 2.1 to condition the value of the optimal segmentation of $\boldsymbol{x}_{1:T}$ on the position of its last change point. Let us denote $F(s)$ the cost of the optimal segmentation on data $\boldsymbol{x}_{1:s}$, and $\mathcal{S}_s = \mathcal{S}_{\boldsymbol{x}_{1:s}}$ the set of all possible segmentations of that data. We can rewrite

$$F(s) = -\beta + \min_{\mathcal{T} \in \mathcal{S}_s} \sum_{k=0}^{|\mathcal{T}|} \Big[c(\boldsymbol{x}_{t_k:t_{k+1}}) + \beta \Big] \tag{2.23}$$

$$= -\beta + \min_{1 \leq t < s} \left\{ \min_{\mathcal{T} \in \mathcal{S}_t} \sum_{k=0}^{|\mathcal{T}|} \Big[c(\boldsymbol{x}_{t_k:t_{k+1}}) + \beta \Big] + c(\boldsymbol{x}_{t:s}) + \beta \right\} \tag{2.24}$$

$$= \min_{1 \leq t < s} \Big\{ F(t) + c(\boldsymbol{x}_{t:s}) + \beta \Big\}. \tag{2.25}$$

Intuitively, Equation 2.25 shows that the optimal segmentation on the segment $\boldsymbol{x}_{1:s}$ can be easily computed if the optimal segmentations on the segments $\boldsymbol{x}_{1:t}$, $1 \leq t < s$ are known. The dynamic programming approach of OP then consists in setting $F(1) = -\beta$, and successively computing $F(s)$ for $s = 2, \ldots, T$, in order to obtain the cost $F(T)$ of the optimal segmentation on the full time-series $\boldsymbol{x}_{1:T}$.

Assuming the costs $c(\boldsymbol{x}_{i:j})$ can be computed or accessed in $\mathcal{O}(1)$, the cost of solving the recursion for index s is linear $\mathcal{O}(s)$. It follows that the computational cost of finding $F(T)$ and the optimal segmentation of $\boldsymbol{x}_{1:T}$ is $\mathcal{O}(T^2)$. Optimal Partitioning is thus better suited than SN for solving the penalized problem with a linear penalty term.

Note that OP can actually be used to solve any problem with an additive criterion as defined in (2.1). However, it is not recommended to perform change point detection with no penalty. The penalty term linear in the number of change points is the simplest option and is very easily incorporated into an additive criterion as in (2.22), which is the reason why OP is mostly used with a linear penalty term.

Pruned Exact Linear Time The Pruned Exact Linear Time (PELT) algorithm, introduced in [11], improves on the OP algorithm by adding a pruning rule. This rule discards values of t that can never be solutions of the problem (2.25) to be solved at each iteration. It is based on the assumption about the cost function $c(\cdot)$ that the cost associated to a given segment is reduced when a change point is introduced into it. More formally this is expressed as

$$\exists K \text{ such that } \forall i < j < k,\ c(\boldsymbol{x}_{i:j}) + c(\boldsymbol{x}_{j:k}) + K \leq c(\boldsymbol{x}_{i:k}). \tag{2.26}$$

It is then proved in Section 5 of [12] that if

$$F(i) + c(\boldsymbol{x}_{i:j}) + K \geq F(j), \tag{2.27}$$

for $i < j$, then for any future $k > j$, i can never be the optimal last change point. We can thus add a pruning step at each iteration of the OP algorithm, that looks for past time indices i that verify (2.27) with the current time index $j = s$. This time index can then be ignored when solving the minimization (2.25) in future iterations.

The hypothesis (2.26) is not very restrictive, as it holds for a wide range of classic cost functions, such as penalized likelihood or sum of squares. It is proved that in cases where the number of change points increases linearly with the number of samples T, the computational cost of PELT is $\mathcal{O}(T)$. Some other conditions are required, but they are not very restrictive. See Section 3.1 of [11] for more details. The worst-case complexity is when no time indices can be pruned. The complexity is then equivalent to the basic OP algorithm $\mathcal{O}(T^2)$.

Other methods A recent body of literature, of which PELT is a part, focuses on improving the complexity of the exact algorithms presented above. We can cite the pDPA algorithms, proposed in [46], which introduces a pruning rule in the SN algorithm. The type of pruning rule, that they call functional pruning, is very different from the inequality-based rule of PELT, and sees the cost of a segment as a function of its specific model parameter. It imposes some restrictions on the type of cost functions compatible, and can only detect changes in one parameter. It has an empirical complexity of $\mathcal{O}(n \log n)$, and the worst-case complexity is the one of the SN algorithm. In some cases, it has been shown to be faster than PELT on datasets with few change points. Some work in that direction is also proposed in [47], where the authors develop new pruning techniques by combining the ideas of PELT and pDPA, that result in the new algorithms FPOP and SNIP.

2.5.2 Approximate methods

Binary Segmentation Binary Segmentation (BS) is an approximate method originally introduced in [48] and often used in the literature as a substitute for exact methods. It is a greedy algorithm, that sequentially adds to the current estimate of the segmentation the one change point that reduces the most the global cost. It can be seen as a way to adapt a single change point detection method for the problem of multiple change point detection. Consider the segmentation of the time-series $x = \{\boldsymbol{x}_t\}_{t=1}^{T}$. Let us first define the problem $P(\boldsymbol{x}_{i:j})$ of finding the change point within $\boldsymbol{x}_{i:j}$ that will reduce the most the global segmentation cost on segment $\boldsymbol{x}_{i:j}$,

$$P(\boldsymbol{x}_{i:j}) : z_{\boldsymbol{x}_{i:j}} = \min_{i<t<j-1} c(\boldsymbol{x}_{i:t}) + c(\boldsymbol{x}_{t:j}). \tag{2.28}$$

Denote $t^*_{\boldsymbol{x}_{i:j}}$ the position of this optimal change point, and $\mathcal{T}^k$ the current segmentation at iteration k. At the first iteration, the current segmentation consists of the single segment $\boldsymbol{x}_{1:T}$

($\mathcal{T}^1 = \emptyset$). The first iteration consists of dividing the single segment $\boldsymbol{x}_{1:T}$ in two. This is done by estimating the position $t^*_{\boldsymbol{x}_{1:T}}$ of the best change point to introduce by solving $P(\boldsymbol{x}_{1:T})$, and adding it to the current segmentation for the next iteration ($\mathcal{T}^2 = \mathcal{T}^1 \cup \{t^*_{\boldsymbol{x}_{1:T}}\}$). At iteration $k > 1$, the best change point to introduce is estimated within every segment of the current segmentation by solving $P(\boldsymbol{x}_{t_i:t_{i+1}})$, $\forall\, i = 0, \ldots, |\mathcal{T}^k|$. Among all the segments, we divide the one for which the addition of the corresponding new change point reduces the most the global cost. This is done by solving

$$\max_{i=0,\ldots,|\mathcal{T}^k|} c(\boldsymbol{x}_{t_i:t_{i+1}}) - z_{\boldsymbol{x}_{t_i:t_{i+1}}}. \tag{2.29}$$

Finally, the corresponding change point $t^*_{\boldsymbol{x}_{t^*_i:t^*_{i+1}}}$ is added to the current segmentation for the next iteration, $\mathcal{T}^{k+1} = \mathcal{T}^k \cup \{t^*_{\boldsymbol{x}_{t^*_i:t^*_{i+1}}}\}$. The algorithm iterates until a criterion is met.

Assuming the costs can be computed or accessed in $\mathcal{O}(1)$, the complexity of this method is $\mathcal{O}(T \log T)$. This gain in computational complexity is at the expense of optimality. The two main drawbacks of this method is that change points are estimated from non-homogeneous segments, and that the position of the estimated change points depend on the positions of the previous ones. Change points that are close to each other are especially imprecisely detected.

This method can be used for solving both the penalized (2.4) and the constrained (2.3) version of the problem, depending on the stopping criterion. For the constrained problem, the algorithm is stopped when the right number of change points is reached. In case the true number of change points is unknown, the criterion can be designed to be equivalent to the penalty term, or other strategies can be used.

Sliding adjacent windows Like BS, the Sliding Adjacent Windows (SAW) method can be seen as an adaptation of a single change point detection method to the problem of multiple change point detection. It is based on a measure of the discrepancy between two consecutive segments. The definition of this discrepancy measure relies on the cost function $c : P_{\boldsymbol{x}} \mapsto \mathbb{R}$ defined in Section 2.4. This cost function can itself use any appropriate underlying model for the time-series. The discrepancy function is defined as

$$d(\boldsymbol{x}_{i:t}, \boldsymbol{x}_{t:j}) = c(\boldsymbol{x}_{i:j}) - [c(\boldsymbol{x}_{i:t}) + c(\boldsymbol{x}_{t:j})]. \tag{2.30}$$

The intuition is that, when the index t is close to the position of a true change point, the two segments $\boldsymbol{x}_{i:t}$ and $\boldsymbol{x}_{t:j}$ are dissimilar. In this case, the segment $\boldsymbol{x}_{i:j}$ is not homogeneous, which should lead to a high cost $c(\boldsymbol{x}_{i:j})$. On the other hand, the two segments taken individually should each be rather homogeneous, meaning low values of their costs $c(\boldsymbol{x}_{i:t})$ and $c(\boldsymbol{x}_{t:j})$.

These two effects combined ensure that the discrepancy value $d(\boldsymbol{x}_{i:t}, \boldsymbol{x}_{t:j})$ is high when the index t is close to a true change point.

The SAW method consists of computing the discrepancy function for pairs of segments of constant length L, that are sliding along the time-series. Once these discrepancies are computed, we can plot the curve $d(\boldsymbol{x}_{t-L:t}, \boldsymbol{x}_{t:t+L})$ as a function of $t = L, \dots, T - L$. We expect this curve to exhibit peaks centered around the true change points. The estimated change points can then be computed by performing a peak search procedure or applying a threshold.

The choice of the length L of the windows is important in order to obtain good results. It should be chosen smaller than the length of the shortest segment in the true segmentation, so that the two segments are homogeneous when located around change points. However, it should not be chosen too small, so that each window contain enough samples to be statistically significant when computing their associated costs.

The advantage of this approximate method is again its low computational cost. Its complexity is linear in the number of samples, assuming the discrepancy values can be computed or accessed in $\mathcal{O}(1)$.

Bottom-up approach The bottom-up approach is the counterpart of the Binary Segmentation method. Instead of starting with an empty segmentation and sequentially adding one optimal change point at a time, the initial segmentation is one with more candidate change point than true change point. A classical approach is to start with a segmentation consisting of a given percentage of the points in the time-series, equally spaced. Then change points are sequentially eliminated until a criterion is met. At each iteration, the change point eliminated is the one with the lowest value of discrepancy between the two segments it separates. The measure of discrepancy is the one defined for the SAW method (2.30).

Assuming the discrepancy values can be computed or accessed in $\mathcal{O}(1)$, the computational cost of this method is linear in the number of candidates. However, this gain in computational complexity is a the expense of optimality, as this method suffers form a few drawbacks. First, if a true change point is not among the candidate change points selected at the beginning, it will never be estimated precisely. Moreover, the first iterations usually tends to be unstable, as the segments on which the discrepancies are computed are rather small, impacting statistical significance.

Other methods As mentioned in the introduction of [46], another common idea for approximate fast algorithms is to develop a fast heuristic for identifying a restricted set of candidates change points, on which to run an exact algorithms.

2.5.3 Reducing the search space

In order for the algorithms presented above to run in shorter times, some strategies are commonly used in the literature to reduce the space of admissible segmentations. These strategies use prior knowledge on the time-series and the performance requirement of the application. We present here two simple strategies that can be used with any of the exact or approximate algorithms presented above, and are implemented in the Python package for change point detection proposed in [49].

Minimum segment length In many application, prior knowledge is available about the minimum time separating two consecutive change points. Let S be the value of this minimum spacing between change points in number of samples. The strategy simply consist in adding the constraint $t_{k+1} - t_k \geq S, \ \forall k = 0, \dots, m$ to the optimization problem being solved. All the algorithms presented above can be modified in a simple way in order to account for this new constraint.

Resolution Depending on the application, we might not need to estimate the position of the change points with a resolution of one sample. In such cases, it is possible to only consider a fraction of the time indices of the time-series as potential change point candidates. For example, one can choose to consider as candidates only the indices that are multiples of a given integer R, called the resolution parameter. An admissible segmentation is then defined as $\mathcal{T} = \{t_k\}_{k=1}^{m} \subseteq \{R * i \mid i = 1, \dots, \lfloor \frac{T}{R} \rfloor\}$. All the algorithms presented above can also be easily modified in order to take this restriction on the candidate change points into account. Note that all the samples of the time-series are still used for computing the costs, and only the set of candidate change points have been reduced.

2.6 Estimating the number of change points

In cases where the number of change points m^* is known a priori, change point detection is simply performed by solving the constrained problem (2.3). However, in most practical applications, no prior knowledge about the number of change points is available. Two strategies exist for estimating that number :

- jointly estimating the number of change points and their positions by solving the penalized problem (2.4); or
- computing the optimal segmentation with different numbers of change points by solving

the constrained problem (2.3) multiple times, and then selecting one of these segmentations using an appropriate criterion.

The distinction between these two strategies is not perfectly clear. Indeed when the penalty term only depends on the number of change points, solving the penalized problem can always be done using the second strategy, with the criterion being the penalized cost in (2.4). For some complex penalty functions, it actually is the only way to solve (2.4). On the contrary, depending on the criterion used, the second strategy cannot always be expressed as a penalized problem.

In the following, we discuss different methods for estimating the number of change points, organized by the type of penalty they correspond to. We also discuss methods for computing optimal segmentations with different numbers of change points. We finish with a peculiar method, not based on penalty.

2.6.1 Linear penalty

The use of a penalty term linear in the number of change points is very common in the literature. This is more formally written as

$$pen_{lin}(\mathcal{T}) = \beta|\mathcal{T}|, \tag{2.31}$$

where β is a smoothing parameter. Thanks to the additive property (2.1) of the criterion $V(\mathcal{T}, \boldsymbol{x})$, the penalty term can be distributed into to segment specific costs as in (2.22), usually allowing for fast algorithms such as Optimal Partitioning.

The choice of the smoothing parameter is very important, as it controls the number of change points that will be estimated. A low value will favor segmentations with many change points, whereas a high value will favor segmentations with few change points. In the following, we describe different methods for choosing this parameter.

Model selection One of the reasons why the linear penalty is so widespread in the literature is that it generalizes some classic model selection methods, the most notable of which is the Bayesian information criterion (also known as the Schwarz Information Criterion).

The Bayesian information criterion (BIC), originally introduced in [50], is a criterion for selecting among a finite set of models of some data $\mathcal{D}$, the one that will best generalize to other data drawn from the same distribution. The preferred model is the one with the lowest

BIC value, which is defined as

$$BIC = \log(n)k - 2\log\widehat{\mathcal{L}}, \tag{2.32}$$

where $\widehat{\mathcal{L}}$ is the maximum of the likelihood of the model on the data $\mathcal{D}$, k is the number of parameters of that model, and n is the number of observations contained in $\mathcal{D}$. It is designed to reach a trade-off between the goodness of fit of the model on data $\mathcal{D}$, represented by the log-likelihood $\log\widehat{\mathcal{L}}$, and its complexity, represented by the number of its parameters k. The value of the coefficient $\log n$ is justified by a Bayesian argument, hence the name of the criterion.

Estimating the number of change points within a given time-series can be framed as a selection problem between different piecewise stationary models with different numbers of segments. A compromise has to be achieved between the goodness of fit and the complexity of the model. When using the model (2.5), the criterion $V(\mathcal{T}, \boldsymbol{x})$ represents the negative log-likelihood $-\log\widehat{\mathcal{L}}$ of the piecewise model associated to $\mathcal{T}$ on the data $\boldsymbol{x}_{1:T}$. The number of parameters of the model is simply the number of its segments $m = |\mathcal{T}|$ multiplied by the number of parameters of each segment specific model p. Hence using the BIC for estimating the number of change points is equivalent to setting the smoothing parameter as

$$\beta = \frac{p}{2}\log T, \tag{2.33}$$

where T is the number of samples in the time-series, and p is the number of parameters of each segment specific model. This idea of using the BIC to estimate the number of change points has first been introduced in [18] for the detection of changes in mean within samples normally distributed with constant variance.

Other model selection criteria can be used for estimating the number of change points. For instance the Akaike information criterion (AIC), introduced in [51], is defined as

$$AIC = 2k - 2\log\widehat{\mathcal{L}}, \tag{2.34}$$

where $\widehat{\mathcal{L}}$ is the maximum of the likelihood of the model on the data $\mathcal{D}$, and k is the number of parameters of that model. The corresponding value for the smoothing parameter is

$$\beta = p. \tag{2.35}$$

The Hannan–Quinn information criterion (HQC), introduced in [52] is an other option. It is defined as

$$HQC = 2k\log(\log n) - 2\log\widehat{\mathcal{L}}, \tag{2.36}$$

where $\widehat{\mathcal{L}}$ is the maximum of the likelihood of the model on the data $\mathcal{D}$, k is the number of parameters of that model, and n is the number of observations contained in $\mathcal{D}$. The corresponding value for the smoothing parameter is

$$\beta = p\log(\log T). \tag{2.37}$$

Other methods for choosing the smoothing parameter In Section 5 of [22], the authors observe that using the BIC largely overestimates the number of change points on a real-world time-series. They argue that the inadequacy of the BIC is due to the fact that the model used is over-simplistic for the data considered. This intuition seems to make sense. Indeed, if the model is over-simplistic for the data, the value of the goodness of fit measure to be minimized will be rather high, requiring in turn a higher penalty value to counteract the overfitting effect. The authors of [53] also argue in the introduction that the violation of assumptions such as Gaussianity or independence often lead to poor performances of methods based on the classic model selection criteria for estimating the number of change points on real data. Moreover the methods based on BIC and AIC only work with the model (2.5), where the criterion being optimized represent the likelihood of the piecewise model on the time-series. This shows that the classic model selection approaches have a limited range of applications, and it can be interesting to explore other means of tuning the smoothing parameter β.

A first alternative is to use a procedure based on cross-validation [54], for example using the reconstruction error. In the limited cases where some annotated time-series are available for testing, supervised approaches can also be used such as [53, 55].

