UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

GREEDY ONLINE CHANGE POINT DETECTION

TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN CIENCIAS DE LA INGENIERÍA, MENCIÓN ELÉCTRICA

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERA CIVIL ELÉCTRICA

JOU-HUI HO KU

PROFESOR GUÍA:
FELIPE ARTURO TOBAR HENRÍQUEZ

MIEMBROS DE LA COMISIÓN:
JORGE FELIPE SILVA SÁNCHEZ
KARINE MARIE ANNE BERTIN

Este trabajo ha sido parcialmente financiado por:
Google
Agencia Nacional de Investigación y Desarrollo (ANID)

SANTIAGO DE CHILE
2022

RESUMEN DE LA TESIS PARA OPTAR
AL GRADO DE MAGÍSTER EN CIENCIAS
DE LA INGENIERÍA, MENCIÓN ELÉCTRICA
POR: JOU-HUI HO KU
FECHA: 2022
PROF. GUÍA: FELIPE TOBAR HENRÍQUEZ

DETECCIÓN VORAZ DE PUNTOS DE CAMBIO A TIEMPO REAL

La detección online de puntos de cambio tiene el propósito de detectar cambios abruptos en series de tiempo a tiempo real, que es crucial en aplicaciones donde se requieran respuestas inmediatas, como finanzas y monitoreo de señales médicas. Los métodos existentes de detección de puntos de cambio basan su decisión en la probabilidad de las últimas observaciones, la cual puede cambiar abruptamente ante la presencia de puntos aislados o *outliers*, retornando una alta tasa de falsos positivos.

En esta tesis, se propone *Detección voraz de puntos de cambio a tiempo real* (en sus siglas en inglés, GOCPD) — un algorithmo computacionalmente eficiente que opera de forma *greedy*. Concretamente, GOCPD usa búsqueda ternaria para buscar el punto de cambio óptimo que maximiza la verosimilitud de los modelos que representan la distribución de los datos antes y después del punto de cambio.

De esta forma, las contribuciones de esta tesis son las siguientes: i) se analizan las desventajas de métodos clásicos para detección de cambios, (ii) se introduce un modelo de detección de puntos de cambio que usa dos criterios intuitivos para buscar y declarar un cambio respectivamente, (iii) se propone una solución eficiente para la búsqueda del punto de cambio candidato usando búsqueda ternaria, (iv) se valida la robustez de GOCPD con procesos gaussianos en escenarios del mundo real, en los cuales GOCPD supera los métodos clásicos en términos de tasa de falsos positivos, y (v) se libera una implementación pública de GOCPD.

RESUMEN DE LA TESIS PARA OPTAR
AL GRADO DE MAGÍSTER EN CIENCIAS
DE LA INGENIERÍA, MENCIÓN ELÉCTRICA
POR: JOU-HUI HO KU
FECHA: 2022
PROF. GUÍA: FELIPE TOBAR HENRÍQUEZ

GREEDY ONLINE CHANGE POINT DETECTION

Online change point detection (CPD) aims to detect abrupt changes in time series in real-time, which is crucial for applications that need immediate responses, such as finance and medical signals monitoring. Existing online CPD methods rely their decision on the probability of the latest observations, which can change abruptly when outliers appear, thus leading to high false positive rates.

In this thesis, we propose *Greedy Online Change Point Detection* (GOCPD) — a computationally appealing algorithm that greedily finds a change point. Concretely, it uses ternary search to look for the optimal timestamp that maximizes the likelihood of the models that summarize the data before and after it.

In this regard, the contribution of this thesis is five-fold: (*i*) we revisit existing online CPD methods and address their drawbacks, (*ii*) we introduce a simple and robust online CPD method that uses two intuitive criteria to search and declare a change point, (*iii*) we propose a computationally efficient approach to search for the candidate change point, (*iv*) we validate the robustness of GOCPD with Gaussian processes in real-world scenarios, in which GOCPD outperforms existing methods in terms of false discovery rates, and (*v*) we release a public implementation of GOCPD.

This too shall pass.

Agradecimientos

A mi profesor guía Felipe Tobar por la generosidad y las sugerencias constructivas durante el trabajo de tesis. Al profesor Jorge Pérez, el más inspirador, y a la profesora Jocelyn Dunstan, la más generosa. A los tres por inspirarme en el camino de la investigación en Machine Learning.

A mi familia, por estar, por su sabiduría, por el inmenso apoyo y la infinita tolerancia. A mi papá, por mostrarme el camino de la ingeniería y el ejemplo de un verdadero ingeniero. A mi mamá, por enseñarme lo que es correcto. A mi hermana Jou-I, por prestarme sus audífonos durante la escritura de esta tesis. A mi hermana Jou-Jin, por preguntarme cada dos días cuándo termino la tesis.

A mis amigos, por la buena compañía, por las conversaciones, por los almuerzos, por las juntas, por escuchar todas mis quejas y por las palabras de aliento. A Hojin, por ser el mejor del universo, por tanta buena compañía y por enseñarme la proactividad, la valentía y la generosidad. A Tamara, por apoyarnos en en este camino, por las conversaciones que siempre terminaban siendo sobre comida, por enseñarme la empatía y la humildad. A Martín, por ser el gran amigo de siempre, por recordarme la motivación por la que entré a la carrera, por recordarme que no todo es descansar, por enseñarme el sentido de la responsabilidad. A Bárbara, por la generosidad, por las conversaciones, por la confianza. A Lucas, por enseñarme a cuidar las amistades que tengo y a mejorar mi memoria. A Josefina, por enseñarme a salir de mi zona de confort. A Jorge, por enseñarme la generosidad. A Pablo, por enseñarme que no todo es trabajar tampoco. A Bryan, por ilustrarme un buen ejemplo a seguir. A Matías, por ponerme en los agradecimientos de su tesis.

A cada persona que me dejó un aprendizaje.

A Dios, por todo.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1. Motivation

Detection of abrupt changes in time series is relevant in many fields, such as genomics [1], speech recognition [2], finance [3, 4] and medical signals monitoring [5, 6, 7]. In particular, detecting these changes in real-time is crucial for applications that require immediate responses, such as medical and financial applications. In the context of online detection, preprocessing or filtering the existing outliers in the dataset is not applicable. Consequently, a crucial property of online change point detection (CPD) methods is their robustness to outliers.

Existing online CPD methods are coarsely categorized into two types: Bayesian or non-Bayesian. The former provides uncertainty quantities of the detection, while the latter mainly focuses on measuring the discrepancy of the data statistics before and after the change point. Many of the Bayesian methods for online CPD [8, 9, 10, 11] rely on the standard Bayesian Online Change Point Detection (BOCPD) [12] that recursively models the posterior probability of the elapsed time since the last change. In the formulation of BOCPD, the change point probability depends on the probability of the streaming observations. Therefore, outliers and noisy data may lead to a high false positive rate. Additionally, BOCPD uses the probabilities for each arriving sample as fixed values for the upcoming iterations, which hinders robustness to outliers. On the other hand, non-Bayesian methods mainly rely on the likelihood ratio test [13, 14], which also leads to false positives when the probability of the latest observations decreases given an outlier. Therefore, there is a need for a robust algorithm for detecting changes in real-world time series.

For this purpose, we hypothesize that the change declaration should rely on the complete segment of the so-far observed data since the last change point. Besides, in real-world scenarios, the distribution that summarizes observations between change points is unknown. Therefore, the CPD method needs training the underlying predictive model for each segment.

Additionally, human reasoning may benefit the design of an online CPD method. An

intuitive way to detect a change is to look at the arriving samples and decide whether the so-far observed data is more likely to follow a single model or two different models. When the last observations seem to be distributed differently from the old ones, one does not notify the change instantly to avoid making a false discovery. Instead, a human annotator waits for more observations and, if the upcoming data continue to behave differently, then the human notifies a change as an informed decision.

1.2. Hypothesis

In this thesis, we validate the following hypotheses:

- The online detection of a change point cannot only rely on the last observed data.
- Human reasoning is beneficial for online CPD in the sense of including a period of verification to approve a change point, which leads to more robust detections.

1.3. General objective

The general objective of this thesis is *to propose a simple and robust method for online change point detection that can outperform existing algorithms in the unsupervised context*. In the unsupervised learning context, the labeled change points are not available for the model, then the method has to learn patterns from the unlabeled data. Remarkably, this method cannot assume prior knowledge of the number of change points, nor the parameters of the distributions, nor the length of each segment in the time series.

1.4. Specific objectives

The specific objectives of this thesis are the following:

- To revisit state-of-the-art online CPD methods and identify their common drawbacks.
- To propose a computationally appealing method for online change point detection.
- To assess the proposed method in univariate and multivariate real-world scenarios, such as human activity segmentation and change point detection in electroencephalography signals.
- To release a publicly available implementation of the proposed online CPD method.

1.5. Contributions

The main contributions of this thesis are the following:

1. The proposal of *Greedy Online Change Point Detection* (GOCPD) — an unsupervised online CPD method that uses two simple and intuitive likelihood-based criteria to search and declare a change point, respectively.
2. A computationally efficient approach to search the candidate change point via ternary search, which operates in logarithmic time.
3. The validation of GOCPD using Gaussian processes in real-world applications on online CPD, such as detecting abrupt changes in electroencephalography signals and human activity monitoring datasets.
4. The construction of a public Python implementation of GOCPD with Gaussian processes that allows GPU-acceleration.

The remainder of the thesis is structured as follows. In Chapter 2, we introduce the problem statement and notations. In addition, we provide an overview of some of the most relevant methods for unsupervised online CPD and their limitations. In Chapter 3, we describe our proposal, GOCPD, that performs the change location search efficiently and is robust to outliers. We validate our model with experimental results in both synthetic and real-world data in Chapter 4, in which we compare GOCPD against classic and extended methods. Next, in Chapter 5, we provide further analysis of GOCPD in terms of time complexity, number of evaluations per iteration, and limitations. Lastly, we summarize our work in Chapter 6 and give possible directions for future work.

Chapter 2

Background

2.1. Problem statement

Let $\mathcal{D} = \{(x_t, y_t)\}_{t=1}^{T}$ denote observations of a time series, where $x_t \in \mathbb{R}^D$ is the input at time t, usually set to the timestamp of the observation, and $y_t \in \mathbb{R}^C$ is the multivariate output with C channels. Also, let $\mathcal{D}_{t_i:t_j} = \{(x_t, y_t)\}_{t=t_i}^{t_j}$ denote the observations between $t = t_i$ and $t = t_j$. Let us consider that the underlying distribution of this time series changes abruptly at specific times, called *change points*. We denote the ordered set of m change points in the time series as $\mathcal{C} = \{c_1^*, c_2^*, \cdots, c_n^*\}$, so that $\forall i \in \{2, \ldots, n-1\}$, the sequences of observations $\mathcal{D}_{c_{i-1}^*:c_i^*-1}$ and $\mathcal{D}_{c_i^*:c_{i+1}^*-1}$ are independent and come from different models.

Online CPD aims to detect the change points in $\mathcal{C}$ in an online fashion. Therefore, for each arriving batch of data $\mathcal{D}_{t_i:t_{i+1}}$, the method should notify if a change occurred in the so-far observed data and output the location of the detected change point if applies. In this thesis, we consider that the parameters of the model before and after the change point are unknown. We aim to detect changes in the context of unsupervised learning, i.e., the algorithm should discover the patterns of the time series in unlabeled data.

In the following subsections, we provide an overview of some of the most relevant methods for online change point detection. We coarsely categorize them into probabilistic or Bayesian methods and non-Bayesian or model-free methods. For a in-depth survey of both probabilistic and non-probabilistic online change point detection methods, we refer to [15, 16].

2.2. Bayesian methods

Bayesian methods are particularly relevant for CPD, as they provide uncertainty measures for each detection. The classic work of online CPD from a Bayesian perspective is based on the Product Partition Model and introduced independently by Adams and Mackay [12] and

Fearnhead and Liu [17].

The seminal work of Adams and Mackay [12] introduced the Bayesian Online Change Point Detection (BOCPD) algorithm for handling piece-wise stationary processes. In their formulation, BOCPD estimates the posterior distribution of the current *run-length* r_t, which is the elapsed time since the last change point.

At the same time, Fearnhead and Liu [17] proposed an online CPD algorithm for Bayesian product partition models that requires prior knowledge of the parameters. For each t, they define a state $C_t \in \{0, 1, \ldots, t-1\}$ that corresponds to the time of the most recent change point prior to t. The sequence of states $C_1, C_2, \ldots C_t$ is considered as a Markov chain. Similar to BOCPD, the aim of the work of Fearnhead and Liu [17] is to estimate the posterior distribution for C_t given the observations. This algorithm scales quadratically if these distributions are stored for all t. To decrease the time complexity, they proposed to use particle filters by sampling values of C_t for each t, at the expense of introducing error in the model.

Since both methods essentially model the posterior of the elapsed time since the last change point, and the proposal of Fearnhead and Liu [17] includes sampling techniques that introduce error, we mainly compare our method with BOCPD in what follows.

2.2.1. Bayesian online change point detection (BOCPD)

Adams and Mackay [12] introduced the Bayesian Online Change Point Detection (BOCPD) algorithm, which estimates the posterior distribution of the current run-length r_t given the observations. In their formulation, the observations in each segment are considered i.i.d. with respect to a distribution.

Consider the reasonable assumption that the last change point occurred at time $t = 0$. Let $\boldsymbol{y}_t \in \mathbb{R}^{t \times C}$ be the vector of observations up to time t after the last change point, and r_t the run-length at time t. BOCPD maintains an estimation of the run-length posterior distribution to solve the following optimization problem:

$$r_t^* = \arg\max_{r_t} p(r_t|\boldsymbol{y}_t) = \arg\max_{r_t} \frac{p(r_t, \boldsymbol{y}_t)}{\sum_{r_t'} p(r_t', \boldsymbol{y}_t)}, \tag{2.1}$$

where the joint probability between the run-length and the observations is recursively calculated as

$$p(r_t, \boldsymbol{y}_t) = \sum_{r_{t-1}} p(r_t|r_{t-1}) p(y_t|r_{t-1}, \boldsymbol{y}_{t-1}) p(r_{t-1}, \boldsymbol{y}_{t-1}). \tag{2.2}$$

In Equation (2.2), $p(y_t|r_{t-1}, \boldsymbol{y}_{t-1})$ is the predictive probability of the model given previous observations, $p(r_{t-1}, \boldsymbol{y}_{t-1})$ results from the previous iteration of the recursion, and $p(r_t|r_{t-1})$ is the conditional prior of the run-length, given by

$$p(r_t|r_{t-1}) = \begin{cases} H(r_{t-1}+1) & \text{if } r_t = 0 \quad \text{(change detection resets } r_t\text{)} \\ 1 - H(r_{t-1}+1) & \text{if } r_t = r_{t-1}+1 \quad \text{(no change)} \\ 0 & \text{otherwise,} \end{cases} \tag{2.3}$$

where $H(r) = \frac{p_{\text{change}}(r)}{\sum_{s=r}^{\infty} p_{\text{change}}(s)}$ is the hazard function and $p_{\text{change}}(\bullet)$ is the prior probability of change. The hazard function defines the event rate at time t conditioned on that the event did not occur before t, i.e., it will be at time t or after t. Concretely, in the context of change point detection, the hazard function describes how likely a change point occurs given an observed run-length r_t. BOCPD computes this probability for each r_t, which leads to a time complexity of $\mathcal{O}(T)$.