These methods usually require the computation of all the optimal segmentations (with different numbers of change points) for a range of smoothing parameter $\beta \in [\beta_0, \beta_1]$. Arbitrarily sampling some values of β and solving the penalized problem for all of them is not efficient and can get very expensive in terms of computation. A much better approach, called the Changepoints for a Range Of PenaltieS (CROPS) algorithm, is proposed in [16]. This algorithm yields all the segmentations that are optimal for any value of the parameter $\beta \in [\beta_0, \beta_1]$, while requiring to solve the linearly penalized problem (2.4) a limited amount of times (which is linear in the difference between the number of change points corresponding to β_0 and β_1). It uses the linear relationship between the penalty value and the penalized cost of a given segmentation, as well as the link between the penalized and constrained problems, to efficiently explore to penalty range.

2.6.2 Other types of penalties

Complex penalty terms We describe here two complex penalty terms that have been derived from theoretical considerations. They both assume the univariate mean-shift model for the time-series (corresponding to the cost function c_{L_2}).

- In [56], a modified BIC criterion is introduced, that is supposed to be more robust to irregularities in the likelihood function. It is derived by asymptotic approximation of the Bayes factor. It is similar to the classic BIC as the term representing the goodness of fit is still the log-likelihood, but the term for penalizing model complexity is different.

  It can be formulated as the penalized problem (2.4) with the penalty term

  $$pen_{mBIC}(\mathcal{T}) = 3|\mathcal{T}| \log T + \sum_{k=0}^{m+1} \log\Big(\frac{t_{k+1} - t_k}{T}\Big), \tag{2.38}$$

  where T is the number of samples in the time-series. This penalty term depends on the number of change points as well as their positions, and can be intuitively interpreted as favoring the segmentations with evenly spaced change points.

- In [57], the following penalty term is derived from a model selection procedure

  $$pen_{Leb}(\mathcal{T}) = \frac{|\mathcal{T}| + 1}{T} \sigma^2 \Big[a_1 \log\Big(\frac{|\mathcal{T}| + 1}{T}\Big) + a_2\Big], \tag{2.39}$$

  where $a_1 > 0$ and $a_2 > 0$ are parameters, and σ^2 is the noise variance. The strength of this procedure is that, contrary to every other model selection procedures presented here, its theoretical justification does not rely on the asymptotic setting, where the number of samples is assumed to tend to infinity. It should thus perform well in practical cases.

The drawback of these complex penalty terms is that directly solving the penalized problem (2.4) is intractable. In practice, the optimal segmentations with m change points are computed using (2.3) for $m = 1, \dots, M_{MAX}$, where M_{MAX} is an upper bound on the number of change points, and the one that minimizes the penalized cost is selected.

Adaptive choice of penalization parameter A method for estimating the number of change points is proposed in [58] for cases where the penalty term is of the form $pen(\mathcal{T}) =$

$\beta f(|\mathcal{T}|)$, where β is a penalization parameter and $f(|\mathcal{T}|)$ is a function that increases with the number of change points in $\mathcal{T}$, and only depends on this number.

Let c_k the optimal value of the constrained problem with k change points, $p_k = f(|\mathcal{T}| = k)$ the penalty value for a segmentation with k change points, and $k(\beta)$ the number of change points in the optimal segmentation with penalization coefficient β. The following proposition is proved in [58].

Proposition 2.2. *There exists a sequence* $1 = k_1 < k_2 < \dots$ *and a sequence* $\infty = \beta_0 > \beta_1 > \dots$ *with*

$$\beta_i = \frac{c_{k_i} - c_{k_{i+1}}}{p_{k_{i+1}} - p_{k_i}}, \quad i \geq 1, \tag{2.40}$$

such that $k(\beta) = k_i, \quad \forall \beta \in (\beta_i, \beta_{i-1})$.

The authors argue that the selected segmentation should be the most stable one, in the sense that it should not strongly depend on the penalization coefficient β. However, directly choosing the number of change points k_i corresponding to the interval $[\beta_i, \beta_{i-1}]$ of greatest length $l_i = \beta_{i-1} - \beta_i$ tend to underestimate the number of change points, therefore the solution they propose is to choose the greatest k_i such as $l_i \gg l_j, \quad \forall j > i$.

This criterion has a more visual interpretation when plotting the evolution of the unpenalized cost c_k as a function of the penalty value p_k. Equation 2.40 tells us that the slope between points (p_{k_i}, c_{k_i}) and $(p_{k_{i+1}}, c_{k_{i+1}})$ is $-\beta_i$. The heuristic thus consist in selecting the point on this plot where the values c_k cease to decrease significantly. Indeed, the length l_i is loosely equivalent to the second derivative, and we are looking for the point of maximum curvature or in other words a break in the slope of the curve.

Note that the linear penalty is a special case, where $f(|\mathcal{T}|) = |\mathcal{T}|$. We have $p_k = k$, and the graphical interpretation is performed on the plot of the unpenalized cost c_k as a function of the number of change points k.

2.6.3 Efficiently computing optimal segmentations with different numbers of change points

One of the strategies for estimating the number of change points is to first compute multiple optimal segmentations with different number of change points, and then discriminate among them using a given criterion. To that end, it is useful to have efficient algorithms for computing those multiple optimal segmentations. The next two sections discuss such algorithms in the cases of approximate and optimal methods.

Approximate methods For the iterative algorithms, such as top-down or bottom-up algorithms, it is fairly cheap to compute segmentations with different number of change points. Indeed, at each iteration, change points are added or remove to the current estimated segmentation until a stopping criterion is met. It is thus enough to just store those segmentations at each iteration. For the SAW method, computing the estimated segmentations with different numbers of change points is a matter of running the peak search method with different parameters. The computational cost of the peak search procedure is usually rather cheap, so running it multiple times is not a problem, at least for offline methods.

Exact methods When using the constrained problem (2.3), the SN algorithm itself is able to compute all the optimal segmentations with $m = 1, \dots, M_{max}$ number of change points. Its computational complexity is $\mathcal{O}(M_{max}T^2)$, linear in the upper bound on the number of change points M_{max}, and quadratic in the length of the time-series T.

Another alternative is to use the CROPS algorithm discussed on page 24 and introduced in [16], combined with the penalized problem (2.4) with linear penalty (2.31). This algorithm yields all the segmentations that are optimal for any value of β within a given range $[\beta_0, \beta_1]$. Let $m(\beta)$ be the number of change points in the optimal segmentation with smoothing parameter β. CROPS requires to solve the penalized problem (2.4) a number of times linear in the difference $\Delta_{\beta_0\beta_1} = m(\beta_0) - m(\beta_1)$. Moreover, provided the number of change points is linear in the length of the time-series T, the PELT algorithm is able to solve the penalized problem in an amount of time linear in the number of samples T. Under these conditions, the CROPS algorithm has a computational complexity $\mathcal{O}(\Delta_{\beta_0\beta_1}T)$, and is thus a computationally efficient way of computing segmentations with different numbers of change points. Note that it does not in general yield *all* the segmentations with *all* number of change points between $m(\beta_1)$ and $m(\beta_0)$. Indeed, a given optimal segmentation for the constrained problem with m change points, $m(\beta_1) \leq m \leq m(\beta_0)$, might never be optimal for the penalized problem for any value of the smoothing parameter $\beta \in [\beta_0, \beta_1]$. However, a good portion of those segmentations are usually recovered.

2.6.4 Method not based on penalty

In [59], a generalization of the SN method of [44] is derived, which computes the N most probable segmentations (i.e. with lowest sum of costs). The computational complexity of this algorithm is $\mathcal{O}(NKT^2)$, where $N \geq 1$ is the number of segmentations computed, K is the number of change points in the segmentations and T is the number of samples in the time-series. This algorithm is interesting to compare the N "near-optimal" segmentations among themselves. These N segmentations can be aggregated into one, by only keeping the most likely change points, where a change point is deemed likely if it is present in many of the top segmentations. When using this method, the number of change points is automatically estimated, without the intervention of a penalty term.

2.7 Other methods

2.7.1 Cumulative Sum (CUSUM)

Theoretical justification The Cumulative Sum (CUSUM) algorithm is one of the first methods developed to tackle the problem of change point detection. It is a simple algorithm, making it very useful in online applications. It has first been proposed by E.S. Page in [13], and a more rigorous interpretation has then been made in [60] and [61]. We will use it to detect an abrupt change in the mean of the signal.

Let $\boldsymbol{x} = (x_n)_{n=0}^{k}$ be a time-series with independent and identically distributed (i.i.d.) samples x_n modeled as random variables X_n following a given probability distribution $p(x_n, \theta)$ where θ is a parameter. We want to decide whether or not the signal contains an abrupt change of the parameter θ, and if so, at what time n_c this change occurs. Let θ_0 be the value of the parameter before change, and θ_1 it's value after n_c, if there is a change. We denote H_0 the hypothesis of no change occurring, and H_1 the hypothesis of a change occurring at time n_c. Under H_0, the likelihood of the signal from the first sample x_0 to the current sample x_k is

$$\mathcal{L}(\boldsymbol{x}|H_0) = \prod_{n=0}^{k} p(x_n, \theta_0).$$

Under H_1, the likelihood becomes

$$\mathcal{L}(\boldsymbol{x}|H_1) = \prod_{n=0}^{n_c-1} p(x_n, \theta_0) \prod_{n=n_c}^{k} p(x_n, \theta_1).$$

The idea of Page was to sequentially apply a likelihood ratio test, in order to decide between H_0 and H_1 after each sample. Once H_1 has been decided, we also want to define an estimator for the time of change n_c. We define the log-likelihood ratio $\mathcal{L}_{\boldsymbol{x}}$ as

$$\mathcal{L}_{\boldsymbol{x}} = \log \frac{\mathcal{L}(\boldsymbol{x}|H_1)}{\mathcal{L}(\boldsymbol{x}|H_0)}.$$

Note that the log-likelihood ratio $\mathcal{L}_{\boldsymbol{x}}$ is negative as long as H_0 is true. If H_1 becomes true, then $\mathcal{L}_{\boldsymbol{x}}$ becomes positive. Therefore we will decide H_1 once $\mathcal{L}_{\boldsymbol{x}} > h$, where h is a threshold to be set by the user. At current sample x_k, the expression becomes

$$\mathcal{L}_{\boldsymbol{x}}(k, n_c) = \sum_{n=n_c}^{k} \log \frac{p(x_n, \theta_1)}{p(x_n, \theta_0)}.$$

However, this expression depends on the unknown change time n_c. The solution is to replace n_c by it's maximum likelihood estimate. Therefore we define what is called the generalized log-likelihood ratio $G_{\boldsymbol{x}}$ as

$$G_{\boldsymbol{x}}(k) = \max_{1 \leq n_c \leq k} \mathcal{L}_{\boldsymbol{x}}(k, n_c) = \max_{1 \leq n_c \leq k} \sum_{n=n_c}^{k} \log \frac{p(x_n, \theta_1)}{p(x_n, \theta_0)}.$$

And the maximum likelihood estimate of the change time is

$$\hat{n}_c = \underset{1 \leq n_c \leq k}{\operatorname{argmax}} \sum_{n=n_c}^{k} \log \frac{p(x_n, \theta_1)}{p(x_n, \theta_0)}.$$

In the final algorithm, we want to compute $G_{\boldsymbol{x}}$ at each new sample. For that, let us define the instantaneous log-likelihood ratio at sample x_n,

$$s_n = \log \frac{p(x_n, \theta_1)}{p(x_n, \theta_0)}.$$

In practice, we will be able to compute the generalized log-likelihood $G_{\boldsymbol{x}}$ through the cumulative sum of the instantaneous log-likelihood from the first sample x_0 to the current one x_k,

$$S_k = \sum_{n=0}^{k} s_n.$$

Indeed, we have $\mathcal{L}_{\boldsymbol{x}}(k, n_c) = S_k - S_{n_c - 1}$, which gives

$$G_{\boldsymbol{x}}(k) = S_k - \min_{1 \leq n_c \leq k} S_{n_c - 1},$$

$$\hat{n_c} = \underset{1 \leq n_c \leq k}{\operatorname{argmin}} S_{n_c - 1}.$$

Notice that the decision function $G_{\boldsymbol{x}}(k)$ only depends on the current value of S_k and its current minimum value. Moreover, in the algorithm, we can compute S_k through

$$S_k = S_{k-1} + s_k.$$

As the decision function $G_{\boldsymbol{x}}$ is compared to a positive value h, we can rewrite it as

$$G_{\boldsymbol{x}}(k) = \max(0, G_{\boldsymbol{x}}(k-1) + s_k).$$

This yields the CUSUM algorithm, which consists of computing s_k at each new sample, and using it to compute $G_{\boldsymbol{x}}(k)$. This quantity should stay close to 0 as long as no change occurs. If its value cross the limit h set by the user, a change is detected, and the estimated time of the change is the timestep right after the last one where $G_{\boldsymbol{x}}(k) = 0$.

Application for detection of a change in mean in an i.i.d. Gaussian signal CUSUM is very often used to detect a change in the mean of a signal. We assume the samples of our signal to be i.i.d. and to follow a Gaussian distribution with mean μ and variance σ^2. Moreover, the signal possibly undergoes a change in mean from μ_0 to μ_1 at time n_c. The probability density of each sample X_n is written as $p(x_n, \mu) = \frac{1}{\sigma\sqrt{(2\pi)}} \exp(-\frac{(x_n-\mu)^2}{2\sigma^2})$, where μ takes the value μ_0 or μ_1 depending of whether the sample is before or after the change. The instantaneous log-likelihood ratio can then be computed for any sample x_n by

$$s_n = \frac{\mu_1 - \mu_0}{\sigma^2}\left(x_n - \frac{\mu_0 + \mu_1}{2}\right).$$

Practical considerations To further simplify the algorithm, we can remove the multiplicative constant in front of s_n. This doesn't change the behavior of the algorithm, provided we modify the value chosen for h accordingly. The accumulated value s_n then becomes $s_n = x_n - \frac{\mu_0+\mu_1}{2}$.

Another problem is that the computation of s_n requires prior knowledge of the mean of the signal before (μ_0) and after (μ_1) the change. One way to solve this problem is by replacing those values by their maximum likelihood estimates [62]. However, this make it impossible to use the recursive form of the algorithm, and make the complexity of the algorithm grow with the number of samples. Another solution often used in practice and that allow us to use the recursive form of the algorithm is to ask the user to set *a priori* the value of the parameter after change relative to the value before change. Indeed, if we define $\delta = \mu_1 - \mu_0$, then the instantaneous log-likelihood can be rewritten as $s_n = x_n - \mu_0 - \frac{\delta}{2}$. The user can set the value for δ by think of it as the minimal shift that one wants to be able to detect. The problem of setting μ_0 remains, although this is less concerning, as it can be estimated from the samples from x_0 to x_k, and if using recursive estimators the impact on the complexity is minimal.

In order to choose the value of the threshold h, one can consider a performance criteria called

Avergage Run Length (ARL). This is defined as the expected number of samples before a change is detected by the algorithm

$$ARL = \mathop{\mathbb{E}}_{\theta}[N_d],$$

where $N_d = n_d - n_c$, n_d being the time when the change has been detected. The values of the ARL in two different cases are especially interesting :

- when $\theta = \theta_0$, $ARL_0 = \mathbb{E}_{\theta_0}[N_d]$ is the expected time before the algorithm detects a change, under the assumption that no change has actually occurred. It can be interpreted as the time before the algorithm produces a false positive.
- when $\theta = \theta_1$, $ARL_1 = \mathbb{E}_{\theta_1}[N_d]$ is the expected time before the algorithm detects a change, under the assumption that a change has actually occurred. It can be interpreted as the delay in the detection of an actual change.

In [13, 60], a relationship between the threshold value h and the ARL is given. The goal for the user is to set h low enough so that the algorithm detects a change quickly (short ARL_1), but high enough to limit the frequency of false positive (long ARL_0). In the offline case, having a short ARL_1 is less important, and we can afford having a high value for h.

It is also important to notice that the algorithm exposed above only works for an increase in the mean of the signal. It is therefore called a one-sided algorithm. To be able to detect both increases and decreases, a very common solution is to use two one-sided algorithms : one for detecting a change in each direction. The only difference between the two algorithms is the expression of the instantaneous log-likelihood. If we call the one corresponding to an increase s_n^i and the one corresponding to a decrease s_n^d, their expression are :

$$s_n^i = x_n - \mu_0 - \frac{\delta}{2} \quad \text{and} \quad s_n^d = x_n - \mu_0 + \frac{\delta}{2}$$

The expression of the ARL as a function of the threshold h (which can take different values for each direction) is modified in the case of the two-sided CUSUM, but it is once again given in [13, 60].

As a final remark, the algorithm as defined by [13] stops once a change has been detected. In his article, the author had in mind an application to a production line, where an action had to be taken immediately after a change is detected. In the case where the goal is to do multiple change point detection, it is very common in the literature to just restart the algorithm once a change has been detected, in order to detect the future ones.

CHAPTER 3 METHODOLOGY

In this work, we propose a new approach to change point detection that consists in maximizing the discrepancy of the statistical properties between consecutive segments of a segmentation $\mathcal{T}$. This is by opposition to the methods from the literature, which are designed to maximize the homogeneity within individual segments. In the previous chapter we have seen that such methods usually consist in minimizing an objective function over a set $\mathcal{S}_{\boldsymbol{x}}$ of feasible segmentations of the time-series $\boldsymbol{x}$

$$\min_{\mathcal{T} \in \mathcal{S}_{\boldsymbol{x}}} V(\mathcal{T}, \boldsymbol{x}). \tag{3.1}$$

The objective function V associates to a segmentation $\mathcal{T} \in \mathcal{S}_{\boldsymbol{x}}$ a global cost that consist of a sum over the segments $\boldsymbol{x}_{t_k:t_{k+1}}$ of segment specific costs $c(\boldsymbol{x}_{t_k:t_{k+1}})$. This is formally expressed as

$$V(\mathcal{T}, \boldsymbol{x}) = \sum_{k=0}^{m} c(\boldsymbol{x}_{t_k:t_{k+1}}). \tag{3.2}$$

The function $c(\cdot)$ associates a cost $c(\boldsymbol{x}_{i:j})$ to any segment $\boldsymbol{x}_{i:j}$, and is derived from a model representing the time-series $\boldsymbol{x}$. It measures the goodness of fit of the data within the segment $\boldsymbol{x}_{i:j}$ to this model, usually through prediction error or likelihood.