Then, given Equation (2.3), we can split Equation (2.2) into two cases:

$$p(r_t = 0, \boldsymbol{y}_t) = \Big(1 - H(r_{t-1}+1)\Big) p(y_t|r_{t-1}, \boldsymbol{y}_{t-1}) p(r_{t-1}, \boldsymbol{y}_{t-1}) \tag{2.4}$$

$$p(r_t = r_{t-1}+1, \boldsymbol{y}_t) = \sum_{r_{t-1}} H(r_{t-1}+1) p(y_t|r_{t-1}, \boldsymbol{y}_{t-1}) p(r_{t-1}, \boldsymbol{y}_{t-1}), \tag{2.5}$$

where Equation (2.4) can be interpreted as the change point joint probability at time t, referred to as the *change point probability* and Equation (2.5) is the joint probability of the no-change scenario, referred to as the *growth probability*. We denote these two expressions as P_t^{cp} and P_t^g, respectively.

The space and time complexity per time step is linear in the number of observations so far observed, which scales to $\mathcal{O}(T)$, where T is the total number of data points. If we only choose the K most probable run-lengths, then the per-step complexity would be $\mathcal{O}(K)$.

Note that for numerical precision issues, a common practice is to implement Equation (2.1) in its logarithmic version.

2.2.2. Practical consequences of BOCPD

BOCPD operates as an online algorithm by calculating r_t^* in Equation (2.1) given the observations up to t. Note that the algorithm does not explicitly perform a change detection. In practice, for the current time t, a change point is declared if the change probability P_t^{cp} surpasses the growth probability P_t^g at that time. However, this detection criterion may be counterintuitive since a change point cannot be detected by only looking at the last set of observations, which can be of size 1 in the extreme case. This criterion may lead to many false positives when outliers appear as the model accepts noise as change points.

In practice, $p(r_t, \boldsymbol{y}_t)$ can be expressed as a matrix indexed by timestamps (rows) and run-lengths (columns). Since each r_t cannot take values that are higher than $t - c_{-1}^*$, where c_{-1}^* is the last change point before t, the matrix $p(r_t, \boldsymbol{y}_t)$ is composed of square blocks, which size

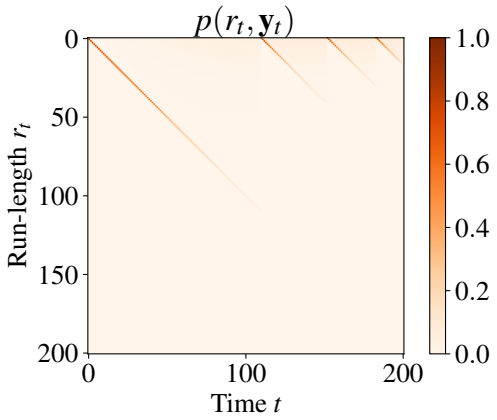


Figure 2.1: Example of diagonal dominant property of $p(r_t, \boldsymbol{y}_t)$ on the first 200 observations of the Well-log dataset [18].

is the number of timestamps between detections. Moreover, each block is diagonally dominant, i.e., its diagonal has larger values than the off-diagonal terms. Figure 2.1 illustrates an example of this property in the first 200 observations of the Well-log dataset [18]. Therefore, the optimization in Equation (2.1) for a given t is in practice between $p(r_t = 0, \boldsymbol{y}_t)$ and $p(r_t = t - c^*_{-1}, \boldsymbol{y}_t)$. Due to the diagonally dominant property of $p(r_t, \boldsymbol{y}_t)$, and given a constant Hazard function, the change probability will eventually surpass the growth probability after a sufficiently long time, independently of the observations.

Besides, as Equation (2.2) is formulated as a recursion, then the predictive distribution $p(y_t|r_{t-1}, \boldsymbol{y}_{t-1})$ may lead to extremely low probabilities when outliers appear. Since the CPD model is recursive, this will also turn all the distribution for future timesteps into very small values.

On the other hand, the change point and growth probabilities of each t are only conditioned to the past of the change point, which prevents verifying the detected change point with new observations. These probabilities are computed only once and cannot be modified in the next timestamps, which limits the possibility of correcting wrong detections.

Additionally, we argue that the prior probability of change $p(r_t|r_{t-1})$ is difficult to parametrize. In real-world settings, the probability of change is often unknown and varies depending on the context.

For these reasons, we conjecture that the resulting r^*_t found by BOCPD may not necessarily reflect the real change points but would tend to be false positives instead.

2.3. Improvements and extensions of BOCPD

As stated in subsection 2.2.2, BOCPD seems to have scarce robustness to outliers, thus leading to high false discovery rates. Therefore, built on BOCPD, there have been many

efforts to propose improvements for this method, either to increase its robustness [10] or to increase its modeling capacity [19, 11]. We describe some of these improvements in the following subsections.

2.3.1. Gaussian process change point detection (GPCPD)

BOCPD assumes observations in each regime are i.i.d., then temporal dependencies are not taken into account. To overcome this drawback, Saatçi *et al.* [11] introduced a nonparametric time series change point detection using Gaussian processes to capture temporal correlations and detect more complex changes, such as changes in the covariance structure of the underlying model.

The Gaussian process (GP) [20] is a Bayesian nonparametric generative model for functions $f : \mathbb{R}^D \to \mathbb{R}$, which is the infinite-dimensional extension of the multivariate normal distribution. A GP, denoted $f \sim \mathcal{GP}(m, K)$, is defined through its mean function $m(\bullet) : \mathbb{R}^D \to \mathbb{R}$ and a covariance kernel $K(\bullet, \bullet)$, which denotes the covariance between two inputs.

In the context of change point detection in time series, the input is the time $x_t = t$, and the output is the observation of the time series y_t, which is given by

$$y_t = f(t) + \epsilon_t, \;\; f \sim \mathcal{GP}(0, K_\theta), \;\; \epsilon_t \sim \mathcal{N}(0, \sigma_n^2), \tag{2.6}$$

where the mean function is set to zero, and the kernel K_θ is parameterized by a set of hyperparameters θ.

Then, the predictive distribution of Equation (2.2) is now given by

$$p(y_t | r_{t-1}, \boldsymbol{y}_{t-1}) = \mathcal{N}(m_t, v_t), \tag{2.7}$$

where

$$\begin{aligned} m_t &= K(\boldsymbol{y}_{t-1}, \boldsymbol{y}_t)^\top \Big(K(\boldsymbol{y}_{t-1}, \boldsymbol{y}_{t-1}) + \sigma_n I \Big)^{-1} \boldsymbol{y}_{t-1} \\ v_t &= K(\boldsymbol{y}_t, \boldsymbol{y}_t) - K(\boldsymbol{y}_{t-1}, \boldsymbol{y}_t)^\top \Big(K(\boldsymbol{y}_{t-1}, \boldsymbol{y}_{t-1}) + \sigma_n^2 I \Big)^{-1} K(\boldsymbol{y}_{t-1}, \boldsymbol{y}_t) \end{aligned} \tag{2.8}$$

Having defined the mean and the kernel functions, we set the GP hyperparameters via maximum likelihood, which has a computational complexity that scales to $\mathcal{O}(T^3)$, where T is the total number of observations.

GPCPD assumes that the observations are samples from a GP and uses the same detection criteria as BOCPD. Standard GPs only model stationary time series, but by including change points in the data, the model can also represent locally smooth time series, which is the case of most real-world time series. The main drawback of this method is its computational complexity, which was $\mathcal{O}(T)$ in BOCPD, but now the GP has to be trained for each iteration, thus scaling to $\mathcal{O}(T^4)$. When pruning is applied, the time complexity is $\mathcal{O}(TKR^2)$, where

K is the number of most-probable run-lengths considered and R is the maximum run-length considered. However, deciding the adequate pruning level is difficult in real-world applications due to the usual lack of prior knowledge about the duration of each state between change points.

2.3.2. Robust Bayesian online change point detection with model selection (RBOCPDMS)

Standard Bayesian models minimizes the Kullback-Leibler (KL) divergence between the learned model and the observations to fit the data, which is not robust to outliers, since its influence function is strictly increasing as the observations move away from the model posterior. Knoblauch *et al.* [21] introduced Robust BOCPD with Model Selection (RBOCPDMS) — a generalization to online model selection with nonstationary data using Generalized Bayesian Inference (GBI) [22, 23], which generalizes standard Bayesian inference to a family of divergences. In GBI, for a model m and a divergence D, the posterior of the model is given by

$$\pi_m^D(\theta_m|\boldsymbol{y}_{(t-r_t):t}) \propto \pi_m^D(\theta) \exp\Big\{ - \sum_{i=t-r_t}^{t} \ell^D(\theta_m|\boldsymbol{y}_i)\Big\}. \tag{2.9}$$

The choice of the divergence D decides the robustness of the model. RBOCPDMS uses the β-divergence between the learned model and the observations, instead of the KL divergence. The authors show that the influence function of the β-divergence does not increase abruptly with the presence of an outlier. Therefore, the posteriors in Equation (2.2) do not decrease abruptly, hence avoiding false detections.

Concretely, instead of minimizing the negative log-likelihood in the case of the KL divergence, i.e., $\ell^{KL}(\theta_m|\boldsymbol{y}_t) = -\log f_m(\boldsymbol{y}_t|\theta_m)$, RBOCPDMS minimizes the Tsallis score, given by

$$\ell^{\beta}(\theta_m|\boldsymbol{y}_t) = -\Big(\frac{1}{\beta} f_m(\boldsymbol{y}_t|\theta_m)^{\beta} - \frac{1}{1+\beta}\int_{\mathcal{Y}} f_m(\boldsymbol{z}|\theta_m)^{1+\beta} d\boldsymbol{z}\Big). \tag{2.10}$$

In this formulation, the likelihood is exponentially down-weighted, thus attaching less influence to observations in the tail of the distribution [21]. Similarly, the log predictive posterior $\log p(y_t|r_{t-1}, \boldsymbol{y}_{t-1})$ is replaced by its β-divergence counterpart:

$$\ell^{\beta}(r_{t-1}, \boldsymbol{y}_{t-1}|\boldsymbol{y}_t) = -\Big(\frac{1}{\beta} f_m(\boldsymbol{y}_t|r_{t-1}, \boldsymbol{y}_{t-1})^{\beta} - \frac{1}{1+\beta}\int_{\mathcal{Y}} f_m(\boldsymbol{z}|r_{t-1}, \boldsymbol{y}_{t-1})^{1+\beta} d\boldsymbol{z}\Big). \tag{2.11}$$

In practice, the integrals in these formulations are intractable. Therefore, Knoblauch *et al.* [21] proposed to leverage that the KL-divergence versions of these posteriors are available in closed form to approximate the β-divergence posterior with a KL-divergence posterior.

Therefore, the β-D posterior for model m and run-length r_t can be approximated as

$$\hat{\pi}_m^{\beta}(\theta_m) = \underset{\pi_m^{KL}(\theta_m)}{\arg\min} \left\{ KL\Big(\pi_m^{KL}(\theta_m) \Big\| \pi_m^{\beta}(\theta_m | \boldsymbol{y}_{(t-r_t):t})\Big)\right\}, \tag{2.12}$$

which is equivalent to maximize the Evidence Lower Bound (ELBO) of m, that is available in closed form for m belonging to the exponential family.

The authors also provide a variation for the optimization algorithm. Full optimization of the parameters for all run-lengths and timestamps requires quadratic time complexity. On the other hand, optimizing with Stochastic Gradient Descent makes the optimization faster, at the expense of lower accuracies. Then, the authors propose a pipeline that mixes both approaches: perform full optimization for the first iterations, and then incrementally refine the model in the next iterations with sampled observations. This optimization procedure is the so-called Stochastic Variance Reduced Gradient (SVRG) inference.

2.3.3. Restarted Bayesian online change point detection (R-BOCPD)

Alami *et al.* [9] introduced Restarted Bayesian Online Change Point Detection (R-BOCPD), which is a pruned version of BOCPD applicable for univariate Bernoulli-distributed samples with changes in the mean of the distribution. In their work, they provide a formal analysis of the change point detection performance in terms of false alarm and detection delay.

In their work, the authors define new concepts for their analysis. Instead of dealing with a run-length, they consider the sequence $\boldsymbol{y}_{s:t}$, whith $s = t - r_t$, thus the predictive probability $p(y_t|r_{t-1}, \boldsymbol{y}_{t-1})$ is expressed as $p(y_t|\boldsymbol{y}_{s:t-1})$. Given that the observations are assumed to follow a Bernoulli distribution $y_t \sim \mathcal{B}(\mu_t)$, the underlying predictive distribution $p(y_t|\boldsymbol{y}_{s:t-1})$ is the Laplace predictor, which is given by

$$\texttt{Lp}(y_t|\boldsymbol{y}_{s:t-1}) = \begin{cases} \dfrac{\sum_{i=t-r_t}^{t} y_i + 1}{r_t + 2} & \text{if } y_t = 1 \\ \dfrac{\sum_{i=t-r_t}^{t} (1 - y_i) + 1}{r_t + 2} & \text{if } y_t = 0. \end{cases} \tag{2.13}$$

A forecaster s is a successive product of $(t-s)$ Laplace predictors starting at time $s = t-r_t$. Note that for each run-length in the original formulation of BOCPD, exists an equivalent forecaster. Then, the loss of this forecaster has the following form:

$$\hat{L}_{s:t} = -\sum_{s'=s}^{t} \log \texttt{Lp}(y_t|\boldsymbol{y}_{s':t-1}) = \sum_{s'=s}^{t} \ell_{s':t}, \tag{2.14}$$

where $\ell_{s:t} = -\log \texttt{Lp}(y_t|\boldsymbol{y}_{s':t-1})$ is the instantaneous loss of the forecaster created at time s.

R-BOCPD assigns to each forecaster a weight $v_{s,t} = p(r_t = t - s|\boldsymbol{y}_{s:t})$. Then, given a constant Hazard function $H(\cdot) = h$ for the sake of clarity, both change and growth probability

in Equations 2.4 and 2.5 can be expressed in terms of this weight as follows:

$$
\begin{aligned}
v_{s,t} &= (1-h)\texttt{Lp}(y_t|\boldsymbol{y}_{s:t-1})v_{s,t-1} && (2.15)\\
&= (1-h)^{t-s+1}h^{\mathbb{I}\{s\neq 1\}}\exp\Big(-\hat{L}_{s:t}\Big)V_s && (2.16)\\
v_{t,t} &= h\sum_{i=1}^{t-1}\texttt{Lp}(y_t|\boldsymbol{y}_{i:t-1})v_{i,t-1}, && (2.17)\\
&= hV_t. && (2.18)
\end{aligned}
$$

where $V_t = \sum_{s=1}^{t} v_{s,t} = \sum_{s=1}^{t-1}\texttt{Lp}(y_t|\boldsymbol{y}_{s:t-1})v_{s,t-1}$ is the initial weight given to the forecaster created at time t, i.e., it is the equivalent to the posterior probability of the run-length equal to 0 in BOCPD.