We propose instead to replace this function $c(\cdot)$ representing goodness of fit within individual segments by a score function $s(\cdot)$ representing the difference in statistical properties between two segments, defined for pairs of consecutive segments of the form $(\boldsymbol{x}_{t_i:t_j}, \boldsymbol{x}_{t_j:t_k})$ with $i < j < k$. It is derived from a predictive model of the time-series $\boldsymbol{x}$, by taking the prediction error on the right segment $\boldsymbol{x}_{t_j:t_k}$ of the model that has been estimated on the left segment $\boldsymbol{x}_{t_i:t_j}$. The new objective function we propose then associates to a segmentation $\mathcal{T} \in \mathcal{S}_{\boldsymbol{x}}$ a global score that consist of a sum, over the pairs of consecutive segments $(\boldsymbol{x}_{t_{k-1}:t_k}, \boldsymbol{x}_{t_k:t_{k+1}})$ of $\mathcal{T}$, of the scores $s(\boldsymbol{x}_{t_{k-1}:t_k}, \boldsymbol{x}_{t_k:t_{k+1}})$. This is formally expressed as

$$V(\mathcal{T}, \boldsymbol{x}) = \sum_{k=1}^{m} s(\boldsymbol{x}_{t_{k-1}:t_k}, \boldsymbol{x}_{t_k:t_{k+1}}). \tag{3.3}$$

Estimating the segmentation of the time-series $\boldsymbol{x}$ then consist in maximizing this objective function over a set $\mathcal{S}_{\boldsymbol{x}}$ of feasible segmentations of the time-series $\boldsymbol{x}$

$$\max_{\mathcal{T} \in \mathcal{S}_{\boldsymbol{x}}} V(\mathcal{T}, \boldsymbol{x}). \tag{3.4}$$

The intuition is that a model estimated on a segment $\boldsymbol{x}_{t_{k-1}:t_k}$ will capture the statistical properties of the data within that segment. Then, if the statistical properties of the data in the next segment $\boldsymbol{x}_{t_k:t_{k+1}}$ are different, we expect the predictive error of this model to be high. This means that the value of the score $s(\boldsymbol{x}_{t_{k-1}:t_k}, \boldsymbol{x}_{t_k:t_{k+1}})$ is high as well. By maximizing the global score (3.3) over the set of all possible segmentations $\mathcal{S}_{\boldsymbol{x}}$, we should obtain a segmentation $\widehat{\mathcal{T}} = \{\hat{t}_k\}_{k=1}^{\hat{m}}$ in which the statistical properties of the time-series are different from one segment to the next. By definition of a change point, the indices $\hat{t}_k$ should thus be good estimates of the locations of the change points to be detected. In short, the best segmentation is the one that maximizes the sum of errors.

We call this new approach to CPD that we are proposing OTAWA for Optimal Two Adjacent Windows Algorithm. Note that we use the word score instead of costs, as the objective function is maximized rather than minimized. Moreover, because the scores depend on pairs of segments instead of individual segments, the algorithms from the literature are not able to handle this new objective function, and we will have to develop new ones.

In this chapter, we first expose different models for describing the time-series with their associated score function, then we detail two exact algorithms for solving both the linearly penalized and the constrained versions of the optimization problem (3.4). After that, we expose strategies for estimating the number of change points, and finally we analyze the computational complexity of the overall method.

3.1 Score function

The objective function (3.3) can be interpreted as the overall score of the segmentation $\mathcal{T}$. Its formulation is based on a score function $s : P_x \mapsto \mathbb{R}$, defined on the set $P_x = \{(\boldsymbol{x}_{t_i:t_j}, \boldsymbol{x}_{t_j:t_k}) \mid 1 \leq t_i < t_j < t_k \leq T\}$ of all the pairs of consecutive segments of a time-series $\boldsymbol{x}$. This score function characterizes the difference in statistical properties between two consecutive segments, and its definition is derived from a predictive model $\mathcal{M}$ of the time-series. Let $\mathcal{M}(\boldsymbol{x}_{t_i:t_j})$ be the model estimated on segment $\boldsymbol{x}_{t_i:t_j}$. Let $\hat{\boldsymbol{x}}^{\mathcal{M}}$ be the time-series of the prediction of model $\mathcal{M}$, meaning that $\hat{\boldsymbol{x}}_t^{\mathcal{M}}$ is the prediction of the model $\mathcal{M}$ at time $1 \leq t \leq T$. The score associated to the pair of segments $(\boldsymbol{x}_{t_i:t_j}, \boldsymbol{x}_{t_j:t_k})$ is

$$s(\boldsymbol{x}_{t_i:t_j}, \boldsymbol{x}_{t_j:t_k}) = D(\boldsymbol{x}_{t_j:t_k}, \hat{\boldsymbol{x}}_{t_j:t_k}^{\mathcal{M}(\boldsymbol{x}_{t_i:t_j})}), \tag{3.5}$$

where $D(\cdot, \cdot)$ is a distance function on time-series, that is used as a measure of the prediction error.

In plain English, this definition means that the score $s(\boldsymbol{x}_{t_i:t_j}, \boldsymbol{x}_{t_j:t_k})$ associated to the pair $(\boldsymbol{x}_{t_i:t_j}, \boldsymbol{x}_{t_j:t_k})$ of consecutive segments it the prediction error on the right segment $\boldsymbol{x}_{t_j:t_k}$ of the model $\mathcal{M}$, once it has been trained in the left segment $\boldsymbol{x}_{t_i:t_j}$. The choice of the model

$\mathcal{M}$ depends on prior knowledge about the time-series. In the following, we list different predictive models with their associated cost function.

3.1.1 Mean change in independent normally distributed samples

With this model, the samples of the time-series are assumed to be independent and to follow a multivariate normal distribution, with constant covariance matrix and piecewise constant mean. More formally, the model $\mathcal{M}_{iid,\boldsymbol{\mu}}$ assumes that the samples are i.i.d. random variables and follow the distribution

$$\boldsymbol{X}_t \sim \sum_{k=1}^{m^*+1} \mathcal{N}(\boldsymbol{\mu}_{k(t)}, \Sigma), \tag{3.6}$$

where $\mathcal{T}^* = \{t_k^*\}_{k=1}^{m^*}$ is the true segmentation, $k(t) = \min\{k \text{ s.t. } 1 \leq k \leq m^* \text{ and } t_k^* > t\}$ is the index of the segment containing the sample $\boldsymbol{X}_t$, Σ is the constant covariance matrix and $\boldsymbol{\mu}_k$ is the mean that changes abruptly at the time t_k^*.

For computing the score associated to the pair of segments $(\boldsymbol{x}_{t_{k-1}:t_k}, \boldsymbol{x}_{t_k:t_{k+1}})$ we first need to infer the model on the first segment $\boldsymbol{x}_{t_{k-1}:t_k}$. This simply consists in computing the sample mean within the segment $\boldsymbol{x}_{t_{k-1}:t_k}$ as $\hat{\boldsymbol{\mu}}_k = \frac{1}{t_k - t_{k-1}} \sum_{t=t_{k-1}}^{t_k-1} \boldsymbol{x}_t$. Once inferred on $\boldsymbol{x}_{t_{k-1}:t_k}$ the prediction of the model is simply $\hat{\boldsymbol{x}}_t^{\mathcal{M}_{iid,\boldsymbol{\mu}}(\boldsymbol{x}_{t_{k-1}:t_k})} = \hat{\boldsymbol{\mu}}_k \; \forall t = 1, \ldots, T$ and the score value is the error evaluated in terms of mean square error (MSE) of this prediction on the second segment $\boldsymbol{x}_{t_k:t_{k+1}}$

$$s_{iid,\boldsymbol{\mu}}(\boldsymbol{x}_{t_{k-1}:t_k}, \boldsymbol{x}_{t_k:t_{k+1}}) = \frac{1}{t_{k+1} - t_k} \sum_{t=t_k}^{t_{k+1}-1} \|\boldsymbol{x}_t - \hat{\boldsymbol{\mu}}_k\|_2^2, \tag{3.7}$$

where $\|\cdot\|_2$ is the Euclidean norm.

3.1.2 Mean change in independent normally distributed samples with margin

In most of the literature, the statistical properties of the time-series are assumed to be piecewise stationary, with *abrupt* changes at change points. However in practice, the transition of the statistical properties between two stationary segments might happen gradually over the course of multiple samples. The score function that we will detail here is similar to (3.7), but allows for transition periods between segments.

First we introduce a margin parameter M, that represent the length (in number of samples) of the transition periods between segments. The time-series is modeled as independent samples following a multivariate normal distribution with a constant covariance matrix. The mean however is assumed to be constant within segments except for the M first samples of every segment, where it can take arbitrary values. More formally, the samples are modeled as random variables such that :

$$\boldsymbol{X}_t \sim \sum_{k=1}^{m^*+1} \Big[\mathcal{N}(\boldsymbol{\mu}_k, \Sigma)\mathbb{1}(t^*_{k-1} + M \leq t < t^*_k) + \mathcal{N}(\boldsymbol{\mu}_t, \Sigma)\mathbb{1}(t^*_{k-1} \leq t < t^*_{k-1} + M)\Big] \qquad (3.8)$$

where $\mathcal{T}^* = \{t^*_k\}_{k=1}^{m^*}$ is the true segmentation, Σ is the constant covariance matrix, $\boldsymbol{\mu}_k$ is the segment or sample specific empirical mean and

$$k(t) = \begin{cases} t \quad \text{if } \exists k \text{ s.t. } 0 \leq t - t_k < M \\ \min\{k \text{ s.t. } 1 \leq k \leq m^* \text{ and } t^*_k > t\} \quad \text{otherwise} \end{cases}. \qquad (3.9)$$

For computing the score associated to the pair of segments $(\boldsymbol{x}_{t_{k-1}:t_k}, \boldsymbol{x}_{t_k:t_{k+1}})$, we first compute the sample mean within the segment $\boldsymbol{x}_{t_{k-1}+M:t_k}$ as $\hat{\boldsymbol{\mu}}_k^M = \frac{1}{t_k - t_{k-1} - M} \sum_{t=t_{k-1}+M}^{t_k - 1} \boldsymbol{x}_t$. Similarly to the case with no margin (3.7), the score is defined as

$$s_{iid,\boldsymbol{\mu},M}(\boldsymbol{x}_{t_{k-1}:t_k}, \boldsymbol{x}_{t_k:t_{k+1}}) = \frac{1}{t_{k+1} - t_k} \sum_{t=t_k}^{t_{k+1}-1} \|\boldsymbol{x}_t - \hat{\boldsymbol{\mu}}_k^M\|_2^2, \qquad (3.10)$$

where $\|\cdot\|_2$ is the Euclidean norm.

3.1.3 Vector Autoregressive (VAR) model

With this model, we assume that the time-series can be modeled as a piecewise VAR model with constant order p. The VAR model $\mathcal{M}_{VAR}$ is a generalization of an autoregressive model to the case of a multivariate time-series. Formally, the samples are modeled as random variables such that

$$\boldsymbol{X}_t = c_k + \sum_{i=1}^{p} A_{ik} \boldsymbol{X}_{t-i} + \varepsilon_t, \quad \forall t,\, t^*_k + p \leq t < t^*_{k+1}, \quad k = 0, \ldots, m^* \qquad (3.11)$$

where $A_{ik} \in \mathbb{R}^{d \times d}$ is the segment specific matrix of the regression coefficients between $\boldsymbol{x}_t$ and it's i-th lag $\boldsymbol{x}_{t-i}$, $c_k \in \mathbb{R}^d$ is the segment specific intercept and ε_t is an error term.

In order to compute the value of the score associated to a pair of segments $(\boldsymbol{x}_{t_{k-1}:t_k}, \boldsymbol{x}_{t_k:t_{k+1}})$, we first estimate the VAR model on the segment $\boldsymbol{x}_{t_{k-1}:t_k}$. Let us denote $\hat{c}_k$ and $\hat{A}_{ik}$ the parameters estimated on this segment. They can be estimated be any method, such as OLS, GLS, Lasso or Ridge. For any time-index t, the prediction of this model is

$$\hat{\boldsymbol{x}}_t^{\mathcal{M}_{VAR}(\boldsymbol{x}_{t_{k-1}:t_k})} = \hat{c}_k + \sum_{i=1}^{p} \hat{A}_{ik} \boldsymbol{x}_{t-i}. \qquad (3.12)$$

The score associated to the pair of segments is then the error evaluated in terms of MSE of this VAR model on the second segment $\boldsymbol{x}_{t_k:t_{k+1}}$

$$s_{VAR}(\boldsymbol{x}_{t_{k-1}:t_k}, \boldsymbol{x}_{t_k:t_{k+1}}) = \frac{1}{t_{k+1} - t_k} \sum_{t=t_k+p}^{t_{k+1}-1} \|\boldsymbol{x}_t - \hat{c}_k - \sum_{i=1}^{p} \hat{A}_{ik}\boldsymbol{x}_{t-i}\|_2^2, \tag{3.13}$$

where $\|\cdot\|_2$ is the Euclidean norm.

Note that it is very important to avoid overfitting, as it leads to noisy score values, hindering correct estimation of the positions of the change points. Unfortunately, overfitting occurs very often with this model. Indeed, on a segment $\boldsymbol{x}_{t_i:t_j}$, the number of samples available for estimating the model is $(t_j - t_i) - p$, where p is the order of the model. Each component of the time-series is described as a linear combination of the p lagged values of the d components, meaning that there are $p \times d$ regression coefficients to estimate. In order to avoid overfitting, we thus need $(t_j - t_i) - p \gg dp$. In practice this condition is very often violated, especially in cases where the dimension d of the time-series is high. To mitigate this problem, a first solution is to set a high minimum distance between change points $S \gg (d+1)p$. In most applications however this is not a realistic constraint. Therefore, the solution that should be preferred is to introduce regularization in the estimation of the parameters, for example by using Lasso or Ridge regression.

3.2 Optimization

Like all optimization-based methods in the literature, the method that consists in directly solving the optimization problem (3.4) is not able to estimate the correct number of change points. However, similarly to the solution proposed in the literature, we can derive both a penalized and a constrained version of the problem (3.4), that each give us control over the number of change points estimated

- The **constrained problem** forces the number of change points to a value m given a priori

$$\begin{aligned} \max_{\mathcal{T} \in \mathcal{S}_{\boldsymbol{x}}} \; & V(\mathcal{T}, \boldsymbol{x}) \\ \text{s.t.} \; & |\mathcal{T}| = m; \end{aligned} \tag{3.14}$$

- The **penalized problem** introduces a penalty term $pen(\mathcal{T})$ into the objective function, whose role is to penalize segmentations with high numbers of change points. This penalty term usually include a parameter that can be tuned in order to give it more or less importance, thereby giving control over the number of change points estimated.

$$\max_{\mathcal{T}\in\mathcal{S}_{\boldsymbol{x}}} V(\mathcal{T},\boldsymbol{x}) + pen(\mathcal{T}). \tag{3.15}$$

In this section, we propose two exact algorithms for solving both the constrained (3.14) and the penalized (3.15) versions of the problem. They are adaptations of the SN and OP algorithms respectively that have been presented in Section 2.5.1. In Section 3.3, we will discuss methods for estimating the number of change points.

First let us introduce a proposition that will be used in both algorithms.

Proposition 3.1. *Consider an optimization problem* $\max_{\mathcal{T}} V(\mathcal{T},\boldsymbol{x})$ *with an additive objective function as defined in* (3.3)*, which can be written as* $\max_{\mathcal{T}} \sum_{k=1}^{m} s(\boldsymbol{x}_{t_{k-1}:t_k}, \boldsymbol{x}_{t_k:t_{k+1}})$*. Let* $\mathcal{T}_t^* = \{t_k^*\}_{k=1}^{m^*} \in \mathcal{S}_{\boldsymbol{x}_{1:t}}$ *be an optimal segmentation on* $\boldsymbol{x}_{1:t}$*, that is satisfying* $V(\mathcal{T}_t^*,\boldsymbol{x}_{1:t}) \geq V(\mathcal{T},\boldsymbol{x}_{1:t})$, $\forall\mathcal{T} \in \mathcal{S}_{\boldsymbol{x}_{1:t}}$*, and assume that it includes the segment* $\boldsymbol{x}_{r:s}$, $1 \leq r < s \leq t$ *(i.e.* $\exists l \mid t_l^* = r$ *and* $t_{l+1}^* = s$*). Then the sub-segmentation* $\mathcal{T}_{r,s} = \{t_k^*\}_{k=1}^{l+1} = \{t_1,\dots,r,s\} \subseteq \mathcal{T}_t^*$ *of* $\mathcal{T}_t^*$ *is an optimal segmentation of* $\boldsymbol{x}_{1:s}$ *with* $\boldsymbol{x}_{r:s}$ *as the last segment, i.e.* $V(\mathcal{T}_{r,s},\boldsymbol{x}_{1:s}) \geq V(\mathcal{T},\boldsymbol{x}_{1:s})$, $\forall\mathcal{T} \in \{\mathcal{T} \in \mathcal{S}_{\boldsymbol{x}_{1:s}} \mid t_m = r\}$*.*

Proof. If $s = t$, meaning that $\boldsymbol{x}_{r:s}$ is the last segment of $\mathcal{T}_t^*$, the result is trivial as $\mathcal{T}_t^*$ is optimal on $\boldsymbol{x}_{1:t}$. Otherwise, let $\mathcal{T}_2 = \{t_k^*\}_{k=l+1}^{m^*} \in \mathcal{T}_t^*$ be the sub-segmentation of $\mathcal{T}_t^*$ on data $\boldsymbol{x}_{s:t}$. Note that $\mathcal{T}_2$ might only contain one segment if $\boldsymbol{x}_{r:s}$ is the second to last segment in $\mathcal{T}_t^*$. In this case, its corresponding scores is null ($V(\mathcal{T}_2) = 0$). We have $\mathcal{T}_{r,s} \cup \mathcal{T}_2 = \mathcal{T}_t^*$. Let $\mathcal{T}_1 \in \{\mathcal{T} \in \mathcal{S}_{\boldsymbol{x}_{1:s}} \mid t_m = r\}$ be any segmentation of the time-series $\boldsymbol{x}_{1:s}$ with $\boldsymbol{x}_{r:s}$ as last segment. $\mathcal{T}_1 \cup \mathcal{T}_2$ is a segmentation on $\boldsymbol{x}_{1:t}$, and as $\mathcal{T}_t^*$ is optimal on $\boldsymbol{x}_{1:t}$ we have :

$$\begin{gathered}
V(\mathcal{T}_t^*,\boldsymbol{x}_{1:t}) \geq V(\mathcal{T}_1 \cup \mathcal{T}_2,\boldsymbol{x}_{1:t}) \\
V(\mathcal{T}_{r,s} \cup \mathcal{T}_2,\boldsymbol{x}_{1:t}) \geq V(\mathcal{T}_1 \cup \mathcal{T}_2,\boldsymbol{x}_{1:t}) \\
V(\mathcal{T}_{r,s},\boldsymbol{x}_{1:s}) + s(\boldsymbol{x}_{r:s}, x_{s:t_{l+2}^*}) + V(\mathcal{T}_2,\boldsymbol{x}_{s:t}) \geq V(\mathcal{T}_1,\boldsymbol{x}_{1:s}) + s(\boldsymbol{x}_{r:s}, x_{s:t_{l+2}^*}) + V(\mathcal{T}_2,\boldsymbol{x}_{s:t}) \\
V(\mathcal{T}_{r,s},\boldsymbol{x}_{1:s}) \geq V(\mathcal{T}_1,\boldsymbol{x}_{1:s})
\end{gathered}$$

Hence $\mathcal{T}_{r,s}$ is an optimal segmentation of $\boldsymbol{x}_{1:s}$ with $\boldsymbol{x}_{r:s}$ as the last segment. □

Intuitively, this proposition states that if a segmentation of some time-series $\boldsymbol{x}_{1:t}$ is optimal, then any sub-segmentation with $\boldsymbol{x}_{r:s}$ as last segment is optimal among the segmentations of $\boldsymbol{x}_{1:s}$ with $\boldsymbol{x}_{r:s}$ as last segment.

3.2.1 Penalized optimization problem

In this section we are interested in solving the penalized problem (3.15). More specifically, we only consider the case of a linear penalty term $pen(\mathcal{T}) = \beta|\mathcal{T}|$. The objective function then becomes

$$V(\mathcal{T}, \boldsymbol{x}) = \sum_{k=1}^{m} s(\boldsymbol{x}_{t_{k-1}:t_k}, \boldsymbol{x}_{t_k:t_{k+1}}) - \beta|\mathcal{T}|, \tag{3.16}$$

where β is a smoothing parameter to be chosen by the user. In order to keep the additive property of the objective function, the linear penalty term can be distributed into the sum of scores

$$V(\mathcal{T}, \boldsymbol{x}) = \sum_{k=1}^{m} \Big[s(\boldsymbol{x}_{t_{k-1}:t_k}, \boldsymbol{x}_{t_k:t_{k+1}}) - \beta\Big]. \tag{3.17}$$

The algorithm we expose here is an adaptation of the OP algorithm exposed in Section 2.5.1 to our case where a score $s(\boldsymbol{x}_{t_i:t_j}, \boldsymbol{x}_{t_j:t_k})$ is associated to a pair of segments rather than to a single segment $\boldsymbol{x}_{t_i:t_j}$. Its goal is to compute the estimated segmentation of the time-series $\boldsymbol{x}$,

$$\widehat{\mathcal{T}} = \operatorname*{argmax}_{\mathcal{T} \in \mathcal{S}_{\boldsymbol{x}}} V(\mathcal{T}, \boldsymbol{x}), \tag{3.18}$$

that maximizes the objective function $V(\mathcal{T}, \boldsymbol{x})$. Like OP, it uses dynamic programming to efficiently explore the space of possible segmentations $\mathcal{S}_{\boldsymbol{x}}$.

The algorithm Let us denote $F(t)$ the score of the optimal segmentation of the time-series $\boldsymbol{x}_{1:t}$ and $G(r, s)$ the score of the optimal segmentation of $\boldsymbol{x}_{1:s}$ with $\boldsymbol{x}_{r:s}$ as last segment. By definition,

$$F(T) = \max_{1<t<T} G(t, T). \tag{3.19}$$

Moreover, thanks to Proposition 3.1, we have

$$G(s, t) = \max_{1 \le r < s} \{G(r, s) + s(\boldsymbol{x}_{r:s}, \boldsymbol{x}_{s:t}) - \beta\}. \tag{3.20}$$

Intuitively, Equation 3.20 shows that the optimal segmentation with $\boldsymbol{x}_{s:t}$ as last segment is easily computed if all optimal segmentation with $\boldsymbol{x}_{r:s}$ as last segment are known for all $r = 1, \ldots, s$. The dynamic programming approach then consists in successively computing $G(r, s)$ for all (r, s) such that $1 < r < s \le T$. We can then simply apply (3.19) to get $F(T)$. Provided the optimal segmentation with $\boldsymbol{x}_{s:t}$ as last segment is stored at each step, the estimated segmentation on the whole time-series $\boldsymbol{x}_{1:T}$ is easily retrieved. The complete algorithm is detailed in Algorithm 1. It has a computational complexity $\mathcal{O}(T^3)$ cubic in the number of samples T in the time-series.

Algorithm 1 Penalized OTAWA

```
Input: time-series $\boldsymbol{x}_{1:T}$, score function $s(\cdot)$, penalty value $\beta$
  Init $G$ and $S$ two $(T-2)$ by $(T-2)$ $2D$-array
  for $u = 2, \dots, T-1$ do
    Init $G[1, u] = 0$
    Init $S[1, u] = \{1, u\}$
  end for
  for $s = 2, \dots, T-1$ do
    for $t = s+1, \dots, T$ do
      $r^* = \text{argmax}_{1 \leq r < s}\{G[r, s] + s(\boldsymbol{x}_{r:s}, \boldsymbol{x}_{s:t}) - \beta\}$
      $G[s, t] = G[r^*, s] + s(\boldsymbol{x}_{r^*:s}, \boldsymbol{x}_{s:t}) - \beta$
      $S[s, t] = S[r^*, s] \cup t$
    end for
  end for
  $t^* = \text{argmax}_{1<t<T}\, G[t, T]$
Output: $S[t^*, T]$
```

Shortest path point of view This algorithm can also be seen as a shortest path problem on a carefully designed graph. We define a directed acyclic graph (DAG) $G = (V, E)$, with one vertex V_{t_i,t_j} for every possible segment $\boldsymbol{x}_{t_i:t_j}$ in the time-series. A directed edge connects a vertex V_{t_i,t_j} to an other vertex V_{t_k,t_l} if and only if $\boldsymbol{x}_{t_i:t_j}$ and $\boldsymbol{x}_{t_k:t_l}$ are two consecutive segments (i.e. $t_i < t_j = t_k < t_l$). The cost associated to this edge is the score $s(\boldsymbol{x}_{t_i:t_j}, \boldsymbol{x}_{t_j:t_l})$ associated to the corresponding pair of segments. Finally, we define a source vertex, with edges of weight 0 connecting it to all the vertices of the form V_{1,t_j}, as well as a target vertex, with edges of weight 0 connecting every vertices of the form $V_{t_i,T}$ to it. An toy example of such a graph is shown in Figure 3.1 corresponding to a time-series of length 5.

Solving the optimization problem is done by searching for the longest path from the source to the target. However, as the graph G is directed and acyclic, the graph $-G$ constructed by changing every weights in G by its opposite does not contain any negative cycle as it also is a DAG. The longest path problem on G is thus equivalent to the shortest path problem on $-G$, and it can by solved by classical shortest path algorithms. The corresponding optimal segmentation is the one made of the segments corresponding to every vertex on this optimal path.

When computing the shortest path on $-G$, we can capitalize on the fact that we have a DAG. Indeed, the shortest path problem on a DAG is a simpler problem than in the general case. For example, when using the Bellman-Ford algorithm we can perform a single iteration, provided we consider the vertices in topological order, instead of $|V| - 1$ in the general case. We can even simply process the edges in the order $1 \leq i < j < k \leq T$. Indeed, if we consider

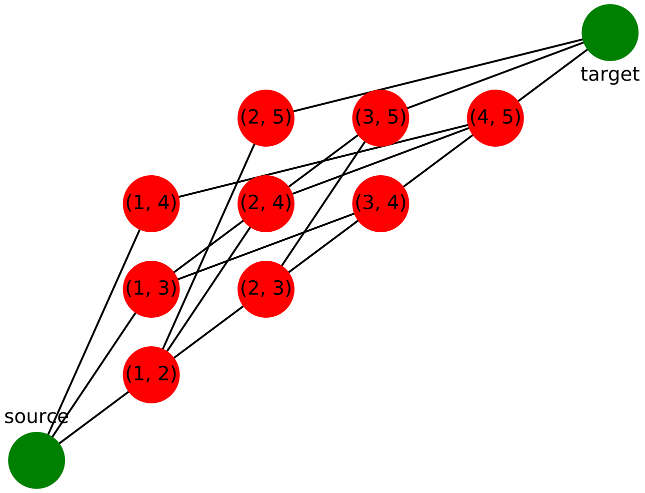

Figure 3.1 Toy example of a DAG associated to the problem of penalized CPD on a time-series $\boldsymbol{x}_{1:5}$ of length 5. Computing the optimal segmentation is done by computing a longest path from source to target.

the edges $(i, j)- > (j, k)$ in the order $1 \leq i < j < k \leq T$, all the edges going into a given node (m, n) will be processed before the ones going out of that node (m, n). In the end, this is equivalent to Algorithm 1.

3.2.2 Constrained optimization problem

In this section we are interested in solving the constrained problem (3.14). The algorithm we expose here is an adaptation of the SN algorithm exposed in Section 2.5.1 to our case where a score $s(\boldsymbol{x}_{t_i:t_j}, \boldsymbol{x}_{t_j:t_k})$ is associated to a pair of segments rather than to a single segment $\boldsymbol{x}_{t_i:t_j}$. Its goal is to compute the estimated segmentation of the time-series $\boldsymbol{x}$ with M change points

$$\begin{aligned} \widehat{\mathcal{T}} = \underset{\mathcal{T} \in \mathcal{S}_{\boldsymbol{x}}}{\operatorname{argmax}} \ & V(\mathcal{T}, \boldsymbol{x}) \\ \text{s.t.} \ & |\mathcal{T}| = M; \end{aligned} \tag{3.21}$$

that maximizes the objective function $V(\mathcal{T}, \boldsymbol{x})$. Like SN, it uses dynamic programming to efficiently explore the space of possible segmentations.

Let M be the user specified number of change points to be detected. Let us denote $F_M(t)$ the score of the optimal segmentation with M change points of the time-series $\boldsymbol{x}_{1:t}$ and $G_M(r, s)$ the score of the optimal segmentation with M change points of $\boldsymbol{x}_{1:s}$, with $\boldsymbol{x}_{r:s}$ as last segment.

By definition,

$$F_M(T) = \max_{1<t<T} G_M(t, T). \tag{3.22}$$

Moreover, thanks to Proposition 3.1, we have

$$G_M(s, t) = \max_{M \leq r < s} \{G_{M-1}(r, s) + s(\boldsymbol{x}_{r:s}, \boldsymbol{x}_{s:t})\} \tag{3.23}$$

Intuitively, Equation 3.23 shows that the optimal segmentation with M change points with $\boldsymbol{x}_{s:t}$ as last segment is easily computed provided all optimal segmentations with $M-1$ change points with $\boldsymbol{x}_{r:s}$ as last segment are known for all r such that $M \leq r < s$. The dynamic programming approach then consists in successively computing $G_m(s, t)$ for all (s, t) such that $1 < s < t \leq T$ and for $m = 1, \ldots, M$. The optimal segmentations with m change points of the time-series $\boldsymbol{x}_{1:T}$ can then simply be computed for $m = 1, \ldots, M$ using (3.22). The complete algorithm is detailed in Algorithm 2. It has a computational complexity $\mathcal{O}(MT^3)$, linear in the number of change points to be detected M, and cubic in the number of samples T in the time-series considered.

Algorithm 2 Constrained OTAWA

```
Input: time-series $\boldsymbol{x}_{1:T}$, score function $s(\cdot)$, number of change points $M$
  Init $G$ and $S$ two $(T-2)$ by $(T-2)$ $2D$-array
  for all $(u, v),\ 2 \leq u < v \leq T$ do
    $G_1[u, v] = s(\boldsymbol{x}_{1:u}, \boldsymbol{x}_{u:v})$
    $S_1[u, v] = \{1, u, v\}$
  end for
  for $m = 2, \ldots, M$ do
    for $s = 1 + m, \ldots, T - 1$ do
      for $t = s + 1, \ldots, T$ do
        $r^* = \text{argmax}_{m \leq r < s}\{G_{m-1}[r, s] + s(\boldsymbol{x}_{r:s}, \boldsymbol{x}_{s:t})\}$
        $G_m[s, t] = G_{m-1}[r^*, s] + s(\boldsymbol{x}_{r^*:s}, \boldsymbol{x}_{s:t})$
        $S_m[s, t] = S[r^*, s] \cup t$
      end for
    end for
  end for
  $t^* = \text{argmax}_{1<t<T}\, G_M[t, T]$
Output: $S_M[t^*, T]$
```

3.3 Estimating the number of change points

In the previous chapter, we have exposed two algorithms for solving both the penalized and the constrained versions of the optimization problem (3.4). Both of these algorithms provide control over to number of change points estimated, either through the penalty term or directly by specifying the desired number of change points. However, we still need a strategy for estimating the correct number of change points. In this section, we present two such strategies. The first one is based on the classical BIC criterion and can be used with both the penalized and the constrained versions of the problem. The second one on the other hand uses the adaptive choice method of penalization parameter described on page 25, and can only be applied with the penalized problem.

Both strategies consist in first computing a set of candidate segmentations with different number of change points. Each such candidate segmentation with m change points must be optimal among the set of segmentations with m change points. A criterion is then applied to discriminate among those candidate segmentations, and the one selected is considered the segmentation with the correct number of change points. In this section, we start by explaining how to compute such a set of candidate segmentations with the penalized or constrained problem, and then expose two different criteria.

3.3.1 Computing multiple candidate segmentations

Penalized problem Our Algorithm 1 for solving the linearly penalized problem is compatible with the CROPS algorithm exposed on page 24. This algorithm is able to efficiently compute all the optimal segmentations obtained for any value of the smoothing parameter $\beta \in [\beta_0, \beta_1]$. The set of candidate segmentations can be computed by first defining a lower and upper bound on the number of change points to be estimated, and explore the penalty range in order to to chose the values β_0 and β_1 corresponding to those bounds. A simple solution, if the chosen lower bound is 1 and upper bound is the maximum possible number of change points is to choose an extremely wide range $[\beta_0, \beta_1]$, as this does not affect the performance of the CROPS algorithm. The CROPS requires Algorithm 1 to be run a number of times linear in the difference between the number of change points corresponding to β_0 and β_1. The overall computational complexity is thus of the order of $\mathcal{O}(M_{max}T^3)$, where M_{max} is the maximum number of change points for a candidate segmentation.

Constrained problem Our Algorithm 2 for solving the constrained problem is directly able to compute all segmentations with $m = 1, \ldots, M_{max}$. The computational complexity $\mathcal{O}(M_{max}T^3)$ is similar to the one when using CROPS.

Difference between the two ways of computing the candidate segmentations Note that in the case of the linear penalty, the penalty term only depends on the number of change points. In such cases, a given segmentation that is optimal for the penalized problem and contains m change points is also optimal for the constrained problem with m change points. This means that any segmentation within the set C_{pen} of candidates computed using the penalized problem and CROPS also belongs to the set C_{const} of candidates computed using the constrained algorithm, and the only difference between these two sets is that some segmentations might be missing from C_{pen}, as not all segmentations with all numbers of change points are optimal for a value of the smoothing parameter β, and we have $C_{pen} \subseteq C_{const}$. In the end, we can say that using the constrained algorithm or the penalized algorithm in conjunction with CROPS for computing the set of candidate segmentations are very similar strategies, but using CROPS first filters out some segmentations that are never optimal for any value β.

3.3.2 Criteria for estimating the number of change points

Bayesian information criterion The BIC is defined as

$$BIC = \log(T)p - 2\log \widehat{\mathcal{L}}, \tag{3.24}$$

where in the context of a time-series segmentation $\widehat{\mathcal{L}}$ is the maximum of the likelihood of the piecewise constant model corresponding to a segmentation $\mathcal{T}$ on the time-series, p is the number of parameters of that piecewise model, and T is the number of samples of the time-series. In the literature, this criterion is often used to directly chose the penalty value β in cases where the objective function represents the maximum likelihood of the piecewise model. However with our method, the score-based objective function does not represent the maximum likelihood of this model. This is the reason why instead of directly choosing the penalty value β, the strategy when using the BIC criterion with Optimal Two Adjacent Windows Algorithm (OTAWA) consists in computing the BIC value associated to every candidate segmentation and choose the one with the lowest BIC value.

For a candidate segmentation $\mathcal{T}$, let $\mathcal{M}_k$ be the segment specific model estimated on $\boldsymbol{x}_{t_k:t_{k+1}}$, and p_k its number of parameters. Note that while usually being the same for all segment $\boldsymbol{x}_{t_k:t_{k+1}}$, the number of parameter p_k can depend on k. Indeed for example with the VAR model (3.11), it is defined as the number of non-zero parameters of $\mathcal{M}_k$. The number of parameters of the global piecewise model is the sum $p = \sum_{k=0}^{m} p_k$. Similarly, its log-likelihood $\log \widehat{\mathcal{L}}$ is the sum over the models $\mathcal{M}_k$ of the log-likelihood they associate to the segment $\boldsymbol{x}_{t_k:t_{k+1}}$ on which it has been estimated. These are used for computing the BIC values associated to every candidate segmentation $\mathcal{T}$, and the one which minimizes it is selected.

Adaptive choice of penalization parameter Contrary to the previous strategy with the BIC, this one can only be used in conjunction with Algorithm 1 for solving the penalized problem. The adaptive choice of penalization parameter method introduced in [58] and described on page 25 is compatible with Algorithm 1. For this reason, estimating the correct number of change points can be as simple as applying this method.

3.4 Computational complexity

In this section, we analyze the computational complexity of the proposed algorithm. Let us consider the estimation of the segmentation of a T samples long time-series $\boldsymbol{x}$. In order to solve either the penalized or the constrained version of the problem (3.4), we need to compute the scores associated to every element in the set $P_x = \{(\boldsymbol{x}_{t_i:t_j}, \boldsymbol{x}_{t_j:t_k}) \mid 1 \leq t_i < t_j < t_k \leq T\}$ of all the pairs of consecutive segments within $\boldsymbol{x}_{1:T}$. The number of scores to be computed is then the cardinality of this set

$$N_{scores} = |P_x| = \frac{1}{6}T^3 - \frac{1}{2}T^2 + \frac{1}{3}T. \tag{3.25}$$

So we can say that the number of scores is of the order $\mathcal{O}(T^3)$. Let us consider that the algorithm is run in two phases

- *Phase 1 :* computing all the scores;
- *Phase 2 :* solving the discrete optimization problem (3.4).

As we have seen in Section 3.2, the complexity of *Phase 2* is $\mathcal{O}(T^3)$ when solving the penalize version of the problem, and $\mathcal{O}(MT^3)$, when solving the constrained problem, where M is the number of scores to be computed. By looking closely at the two algorithms exposed, we can see that they both consist in considering all scores at least once. Algorithm 1 considers each score once, while Algorithm 2 considers scores a number of times of the order of M. This means that the complexity of *Phase 2* will always depend linearly in the number of scores N_{scores} (of the order $\mathcal{O}(T^3)$).