Instead of computing all the combinations of the forecasters' weights to obtain the value of the initial weight V_t, R-BOCPD prunes the contribution of the forecasters created before some time r. Concretely, the initial weight is set to the following:

$$\mathcal{V}_{r:s-1} = \exp\Big(-\hat{L}_{r:s-1}\Big) \quad \text{for some } r < s. \tag{2.19}$$

Now, the new weight assigned to the forecaster $s \geq r$ at time $t \geq s$ that considers the initial weight $\mathcal{V}_{r:s-1}$ is denoted as $\vartheta_{r,s,t}$, and the change detection criterion is defined as

$$\texttt{Restart}_{r:t} = \mathbb{I}\{\exists s \in (r,t] : \vartheta_{r,s,t} > \vartheta_{r,r,t}\}, \tag{2.20}$$

which is the prunned version of the criterion we described for BOCPD. After a change is detected, R-BOCPD restarts a new forecaster at time $r = t+1$ and delete the forecasters created before time t.

Finally, the constant Hazard h is a hyperparameter of the algorithm. Given a range for the constant Hazard function, the false alarm rate and the detection delay are upper bounded. This is intuitively expected: only considering the prior probability of change, as h is low enough, the algorithm will flag more detections, which decreases the detection delay; and as h is high enough, the algorithm will bound its number of detections, which decreases its false alarm rate.

2.3.4. Other variants of BOCPD

In addition to the aforementioned improvements of BOCPD, there were also several attempts to extend it for different scenarios under some assumptions, such as multi-armed bandits [24, 25]. Agudelo-España *et al.* [8] extended BOCPD to future occurrences of the change points by estimating the remaining time of the current state. Moreno-Muñoz *et al.* [19] considered a hierarchical model that assigns latent classes for the observations between change points. Han *et al.* [26] extends BOCPD with Gaussian processes by including a likelihood ratio test that flags a change in the covariance structure for locally smooth time series modeled by Gaussian processes.

2.4. Non-Bayesian methods

A typical line of non-Bayesian online CPD methods is based on the comparison between two consecutive windows of data, and the decision is typically subject to a likelihood ratio thresholding. This method is called Generalized Likelihood Ratio (GLR), which is an extention of the CUSUM strategy [27]. These methods usually do not require prior knowledge of the data distribution, thus referred to as *model-free* methods. Likelihood-ratio-based methods explicitly compare the probability distribution of consecutive time intervals of fixed window size [27, 28, 29] and detects a change point when the likelihood ratio of consecutive windows surpasses a given threshold.

As this approach mainly considers the discrepancy in the likelihood of two consecutive windows, then, when outliers appear, the likelihood of the new window will change abruptly. Therefore, these methods generally have a high false positive ratio.

2.4.1. Cumulative sum (CUSUM)

Page [27] firstly proposed the Cumulative Sum (CUSUM) strategy to detect a change in a parameter of the model for streaming data. This method is commonly applied for normally distributed data, in which the mean value changes. This strategy accumulates deviations relative to a given target of incoming measurements, and flags a detection when the cumulative sum surpasses a threshold. In practice, the CUSUM method assigns a score for each sampled observation, and defines the cumulative score as

$$S_n = \sum_{k=1}^{n} s_k, \tag{2.21}$$

where s_k is the score for the k-th sample, which computation varies depending on the types of changes to be detected, and n is the index for the current sample.

A *one-sided* transition is a deviation of the model parameter in only one direction, and a *two-sided* transition refers to deviations in both directions (either increasing or decreasing). In a general CPD context, one aims to detect two-sided changes. A two-sided change is detected if any of the following condition is satisfied:

$$\text{(i)} \quad S_n - \min_{0 \le i < n} S_i \ge \nu \text{ (increasing direction), or} \tag{2.22}$$

$$\text{(ii)} \quad \max_{0 \le i < n} S_i - S_n \ge \mu \text{ (decreasing direction),} \tag{2.23}$$

given thresholds ν, μ.

To assess the quality of detection, Page [27] defined the *average run-length* (ARL) as the expected number of observations before a change occurs. When the so-far observed data follow the same distribution, the ARL is a measure of the expense incurred by the scheme when if returns false alarms. On the other hand, after a change occurred, the ARL indicates

the detection delay.

After the classic work of Page [27], further extensions have been proposed. Jeske *et al.* [30] extended the CUSUM method to nonstationary time series by including nonparametric techniques for the monitoring process. Aue *et al.* [31] introduced tests that assess the structural stability of cross-volatilities for multivariate time series to detect a change in the signal covariance. Cho and Fryzlewicz [32] proposed a binary segmentation algorithm based on the CUSUM strategy for high-dimensional time series. It computes CUSUM statistics for each channel so that channels without changes do not influence the detection. In addition, to avoid the need of post-processing changes from different channels, their method automatically identifies changes shared across channels.

2.4.2. Likelihood-ratio-based methods

A standard line of online CPD methods [13, 14] considers two models that fit data before and after a timestamp that splits the data into two windows and computes the ratio between both likelihoods. It detects a change when this ratio exceeds a given threshold.

On the other hand, instead of estimating the probability density of the models before and after a change point, Kawahara and Sugiyama [33] proposed to directly estimate the ratio of these probability densities in an online manner. The direct density-ratio estimation extends the Kullback-Leibler Importance Estimation Procedure (KLIEP) [34, 35] to the online setting.

2.4.3. No-prior-knowledge exponential weighted moving average (NEWMA)

More recently, Keriven *et al.* [36] introduced NEWMA, a model-free method that summarizes old and new data in two statistics using the Exponential Weighting Moving Average algorithm with different forgetting factors, and performs a comparison using Maximum Mean Discrepancy.

Formally, NEWMA considers $\Psi : \mathbb{R}^C \to \mathcal{H}$, a mapping from the observations to a normed space $(\mathcal{H}, ||\bullet||)$. This mapping summarizes the observations to a lower-dimensional space. Then, this mapping is utilized to recursively update a statistic that summarizes the information of the so far observed data, given a forgetting factor. To extract time-varying information, i.e., to compare current observations with respect to older ones, NEWMA computes two statistics, given by

$$\boldsymbol{z}_t = (1 - \Lambda)\boldsymbol{z}_{t-1} + \Lambda\Psi(x_t) \tag{2.24}$$

$$\boldsymbol{z}'_t = (1 - \lambda)\boldsymbol{z}'_{t-1} + \lambda\Psi(x_t) \tag{2.25}$$

where λ and Λ are the forgetting factors, which determine the importance of the current observation to the statistic over the old value of the statistic. Then, a change point is flagged

if $||\boldsymbol{z}_t - \boldsymbol{z}'_t|| > \tau$ for some threshold τ, which can be fixed beforehand or adaptive. Note that, in contrast to sliding-window-based methods, NEWMA does not require keeping any sample in memory, as all information is summarized in z_t and z'_t.

NEWMA uses Random Features for Ψ, and the authors show that the Euclidean distance between averaged random features approximates the Maximum Mean Discrepancy with high probability [36].

A drawback of this method is that the saved statistic summarizes the information of all the so far observed data, which includes the current batch of data and the data before the last change point. Therefore, the old statistic is contaminated with both current and previous data information, thus may not uniquely represent the information of the old data.

Our aim is not to merely detect the changes but also to provide good modeling of the observations to infer additional information from it, e.g., to detect seizures in electroencephalography. Therefore, data modeling becomes essential to get this information, as it cannot be provided by model-free methods. In this thesis, to illustrate the difference between these two categories of algorithms for online change point detection, we also compare our method against NEWMA as the representative of model-free methods.

Chapter 3

Greedy Online Change Point Detection

Prior work in online CPD mostly depends on the probability of the last observed data with respect to that of the data from the previously observed window. This may lead to incorrect detections due to the lack of observations from the previous window. The decision of change detection may be different when comparing the new window with all of the previous observations since the last change point, rather than only the previous window.

Also, as pointed out in the previous section, the detection cannot be made by only looking at a small set of observations, which can have size 1 in the extreme case, because the method will fail in false positives when outliers appear.

To avoid high false discovery rates, we propose to follow human intuition for detecting a change. A human observes the arriving data, and when the distribution of the data seems to change, he/she does not notify a change immediately but waits a short period to make sure that the new distribution is maintained. If this short period is satisfied, then the human expert confirms that a change happened at that time.

Moreover, before a change point appears, a human is also capable of selecting the most probable change point given the data observed so far. However, the expert does not notify the change because the observations are more likely to belong to one distribution than two independent ones.

Following this intuition, we propose *Greedy Online Change Point Detection* (GOCPD). The key insight of GOCPD is to *greedily* save the optimal change point location for each arriving batch of observations and approve it as a real change point if it provides enough discrepancy during a given period. This procedure is performed in two steps: (*i*) efficiently search for the optimal change point location for each iteration and (*ii*) declare this optimal location as a real point of transition between states in the time series when the discrepancy with respect to the original model is high enough.

We describe the algorithm by dividing it into four parts: (*i*) a change point location criterion, (*ii*) an efficient greedy search of the optimal location of the change point, (*iii*) the criterion for approving the selected change point as a real change point, and (*iv*) how the previous steps are implemented in an online fashion.

3.1. Notation

Following the notation introduced in Section 2.1, we consider $c_1^* = 0$ under the realistic assumption that the first segment is initialized at the first observation. We denote the detected changes as $\tilde{\mathcal{C}} = \{\tilde{c}_1, \cdots, \tilde{c}_m\}$, where $n = m$ does not necessarily hold. At each timestamp, we denote $\tilde{c} : \mathbb{R} \to \mathbb{R}$ the function that maps each timestamp to the last detected change point before time t, e.g., for $t \in (\tilde{c}_1, \tilde{c}_2)$, it holds that $\tilde{c}(t) = \tilde{c}_1$, which is not necessarily equal to c_1^*.

Note that, in the online setting, the online CPD algorithm is expected to detect a change point before the next change point arrives. Therefore, in what follows, we consider time series with only one change point for our analysis, which is analogous to the scenario with multiple change points.

3.2. Criterion for the candidate change point location

Though our method operates on an arbitrary number of change points, for the sake of simplicity, we present the detection of a single change point, and the case of multiple change points can be naturally extended from this setting. Therefore, consider datapoints $\mathcal{D}_{\tilde{c}(t):t} \subseteq \mathcal{D}$ with a change point at time $c^* \in \{\tilde{c}(t)+1, \tilde{c}(t)+2, ..., t-1\}$. An intuitive procedure for online CPD is to analyze the observations available up to time t, and identify the change point at a timestamp τ *previous* to the current time t, so that the collections $\mathcal{D}_{\tilde{c}(t):\tau}$ and $\mathcal{D}_{\tau:t-1}$ are independent and given by different models.

Intuitively, this detection can be made based on the fact that the probability of the data coming from two different independent models — i.e., $\mathcal{D}_{\tilde{c}(t):\tau-1}$ comes from a pre-change model m_1 and $\mathcal{D}_{\tau:t}$ comes from a post-change model m_2 —, is larger than that of the data coming from a single one, denoted as m_0. For numerical stability, we consider the log-likelihood ratio given by

$$L_t(\tau) = \log \frac{p(\mathcal{D}_{\tilde{c}(t):\tau-1}|\hat{\theta}_1^\tau)p(\mathcal{D}_{\tau:t}|\hat{\theta}_2^\tau)}{p(\mathcal{D}_{\tilde{c}(t):t}|\hat{\theta}_0)} \tag{3.1}$$

$$= \log p(\mathcal{D}_{\tilde{c}(t):\tau-1}|\hat{\theta}_1^\tau) + \log p(\mathcal{D}_{\tau:t}|\hat{\theta}_2^\tau) - \log p(\mathcal{D}_{\tilde{c}(t):t}|\hat{\theta}_0), \tag{3.2}$$

where $\hat{\theta}_0, \hat{\theta}_1^\tau$ and $\hat{\theta}_2^\tau$ are the maximum likelihood estimators (MLE) of the parameters of the parameters in m_0, m_1, m_2, i.e., the models trained with the complete section up to t, the data before and after τ, respectively. Note that $\hat{\theta}_1^\tau$ and $\hat{\theta}_2^\tau$ depend on the chosen split given by τ.

Remarkably, although Equation (3.2) is a log-likelihood ratio, since the last term $\log \bar{p}(\mathcal{D}_{\tilde{c}(t):t}|\hat{\theta}_0)$ is constant for a given t, the optimization is essentially computed only through the first two terms: $L_t(\tau) = \log p(\mathcal{D}_{\tilde{c}(t):\tau-1}|\hat{\theta}_1^\tau) + \log p(\mathcal{D}_{\tau:t}|\hat{\theta}_2^\tau)$. Therefore, this criterion differs from the thresholding of likelihood-ratio-based methods [27, 28] since we directly optimize the likelihood of the data belonging to two independent models instead of a likelihood ratio.

However, as $L_t(\tau)$ is composed of two likelihoods, it depends on the number of observations. Each of both likelihoods decreases as the number of observations involved increases. Then, it may occur that, for instance, if $t - \tau \gg \tau - 1 - \tilde{c}(t)$, i.e., the model m_2 is modeling considerably more observations than m_1, then the likelihood of m_1 will have more influence in $L_t(\tau)$ than m_2, which is not a desired phenomenon, since one wants both models to weight equally. Therefore, to remove dependency on the number of observations, we consider the average log likelihood instead, i.e. $\log \bar{p}(\mathcal{D}|\cdot) = \dfrac{1}{|D|} \log p(\mathcal{D}|\cdot)$. In this way, we define a *change point location metric* as:

$$s_t(\tau) = \log \bar{p}(\mathcal{D}_{\tilde{c}(t):\tau-1}|\hat{\theta}_1^\tau) + \log \bar{p}(\mathcal{D}_{\tau:t}|\hat{\theta}_2^\tau). \tag{3.3}$$

Note that, by using the average log-likelihood of both sections, we consider *equal contribution* from both m_1 and m_2, regardless of the size of the segments that the models are fitting.

Therefore, searching the optimal change point location given current observations is equivalent to finding the optimal τ that maximizes Equation (3.3). We denote this optimal τ as c_t:

$$c_t = \arg \max_{\tau \in \{\tilde{c}(t)+1,\dots,t-1\}} s_t(\tau). \tag{3.4}$$

3.3. Efficient greedy search of the optimal change point

A naive way to solve Equation (3.4) is to compute $s_t(\tau)$ over all values of $\tau \in \{\tilde{c}(t), \dots, t\}$ for every t, but this will take $\mathcal{O}(t)$ per timestep. Moreover, in the streaming scenario, data arrive in batches. Thus, in cases where the data batch size at each iteration is small — in the limit case, it can be of size 1 — the optimizer of s_t and s_{t+1} are close to each other, then computing all values is redundant. Therefore, to circumvent the time complexity of solving Equation (3.4) at each iteration, we leverage some properties of $s_t(\tau)$, which we describe as follows.

Proposition 3.3.1 Let $\mathcal{D}_{\tilde{c}(t):t}$ be observations of a time series with a change point at $c^* \in \{\tilde{c}(t)+1, \tilde{c}(t)+2, ..., t-1\}$, and let $s_t(\tau)$ be the change point location metric $s_t(\tau) = \log \bar{p}(\mathcal{D}_{\tilde{c}(t):\tau-1}|\hat{\theta}_1^\tau) + \log \bar{p}(\mathcal{D}_{\tau:t}|\hat{\theta}_2^\tau)$. Then, $s_t(\tau)$ has a unique maximum at $\tau = c^*$ and monotonically increases for $\tau < c^*$ and decreases for $\tau > c^*$.

Proof. Let $\mathcal{D}_{\tilde{c}(t):t}$ be observations of a time series with a change point at $c^* \in \{\tilde{c}(t)+1, \tilde{c}(t)+2, ..., t-1\}$. Following the recent developments of Chatterjee and Bhattacharya [37], let θ_1 and θ_2 be the true parameters of m_1 and m_2, which model observations before and after the

true change point, respectively, which we consider equal to the estimated parameters given the true change point, i.e., $\hat{\theta}_1^{c^*}$ and $\hat{\theta}_2^{c^*}$, respectively. Since $\log \bar{p}(\mathcal{D}|\cdot)$ does not depend on the size of $\mathcal{D}$, for $\tau < c^*$, the following expression holds:

$$\lim_{\tau \to c^*} \left| \log \bar{p}(\mathcal{D}_{\delta_1:\tau-1}|\hat{\theta}_1^{\tau}) - \log \bar{p}(\mathcal{D}_{\delta_1:c^*-1}|\hat{\theta}_1^{c^*}) \right| = -h(\hat{\theta}_1^{c^*}), \tag{3.5}$$

where $h(\theta)$ is the KL-divergence rate. For GP models with Gaussian-distributed noise, Chatterjee and Bhattacharya [37] have shown that this rate is given by

$$h(\hat{\theta}) = \log \frac{\hat{\sigma}}{\sigma} - \frac{1}{2} + \frac{\sigma^2}{2\hat{\sigma}^2} + \frac{1}{2\hat{\sigma}^2}\mathbb{E}[\hat{\mu} - \mu]^2, \tag{3.6}$$

where μ, σ are the true mean and noise variance of the GP. As we consider that the model approximates the true parameters, we have that $h(\hat{\theta}_1^{c^*}) = 0$.

On the other hand, as $\tau < c^*$, m_2 is also modeling observations that belong to the first segment (prior to c^*), thus the learned model needs to fit observations from the first segment, then its likelihood will be lower than that of the model learned with data from the second segment. Formally, we have

$$\log \bar{p}(\mathcal{D}_{\tau:t-\delta_2}|\hat{\theta}_2^{\tau}) - \log \bar{p}(\mathcal{D}_{c^*:t-\delta_2}|\hat{\theta}_2^{c^*}) < 0. \tag{3.7}$$

Additionally, by definition of the limit in Equation (3.5), for every $\epsilon > 0$, there exists $\delta > 0$ such that $|\tau - c^*| < \delta \implies |\log \bar{p}(\mathcal{D}_{\delta_1:\tau-1}|\hat{\theta}_1^{\tau}) - \log \bar{p}(\mathcal{D}_{\delta_1:c^*-1}|\hat{\theta}_1^{c^*})| < \epsilon$. Taking the last expression of this implication, we have the following:

$$\begin{aligned}
& -\epsilon < \log \bar{p}(\mathcal{D}_{\delta_1:\tau-1}|\hat{\theta}_1^{\tau}) - \log \bar{p}(\mathcal{D}_{\delta_1:c^*-1}|\hat{\theta}_1^{c^*}) < \epsilon \\
\overset{(3.7)}{\implies} \quad & \log \bar{p}(\mathcal{D}_{\delta_1:\tau-1}|\hat{\theta}_1^{\tau}) - \log \bar{p}(\mathcal{D}_{\delta_1:c^*-1}|\hat{\theta}_1^{c^*}) + \log \bar{p}(\mathcal{D}_{\tau:t-\delta_2}|\hat{\theta}_2^{\tau}) - \log \bar{p}(\mathcal{D}_{c^*:t-\delta_2}|\hat{\theta}_2^{c^*}) < \epsilon \\
\implies \quad & \log \bar{p}(\mathcal{D}_{\delta_1:\tau-1}|\hat{\theta}_1^{\tau}) + \log \bar{p}(\mathcal{D}_{\tau:t-\delta_2}|\hat{\theta}_2^{\tau}) < \log \bar{p}(\mathcal{D}_{\delta_1:c^*-1}|\hat{\theta}_1^{c^*}) + \log \bar{p}(\mathcal{D}_{c^*:t-\delta_2}|\hat{\theta}_2^{c^*}) + \epsilon
\end{aligned}$$

Since the last expression holds for every $\epsilon > 0$, then we get the desired result. The case $\tau > c^*$ is analogous. □

For illustration, Figure 3.1 shows a toy example of the unimodal behavior of $s_t(\tau)$. We generated a synthetic dataset up to $t = 100$ with a change point at $c^* = 50$, where $y_{0:49} \sim \mathcal{N}(0, 0.1^2)$ and $y_{50:100} \sim \mathcal{N}(1, 0.1^2)$. This example simulates a batch of streaming data, thus we set $t = 1.0$ for each τ up to $t = 100$, i.e. the criteria for all τ is evaluated over the data up to $t = 100$. We consider a family of models $m_{0,1,2} \in \{\mathcal{N}(\mu, 0.001^2)\}_\mu$, where μ is the parameter of the model, which is learned via maximum likelihood. Recall that for each τ, m_1 models $\boldsymbol{y}_{1:\tau-1}$, m_2 models $\boldsymbol{y}_{\tau:t}$, and m_0 models $\boldsymbol{y}_{1:t}$. Near $\tau = 50$, m_1 and m_2 maximize the modeling data that comes from the same distribution, then $s_t(\tau)$ effectively peaks at $\tau = 50$ and decreases before and after the change point.

Additionally, the fourth plot of Figure 3.1 shows the learned parameter for each split timestamp τ. Before $\tau = 50$, m_1 observes data that takes value near to 0, thus the learned

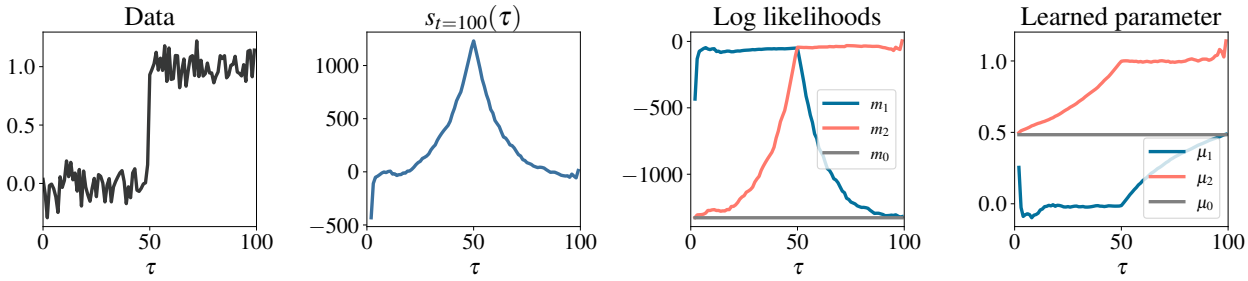


Figure 3.1: **Toy example of unimodal behavior of the change point location metric.** From left to right, the first plot shows synthetic Gaussian observations with a change in the mean value, the second plot shows the change point location metric of Equation (3.3), the third plot shows the log likelihood of $m_{0,1,2}$. Lastly, the fourth plot shows the learned mean of $m_{0,1,2}$ for each τ.

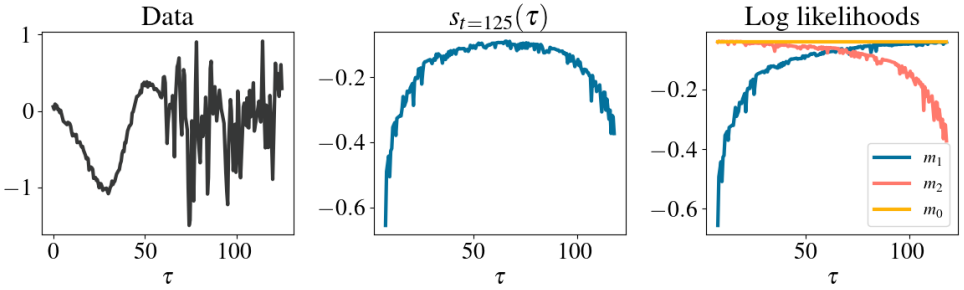


Figure 3.2: **GP example of unimodal behavior of the change point location metric.** From left to right, the first plot shows an synthetic GP data with a change point at $\tau = 60$, the second one shows the change point detection criteria $s_{t=125}(\tau)$, and the third shows the log likelihood of m_1, m_2 and m_0.

parameter μ_1 is close to 0. On the other hand, at $\tau = 0$, m_2 observes the complete dataset. Consequently, the learned parameter takes both segments before and after the change into account, thus μ_2 at $\tau = 1$ is 0.5. As τ approaches $\tau = 50$, the portion of data observed by m_2 that belong to the second segment increases, thus the learned parameter μ_2 starts approaching $\mu_2 = 1$. The scenario after $\tau = 50$ is analogous.

Figure 3.2 illustrates an example of the unimodal behavior of the log-likelihood using GP samples. The data comes from synthetic GP data with a change point at $c^* = 60$. Both segments before and after the change point are generated using an RBF kernel, where the lengthscale changed from $\ell = 10$ to $\ell = 2$. Observe that, effectively, the log-likelihood has a maximum at $\tau = 60$. In this example, $s_t(\tau)$ is slightly noisier than the toy example of Figure 3.1, thus finding its maximum may lead to local maxima. This can be remedied by downsampling $s_t(\tau)$ when optimizing.

Consequently, solving Equation (3.4) is equivalent to finding the maximum of realizations of a unimodal function, which is a well-solved problem in the literature and its solutions have logarithmic complexity [38]. In particular, *ternary search* [38] is a divide-and-conquer

Algorithm 1: TernarySearch(ℓ, r, c_{t-1})

Input: Initial left and right positions $\ell = c_{t-1}, r = t$, previous candidate change point c_{t-1}, tolerance ϵ

Output: $c_t = \arg\max_\tau s_t(\tau)$

1. Divide the sequence in three parts
   $\tau_1 = \ell + (r - \ell)/3$;
   $\tau_2 = r - (r - \ell)/3$;
2. Compute $s_t(c_{t-1}), s_t(\tau_1), s_t(\tau_2)$
3. Search
   if $|\tau_1 - \tau_2| < \epsilon$ **then return** τ_1; // Maximum is found
   // Compare with previous candidate CP
   if $s_t(c_{t-1}) > s_t(\tau_1)$ **then return** TernarySearch(c_{t-1}, τ_1, c_{t-1});
   // Compare τ_1 and τ_2
   if $s_t(\tau_1) < s_t(\tau_2)$ **then return** TernarySearch(τ_1, r, c_{t-1});
   if $s_t(\tau_1) > s_t(\tau_2)$ **then return** TernarySearch(l, τ_2, c_{t-1});
   if $s_t(\tau_1) = s_t(\tau_2)$ **then return** TernarySearch(τ_1, τ_2, c_{t-1});

algorithm that finds the maximum of a unimodal sequence of size n by dividing it into three parts. The time complexity of this algorithm is $\mathcal{O}(\log_3(n))$. In the context of change point detection with GOCPD, the length of the sequence at time t is $t - \tilde{c}(t) + 1$, which can be t in the worst case (when there is no change point yet). Then, instead of evaluating $s_t(\tau)$ on each timestamp, we use ternary search to find its maximum at each iteration, and now the per-iteration complexity of the change searching is $\mathcal{O}(C \log_3(t))$, where C is the cost of computing $s_t(\tau)$, which depends on the chosen underlying predictive model. Note that this time complexity directly stems from using ternary search and does not depend on whether if there is a change point in the so-far observed data or not, as the algorithm will converge to a local maximum. However, local maxima do not hinder the search for the optimal change point, as there is no change point yet. Algorithm 1 summarizes the searching procedure. Note that computing $s_t(\tau)$ requires training models m_1 and m_2 for each iteration.

However, even using a logarithmic-complexity search algorithm, it still has to be performed over all the current observable range of data and needs to be repeated for each iteration of the online scenario. Then, to avoid an exhaustive search over all the data points, we leverage the following property of the optimal change points at each iteration.

Proposition 3.3.2 Given an arbitrary timestamp t, let $c_t = \arg\max_\tau s_t(\tau)$ and $c_{t+1} = \arg\max_\tau s_{t+1}(\tau)$ be the solutions of Equation (3.4) at time t and $t+1$ respectively. Then, it holds that $c_t \leq c_{t+1}$.

Proof. Consider that the time series has a true change point at c^*. Then, in the online setting, two scenarios are possible:

1. $c_t = c^*$, i.e. current candidate is the real change point. Then, given Proposition 1, c_t is the unique maximum of $s_t(\tau)$ and $s_{t+1}(\tau)$. Therefore, $\arg\max_\tau s_t(\tau) = \arg\max_\tau s_{t+1}(\tau) \implies$

$c_t = c_{t+1}$.

2. $c_t \neq c^*$, i.e. current candidate is not the true change point. This yields more cases:

   (*i*) $t + 1 < c^*$, i.e. the true change point is not observed yet. Then, the candidate c_{t+1} can be any timestamp of the observed data, and the variation of the candidate's position is not relevant.

   (*ii*) $t \geq c^*$, i.e. the true change point is observed already but not selected as candidate. Note that in this case $c_t < c^*$. This happens when $c^* - t$ is small, i.e. few datapoints from the new segment are observed so far, thus the change point position at c^* is not conclusive yet. In the next iteration, $\mathcal{D}_{t+1}$ is observed, and since it belongs to the segment after c^*, then more information is given to the algorithm to select the candidate change point, then c_{t+1} will be closer to c^*. Therefore, $c_{t+1} \geq c_t$.

□

This property guarantees that, for a resulting optimal change point c_t of an iteration, the real change point is never located before c_t. Moreover, $c_{t'} \geq c_t$ for all $t' > t$. Therefore, timestamps that are located before the resulting optimal change point of the previous iteration do not need to be evaluated in the current iteration.

Following this result, we save the optimal change point with the highest s_t at each iteration as a *candidate* change point, and for the next iteration, only the timestamps after c_t are evaluated in order to find the maximum of $s_{t+1}(\tau)$. We refer to the resulting interval of timestamps located after the previously saved candidate change point as the *effective* interval size.

3.4. Criterion for the change point approval

When no change point is present in the thus-far observed data, the candidate change point could be located at any of the current timestamps depending on the result of Equation (3.3). And this candidate may not change until the real change point appears. However, this candidate is not the real change point. Therefore, we need a criterion for approving a candidate as a real change point.

Intuitively, the model should not detect any change when the likelihood of m_0 is large enough, where the topology of the neighborhood of m_0 is given by the posterior covariance of the model. Following this intuition, recall that the GP observations $\mathcal{D}_{\tilde{c}(t):t}$ follow a log-density given by a *Mahalanobis distance* between the posterior mean and the observations, defined by

$$d_m(\mathcal{D}) = \sqrt{(\boldsymbol{y} - \boldsymbol{\mu_x})^T \Sigma_{\boldsymbol{xx}}^{-1} (\boldsymbol{y} - \boldsymbol{\mu_x})}, \tag{3.8}$$

where $\boldsymbol{x} = \{x_t\}_t$ and $\boldsymbol{y} = \{y_t\}_t$ are the input and output respectively, and $\boldsymbol{\mu_x}$ and $\Sigma_{\boldsymbol{xx}}$ are the posterior mean and variance of the model m on $\boldsymbol{x}$. This is a multi-dimensional

generalization of measuring how many standard deviations is a point away from a distribution. Note that evaluating with Equation (3.8) is equivalent to evaluate the GP log posterior of m. Nevertheless, this expression brings a clearer intuition of the distance between the observations and the posterior mean.