The complexity of *Phase 1* depends on the complexity of computing a score. In the most advantageous case, when the complexity of computing a score is constant, the complexity of *Phase 1* is $\mathcal{O}(N_{scores} * 1) = \mathcal{O}(T^3)$. However, computing a score involves estimating the parameters of a model and computing its prediction, and this will in general depend on the length of the segments considered, meaning the global complexity of the algorithm is $\mathcal{O}(N_{scores})$ multiplied by the mean complexity of computing scores. Hence the computational complexity of *Phase 1* is at least as high as the one of *Phase 2*, meaning that the global complexity of the whole algorithm is equivalent to the complexity of *Phase 1*. For this reason, we can consider only the complexity of *Phase 1*.

Another strategy for running the algorithm would be to compute the scores on the fly while solving the optimization problem. This doesn't change the global complexity of the algorithm which is still at least $\mathcal{O}(N_{scores}) = \mathcal{O}(T^3)$ since each score is used at least once. Nevertheless, it spares some spacial complexity, which was $\mathcal{O}(T^3)$ with the two phases algorithm, as no scores need to be stored.

Reducing the complexity of the algorithm for solving the optimization problem is non-trivial. When computing all costs in one phase, model estimation can be shared between all scores that have their first segment $\boldsymbol{x}_{t_i:t_j}$ in common, but error evaluation cannot. In any case, the complexity of computing a score depends on the choice of the predictive model used, which should be chosen according to prior knowledge about the time-series. So the best strategy for reducing the global complexity of the algorithm is to reduce the number of scores N_{scores} to be computed. The two methods for reducing the space of possible segmentations exposed in Section 2.5.3 can be used with our method as well, and can help reducing the number of scores N_{scores} to be computed. In the following, we will analyze how these two methods can reduce the number of scores to compute N_{scores} with minimal impact on the quality of the estimated segmentation.

3.4.1 Minimum length of a segment

A solution for reducing the global complexity of our algorithm is to add a constraint on the minimum length of a segment. Adding this constraint greatly reduces the cardinal $|\mathcal{S}_{\boldsymbol{x}}|$ of the set of all possible segmentations, in turn reducing the number of scores to be computed. Let $S \in \mathbb{N}$ be the minimum length of a segment. An admissible segmentation $\mathcal{T}$ is now

$$\mathcal{T} = \{t_k\}_{k=1}^{m} \subset \{1, \dots, T\} \tag{3.26}$$

$$\text{s.t. } t_{k+1} - t_k \geq S, \ \ \forall k = 0, \dots, m. \tag{3.27}$$

Thanks to the new constraint added, we can now omit the computation of $s(\boldsymbol{x}_{t_i:t_j}, \boldsymbol{x}_{t_j:t_k})$ if $t_j - t_j < S$ or $t_k - t_j < S$. Moreover, the new constraint also implies that no change point can exist at indices $1 < i < S$, because the segment $\boldsymbol{x}_{1:i}$ would violate it. For the same reason, no change point can exist at indices $T - S < i < T$. This means we can also omit the computation of $s(\boldsymbol{x}_{t_i:t_j}, \boldsymbol{x}_{t_j:t_k})$ if $0 < t_i < S$ or $T - S < t_k < T$. The remaining number of scores to compute is now

$$N_{scores} = \frac{1}{6}T^3 + T^2(\frac{3}{2} - 2S) + T(\frac{7}{3} + 8S^2 - 10S) + (-\frac{32}{3}S^3 + 17S^2 - \frac{19}{3}S). \tag{3.28}$$

This does not change the order of the number of scores to be computed, but in practice, it still reduces it significantly, as we always try to keep T from being too high.

Adding this constraint on the minimum length of segments makes sense in practical settings. Indeed, the predictive model requires a minimum number of samples in each segment, in order to be significant from a statistical standpoint. Moreover, in most applications, prior knowledge about the phenomenon generating the abrupt changes give a rough value for the minimum time between two change points. For example, if change point detection is used to detect maintenance events on a system, prior knowledge of the frequency of maintenance cycles can help setting the value of S.

3.4.2 Resolution

Another solution for reducing the computational cost of our algorithm is to reduce the set of indices considered as candidates change points. For example we can introduce a parameter $R \in \mathbb{N}$, that we call the *resolution*, and decide to consider as candidate change point only one in R points in the time-series. Let $M_R = \{R * i \mid i = 1, \dots, \left\lfloor \frac{T}{R} \right\rfloor\}$ the set of multiples of R inferior to T. The set of candidate change points is M_R, meaning that an admissible segmentation is now defined as $\mathcal{T} = \{t_k\}_{k=1}^m \subseteq M_R$. It can be interpreted as considering as candidate change point one in R points in the time-series. The number of candidate change points has changed from T to $|M_R| = \left\lfloor \frac{T}{R} \right\rfloor$. The number of scores to compute can be obtained by replacing T by $\left\lfloor \frac{T}{R} \right\rfloor$ in (3.25) :

$$N_{scores} = \frac{1}{6}\left\lfloor \frac{T}{R} \right\rfloor^3 - \frac{1}{2}\left\lfloor \frac{T}{R} \right\rfloor^2 + \frac{1}{3}\left\lfloor \frac{T}{R} \right\rfloor \tag{3.29}$$

The order of the number of scores to compute is $\mathcal{O}(N_{scores}) = \mathcal{O}((\frac{T}{R})^3)$, so we can say that the order of the number of scores to compute is divided by R^3.

The resolution parameter and the minimal length constraint can be used together, to further reduce the amount of scores to be computed. This amount can be computed in cases where S is a multiple of R, by replacing T by $\left\lfloor \frac{T}{R} \right\rfloor$ and S by $\frac{S}{R}$ in (3.28).

The effect of the resolution parameter is to reduce the precision of the positions of the change points estimated. All the information in the time-series $\boldsymbol{x}$ is still used in the sense that all the samples are considered for computing the scores. However, we can intuitively understand that the position of a true change point will now be estimated with a precision of only $\frac{R}{2}$, as its position should be set as the closest point with an index multiple of R.

CHAPTER 4 EVALUATION AND COMPARISON ON REAL DATA

In this chapter, we evaluate the performance of OTAWA on real data, and compare it to the state-of-the-art methods PELT and SN, as well as the approximate method SAW. To perform the comparison we use two different datasets for which the positions of the true change points are known. Note that the methods compared are all unsupervised methods, and we only use the annotations in order to quantify the performances achieved by each of them. The first dataset contains data acquired by a portable three-axis accelerometer placed on the body of a person. The goal of the segmentation is to identify the different activities the person was performing sequentially through time. The times of transition between different activities have been annotated by hand. The second dataset has been supplied by our partner PREDICT. It consists of in-flight measurements performed on an hydraulic system of an aircraft throughout its life-span. The dataset is annotated with the dates at which maintenance has been performed on the system, and the goal of the segmentation is to retrieve the dates of those maintenance events.

In the following, we first present the comparison methodology by listing the metrics used for performance evaluation and detailing the different algorithms against which OTAWA is compared. We then expose the comparison results for each of the two datasets separately.

4.1 Comparison methodology

4.1.1 Metrics

In order to quantify the performance of a given method, we will use different metrics listed in Section 3.2 of [1] measuring how similar the estimated and true segmentations are. The closest a segmentation is from the true segmentation, the better the method which produced it is. Different metrics emphasis different aspects of a good segmentation. We will use these different metrics to discuss performances and discriminate between the change point methods compared. Let us denote $\mathcal{T}^* = \{t_k^*\}_{k=1}^{m^*}$ the true segmentation and $\widehat{\mathcal{T}} = \{\widehat{t}_k\}_{k=1}^{\widehat{m}}$ the estimated segmentation. Note that the number of change points is not necessarily the same in both segmentations.

Annotation Error The AnnotationError is a simple metric used in order to assess whether or not the right number of change points has been estimated. It is simply the difference between the number of change points estimated $\widehat{m}$ and the true number m^*,

$$\textsc{AnnotationError} = |m^* - \widehat{m}|. \tag{4.1}$$

This metric is relevant to evaluate methods in which the number of change points estimated is not constrained.

F1-Score A simple way to quantify the quality of a segmentation is to use the widely used binary classification metric called F1-Score. Let's define a detection radius $R > 0$, and consider a true change point as being detected if a change point has been estimated within R samples of its location. We call true positive the set Tp of actual change points that have been detected,

$$\textsc{Tp}(\mathcal{T}^*, \widehat{\mathcal{T}}) = \{t^* \in \mathcal{T}^* | \exists \hat{t} \in \widehat{\mathcal{T}} \ s.t. \ |t^* - \hat{t}| < R\}. \tag{4.2}$$

We can then define the classical precision and recall measures as

$$\textsc{Prec}(\mathcal{T}^*, \widehat{\mathcal{T}}) = \frac{|\textsc{Tp}(\mathcal{T}^*, \widehat{\mathcal{T}})|}{\widehat{m}}, \tag{4.3}$$

$$\textsc{Rec}(\mathcal{T}^*, \widehat{\mathcal{T}}) = \frac{|\textsc{Tp}(\mathcal{T}^*, \widehat{\mathcal{T}})|}{m^*}. \tag{4.4}$$

and the F1-Score is the harmonic mean of the two,

$$\textsc{F1-Score}(\mathcal{T}^*, \widehat{\mathcal{T}}) = 2 \times \frac{\textsc{Prec}(\mathcal{T}^*, \widehat{\mathcal{T}}) \times \textsc{Rec}(\mathcal{T}^*, \widehat{\mathcal{T}})}{\textsc{Prec}(\mathcal{T}^*, \widehat{\mathcal{T}}) + \textsc{Rec}(\mathcal{T}^*, \widehat{\mathcal{T}})}. \tag{4.5}$$

The best value is 1 and the worst value is 0. Over-segmentation is penalized as it makes the precision tend to zero. Under-segmentation is penalized as well, as it makes the recall small.

Since Prec and Rec are classically defined as percentages, their values must lie within $[0, 1]$. Note that with definition (4.3), to ensure $\textsc{Prec}(\mathcal{T}^*, \widehat{\mathcal{T}}) \leq 1$ we need the detection radius R to be smaller that half the minimal spacing between two consecutive true change points ($R < \frac{t^*_{k+1} - t^*_k}{2}, \forall k \ s.t. \ 0 \leq k \leq m^*$). Indeed, if this is not the case ($\exists k \in [0, m^*] \ s.t. \ R \geq \frac{t^*_{k+1} - t^*_k}{2}$), one estimated change point could lie within the detection radius of two true change points, thus potentially allowing the number of detected change points to be higher than the number of estimated change points.

Hausdorff The Hausdorff metric is based on the Hausdorff distance, which is a distance defined on the set of subsets of a metric space. Here the metric space is the set of indices of the time-series $I = \{1, \dots, T\}$ with the ℓ_1–distance, $d : (x, y) \mapsto |y - x|$. The Hausdorff distance is used to measure the distance between the subsets of I corresponding to the true ($\mathcal{T}^* \subset I$) and estimated ($\widehat{\mathcal{T}} \subset I$) segmentations. Formally it is the largest distance between a true change point and its estimate or between an estimate and the true change point it is estimating :

$$\textsc{Hausdorff}(\mathcal{T}^*, \widehat{\mathcal{T}}) = \max\{\max_{\hat{t} \in \widehat{\mathcal{T}}} \min_{t^* \in \mathcal{T}^*} |\hat{t} - t^*|, \max_{t^* \in \mathcal{T}^*} \min_{\hat{t} \in \widehat{\mathcal{T}}} |t^* - \hat{t}|\} \tag{4.6}$$

If it is null, the two segmentations are equal,

$$\textsc{Hausdorff}(\mathcal{T}^*, \widehat{\mathcal{T}}) = 0 \quad \Rightarrow \quad \widehat{\mathcal{T}} = \mathcal{T}^*.$$

It takes a large value if a change point is estimated far from any true change point, or if no change point is estimated close from a true change point. This means that both over- and under-segmentation are penalized.

RandIndex RandIndex is a similarity metric between partitions of a set very well suited to our case of time-series segmentation. It has initially been introduced in [63] as a metric for evaluating clustering methods. It is defined as the proportion of agreements, where an agreement is a pair of indices in the time-series which are either in the same segment according to both $\mathcal{T}^*$ and $\widehat{\mathcal{T}}$ or in different segments according to both $\mathcal{T}^*$ and $\widehat{\mathcal{T}}$. More formally, for a given segmentation $\mathcal{T}$, let $gr(\mathcal{T})$ be the set of pairs of indices which are in the same segment (grouped) :

$$gr(\mathcal{T}) = \{(i, j), 1 \leq i < j \leq T \; s.t. \; \nexists t_k \in \mathcal{T} \mid i < t_k \leq j\} \tag{4.7}$$

and $ngr(\mathcal{T})$ be the set of pairs of indices which are not in the same segment (not grouped) :

$$ngr(\mathcal{T}) = \{(i, j), 1 \leq i < j \leq T \; s.t. \; \exists t_k \in \mathcal{T} \mid i < t_k \leq j\} \tag{4.8}$$

The RandIndex is :

$$\textsc{RandIndex}(\mathcal{T}^*, \widehat{\mathcal{T}}) = \frac{|gr(\widehat{\mathcal{T}}) \cap gr(\mathcal{T}^*)| + |ngr(\widehat{\mathcal{T}}) \cap ngr(\mathcal{T}^*)|}{T(T-1)/2} \tag{4.9}$$

Note that $T(T-1)/2$ is the total number of pairs of different indices in the time-series, or also the sum of the cardinals of the sets $gr(\mathcal{T})$ and $ngr(\mathcal{T})$ for any $\mathcal{T}$ ($T(T-1)/2 = |gr(\mathcal{T})| + |ngr(\mathcal{T})|, \quad \forall \mathcal{T} \subset \{1, \ldots, T\}$). Hence the maximum value of this metric is 1. Note that the RandIndex has the tendency to get close to 1 when the number of segments increases. Indeed, the more segments there are in the two segmentations compared, the higher the chances are of a pair of points far apart to be in different segments in both of them. This doesn't affect the discrimination power of the metric, but makes it harder to understand how significant a given difference of RandIndex is.

Mean Distance The MeanDistance metric is the mean over every true change points of the distance to the closest estimated change point :

$$\textsc{MeanDistance}(\mathcal{T}^*, \widehat{\mathcal{T}}) = \frac{\sum_{t^* \in \mathcal{T}^*} min_{\hat{t} \in \widehat{\mathcal{T}}} |\hat{t} - t^*|}{|\mathcal{T}^*|} \tag{4.10}$$

The MeanDistance is a positive value, and the lower it is, the better the estimated segmentation is. It is a good measure of the precision with which the estimated change points are positioned. However, it does not penalize over-segmentation.

4.1.2 Algorithms compared against OTAWA

Since the OTAWA algorithm we developed is an exact method, we want to compare it to other exact methods from the literature. We decide to mainly compare the penalized version of OTAWA against the PELT method. As both approaches are exact, this should enable us to analyze the value added by the new score-based objective function we propose. Moreover, PELT is considered a state-of-the-art method for optimization-based change point detection, making it a good candidate for evaluating the value of our method. We also want to compare the constrained version of OTAWA, in order to evaluate how different the performance is from the penalized version. To that end, the constrained version of OTAWA will be compared to the SN algorithm. Finally, we are interested in comparing OTAWA against the SAW method, as it uses a discrepancy-based scoring function as well, but the segmentation is estimated by an approximate peak detection algorithm. This comparison should thus give us information about the benefit of spending the computational power to solve the optimization problem exactly.

4.2 Human activity dataset

This dataset is provided online by the Human Activity Sensing Consortium and is part of a challenge held in 2011 [64]. It contains measurements made by a device such as a smart-phone fixed on the waist of a person. The device measures accelerations with an accelerometer, rotational rate with a gyroscope, and the local magnetic field with a magnetometer. Each of these three quantities is a vector of the physical space with values in $\mathbb{R}^3$, giving a time-series containing a total of nine variables. Measurements are recorded at a frequency of $100Hz$, for about seven minutes. The different activities are performed sequentially and include walking, staying, going up or down some stairs, escalators or in an elevator. The goal is to detect the times at which the activity performed changes, so that every segment in the inferred segmentation corresponds to a single activity.

4.2.1 Data preprocessing

Most of the activities performed in the dataset involve repetitive movements, such as walking, or going up some stairs. It is thus insightful to analyze the spectral information of the time-series, as it can enable us to pick up the frequencies associated with these repetitive movements. For these reasons, we perform a preprocessing of the dataset by computing the Short-time Fourier transform (STFT) of the signal, and keeping only the frequency bins that correspond to the frequencies relevant for the walking motion (between $0.5Hz$ and $5Hz$). This preprocessing method is inspired by the one performed in [4], where the interest is also analysis of accelerometer data measured on people while walking or running.

In practice, we performed the STFT with a window of size 512 samples (approximately $5s$), and an overlapping between windows of 75%.

The raw local magnetic field measurements mostly gives information about the orientation of the person. However, changes in orientation don't necessarily correspond to changes of activity, even if they might some times coincide. The local magnetic measurements might also hold some information about the repetitive walking movements of the person in the frequency domain, but applying a STFT to that signal is not practical, as the sampling frequency is highly variable through time. For these reasons, we decided to discard the local magnetic field information, and only consider the acceleration and rotational rate information for a total of six variables.

The signals measured by the portable accelerometer and gyroscope are very noisy. The STFT preprocessing has the advantage of filtering out this noise, as we are not keeping the frequency bins above $5Hz$, where most of the noise lies. Figure 4.1 shows the time-series corresponding to the acceleration along the x-axis before and after preprocessing. We can clearly see that the STFT preprocessing has the effect of smoothing the signal. Note that for visibility, only 3 of the 23 variables are displayed, for the time-series after preprocessing.