Besides, $d_m(\mathcal{D}_{\tilde{c}(t):t})$ depends on the dimension of $\mathcal{D}_{\tilde{c}(t):t}$, that is $|\mathcal{D}_{\tilde{c}(t):t}| = t - \tilde{c}(t) + 1$, which varies over time. To avoid getting a higher value for the criterion given larger data sizes, normalization of $d_m(\mathcal{D})$ is required. We then approximate the normalization of the Mahalanobis distance as follows:

$$\bar{d}_m(\mathcal{D}) = d_m(\mathcal{D})^{2/|\mathcal{D}|}. \tag{3.9}$$

Lastly, the criterion consists of a threshold over the normalized Mahalanobis distance of both the data before and after the candidate change point with respect to the posterior distribution of m_0:

$$\bar{d}_{m_0}(\mathcal{D}_{\tilde{c}(t):c_t-1}) > \nu_1 \wedge \bar{d}_{m_0}(\mathcal{D}_{c_t:t}) > \nu_2, \tag{3.10}$$

where the thresholds $\nu_{1,2}$ control how strict this criterion is. A higher ν_i implies that a higher discrepancy with respect to the model posterior is allowed without approving a change, thus fewer change points are detected.

This detection criterion differs from using a threshold for the distance of the complete sequence $d_{m_0}(\mathcal{D}_{\tilde{c}(t):t})$. When using a threshold for the normalized Mahalanobis distance for the complete section, the covariance between observations before and after the candidate change point is taken into account. However, in our formulation, as we consider that the observations before and after the change point come from two independent models, the correlation between both sequences is not included in the criterion for the detection approval.

Importantly, setting independent thresholds to both segments prevents false detections with outliers. For instance, if an outlier is present at the second segment $\mathcal{D}_{c_t:t}$, only the criteria for the second segment is satisfied, then the candidate change point is not declared as a real one. For a candidate change point to be declared as real, the second segment $\mathcal{D}_{c_t:t}$ must behave sufficiently different than the previous one so that m_0 changes its parameters largely to fit the complete segment $\mathcal{D}_{\tilde{c}(t):t}$, and then the Mahalanobis distance for the first segment $\mathcal{D}_{\tilde{c}(t):c_t-1}$ with respect to the posterior mean of m_0 exceeds the threshold.

3.5. Online implementation

Algorithm 2 summarizes the detection procedure of GOCPD. To increase robustness to outliers, it notifies a change point when Equation (3.10) is satisfied for a number $k_{\max}$ of iterations, which is a hyperparameter that depends on the context of the CPD task. Since ternary search is inherently implemented as recursion, the method needs to train m_1 and m_2 at each iteration. In practice, we store the learned parameters at each iteration as initial condition for the next stage.

Algorithm 2: Greedy Online Change Point Detection (GOCPD)

Input: $X \in \mathbb{R}^{T,D}, Y \in \mathbb{R}^{T,C}$
Initialize model parameters $\theta_{0,1,2} = \theta_{0,1,2}^{\text{prior}}$, last change point $\tilde{c}(t) = 0$, threshold $\nu_{1,2}$, initial counter $k = 0$, maximum counter $k_{\max}$, initial time t, initial candidate $c_t = 1$, initial window size T_{ini};
for $t < T$ **do**
  // Wait for a window size T_{ini} to start detecting
  if $t - \tilde{c}(t) < T_{ini}$ **then**
    continue;
  end
  // Current observations since last detected change point
  $\boldsymbol{x} = X_{\tilde{c}(t):t}$; $\boldsymbol{y} = Y_{\tilde{c}(t):t}$;
  // Get current candidate change point
  $c_t \leftarrow \mathit{TernarySearch}(x, y, c_{t-1})$;
  // Evaluate change point detection criterion
  if $d_{m_0}(\boldsymbol{y}_{1:c_t}) > \nu_1 \wedge d_{m_0}(\boldsymbol{y}_{c_t+1:t}) > \nu_2$ ***and*** $c_t = c_{t-1}$ **then**
    $k \leftarrow k + 1$;
    if $k > k_{\max}$ **then**
      Change point is detected: $\tilde{c}(t) \leftarrow c_t$;
      Reset model parameters $\theta_{0,1,2} \leftarrow \theta_{0,1,2}^{\text{prior}}$;
      $k \leftarrow 0$
    end
  end
end

In simple words, for each arriving batch of observations, GOCPD updates the optimal change point location by maximizing the likelihood in Equation (3.4) which considers that the data belongs to two independent models $m_{1,2}$. Then, GOCPD evaluates whether this data split provides enough evidence of the data not belonging to only one model. The criterion for this decision is the normalized Mahalanobis distance. If this distance surpasses a context-dependent threshold for enough observations, GOCPD declares the candidate change point as a real detection.

Figure 3.3 illustrates the detection procedure in the online setting. GOCPD saves a candidate change point at each iteration, and only the timestamps located after that candidate are evaluated in the next iteration. Notably, before the real change point arrives, GOCPD selects the optimal location for the candidate change point, but does not approve it as a real change, since the condition of Equation (3.10) is not satisfied (Figure 3.3.a) or the period in which Equation (3.10) is satisfied is yet not long enough (Figure 3.3.b).

Note that every time that a change point is detected, the algorithm waits a time T_{ini} to start detecting again. Therefore, it prevents abrupt changes in the likelihoods when the data before the detected change point are dropped and only a few observable data are left. This is under the assumption that the minimum distance between consecutive change points is larger than T_{ini}.

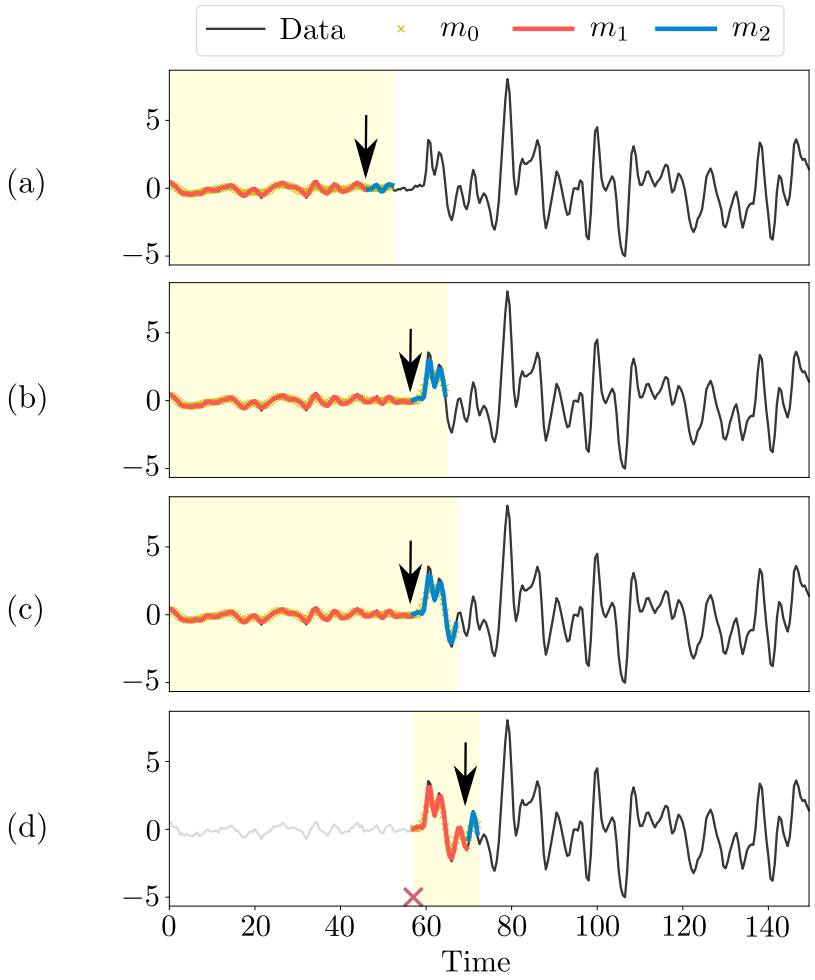


Figure 3.3: **Online detection of GOCPD.** Yellow region shows the observed data. The red cross represents a detected CP. (a) A candidate change point is saved, which is the intersection between red (m_1) and blue (m_2) curves, indicated with a black arrow. and only the blue section accounts for the next iteration. (b) GOCPD updates the candidate change point as observations unfold. (c) The candidate change point is the same as the previous iteration. (d) The candidate change point did not move after a given period, thus GOCPD declares it as a detected change; then, the model waits for a initial window to locate the candidate change point again.

When Equation (3.10) is satisfied for the given period, the candidate change point is identified as a real one (Figure 3.3.d). Following the work of Alami *et al.* [9], after the change detection, since we consider that the data points in different segments are independent, the observations located before the detected change point are *dropped*. In addition, to minimize the regression error, the models $m_{0,1,2}$ takes the parameters of m_2 as initial condition after the change detection, since m_2 has only modeled observations that are located after the declared change point.

A key property of GOCPD is that it detects a change point at a period after the real change occurs. This is essential in change point detection, otherwise, the detected changes

are simply outliers.

Notably, since GOCPD efficiently searches in the timestamps previous to the current time, the resulting detections are not limited to the intersection between fixed windows nor batches of streaming data but can be located in any timestamp since the last change point, in contrary to sliding-window based methods [33, 39, 40], for which the detections are always located at the end of a window. Therefore, in GOCPD, there is no need to choose a proper window size, which is a difficult problem without a universally good solution [36].

On the other hand, although we stated that a change point should be detected before the next one arrives, if a change point c_2^* appears before the previous one c_1^* is detected, GOCPD is still able to at least detect one of the changes, which would be the one that satisfies the detection condition.

Most online CPD methods [10, 12, 11] refer to splitting the data into train and test sets and learning the hyperparameters in the train set. However, we argue that this approach is not applicable in reality when the distributions of new data are unknown. Therefore, the only utility of the training set is to learn the hyperparameters of the first section of the data, and learning is also needed for the upcoming observations, as GOCPD does.

Note that model-free CPD methods first decides if there exists a change point in the data. If so, then it estimates the change point location from the data. We invert this order, as we first locate the most likely location for a change point, and then decide whether to approve or not this candidate as a real change point.

Remark 1. Difference with respect to likelihood-ratio-based methods

We acknowledge that the likelihood ratio has been widely used [28, 29, 27] for change point detection. However, it is important to clarify that GOCPD essentially differs from likelihood-ratio-based methods since it does not compute a likelihood *ratio* but directly optimizes the likelihood of the two models that fits the data before and after a candidate change point respectively to search for the optimal change point location. In addition, unlike likelihood-ratio-based methods, the thresholding in GOCPD is computed over the discrepancy of the observations with respect to the no-change scenario to declare the candidate as a detection.

Remark 2. Change detection for better data forecasting

BOCPD can be viewed as a method to predict the upcoming observations more precisely given the change points. The approach in BOCPD is analogous to predict the data with m_0 in GOCPD. However, since BOCPD does not prune the data before a detected change point, the underlying predictive model $p(y_{t+1}|r_t, y_t)$ may contain observations before the last declared change point depending on the run-length r_t considered, thus providing noisy modeling. In this sense, in addition to the objective of detecting changes, GOCPD also provides models capable of predicting the upcoming observation. After selecting the optimal candidate change

point location for each arriving batch of data, m_1 always models observations that are located before the change point. On the other hand, after a detection, the observations modeled by m_2 are certainly located after the change point. Therefore, the trained m_1 can predict observations before the change point, while after a change point detection the trained m_2 can forecast the data after the change point.

Chapter 4

Experiments

We evaluated our method on both synthetic and real-world datasets that contain abrupt changes in the underlying dynamics. To increase the flexibility when fitting the data, we used nonparametric models as the underlying predictive model. Specifically, we used Gaussian processes [20] for univariate signals and Multi-output Gaussian processes [41] for multivariate data. We include comparisons with baseline models in terms of detection quality and execution time. Our code is implemented with GPyTorch [42], and all experiments were executed on a laptop with an Intel Core i7-9750H 2.6GHz CPU and NVIDIA GTX 1660 Ti 6Gb GPU. We include the code to replicate our experiments in our GitHub repository[1].

4.1. Synthetic data

In this section, we present results in artificially generated time series that contain change points. We show that BOCPD has an extremely low precision by flagging a high number of false positives. In addition, we demonstrate the effectiveness of GOCPD in avoiding high false discovery rates.

Datasets

We generated four time series of 2,000 points from a GP with RBF kernel, i.e., we generated $\{y_t\}_{t=1}^{2000}$ from the following:

$$y_t = f(t) + \epsilon_t, \quad f \sim \mathcal{GP}(\mu, K), \quad \epsilon_t \sim \mathcal{N}(0, \sigma_n^2),$$

$$K(x, x') = \sigma_f \exp\left(-\frac{1}{2\ell^2}\|x - x'\|^2\right),$$

[1] https://github.com/jouhui/GOCPD

where we allowed changes in the hyperparameters $\{\ell, \sigma_f, \mu, \sigma_n\}$, which correspond to the lengthscale of the kernel, the output scale of the kernel, the mean of the Gaussian process, and the noise variance of the likelihood, respectively. The change locations were randomly chosen timestamps with a minimum difference of 50 observations between changes to allow the method to learn sufficiently from the data and make the changes noticeable. We provide details of these datasets in Appendix A.

Experimental setup

We divided each time series in a 30 %-70 % (train-test) split, in which the training section was used to tune the hyperparameters of GOCPD, such as the thresholds $\nu_{1,2}$, the initial window size, and the maximum counter $k_{\max}$. We evaluate the results in terms of sensitivity, or true positive rate (TPR) and precision, or positive predictive value (PPV). The sensitivity measures how many of the real changes did the method detect, and the precision is a metric for how many detections were effectively real changes. These metrics are given by

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \qquad \text{PPV} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{4.1}$$

where TP, FP, and FN are true positives, false positives, and false negatives respectively. The intuitive metric for change point detection is the false positive rate. However, the computation of the false positive rate needs the number of true negatives, which is considerably higher than TP, FP, and FN, since the distribution of the timestamps that correspond to changes versus that of the non-change timestamps is highly imbalanced. Therefore, we use PPV as an alternative metric for false positive ratio.

Results

Figure 4.1 illustrates the detections made by BOCPD and GOCPD over synthetic GP time series. Overall, BOCPD flagged numerous false positives, which is particularly noticeable in the time series with changes in lengthscale.