The STFT preprocessing also greatly reduces the number of samples of the time-series on which to perform CPD, while still using all the available information. With a window size of 512 samples and 75% overlapping, Fourier transforms are computed on local segments centered around one in every $512(1-0.75) = 128$ observations. This means that the number of samples is divided by 128 through the STFT process. For instance, on the acceleration

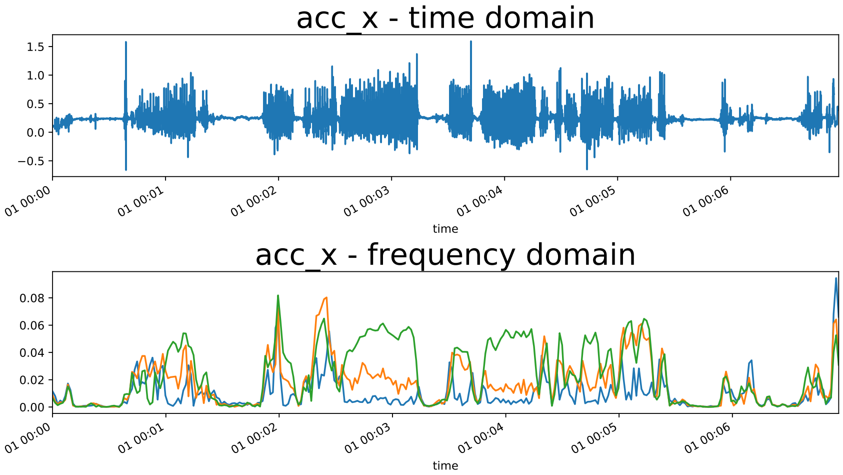

Figure 4.1 (top) signal corresponding to the acceleration measurements along the x-axis (bottom) same signal after STFT preprocessing, only 3 of the 23 variables are displayed for visibility

measurements, the number of samples goes from 39397 down to 308. However, it increases the number of variables of the time-series. With our signal sampled at $100Hz$, when computing the Fourier transform on a window of 512 samples and keeping the frequencies between $0.5Hz$ and $5Hz$, the number of variables is multiplied by 23. To compensate this increase in the number of variables, we only consider values of one quantity along one axis, resulting in only 23 variables. This doesn't seem to affect the performance of the methods. This can be interpreted by the fact that the measurements of the different quantities are correlated. We thus don't lose much information by considering a subset of the available variables.

The measurement dates are not included in the time-series fed to the CPD algorithms. Instead, the observations are assumed to be evenly spaced through time. Also, the time-series is rescaled to the $[0, 1]$ range using min-max scaling.

The different activities performed are labeled as periods, with a start- and end-time, associated with the corresponding name of the activity. These period don't overlap, meaning that no two activities can be performed simultaneously. There are little gaps between periods, where no activity is associated. These gaps are rather short in comparison to the duration of the activities. They make up 17% of the duration of the times-series and there are 24 of them. For this reason, we consider them as transitions, rather than an additional type of activity on their own. We define the position of the true change points as the middle point of those transition segments. In the dataset are also included some one-time events, such as pushing the button of an elevator. These are useful to understand what was happening around the person during the recording, and why the person changed activities. However, these events don't correspond to actual changes in the activities that are not already recorded through the periods, so we ignore them.

4.2.2 Experimental methodology

The performances of all methods are evaluated on the preprocessed data presented above. During a given type of activity, for instance walking or going up some stairs, we can reasonably assume that the repetitive walking motion does not change. This means that the spectral information is stationary within each segment. We will use the scoring function (3.10) associated to the model (3.8) for detecting a change in mean within i.i.d. samples following a normal distribution. This choice is motivated by the fact that it is a simple model, and the normally distributed with piecewise constant mean hypothesis is suited to the data considered.

As we are dealing with real data, the transitions might not be perfectly abrupt. Moreover, the STFT preprocessing tends to smooth the signal. Assuming there exist an abrupt change of a statistical property in the initial signal, as the spectral information is aggregated over segments with 75% overlapping during the STFT process, the transition will happen over at least three samples. With all methods, we use the model (3.8) with a margin parameter of $M = 3$, as it is the minimum values that enables us to account for the smoothing of the STFT preprocessing.

As we decided to only consider the measurements of one quantity along one axis for performing change point detection, we can repeat the experiment using each of the six time-series representing the measures of acceleration and rotational rate along each axis, and average the performances in order to get a more significant comparison of the different methods.

With all the methods compared we use a resolution parameter of $R = 2$ and a minimal space between change points of $S = 8$ samples. For OTAWA and PELT, the number of change points is estimated using the BIC. For SAW, we fix the length of both windows at $L = 10$ samples. We do not use the approximation with our method ($A = 1$). The F1-Score is computed with a detection radius of $R = 6$, meaning that a true change point is considered detected if an estimated change point lies at a distance strictly lower than $R = 6$ samples. After STFT, the acceleration time-series are 308 samples long, and the rotational rate time-series are 306 samples long.

4.2.3 Results

Figures 4.2 and 4.3 compare the performances of the penalized version of OTAWA with the PELT and SAW methods using five metrics that quantify how close the estimated segmentations are to the true segmentation. For each method, the values are averaged across the six different segmentations obtained using the accelerometer and gyroscope measurements along each spacial axis. The standard deviation is also displayed with error bars. Figure 4.2 shows the F1-Score and RandIndex, and Figure 4.3 shows the AnnotationError MeanDistance and Hausdorff. More detailed results for every individual segmentations are given in Table 4.1.

A first observation is that OTAWA outperforms both SAW and PELT according to all of the five metrics. The performance of SAW is particularly poor according to every metrics except for AnnotationError. This means that SAW is almost as good as OTAWA for estimating the number of change points. However the corresponding segmentations are much worse than the ones estimated by PELT or OTAWA. This observation emphasizes the value of using an exact algorithm for solving the optimization problem.

Focusing more on the comparison between our method and PELT, the AnnotationError graph is particularly interesting. It clearly shows that our method is much better at estimating the right number of change points than PELT. Having a closer look at the individual results of every experiments in Table 4.1, all three methods under-estimate the number of change points in every cases. The conclusion we can draw from the AnnotationError graph in Figure 4.3 is that our OTAWA algorithm has less of a tendency to under-estimate the number of change points than PELT. This is the reason for the quite large difference in terms of F1-Score as well. Under-estimating the number of change points does indeed decreases Recall and increases Precision in general. But as the Recall values are closer to zero than the Precision values, they have more impact on the harmonic mean used to compute the F1-Score.

The standard deviation values of the F1-Score, RandIndex and AnnotationError are lower for PELT than for our method. This means that even though the results of PELT are worse than OTAWA on average, the F1-Score and RandIndex values it achieves are more consistent, in the sense that they vary less from one dataset to the other. This effect is especially important for AnnotationError. Looking at the detailed results at Table 4.1, the values range from 6 to 13 for our method, and only from 16 to 17 for PELT. This higher variability of performance might be a drawback in certain circumstances where consistency is important. Nevertheless despite the lower variability, PELT only achieves a higher F1-Score on `gyro-z`, and a higher RandIndex on `gyro-x` and `gyro-z`, and always by a quite small margin. For MeanDistance and Hausdorff, the standard deviation values are similar between OTAWA and PELT, with even a slight advantage for our method.

Figure 4.4 shows the actual segmentations that have been estimated by both PELT and OTAWA on the time-series corresponding to the acceleration measurements along the x-axis. We observe that in general, OTAWA estimates more change points than PELT. While all change points estimated by PELT correspond to actual change points (Precision = 1), OTAWA places some that do not correspond to true change points. However, overall OTAWA is able to detect more of the true change points (higher Recall), especially between indices 190 and 214 and towards the end.

We can conclude that on this dataset, OTAWA achieves overall better results.This might be a sign that the new type of score-based objective function helps achieving better segmentations on real-world time-series.

In Appendix A, graphs similar to Figures 4.2 and 4.3 are shown for the comparison between the constrained version of OTAWA and the SN algorithm. The number of change points is here again estimated via the BIC. The same result as before are reported for the SAW method. We can observe that overall, the results are very similar to those obtained for PELT and the penalized version of OTAWA. This shows that the choice between the penalized and the constrained versions of the problem does not very much influence the nature of the results. However, if we study the values in more detail, we can note that the gap closes slightly between OTAWA and SN, with OTAWA performing a little worse. It gets even bitten by SAW in terms of AnnotationError. Since the only difference between the penalized and constrained versions of OTAWA is that the penalized version filters out some of the candidate segmentations before selection using the BIC, we can interpret that this filtering is useful in this case in order to filter out a segmentation that would be optimal in terms on BIC, but achieving slightly worse performance. Also, the standard deviations of OTAWA tends to increase.

4.3 Hydraulic system dataset

This dataset has been supplied by our partner PREDICT. It contains pressure and temperature values measured on an hydraulic system of an aircraft over a period of 716 days (approximately two years). Each data point corresponds to values averaged over a short period that can vary in length. Those periods are called flight phases, and are defined as a portion of a flight during which the speed and altitude of the aircraft are constant. On this particular instance, the company PREDICT knows the dates of the eight maintenance

Table 4.1 Detailed performance measures for OTAWA, PELT and SAW on each of the six available signals of acceleration and rotational rate.

method	**signal**	**randindex**	**f1**	**precision**	**recall**	**meandist**	**hausdorff**	**annotation**	**nbcps**
PELT	acc-x	0.919	0.514	**1**	0.346	13.6	62	17	9
OTAWA	acc-x	**0.968**	**0.739**	0.85	**0.654**	**2.69**	**11**	**6**	20
SAW	acc-x	0.733	0.238	0.312	0.192	54.4	148	10	16
PELT	acc-y	0.924	0.514	**1**	0.346	13.4	62	17	9
OTAWA	acc-y	**0.942**	**0.622**	0.737	**0.538**	**4.92**	**31**	**7**	19
SAW	acc-y	0.646	0.238	0.312	0.192	75.1	175	10	16
PELT	acc-z	0.933	0.514	**1**	0.346	8.92	**27**	17	9
OTAWA	acc-z	**0.937**	**0.6**	0.857	**0.462**	**7.08**	31	12	14
SAW	acc-z	0.734	0.255	0.286	0.231	53.9	148	**5**	21
PELT	gyro-x	**0.934**	0.556	**1**	0.385	8.04	**27**	16	10
OTAWA	gyro-x	0.93	**0.564**	0.846	**0.423**	**7.58**	31	13	13
SAW	gyro-x	0.733	0.244	0.333	0.192	53.5	147	**11**	15
PELT	gyro-y	0.925	0.457	**0.889**	0.308	8.42	27	17	9
OTAWA	gyro-y	**0.96**	**0.636**	0.778	**0.538**	**3.04**	**11**	**8**	18
SAW	gyro-y	0.73	0.195	0.267	0.154	54.2	148	11	15
PELT	gyro-z	**0.936**	**0.556**	**1**	0.385	8.35	**27**	16	10
OTAWA	gyro-z	0.918	0.55	0.786	**0.423**	**8.04**	37	**12**	14
SAW	gyro-z	0.708	0.2	0.286	0.154	60.3	156	**12**	14

events that occurred on the system during the considered period. However this is not always the case. Since in many applications they don't have access to additional information about maintenance, the engineers at PREDICT often need to retrieve the dates of maintenance events from raw measurement data, without any other prior knowledge. Moreover, they are interested in automating this task as much as possible. Our goal here is to use our OTAWA algorithm as well as exact and approximate methods from the literature in order to perform this task of retrieving the maintenance events. This is a good exercise to assess how well the CPD method we developed is able to perform this task, and thus how useful it can be to our partner PREDICT in order to automate it. We make here the assumption that the maintenance events will modify the behavior of the system, and that these modifications should be detectable from the measurements recorded. PREDICT can tell from experience that this is usually a reasonable assumption, provided the measurement of the right physical quantities are available.

4.3.1 Data preprocessing

The dataset initially contains 10399 observations. In order for the change point detection methods to run in a reasonable amount of time, we first need to reduce the number of samples of the time-series they will analyze. To do so, we perform a sub-sampling by averaging the measurement values over the period of one day. Out of the 716 days, 302 have not measurements. They are distributed across 123 periods, 2.46 days long on average. For those days, we set the values of the measurements equal to the previous valid measurement values. Thanks to this preprocessing, we reduce the number of samples of the time-series to 716. We still retain temporal information in the sens that all the samples are evenly spaced. Also, the time-series is rescaled to the $[0, 1]$ range using min-max scaling.

4.3.2 Experimental methodology

The maintenance events are on average separated by three month. Over such a time-scale of the order of the month, we expect the behavior of the hydraulic system to slowly drift due to normal degradations. Contrary to this slow drift, the maintenance events will translate into abrupt changes in the measurements, as they are performed between two observations. The slow drift of the behavior of the system can correspond to a smooth change of the mean value of the measurements over time. For this reason, the models (3.6) and (3.8) that assume normally distributed samples with piecewise constant mean are not suited. They might indeed detect a change in mean within segments where there only is a drift of the mean values of the measurement. We decide to use a VAR model (3.11) instead, as we hope that the linear relationship between a sample and its lagged values will be able to capture the drift of the mean value. If this assumption holds, the model should not be surprised by a slow drift. This means that the prediction error of the model should not increase at locations where there only is a drift with not abrupt change, thus preventing the detection of a false change point. This is corroborated by our experiments, as we observed that the VAR model (3.11) indeed performed better than both (3.6) and (3.8) models with piecewise constant mean. With all methods, we thus use a VAR model of order $p = 3$ estimated via Lasso with a regularization parameter of $\alpha = 10^{-2}$.

With all the methods compared we use a resolution parameter of $R = 5$ and a minimal space between change points of $S = 50$ samples. For SAW, we fix the length of both windows at $L = 20$ samples. With OTAWA, we do not use the approximation ($A = 1$). The F1-Score is computed with a detection radius of $R = 10$, meaning that a true change point is considered detected if an estimated change point lies at a distance strictly lower than $R = 10$ samples.

4.3.3 Results

Figures 4.5 and 4.6 compare the performances of the OTAWA, PELT and SAW methods by showing the values of different metrics. For PELT and OTAWA, the number of change points is estimated via BIC. Figure 4.5 shows the RandIndex, F1-Score, Prec and Rec metrics, while Figure 4.6 show the AnnotationError MeanDistance and Hausdorff metrics.

First of all we observe that OTAWA outperforms both PELT and SAW according to all metrics except for AnnotationError. OTAWA estimates 13 change points, while PELT estimates 6, and SAW is able to estimate the correct number of 8 change points. However despite over-estimating the number of change points, OTAWA still achieves the best Precision value among the three methods. This means that despite having estimated five more change points than necessary, the percentage of them actually corresponding to true change points is higher than with PELT and SAW.

When using CPD in scenarios where humans will manually analyze the results of the CPD algorithm – which is the case for the detection of maintenance events performed by PREDICT – over-estimating the number of change points might actually be preferred to under-estimation, as we give more importance to Recall rather than Precision. Indeed, it is significantly more efficient for a human to assess whether or not a maintenance event occurred at a given time, rather than to detect the times at which such events occurred. We thus can afford to trigger false alarms (false positive) that a human will be able to dismiss, as long as we minimize the omission of potential change points. CPD can then be a tool helping the human to pick potential change point candidates. Here, we observe that Recall is four times higher with OTAWA in comparison to the other two methods, making it well suited for this scenario.

On top of a higher F1-Score, OTAWA also achieves a significantly higher RandIndex value than the PELT and SAW. Finally, OTAWA is the best method according to the distance metrics Hausdorff and MeanDistance, which is expected when over-estimating the number of change points. Another interesting thing to note is that there is a very big gap between the performances of OTAWA and SAW. The sliding adjacent windows method achieve indeed very low performance in comparison to OTAWA, suggesting once again that it is very valuable to solve the optimization problem exactly, despite the higher computational cost.

In Appendix B, graphs similar to Figures 4.5 and 4.6 are shown for the comparison between the constrained version of OTAWA and the SN algorithm. The number of change points is also estimated via the BIC. The same result as before are reported for the SAW method. The overall results are relatively similar, with OTAWA still performing the best. Actually the exact same segmentation is estimated with both the penalized and constrained versions of the OTAWA method, and this is the reason why the results are identical in both cases. On the other hand, the segmentation estimated by SN is quite better than with PELT, enabling the gap with OTAWA to close significantly.

Adaptive choice of penalization parameter Figures 4.7 and 4.8 compare the performances of OTAWA and PELT when the number of change points is estimated via the adaptive choice of penalization parameter method presented in Section 2.6.2. For the the SAW method, the same results as before are reported for comparison. We observe that with this other way of estimating the number of change points, PELT performs better according to all metrics. The performance of OTAWA also increases in terms of Precision (and thus F1-Score), as well as for AnnotationError. The number of change points is indeed overestimated by only 2, instead of 5 with the BIC, while the Recall is unchanged. The performance slightly decreases in terms of MeanDistance and Hausdorff, and the RandIndex stays almost the same. Despite the changes in performances, the ranking of the methods is still the same as when using the BIC, with OTAWA still being the best out of the three methods.

Figure 4.9 shows the segmentations estimated by both PELT and OTAWA using the adaptive choice of penalization parameter method. First we can observe that both methods position change points at indices 465 and 665, even though they do not correspond to maintenance events. But these locations for change points are not surprising, as we visually see a big change in the time-series. Other than that, we can see that contrary to PELT, OTAWA is able to detect the first and third true change points. The change point estimated at index 415 is not counted as detected, since the true change point is 12 samples later at index 427. Overall, both segmentations are similar in the second half of the time-series, while

In conclusion, on this dataset, OTAWA appears the best method for accurately detecting the maintenance events according to every metrics except for AnnotationError. The performances of both PELT and OTAWA improve when using the adaptive choice of penalization parameter method for estimating the number of change points. However, this does not changes the advantage OTAWA has over PELT in terms of performance. This can be interpreted as OTAWA being able to more accurately detect change points than its competitors, only at the expense of a higher tendency to overestimate the number of change points. Finally, the importance of solving the optimization problem is highlighted, as the approximate method SAW clearly underperforms in comparison to both exact methods.

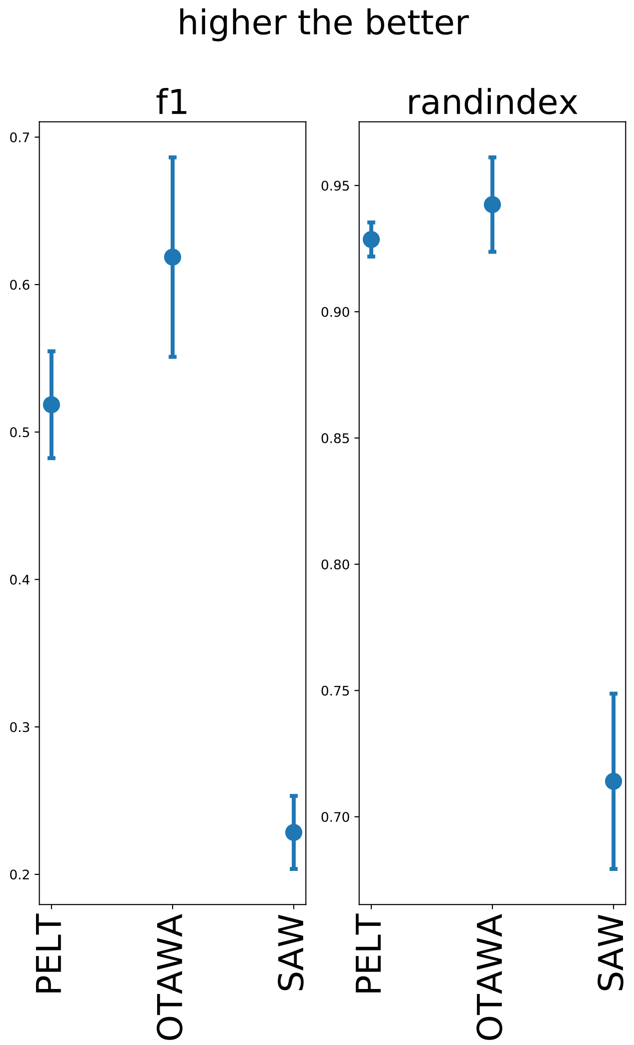

Figure 4.2 F1-Score and RandIndex for the OTAWA, PELT and SAW methods (higher the better). Number of change points estimated via BIC. Center value is the mean, error bars represent the standard deviation.