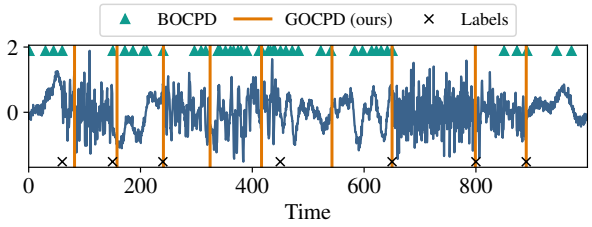
On the other hand, for GOCPD, changes in lengthscale, output scale, and mean were successfully detected. Nonetheless, GOCPD also detected some false positives for these three time series. However, although those points were not labeled change points, the dynamic of the underlying GP at those points clearly changed, e.g., at $t = 400$ in the time series with changes in mean and at $t \approx 550$ in the time series with changes in lengthscale. Figure 4.1.d shows the detections for the case of change in the noise magnitude, in which GOCPD did not detect the first and third change points. However, this corroborates the robustness of our proposal since it is capable of effectively recognizing the change in variance as simply noise.

Table 4.1 summarizes the results on synthetic datasets. BOCPD achieved perfect TPR in two datasets, but its PPV was considerably lower. In contrast, our method returned a lower TPR than that of BOCPD, but consistently outperformed BOCPD in terms of PPV

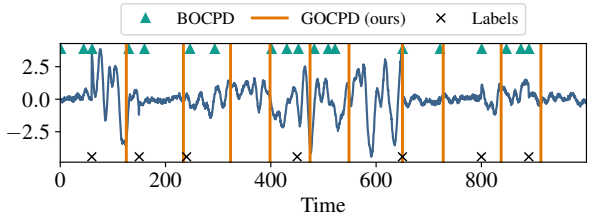
Table 4.1: **True positive rate (TPR) and positive predictive value (PPV).** BOCPD detected all of the real changes in two datasets, but returns considerably low PPV. In contrast, GOCPD outperformed BOCPD in terms of PPV for all datasets.

Method	Change in ℓ		Change in σ_f		Change in μ		Change in σ_n	
	TPR	PPV	TPR	PPV	TPR	PPV	TPR	PPV
BOCPD	0.86	0.13	**1.00**	0.35	**1.00**	0.33	**0.86**	0.26
GOCPD (ours)	**1.00**	**0.78**	0.86	**0.60**	0.86	**0.67**	0.71	**0.83**

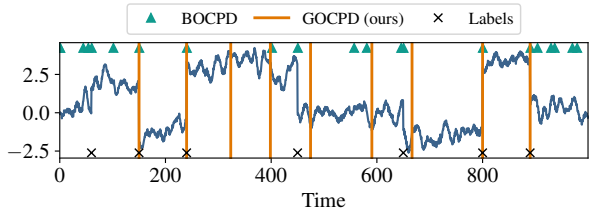
for all synthetic time series, i.e., most of the detected changes of GOCPD corresponded to real labels.



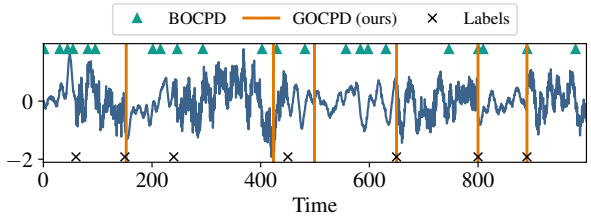
(a) Change in lengthscale.



(b) Change in output scale.



(c) Change in mean value.



(d) Change in noise variance.

Figure 4.1: **Change point detection on synthetic GP data.** In each figure, the change points consist of changes in each of the parameters of the RBF kernel. Black crosses represent the labeled changes, green triangles represent detections of BOCPD, and orange lines represent the detections of GOCPD. Overall, BOCPD notified a high number of false detections, while GOCPD returned the highest precision.

4.2. Real world data

In this section, we present the validation of GOCPD in real-world scenarios. We show that our proposal reaches the lowest false discovery rate in all datasets, while the baseline methods either flag numerous false positives or miss many labeled changes.

Datasets

We validated our proposed GOCPD method on four real-world benchmarks in both univariate and multivariate scenarios:

- **Neonatal Electroencephalography (EEG) dataset** [43]. It is a dataset of multichannel EEG recordings from 79 human neonates with brain seizures admitted to the Neonatal Intensive Care Unit (NICU) at the Helsinki University Hospital. It also includes the visual interpretation of the EEG by three human experts that label the segments that correspond to brain seizure. We considered a consensus of at least one expert to approve a timestamp as a seizure. Then, the change points corresponded to the timestamps in which the signal changed from seizure to non-seizure or vice versa. We present results with the first channel of the subject #1.
- **Human Activity dataset**. It is a subset of the Human Activity Sensing Consortium (HASC) challenge 2011[2]. It provides human activity information collected by portable three-axis accelerometers, in which the person is asked to do six actions: *stay, walk, jog, skip, stair up* and *stair down*. The change points in the dataset correspond to transitions between human activity actions. We present detection results for the subject #671 in both univariate and multivariate scenarios by using one (x-axis) channel and three channels.
- **Respiration dataset**. It is a univariate dataset from the UCR Time Series Data Mining Archive[3]. It contains recordings of patients' respiration measured by thorax extension as they wake up. The changes were manually labeled by a medical expert.
- **Well-log dataset** [18]. It is a classic benchmark of univariate change point detection that records 4,050 nuclear magnetic resonance measurements while drilling a well. Changes in this dataset correspond to abrupt changes in the resonance measurements.

Models

For GOCPD, we used Gaussian processes [20] for the univariate datasets and Multi-output Gaussian processes [41] for the multivariate dataset. We consider RBF kernels for all datasets except for Well-log, in which we used the Dirac delta function for the covariance $K(x, x') = \mathbb{1}_{x=x'}$ due to observed uncorrelateness.

We compared our results against three Bayesian methods and one model-free method. We considered BOCPD [12], GPCPD [11], and RBOCPDMS [21] as our Bayesian baselines, and NEWMA [36] as the representative baseline of model-free methods. For BOCPD and RBOCPDMS, we considered their pruned version by selecting the $K = 50$ most probable run lengths. Following [12], we considered constant hazard functions for BOCPD-based methods.

[2] http://hasc.jp/hc2011/index-en.html
[3] https://www.cs.ucr.edu/~eamonn/time_series_data_2018/

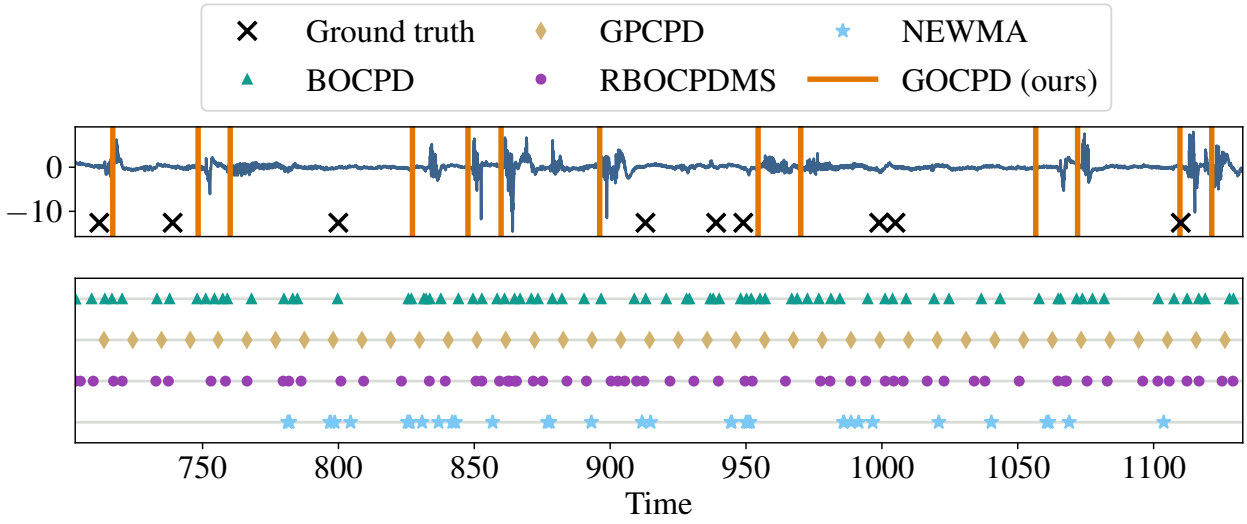


Figure 4.2: **Change point detection on the EEG data**. For the sake of clarity, we illustrate the detected changes of the baseline methods in the plot at the bottom. Baseline methods detected a considerably high number of false positives, while GOCPD is robust to slight changes.

Regarding NEWMA, it declares a timestamp t as a change point whenever the distance between both old and current statistics surpasses a threshold. Therefore, NEWMA may consider all timestamps after a real change point as detections, as the old statistic still contains information from the observations before the change point, which would lead to an extremely high false positive rate. To remedy this issue, we considered a flagged timestamp t as a detected change point only if $t-1$ was not flagged. We used windows of size 100 for all datasets, except for the EEG dataset, for which we considered size 300, and the Well-log dataset, for which we used size 50. We chose these values by considering the minimum distance between consecutive change points in the dataset. We report the implementation details and the hyperparameter choices in Appendix B. For BOCPD, RBOCPD, and NEWMA, we relied on officially released codes.

Experimental setup

We evaluated the detection performance through the resulting negative log-likelihood (NLL) and mean absolute error (MAE) of the learned models evaluated on the dataset split by the detected change points. In addition, to quantitatively compare the number of false detections, we report the false discovery rate FDR, computed as $\text{FDR} = 1 - \text{PPV}$. This metric measures the proportion of false detections in the total set of detected changes. For all methods, we report the averaged values in five independent runs, in which the parameters of the models are randomly initialized. To visualize the detected changes of each method, we select the best performing run among the five repetitions.

4.2.1. EEG dataset

Figure 4.2 shows the detected changes for the EEG dataset. Most of the detections of NEWMA corresponded to false positives, and some of the real abrupt changes are not detected, for instance, at timestamps before $t = 770$.

Both BOCPD and RBOCPDMS detected all changes in the ground truth. However, aside from the ground truth, most detections were false positives, even when the signal did not have any evident change, e.g., for $t \in (1000, 1050)$. The false discoveries are particularly noticeable when the variance of the signal increases, e.g., for $t \in (850, 900)$. Furthermore, even allowing not-labeled changes that consist of sudden peaks in the signal, e.g., for $t \in (850, 900)$, many of the detections of BOCPD and RBOCPDMS were essentially only one, but many consecutive detections were flagged. The case of GPCPD is similar, as it mostly detected false changes, which seems to follow the prior probability of change, as outlined in subsection 2.2.2.

In contrast, GOCPD identified considerably fewer detections. GOCPD missed some labeled changes, which generally corresponded to transitions between high-variance data to low-variance data, e.g., the changes at $t \approx 800$ and $t \approx 1000$.

Recall that the ground truth derived from the timestamps in which the consensus of the experts changes between seizure and non-seizure. These annotations were not absolutely accurate in the sense that the change occurs precisely at the labeled timestamp. For instance, for the labeled change near $t = 750$, GOCPD detected a change slightly after the ground truth, but it was qualitatively more accurate to the abrupt change that appeared after $t = 750$. A similar phenomenon occurred for the change at $t = 950$.

In addition, the annotations only partition the signal into seizure and non-seizure segments. However, noticeable changes may also arise inside both categories, for instance, in the intervals $(800, 900)$ and the remaining interval after $t = 1100$, which were detected by GOCPD. In contrast to the baseline methods, the false positives of GOCPD effectively corresponded to sudden jumps of the signal, which were qualitatively acceptable not-labeled changes. Notably, our method did not notify a change whenever an abrupt jump appeared, e.g., in the interval $t \in (860, 900)$. These results highlight that our proposal is more robust than model-free and BOCPD-based methods, and it is capable of defining more accurate starting points of change.

4.2.2. Human Activity dataset

Figure 4.3 pictures the performance of GOCPD against baseline methods in the univariate Human Activity dataset. It shows that BOCPD could detect the most evident changes, e.g., the transition between stillness and significant movements. Again, detections of GPCPD were not accurate. Besides, the figure shows that NEWMA has considerably lower accuracy than the other methods, as it returned some duplicated consecutive changes and missed most of the labeled changes. On the other hand, the false positives of GOCPD that are located between

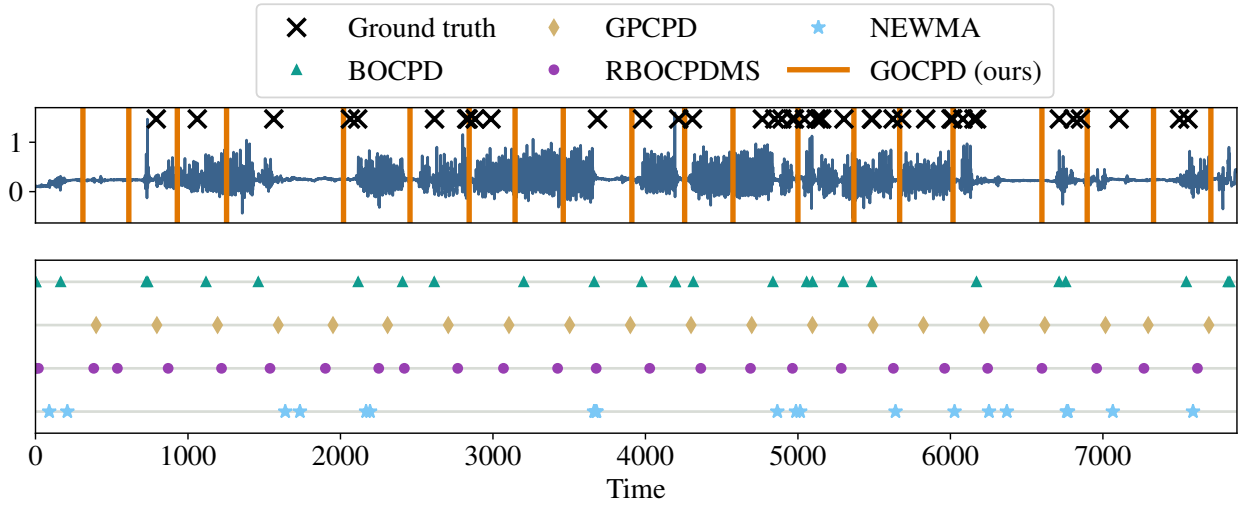


Figure 4.3: **Results in univariate Human Activity dataset**.

$t = 3000$ and $t = 4000$ are attributable to the chosen hyperparameters in the training set. As shown in Figure 4.3, the changes in the signal variance in the last 70 % of the dataset have higher magnitudes than the first 30 % of the dataset. Therefore, the tuned thresholds had smaller values than those required in test data. These results evidence the discrepancy of the change point detection datasets, which suggests the need for adaptive thresholds, which we leave for future work.

Figure 4.4 illustrates the multivariate case of the same dataset. Similar to the univariate experiment, most of the results of NEWMA corresponded to real changes, but it also missed many of the labeled changes. In this case, most of the detections of BOCPD corresponded to false discoveries, as we discussed in subsection 2.2.2, while GOCPD only flagged two false detections. On the other hand, RBOCPDMS effectively had a more robust behavior in comparison with BOCPD, as it only detected two false positives, similar to GOCPD. However, in this case, RBOCPDMS is too robust and produced more false negatives. This may be due to the use of β-divergences, as it has a unique maximum and observations that are farther from the model posterior than that maximum are considered as outliers [21]. Therefore, depending on β, RBOCPD would consider some extremely abrupt changes as outliers, thus leading to false negatives.