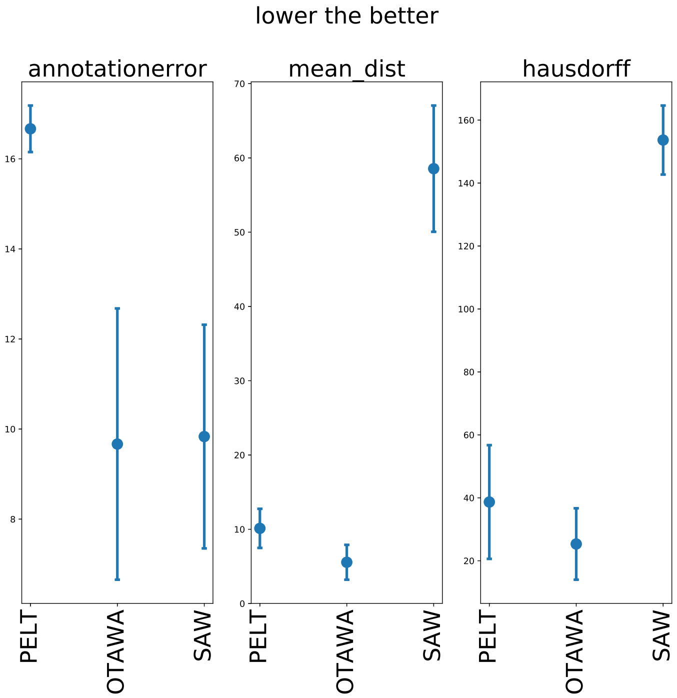

Figure 4.3 AnnotationError, MeanDistance and Hausdorff for the OTAWA, PELT and SAW methods (lower the better). Number of change points estimated via BIC. Center value is the mean, error bars represent the standard deviation.

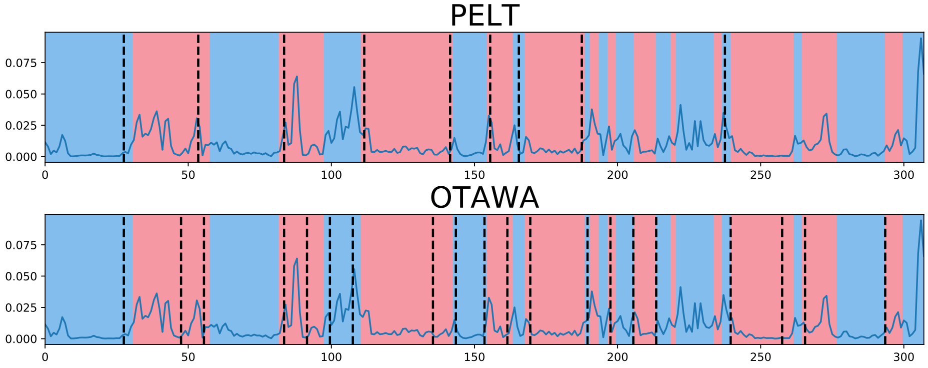

Figure 4.4 Segmentations estimated by both PELT and OTAWA on the time-series corresponding to the acceleration measurements along the x-axis. Color changes in the background indicate the true segmentation, vertical dotted lines indicate the estimated segmentation. Only one component of the time-series is shown for visibility.

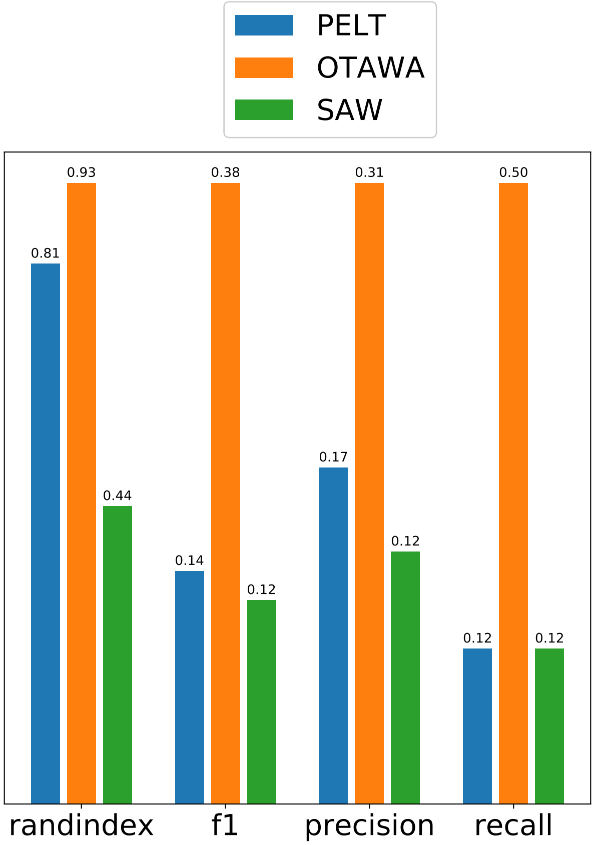

Figure 4.5 RandIndex, F1-Score, Prec and Rec for the OTAWA, PELT and SAW methods (higher the better). Number of change points estimated via BIC.

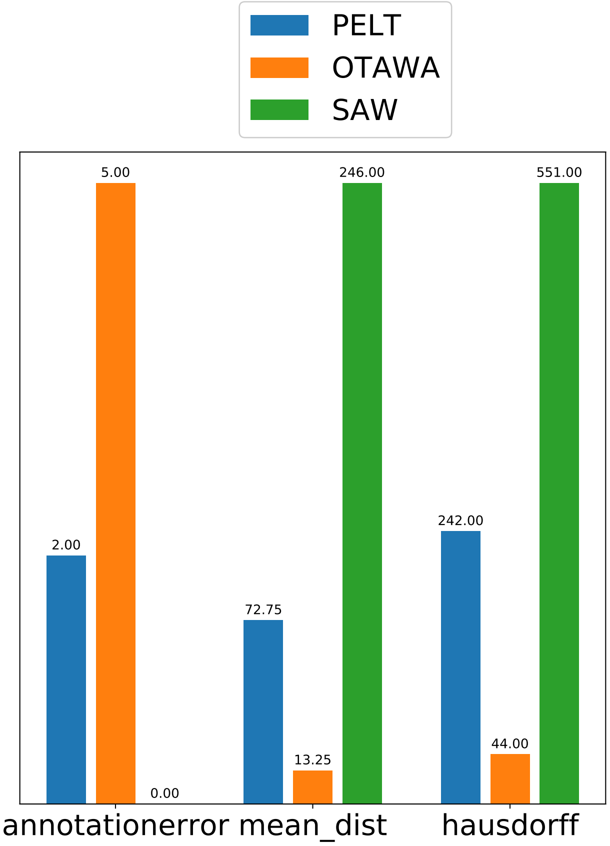

Figure 4.6 AnnotationError MeanDistance and Hausdorff for the OTAWA, PELT and SAW methods (lower the better). Number of change points estimated via BIC.

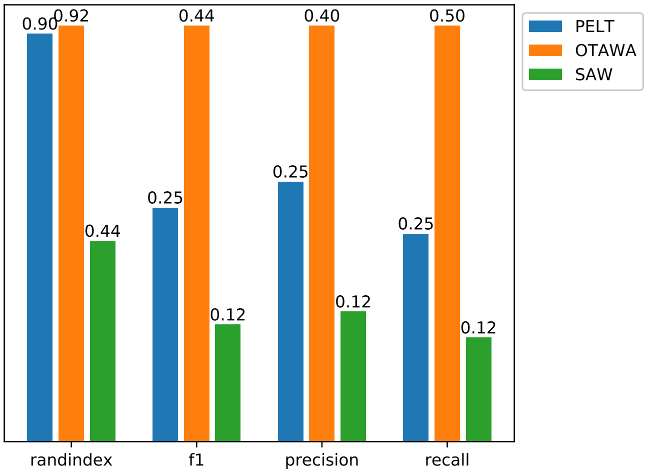

Figure 4.7 RandIndex, F1-Score, Prec and Rec for the OTAWA, PELT and SAW methods (higher the better). Number of change points estimated via the adaptive choice of penalization parameter method.

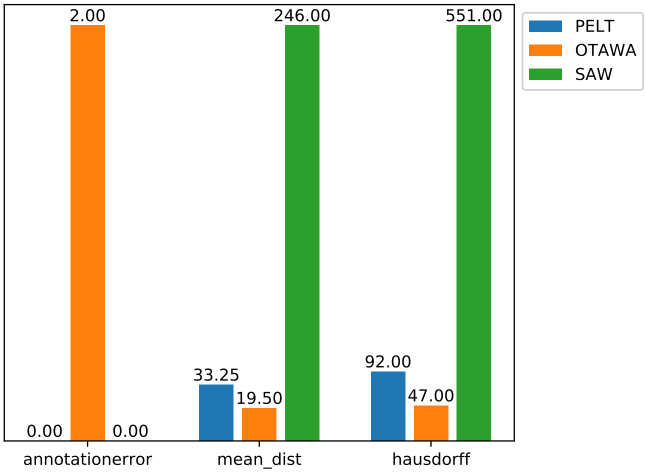

Figure 4.8 AnnotationError MeanDistance and Hausdorff for the OTAWA, PELT and SAW methods (lower the better). Number of change points estimated via the adaptive choice of penalization parameter method.

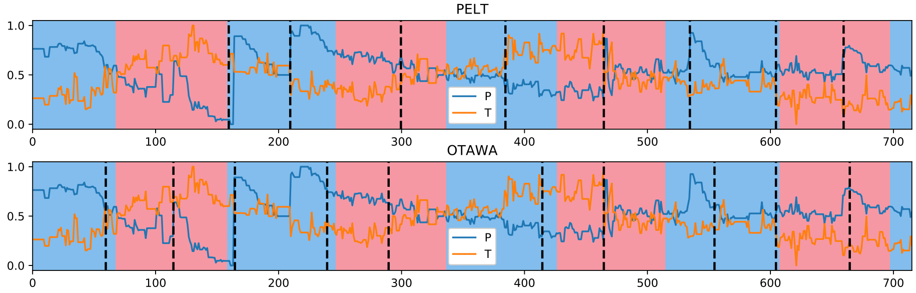

Figure 4.9 Segmentations estimated by both PELT and OTAWA on the hydraulic system dataset. Color changes in the background indicate the true segmentation, vertical dotted lines indicate the estimated segmentation.

CHAPTER 5 CONCLUSION AND RECOMMENDATIONS

5.1 Summary

In this work we addressed the problem of detecting changes in the operating mode of an equipment being monitored, with the constraint of only using the information contained within the time-series representing the quantities measured on the equipment. We framed this problem as a change point detection problem, and propose a novel approach to this problem called OTAWA. This approach is offline, as it assume that all measurement data has been collected before analysis. It is also unsupervised in the sense that no dataset is required where the true change points annotated. We tested OTAWA on two real-world datasets. The first dataset contains accelerations and rotation rates data from an inertial measurement unit (IMU) fixed on the body of a person while performing different activities. Change point detection is used for the task of segmenting the time-series into periods corresponding to the different activities, that can be considered as "operating modes" of the human being monitored by the inertial unit. The second dataset contains temperature and pressure measurements acquired on an hydraulic system of an aircraft over a period of two years. The task of change point detection here is to detect the maintenance events that happened on the system during that period. This dataset has been supplied by our partner PREDICT, and corresponds to an actual case where they could use change point detection in order to retrieve the dates of the maintenances, that are considered as operating mode changes. We used these two dataset in order to compare the performance of OTAWA against three other methods form the literature

- Pruned Exact Linear Time and Segment Neighbourhood, two popular exact optimization-based change point detection method using an objective function characterizing goodness of fit; and
- Sliding Adjacent Windows, an approximate method using an objective function characterizing discrepancies in statistical properties.

The results clearly show that OTAWA outperforms the approximate method SAW by a good margin in terms of RandIndex, F1-Score, MeanDistance and Hausdorff distance, emphasizing the value of solving the optimization problem exactly. Moreover, OTAWA also outperforms both exact methods PELT and SN by a significant margin according to those same metrics. We can interpret that as showing the value of the discrepancy-based objective function we proposed.

5.1.1 Value for our partner PREDICT

Here is a quote from the company PREDICT, about how they benefited form the collaboration and the conclusions they draw from this project.

> "Les travaux menés nous ont permis de prendre connaissance de la problématique de change point detection qui permet une approche multivarié basé sur le machine learning pour la détection de rupture sur un ensemble séries temporelles représentant des mesures effectuées sur un équipement alors que jusqu'à présent nous travaillions des méthodes de détection monovarié issue du traitement du signal.
>
> Les travaux qui ont été menés montrent la complexité du problème et permettent d'avoir des résultats intéressants et prometteurs sur un jeu de données réels comparé à des méthodes existantes. Ces résultats sont notamment intéressant pour de une application à la fouille d'historiques de données.
>
> En terme de perspectives d'amélioration, le temps de calcul augmente rapidement avec le nombre de point. Il peut dans certains cas être réduit, mais celà nécessite des connaissance a priori sur les données afin de régler des paramètres supplémentaires. Cela rend difficile l'utilisation en l'absence de telles connaissances. Idéalement, nous souhaiterions n'avoir aucun paramètre à régler pour l'usage."

5.2 Limitations

One of the main limitations of our OTAWA method lies in the estimation of the number of change points. It is done by selecting among a set of candidate optimal segmentations with different numbers of change points using a criterion, and we propose two different criteria. Now, on the annotated datasets we used for comparison we can evaluate the performance of every candidate segmentations, and when comparing the performance of the segmentation selected by either criteria to some other candidate segmentations that we picked by hand, we observe a clear gap, meaning that there is room for a new type of criterion to select a better segmentation. This would improve the overall performance of the method without requiring any modification to the objective function or the optimization algorithm. However, developing a better criterion seems hard, as all methods from the literature suffer of the same problem. For instance, for the PELT and SN methods we compared against, we observed this performance gap between the best segmentation a posteriori and the selected one as well. Coming up with new and better criteria is actually one of the main focus of the recent literature in the field.

An other drawback that is this time specific to our approach is the computational cost of the OTAWA algorithm. It is indeed $\mathcal{O}(T^3)$ cubic in the number of samples T in the time-series considered, compared to a quadratic complexity $\mathcal{O}(T^2)$ with exact methods from the literature. Moreover, this complexity can even get worse when using complex models for the time-series, as it must be multiplied by the computational complexity of the algorithm for model estimation.

5.3 Future Research Directions

As mentioned in the limitations above, some interesting research directions lie in improving on the computational cost of the OTAWA algorithm, as well as developing a criterion that is able to better estimate the number of change points. Improving on the computational cost could be done by developing an approximate algorithm, or by adapting pruning techniques to our approach, such as the one used in PELT.

On top of that, we can think of two potential ways of improving OTAWA. First, the discrepancy-based objective function has an intrinsic notion of direction. Indeed, for any pair of consecutive segments, it consists in estimating a model on the left segment, and measuring its prediction error on the right segment, imposing a "forward" direction. However, in the offline case where all samples of the considered time-series are available, there is no reason in general not to define the objective function the other way around in a "backward" manner. An interesting direction might then lie toward modifying the objective function into a "bidirectional" objective function that would take into account both "forward" and "backward" directions, similarly to bidirectional recurrent neural networks, or bidirectional inference networks. We note that while this is perfectly grounded for prediction of the future, autoregressive models have such an intrinsic directional bias as well. A second potential research direction could be interested in developing an hybrid method between the discrepancy-based objective function of OTAWA and the one based on goodness of fit such as used in PELT or SN. Indeed, while our approach seems promising in the sense that it outperformed the existing methods we compared it to, it does not discard the approaches based on goodness of fit. A wide body of literature has proven that they achieve good performance on a wide variety of datasets, and with a wide variety of underlying models. It might thus be interesting to study the combination of the two approaches into a hybrid method that wood optimize both goodness of fit and discrepancy at the same time.

Finally, it could be interesting to adapt our approach to models with latent or hidden variables by looking for changes of this variable. This could allow to relax the independence hypothesis being made with piecewise i.i.d. models, which can often be to restrictive.