This dataset indicates the type of changes that GOCPD can and cannot detect. Since the underlying predictive model is a GP in this case, when the current observations correspond to a GP with small variance, if the time series changes to a state with higher variance, the original m_0 — which is a GP with small variance — will have to change its hyperparameters to fit both segments of observations with small and high variance respectively. Therefore, the normalized Mahalanobis distance will be higher, then GOCPD will detect a change point. On the other hand, if the current observations correspond to noisier samples of the time series, the GP will have a higher variance. However, when the time series changes to a less noisy state, the original GP can also model data with smaller variance. Consequently, the posterior mean of the model still closely follows the new observations, then GOCPD may miss this change.

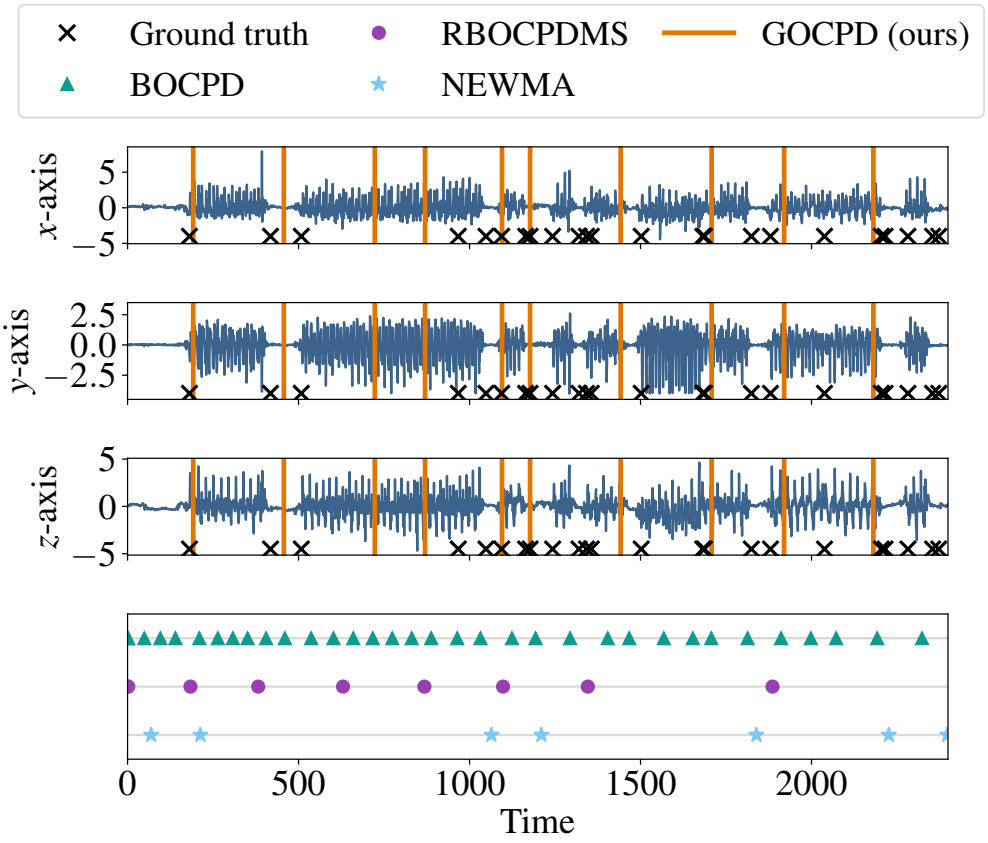


Figure 4.4: **Results in multivariate Human Activity dataset**. Each channel corresponds to the acceleration in $x-y-z$ axis respectively. For the sake of clarity, we present the detections of baseline methods in a different plot at the bottom.

4.2.3. Respiration dataset

Figure 4.5 shows the detection results for the Respiration dataset. Again, NEWMA detected many false positives, and some of its detected changes did not correspond to the real transitions, e.g., at $t \approx 750$. Regarding BOCPD, most of its detections were false positives and corresponded to very small changes of the signal magnitude, e.g., at $t < 500$. Surprisingly, instead of returning more robust results than BOCPD, RBOCPDMS identified even more false positives, which may be attributable to the high variance of the signal, which is hard to model using Bayesian Linear Regressions. On the contrary, the detections of GOCPD were overall closer to the perceptible changes, with some false positives, e.g., at $t \approx 1,700$. Nonetheless, the false positive rate was considerably lower than those of the BOCPD-based methods.

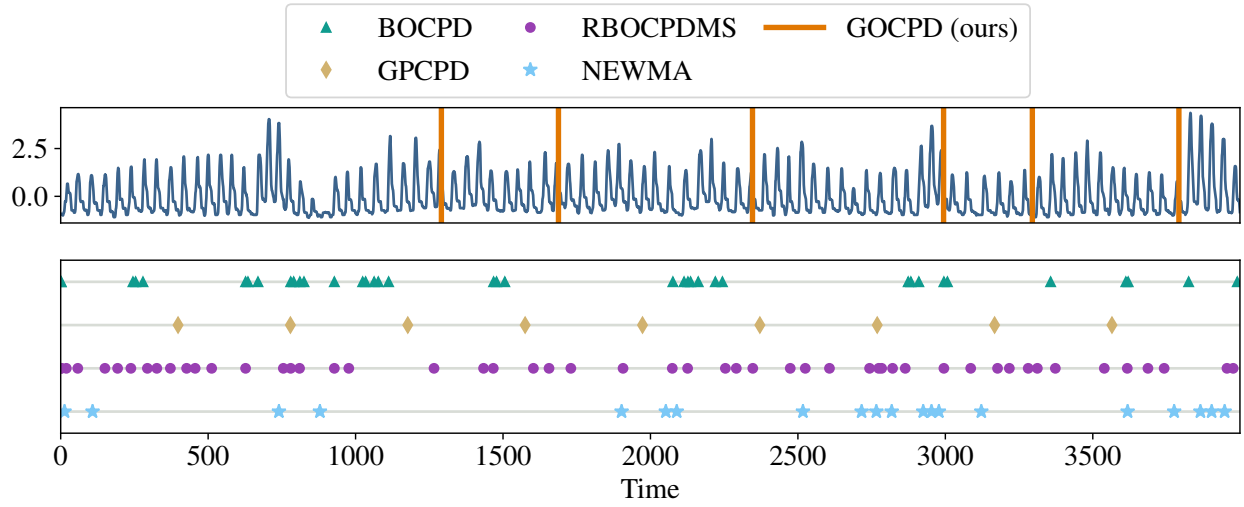


Figure 4.5: **Results in the Respiration dataset.** GOCPD effectively detects changes in the distribution, while BOCPD and ROCPDMS notified a large number of false detections.

4.2.4. Well-log dataset

Figure 4.6 shows the results of the detections for the Well-log dataset. The detections of GPCPD followed the practical consequences we described in subsection 2.2.2, as its detections were in practice owing to the prior probability of change. Therefore, it practically flags changes every a certain period, independent of the nature of the signal. On the other hand, NEWMA failed to detect real changes, and most of its detections were false alarms. As shown in [21], RBOCPDMS effectively returned a more robust detection than that of BOCPD but also missed some real changes, e.g., the set of changes located at $t \approx 2500$. Regarding GOCPD, it missed some of the real changes due to the dissimilarity of the changes between the training set and the test set. Interestingly, the changes missed by GOCPD were also missed by RBOCPDMS. Note that, unlike BOCPD and GPCPD, our method was robust to the outliers in $t \in (1000, 1500)$. Since the training set includes the outlier located at $t \approx 1200$, GOCPD learned to consider those types of shifts as outliers. Therefore, this is the same reason for which GOCPD did not detect the last labeled change at $t \approx 4000$, as it considered it as an outlier.

Table 4.2 summarizes the detection results for those datasets that provided labels of the true change points. Importantly, GOCPD achieved the lowest FDR for all datasets. In particular, for the multivariate Human Activity dataset, the FDR of GOCPD was less than 25 % of that of BOCPD. In contrast, BOCPD attained the highest FDR, followed by RBOCPDMS. Both these methods reached a high TPR due to the large number of detections, which includes the true change points. Recall that the results of Figures 4.2, 4.3 and 4.4 highlight the property of BOCPD-based methods of notifying a detection simply due to the prior probability of change. Therefore, although BOCPD and RBOCPDMS attained a high TPR, the large number of false discoveries turn these models into unreliable methods for change detection in these datasets. Regarding NEWMA, it either committed a similar or higher FDR than

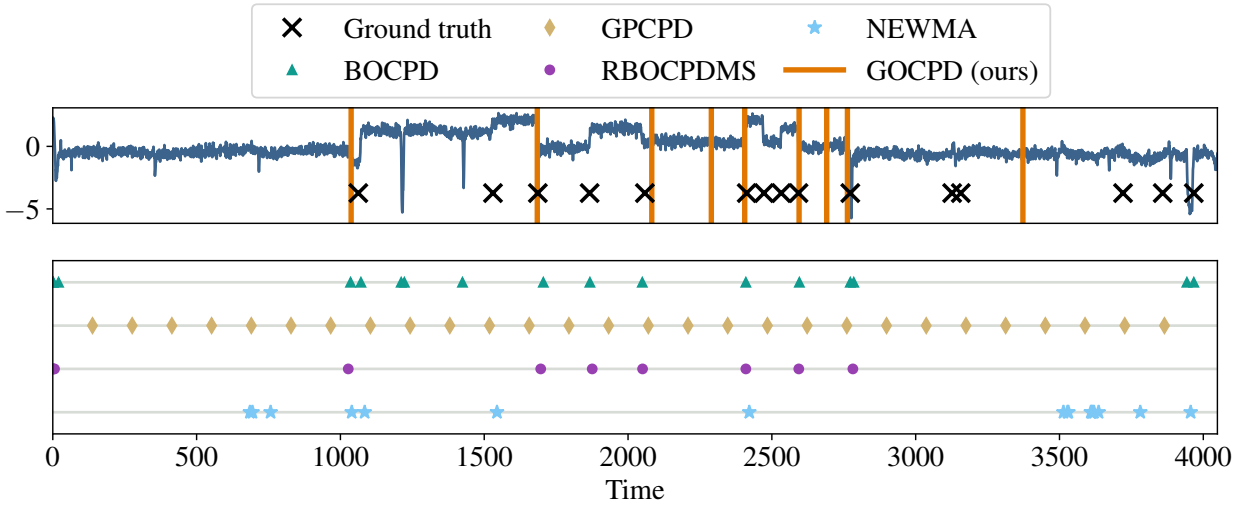


Figure 4.6: **Change point detection in Well log data**. Both GOCPD and RBOCPDMS were robust to outliers located at $t \in (1000, 1500)$, while BOCPD flaggled numerous false positives.

Table 4.2: **True positive rate (TPR) and false discovery rate (FDR).** Baseline methods reached higher TPR than GOCPD but returned considerably high FDR in all datasets. In contrast, GOCPD achieved the lowest FDR overall.

Method	**EEG**		**Human Act.** (uni)		**Human Act.** (multi)		**Well-log**	
	TPR	FDR	TPR	FDR	TPR	FDR	TPR	FDR
BOCPD	**1.00**	0.88	0.45	0.33	**0.86**	0.47	0.53	0.53
RBOCPD	**1.00**	0.86	**0.54**	0.22	0.38	0.33	**0.70**	**0.30**
NEWMA	0.78	0.81	0.38	0.25	0.27	0.14	0.33	0.71
GOCPD (ours)	0.67	**0.54**	0.42	**0.15**	0.47	**0.10**	0.54	**0.30**

BOCPD-based methods (in the EEG dataset and the Well-log dataset) or achieved lower FDR than BOCPD-based methods but failed to detect most real changes (in multivariate and univariate Human Activity datasets). This summary highlights the effectiveness of our proposal for avoiding high false discovery rates.

Table 4.3: **Comparison of negative log-likelihood (NLL) and mean absolute error (MAE) in univariate datasets.** All numbers were averaged over five independent runs. Overall, GOCPD reached the lowest MAE. *Corresponds to the original standard deviation 0.01 rounded to the second decimal.

Method	**Well-log**		**EEG**		**Human Act** (uni)		**Respiration**	
	NLL	MAE	NLL	MAE	NLL	MAE	NLL	MAE
BOCPD	9.99±0.20	4303.12±644.83	0.29±0.14	0.69±0.08	**−0.60±0.10**	0.09±0.00*	1.12±0.24	0.89±0.07
RBOCPDMS	12.26±2.70	6548.87±338.61	**0.16±0.24**	0.77±0.03	−0.40±0.22	0.10±0.01	**0.87±0.41**	0.97±0.05
GOCPD (ours)	**0.74±0.01**	**0.28±0.01**	0.99±0.00*	**0.26±0.00***	0.88±0.05	**0.05±0.01**	0.93±0.00*	**0.28±0.00***

Table 4.4: **Comparison of negative log-likelihood (NLL) and mean absolute error (MAE) in the multivariate dataset.** Results were averaged over five independent runs. Overall, GOCPD reached the lowest MAE. *Corresponds to the original standard deviation 0.007 rounded to the second decimal.

Method	**Human Act** (multi)	
	NLL	MAE
BOCPD	**2.87**$\pm$**0.20**	0.49$\pm$0.00*
RBOCPDMS	3.51$\pm$0.38	0.62$\pm$0.06
GOCPD (ours)	148.91$\pm$9.42	**0.35**$\pm$**0.02**

Tables 4.3 and 4.4 summarize the comparison of our model against Bayesian methods in both univariate and multivariate datasets respectively. Notably, our proposed GOCPD provided the lowest MAE in all datasets, while the best-performing method in terms of negative log-likelihood varies. It is important to note that the higher amount of detections imply a lower amount of observations for each segmented section of the time series, thus the model can fit better on the data, which may be the case of BOCPD and RBOCPDMS in the EEG and Respiration datasets, as it returned a higher number of false positives than our method. Recall that the change detection results considered the best run among five repetitions, and we observed that some of the repetitions of BOCPD and RBOCPDMS returned a very high number of false positives, which may explain the small values of the negative log-likelihood in some cases. Therefore, we argue that the negative log-likelihood is only a metric to measure the quality of the data fitting but does not account for a comparison of the change point detection performance.

Chapter 5

Deeper insights

In this section, we present more in-depth insights into GOCPD in terms of time complexity in comparison with existing CPD methods. We also provide results of the number of evaluations to support the time complexity analysis. Finally, we present the limitations of our method and potential solutions, which we leave for future work.

5.1. Time complexity

Since the implementation in the released code of BOCPD and RBOCPDMS [21] does not include GPU acceleration, we executed the training process in GOCPD on CPU to provide a fair comparison in terms of their computational cost. We considered the original version of BOCPD and RBOCPDMS without pruning for a fair comparison with GOCPD. To have a comparable method to the pruning technique in BOCPD for GOCPD, it would have to include some prior knowledge of the change point locations. We kept model-free methods away from this analysis since those method does not need to model the observations, thus the execution time is expected to be smaller than that of the methods that require data modeling.