REFERENCES

[1] C. Truong, L. Oudre, and N. Vayatis, "Selective review of offline change point detection methods," jan 2018. [Online]. Available: http://arxiv.org/abs/1801.00718

[2] S. Aminikhanghahi and D. J. Cook, "A survey of methods for time series change point detection," *Knowledge and Information Systems*, vol. 51, no. 2, pp. 339–367, may 2017. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/28603327http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC5464762

[3] B. Jackson, J. D. Scargle, D. Barnes, S. Arabhi, A. Alt, P. Gioumousis, E. Gwin, P. Sangtrakulcharoen, L. Tan, and T. T. Tsai, "An algorithm for optimal partitioning of data on an interval," *IEEE Signal Processing Letters*, vol. 12, no. 2, pp. 105–108, feb 2005. [Online]. Available: http://ieeexplore.ieee.org/document/1381461/

[4] L. Oudre, A. Lung-Yut-Fong, and P. Bianchi, "Segmentation automatique de signaux issus d'un accéléromètre triaxial en période de marche," Tech. Rep., 2011. [Online]. Available: http://www.laurentoudre.fr/publis/OLB-GRETSI-11.pdf

[5] K. Haynes, P. Fearnhead, and I. A. Eckley, "A computationally efficient nonparametric approach for changepoint detection," *Statistics and Computing*, vol. 27, no. 5, pp. 1293–1305, sep 2017. [Online]. Available: http://link.springer.com/10.1007/s11222-016-9687-5

[6] A. L. Schröder and H. Ombao, "FreSpeD: Frequency-Specific Change-Point Detection in Epileptic Seizure Multi-Channel EEG Data," *Journal of the American Statistical Association*, pp. 1–14, oct 2018. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/01621459.2018.1476238

[7] M. Bosc, F. Heitz, J.-P. Armspach, I. Namer, D. Gounot, and L. Rumbach, "Automatic change detection in multimodal serial MRI: application to multiple sclerosis lesion evolution," *NeuroImage*, vol. 20, no. 2, pp. 643–656, oct 2003. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/14568441https://linkinghub.elsevier.com/retrieve/pii/S1053811903004063

[8] T. D. Hocking, G. Schleiermacher, I. Janoueix-Lerosey, V. Boeva, J. Cappo, O. Delattre, F. Bach, and J.-P. Vert, "Learning smoothing models of copy number profiles using breakpoint annotations." *BMC bioinformatics*, vol. 14, p. 164, may 2013. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/23697330http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3712326

[9] E. Keogh, S. Chu, D. Hart, and M. Pazzani, "An Online Algorithm for Segmenting Time Series," in *Proceedings of the 2001 IEEE International Conference on Data Mining.* IEEE Computer Society, 2001, p. 677. [Online]. Available: https://dl.acm.org/citation.cfm?id=657889

[10] J. Reeves, J. Chen, X. L. Wang, R. Lund, Q. Q. Lu, J. Reeves, J. Chen, X. L. Wang, R. Lund, and Q. Q. Lu, "A Review and Comparison of Changepoint Detection Techniques for Climate Data," *Journal of Applied Meteorology and Climatology*, vol. 46, no. 6, pp. 900–915, jun 2007. [Online]. Available: http://journals.ametsoc.org/doi/abs/10.1175/JAM2493.1

[11] R. Killick, P. Fearnhead, and I. A. Eckley, "Optimal Detection of Changepoints With a Linear Computational Cost," *Journal of the American Statistical Association*, vol. 107, no. 500, pp. 1590–1598, dec 2012. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/01621459.2012.737745

[12] ——, "Supplemental material: Optimal detection of changepoints with a linear computational cost," vol. 2, no. February 2015, pp. 37–41, 2012.

[13] E. S. Page, "Continuous Inspection Schemes," *Biometrika*, vol. 41, no. 1-2, pp. 100–115, jun 1954. [Online]. Available: https://www.jstor.org/stable/2333009?origin=crossrefhttps://academic.oup.com/biomet/article-lookup/doi/10.1093/biomet/41.1-2.100

[14] R. J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam, "Image change detection algorithms: a systematic survey." *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, vol. 14, no. 3, pp. 294–307, mar 2005. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/15762326

[15] M. F. R. Chowdhury, S.-A. Selouani, and D. O'Shaughnessy, "Bayesian on-line spectral change point detection: a soft computing approach for on-line ASR," *International Journal of Speech Technology*, vol. 15, no. 1, pp. 5–23, mar 2012. [Online]. Available: http://link.springer.com/10.1007/s10772-011-9116-2

[16] K. Haynes, I. A. Eckley, and P. Fearnhead, "Computationally Efficient Changepoint Detection for a Range of Penalties," *Journal of Computational and Graphical Statistics*, vol. 26, no. 1, pp. 134–143, jan 2017. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/10618600.2015.1116445

[17] A. Sen and M. S. Srivastava, "On Tests for Detecting Change in Mean," *The Annals of Statistics*, vol. 3, no. 1, pp. 98–108, jan 1975. [Online]. Available: http://projecteuclid.org/euclid.aos/1176343001

[18] Y.-C. Yao, "Estimating the number of change-points via Schwarz' criterion," *Statistics & Probability Letters*, vol. 6, no. 3, pp. 181–189, feb 1988. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0167715288901186

[19] M. Lavielle and E. Moulines, "Least-squares Estimation of an Unknown Number of Shifts in a Time Series," *Journal of Time Series Analysis*, vol. 21, no. 1, pp. 33–59, jan 2000. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/1467-9892.00172

[20] H. Keshavarz, C. Scott, and X. Nguyen, "Optimal change point detection in Gaussian processes," *Journal of Statistical Planning and Inference*, vol. 193, pp. 151–178, feb 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S037837581730160X

[21] S. I. M. Ko, T. T. L. Chong, and P. Ghosh, "Dirichlet Process Hidden Markov Multiple Change-point Model," may 2015. [Online]. Available: http://arxiv.org/abs/1505.01665http://dx.doi.org/10.1214/14-BA910

[22] K. Haynes, I. A. Eckley, and P. Fearnhead, "Efficient penalty search for multiple changepoint problems," dec 2014. [Online]. Available: http://arxiv.org/abs/1412.3617

[23] M. Lavielle, "Detection of multiple changes in a sequence of dependent variables," *Stochastic Processes and their Applications*, vol. 83, no. 1, pp. 79–102, sep 1999. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S030441499900023X

[24] A. Aue and L. Horváth, "Structural breaks in time series," *Journal of Time Series Analysis*, vol. 34, no. 1, pp. 1–16, jan 2013. [Online]. Available: http://doi.wiley.com/10.1111/j.1467-9892.2012.00819.x

[25] V. Jandhyala, S. Fotopoulos, I. MacNeill, and P. Liu, "Inference for single and multiple change-points in time series," *Journal of Time Series Analysis*, vol. 34, no. 4, pp. 423–446, jul 2013. [Online]. Available: http://doi.wiley.com/10.1111/jtsa.12035

[26] S. Chib, "Estimation and comparison of multiple change-point models," *Journal of Econometrics*, vol. 86, no. 2, pp. 221–241, oct 1998. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0304407697001152

[27] D. Angelosante and G. B. Giannakis, “Group lassoing change-points in piecewise-constant AR processes,” *EURASIP Journal on Advances in Signal Processing*, vol. 2012, no. 1, p. 70, dec 2012. [Online]. Available: https://asp-eurasipjournals.springeropen.com/articles/10.1186/1687-6180-2012-70

[28] R. A. Davis, T. C. M. Lee, and G. A. Rodriguez-Yam, “Structural Break Estimation for Nonstationary Time Series Models,” *Journal of the American Statistical Association*, vol. 101, no. 473, pp. 223–239, mar 2006. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1198/016214505000000745

[29] M. Hušková, Z. Prášková, and J. Steinebach, “On the detection of changes in autoregressive time series I. Asymptotics,” *Journal of Statistical Planning and Inference*, vol. 137, no. 4, pp. 1243–1259, apr 2007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378375806000826

[30] E. Gombay, “Change detection in autoregressive time series,” *Journal of Multivariate Analysis*, vol. 99, no. 3, pp. 451–464, mar 2008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0047259X07000061

[31] C. F. H. Nam, J. A. D. Aston, and A. M. Johansen, “Quantifying the uncertainty in change points,” *Journal of Time Series Analysis*, vol. 33, no. 5, pp. 807–823, sep 2012. [Online]. Available: http://doi.wiley.com/10.1111/j.1467-9892.2011.00777.x

[32] S. Chakar, E. Lebarbier, C. Lévy-Leduc, and S. Robin, “A robust approach for estimating change-points in the mean of an AR(1) process,” *Bernouilli Society for Mathematical Statistics and Probability*, vol. 23, no. 2, pp. 1408–1447, may 2017. [Online]. Available: http://projecteuclid.org/euclid.bj/1486177403

[33] Z. Qu and P. Perron, “Estimating and Testing Structural Changes in Multivariate Regressions,” pp. 459–502, 2007. [Online]. Available: https://www.jstor.org/stable/4501997

[34] J. Bai, “Vector Autoregressive Models with Structural Changes in Regression Coefficients and in Variance-Covariance Matrices,” *Annals of Economics and Finance*, vol. 1, no. 2, pp. 303–339, 2000. [Online]. Available: https://ideas.repec.org/p/cuf/wpaper/24.html

[35] ——, “Testing for Parameter Constancy in Linear Regressions: An Empirical Distribution Function Approach,” *Econometrica*, vol. 64, no. 3, p. 597, may 1996. [Online]. Available: https://www.jstor.org/stable/2171863?origin=crossref

[36] P. Perron and P. Perron, “Dealing with Structural Breaks,” *Palgrave handbook of econometrics*, vol. 1, no. 2, pp. 278—-352, 2006. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.148.3909

[37] J. Bai and P. Perron, “Critical values for multiple structural change tests,” *The Econometrics Journal*, vol. 6, no. 1, pp. 72–78, jun 2003. [Online]. Available: https://academic.oup.com/ectj/article/6/1/72-78/5074163

[38] J. Bai, “Likelihood ratio tests for multiple structural changes,” Tech. Rep., 1999. [Online]. Available: http://www.columbia.edu/~jb3064/papers/1999_Likelihood_ratio_tests_for_multiple_structural_changes.pdf

[39] B. M. Doyle and J. Faust, “Breaks in the Variability and Comovement of G-7 Economic Growth,” *Review of Economics and Statistics*, vol. 87, no. 4, pp. 721–740, nov 2005. [Online]. Available: http://www.mitpressjournals.org/doi/10.1162/003465305775098134

[40] J. Cabrieto, F. Tuerlinckx, P. Kuppens, F. H. Wilhelm, M. Liedlgruber, and E. Ceulemans, “Capturing correlation changes by applying kernel change point detection on the running correlations,” *Information Sciences*, vol. 447, pp. 117–139, jun 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025516316929

[41] Z. Harchaoui and O. Cappe, “Retrospective Mutiple Change-Point Estimation with Kernels,” in *2007 IEEE/SP 14th Workshop on Statistical Signal Processing*. IEEE, aug 2007, pp. 768–772. [Online]. Available: http://ieeexplore.ieee.org/document/4301363/

[42] S. Arlot, A. Celisse, and Z. Harchaoui, “A Kernel Multiple Change-point Algorithm via Model Selection,” feb 2012. [Online]. Available: http://arxiv.org/abs/1202.3878

[43] A. Celisse, G. Marot, M. Pierre-Jean, and G. Rigaill, “New efficient algorithms for multiple change-point detection with reproducing kernels,” *Computational Statistics & Data Analysis*, vol. 128, pp. 200–220, dec 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167947318301683

[44] I. E. Auger and C. E. Lawrence, “Algorithms for the optimal identification of segment neighborhoods,” *Bulletin of Mathematical Biology*, vol. 51, no. 1, pp. 39–54, jan 1989. [Online]. Available: http://link.springer.com/10.1007/BF02458835

[45] Bellman R, “The theory of dynamic programming,” *Bulletin of the American Mathematical Society*, vol. 60, no. 6, pp. 503–515, 1954.

[46] G. Rigaill, “A pruned dynamic programming algorithm to recover the best segmentations with 1 to K_max change-points.” *Journal de la Société Française de Statistique*, vol. 156, no. 4, pp. 180–205, 2015. [Online]. Available: http://journal-sfds.fr/article/view/485

[47] R. Maidstone, T. Hocking, G. Rigaill, and P. Fearnhead, “On optimal multiple changepoint algorithms for large data,” *Statistics and Computing*, vol. 27, no. 2, pp. 519–533, mar 2017. [Online]. Available: http://link.springer.com/10.1007/s11222-016-9636-3

[48] A. J. Scott and M. Knott, “A Cluster Analysis Method for Grouping Means in the Analysis of Variance,” *Biometrics*, vol. 30, no. 3, p. 507, sep 1974. [Online]. Available: https://www.jstor.org/stable/2529204?origin=crossref

[49] C. Truong, L. Oudre, and N. Vayatis, “ruptures: change point detection in Python,” jan 2018. [Online]. Available: http://arxiv.org/abs/1801.00826

[50] G. Schwarz, “Estimating the Dimension of a Model,” *The Annals of Statistics*, vol. 6, no. 2, pp. 461–464, mar 1978. [Online]. Available: http://projecteuclid.org/euclid.aos/1176344136

[51] H. Akaike, “A new look at the statistical model identification,” *IEEE Transactions on Automatic Control*, vol. 19, no. 6, pp. 716–723, dec 1974. [Online]. Available: http://ieeexplore.ieee.org/document/1100705/

[52] E. J. Hannan and B. G. Quinn, “The Determination of the Order of an Autoregression,” pp. 190–195, 1979. [Online]. Available: https://www.jstor.org/stable/2985032

[53] T. Hocking, G. Rigaill, J.-P. Vert, and F. Bach, “Learning Sparse Penalties for Change-point Detection using Max Margin Interval Regression,” in *International Conference on Machine Learning (ICML)*, feb 2013, pp. 172–180. [Online]. Available: http://proceedings.mlr.press/v28/hocking13.html

[54] S. Arlot and A. Celisse, “A survey of cross-validation procedures for model selection,” *Statistics Surveys*, vol. 4, no. 0, pp. 40–79, 2010. [Online]. Available: http://projecteuclid.org/euclid.ssu/1268143839

[55] C. Truong, L. Gudre, and N. Vayatis, “Penalty learning for changepoint detection,” in *2017 25th European Signal Processing Conference (EUSIPCO)*. IEEE, aug 2017, pp. 1569–1573. [Online]. Available: http://ieeexplore.ieee.org/document/8081473/

[56] N. R. Zhang and D. O. Siegmund, “A Modified Bayes Information Criterion with Applications to the Analysis of Comparative Genomic Hybridization Data,”

Biometrics, vol. 63, no. 1, pp. 22–32, mar 2007. [Online]. Available: http://doi.wiley.com/10.1111/j.1541-0420.2006.00662.x

[57] E. Lebarbier and E., "Detecting multiple change-points in the mean of Gaussian process by model selection," *Signal Processing*, vol. 85, no. 4, pp. 717–736, apr 2005. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0165168404003196

[58] M. Lavielle, "Using penalized contrasts for the change-point problem," *Signal Processing*, vol. 85, pp. 1501–1510, 2005. [Online]. Available: www.elsevier.com/locate/sigpro

[59] Y. Guédon, "Exploring the latent segmentation space for the assessment of multiple change-point models," *Computational Statistics*, vol. 28, no. 6, pp. 2641–2678, dec 2013. [Online]. Available: http://link.springer.com/10.1007/s00180-013-0422-9

[60] M. Basseville and I. V. Nikiforov, *Detection of abrupt changes: theory and application*, ser. Prentice Hall information and system sciences. Prentice Hall, 1993. [Online]. Available: http://books.google.de/books?id=Vu5SAAAAMAAJ

[61] P. Granjon, "The CuSum algorithm - a small review," Tech. Rep., 2013. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00914697

[62] G. Lorden, "Procedures for Reacting to a Change in Distribution," *The Annals of Mathematical Statistics*, vol. 42, no. 6, pp. 1897–1908, dec 1971. [Online]. Available: http://projecteuclid.org/euclid.aoms/1177693055

[63] W. M. Rand, "Objective Criteria for the Evaluation of Clustering Methods," *Journal of the American Statistical Association*, vol. 66, no. 336, p. 846, dec 1971. [Online]. Available: https://www.jstor.org/stable/2284239?origin=crossref

[64] "Hasc Challenge 2011," 2011. [Online]. Available: http://hasc.jp/hc2011/

APPENDIX A COMPARISON OF CONSTRAINED ALGORITHMS ON THE HUMAN ACTIVITY DATASET

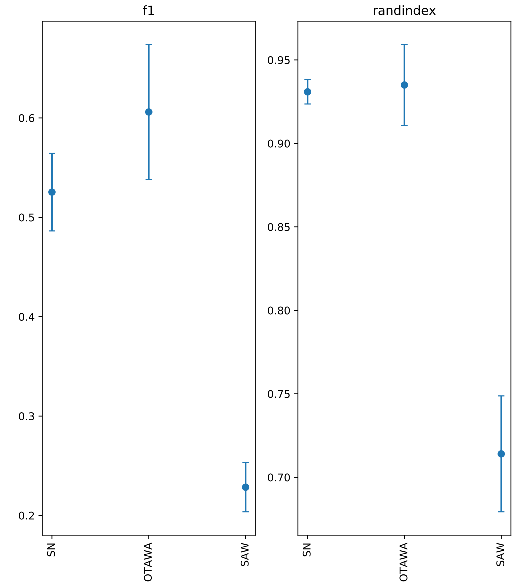

Figure A.1 F1-Score and RandIndex for the OTAWA, SN and SAW methods (higher the better). Center value is the mean, error bars represent the standard deviation.

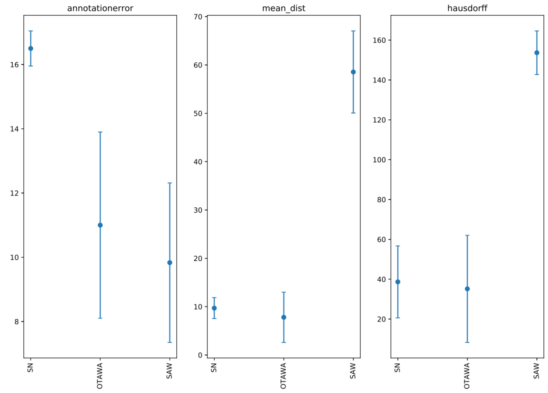

Figure A.2 AnnotationError MeanDistance and Hausdorff for the OTAWA, SN and SAW methods (lower the better). Center value is the mean, error bars represent the standard deviation.

APPENDIX B COMPARISON OF CONSTRAINED ALGORITHMS ON THE HYDRAULIC SYSTEM DATASET

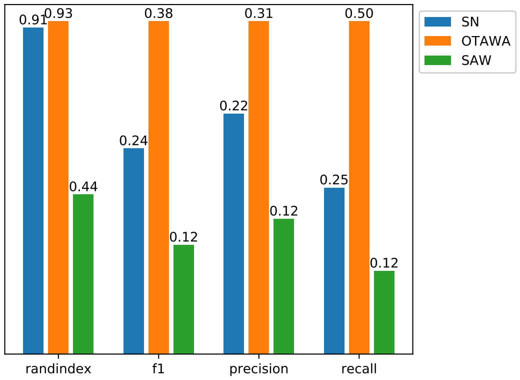

Figure B.1 RandIndex, F1-Score, Prec and Rec for the OTAWA, SN and SAW methods (higher the better).

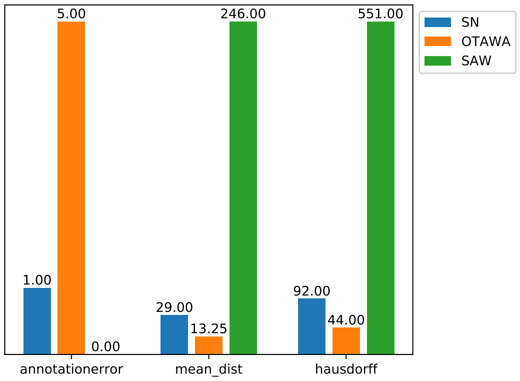

Figure B.2 AnnotationError, MeanDistance and Hausdorff for the OTAWA, SN and SAW methods (lower the better).