As shown in Table 5.1, GOCPD was the fastest method overall, followed by BOCPD, due to its efficient recursive formulation and because the predictive probabilities are directly expressed in closed form, then it does not require any approximations nor sampling procedures. On the other hand, the execution time of RBOCPDMS was almost 170 times slower than that of BOCPD, due to its steps of full optimization and sampling. However, since we implemented GOCPD with Gaussian processes, the fair comparison was against GPCPD, for which GOCPD was at least 5× faster. Nonetheless, even comparing against RBOCPDMS, which used GLR as the underlying predictive model, GOCPD presented a 170× speedup.

Additionally, Table 5.1 summarizes the theoretical complexity of each CPD method. It shows that GOCPD had considerably higher complexity than BOCPD and RBOCPDMS. However, the time complexity in practice generally differed from the theoretical complexity

Table 5.1: **Average per-batch CPU execution time on Well-log dataset**. We report time measures for the non-pruned version of BOCPD-based methods. **Boldface** indicate the fastest method and underline highlights the second fast method. GOCPD outperforms RBOCPDMS by a $\approx 170\times$ speedup and is $\approx 5\times$ faster than GPCPD.

Method	**Time complexity**	**Time/#points** [s]
BOCPD	$\mathcal{O}(T)$	<u>0.156</u>
GPCPD	$\mathcal{O}(T^4)$	0.404
RBOCPDMS	$\mathcal{O}(1)$	13.293
GOCPD (ours) + GP	$\mathcal{O}(T^3 \log(T))$	**0.075**

ranking, as GOCPD does not search over the complete segment of the so-far data. Therefore, the theoretical complexity accounts for the worst scenario, in which the saved candidate change point of GOCPD always stays at $t = 0$, which is improbable in reality. We present a deeper discussion of this result in the next subsection.

It is important to note that the execution time of these methods depends on the detections because the model training is performed over the so-far observed data since the last detected change. Therefore, if the method detects many changes, its execution time will be lower, as the model needs to fit fewer observations.

These results highlight the advantage of minimizing the searching interval by storing the best candidate at each iteration and using ternary search to identify the optimal location of the change point, which reduces the linear per-timestep complexity of computing all possible change point locations to a logarithmic complexity. To illustrate this, let us consider the univariate Human Activity dataset, in which the minimum gap between detected change points (see Figure 4.3) is $\Delta \approx 200$. Therefore, without saving the candidate nor identifying the change point with ternary search, a coarse approximation of the total number of evaluations would be $\sum_{k=1}^{\Delta=200} k = 20,100$. By saving the candidate change point, the average searching interval size per time-step was reduced to 81.6 ± 5.6 (we include details in Appendix C). Moreover, by applying ternary search, the real average number of evaluations is 5.6 ± 3.2, which is equivalent to $5.6 \times \Delta \approx 1,120$ evaluations. In other words, in the univariate Human Activity dataset, the proposed components for GOCPD decrease the time complexity of the optimal change point searching procedure by at least 20 times with respect to the naive setting. We provide a complete discussion of the effect of the components of GOCPD on the number of evaluations at each iteration in the next subsection.

5.2. Effectively evaluated timestamps

Figure 5.1 shows the distribution of the original interval size at each iteration, i.e., $t - \tilde{c}(t)$, and the effective interval size after saving the candidate change point, i.e., $t - c_t$. Since timestamps located before the candidate change point are not potential timestamps to be

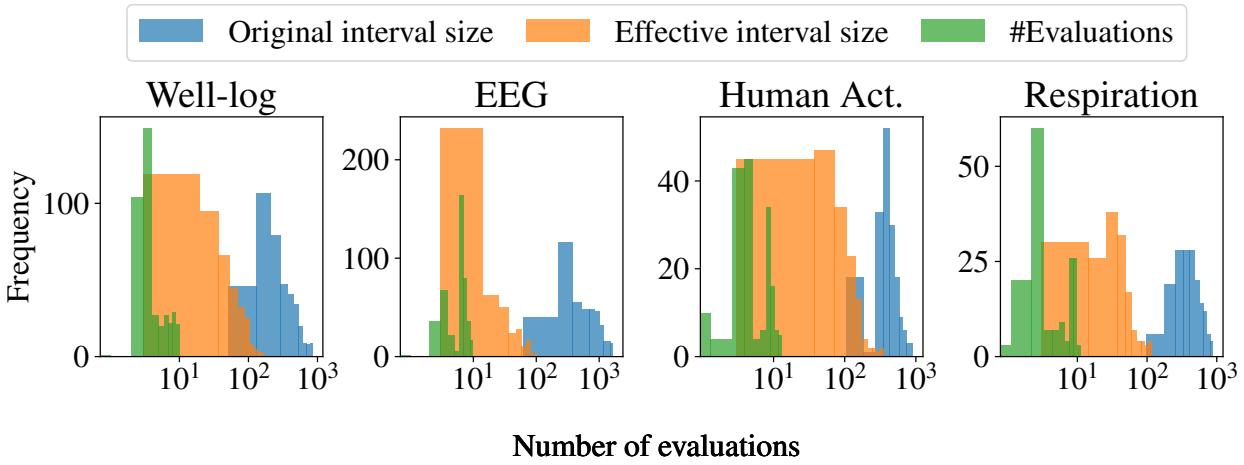


Figure 5.1: **Histogram of the size of the original interval since the last detected change point, the effective interval size, and the real number of evaluations.** Notably, by saving the candidate change point and using ternary search to maximize the change point location metric, the number of evaluations decreased in two orders of magnitude.

evaluated in the procedure of ternary search that maximizes the change point location metric, only the last $t - c_t$ timestamps are the possible options in the maximization procedure. In practice, we observed that $t-c_t \ll t-\tilde{c}(t)$ in the datasets included in this work. The difference of both distributions was of at least one order of magnitude, which illustrates the benefit of saving the candidate change point at each iteration.

On the other hand, the difference between the distribution of the effective interval size (after saving the candidate change point) and that of the real number of evaluations per iteration demonstrates the effect of optimizing the change point location metric using ternary search. Overall, joining both components, GOCPD decreased the number of evaluations needed to optimize the change point location metric in approximately two orders of magnitude with respect to the original case, that has quadratic time complexity.

We provide mean and standard deviations of the number of evaluations as well as the total interval size and the effective interval size in Appendix C.

5.3. Limitations

GOCPD is motivated by the intuition of searching a change point in the thus-far observed data before the current time and waiting for a minimum period of verification to effectively notify a change point. The verification period increases the robustness of detection but also creates an irreducible detection delay. As a consequence, there is a trade-off between robustness and detection delay for tuning this hyperparameter.

In addition, experimental results showed that using GPs, changes from high to small

variance were more difficult to detect with GOCPD since a model with high variance (m_1) can still fit data with lower variance. Therefore, the likelihood of m_1 does not decrease abruptly. Using models that are symmetric in terms of the variance or including regularizations in the model may circumvent this issue. Also, the experimental results show that sometimes the detected changes were not exactly located in the labeled timestamp. This result may be due to the shape of $s_t(\tau)$ near the change point, which did not have an evident peak. Therefore, the ternary search algorithm returned only an approximation of the real maximum.

If a new change point c_2^* appears before detecting the previous one c_1^*, Proposition 3.3.2 ensures that the current candidate change point is located before c_1^*. However, this may not happen in reality since the detection condition of Equation (3.10) must hold for a given period. Therefore, since there are two change points in the so-far observed data, the ternary search will output the one that provides the split with maximum likelihood, which may be c_2^* and then GOCPD fails to detect c_1^*. This phenomenon explains the false negatives in the experimental results.

Chapter 6

Conclusions

In this thesis, we have introduced GOCPD, an online CPD method that uses two simple criteria simulating human intuition when making online change detection: one for identifying the timestamp that is more likely to be a change point, and one for approving it as a real change point. GOCPD uses a computationally appealing searching algorithm to find the most likely change point location, and leverages the unimodal property of the likelihood of the observations coming from two different distributions. By greedily saving candidate change points, the space of possible change points is considerably reduced.

We have validated the hypotheses stated in Section 1.2 with the experimental results, which have shown that GOPCD can effectively detect changes in synthetic datasets and are competitive with – and in some cases outperforms – Bayesian methods in both univariate and multivariate real-world data. Overall, our method returns robust detections, thus decreasing the false alarm rate. We have also analyzed the computation complexity of the method, which is two orders of magnitude faster than the method without the proposed components in GOCPD.

Nonetheless, detecting changes in high-dimensional data is still time-consuming when using GPs as the underlying predictive model. Therefore, future work includes incorporating sparse GP approximations [44] to further decrease the training cost and a representation with latent variables in a lower-dimensional space. On the other hand, since our method retrains models in the searching algorithm, online models that only need retraining over the new set of observations are beneficial for GOCPD [45]. Moreover, at each step of the ternary search, one model steps ahead and observes new data, while the other model takes a step back. Consequently, it needs to *forget* datapoints. Therefore, probing these type of models are also a promising future research direction.

In conclusion, we have presented GOCPD as a new approach for detecting changes in an online manner different from the standard BOCPD-based or likelihood-ratio-based methods. We believe our proposal brings the CPD problem closer to a human-like solution, which may benefit many machine learning methods. We hope this work contributes to broadening the CPD researches and provides significant intuition for future work.

Bibliography

[1] F. Caron, A. Doucet, and R. Gottardo, "On-line changepoint detection and parameter estimation with application to genomic data," *Statistics and Computing*, vol. 22, pp. 579–595, Mar 2012.

[2] L. Lu and H.-J. Zhang, "Speaker change detection and tracking in real-time news broadcasting analysis," in *Proceedings of the Tenth ACM International Conference on Multimedia*, MULTIMEDIA '02, (New York, NY, USA), p. 602–610, Association for Computing Machinery, 2002.

[3] K. J. Oh and I. Han, "Using change-point detection to support artificial neural networks for interest rates forecasting," *Expert Systems with Applications*, vol. 19, no. 2, pp. 105–115, 2000.

[4] K. J. Oh and I. Han, "An intelligent clustering forecasting system based on change-point detection and artificial neural networks: application to financial economics," in *Proceedings of the 34th Annual Hawaii International Conference on System Sciences*, p. 8, 2001.

[5] G. Chen, G. Lu, W. Shang, and Z. Xie, "Automated change-point detection of eeg signals based on structural time-series analysis," *IEEE Access*, vol. 7, pp. 180168–180180, 2019.

[6] R. Malladi, G. P. Kalamangalam, and B. Aazhang, "Online Bayesian change point detection algorithms for segmentation of epileptic activity," in *2013 Asilomar Conference on Signals, Systems and Computers*, pp. 1833–1837, 2013.

[7] P. Yang, G. Dumont, and M. Ansermino, "Adaptive change detection in heart rate trend monitoring in anesthetized children," *IEEE Transactions on Bio-medical Engineering*, vol. 53, pp. 2211–9, 12 2006.

[8] D. Agudelo-España, S. Gomez-Gonzalez, S. Bauer, B. Schölkopf, and J. Peters, "Bayesian online prediction of change points," in *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI)* (J. Peters and D. Sontag, eds.), vol. 124, pp. 320–329, PMLR, 03–06 Aug 2020.

[9] R. Alami, O. Maillard, and R. Feraud, "Restarted Bayesian online change-point detector achieves optimal detection delay," in *Proceedings of the 37th International Conference on Machine Learning* (H. D. III and A. Singh, eds.), vol. 119, (Virtual), pp. 211–221, PMLR, 13–18 Jul 2020.

[10] J. Knoblauch and T. Damoulas, "Spatio-temporal Bayesian on-line changepoint detection with model selection," in *Proceedings of the 35th International Conference on Ma-*

chine Learning (J. Dy and A. Krause, eds.), vol. 80, pp. 2718–2727, PMLR, 10–15 Jul 2018.

[11] Y. Saatçi, R. Turner, and C. E. Rasmussen, "Gaussian process change point models," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, (Madison, WI, USA), p. 927–934, Omnipress, 2010.

[12] R. P. Adams and D. J. C. MacKay, "Bayesian online changepoint detection," *arXiv e-prints*, p. arXiv:0710.3742, Oct. 2007.

[13] F. Gustafsson, "The marginalized likelihood ratio test for detecting abrupt changes," *IEEE Transactions on Automatic Control*, vol. 41, no. 1, pp. 66–78, 1996.

[14] F. Gustafsson, "Adaptive filtering and change detection," 2000.

[15] S. Aminikhanghahi and D. J. Cook, "A survey of methods for time series change point detection," *Knowledge and information systems*, vol. 51, pp. 339–367, May 2017.

[16] G. J. J. van den Burg and C. K. I. Williams, "An evaluation of change point detection algorithms," 2020.

[17] P. Fearnhead and Z. Liu, "On-line inference for multiple changepoint problems," *Journal of the Royal Statistical Society Series B*, vol. 69, pp. 589–605, 09 2007.

[18] J. J. K. Ó Ruanaidh and W. J. Fitzgerald, *Numerical Bayesian Methods Applied to Signal Processing*. Springer, 1996.

[19] P. Moreno-Muñoz, D. Ramírez, and A. Artés-Rodríguez, "Continual learning for infinite hierarchical change-point detection," in *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3582–3586, 2020.

[20] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*, vol. 2. The MIT Press, 2006.

[21] J. Knoblauch, J. E. Jewson, and T. Damoulas, "Doubly robust Bayesian inference for non-stationary streaming data with β-divergences," in *Advances in Neural Information Processing Systems* (S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, eds.), vol. 31, pp. 64–75, Curran Associates, Inc., 2018.

[22] P. G. Bissiri, C. C. Holmes, and S. G. Walker, "A general framework for updating belief distributions," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 78, p. 1103–1130, Feb 2016.

[23] J. Jewson, J. Q. Smith, and C. Holmes, "Principles of Bayesian inference using general divergence criteria," *Entropy*, vol. 20, no. 6, 2018.

[24] R. Alami, O. Maillard, and R. Féraud, "Memory Bandits: a Bayesian approach for the Switching Bandit Problem," in *31st Conference on Neural Information Processing Systems*, (Long Beach, United States), Dec. 2017.

[25] J. Mellor and J. Shapiro, "Thompson sampling in switching environments with bayesian online change detection," in *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics* (C. M. Carvalho and P. Ravikumar, eds.), vol. 31, (Scottsdale, Arizona, USA), pp. 442–450, PMLR, 29 Apr–01 May 2013.

[26] J. Han, K. Lee, A. Tong, and J. Choi, "Confirmatory Bayesian online change point

detection in the covariance structure of gaussian processes," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI'19, p. 2449–2455, AAAI Press, 2019.

[27] E. S. Page, "Continuous inspection schemes," *Biometrika*, vol. 41, no. 1/2, pp. 100–115, 1954.

[28] A. Huitson and C. S. Bruyn, "Cumulative sum tests: Theory and practice," *The Statistician*, vol. 19, pp. 357–357, 1970.

[29] L. I. Kuncheva, "Change detection in streaming multivariate data using likelihood detectors," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 5, pp. 1175–1180, 2013.

[30] D. R. Jeske, V. Montes De Oca, W. Bischoff, and M. Marvasti, "Cusum techniques for timeslot sequences with applications to network surveillance," *Computational Statistics & Data Analysis*, vol. 53, no. 12, pp. 4332–4344, 2009.

[31] A. Aue, S. Hörmann, L. Horváth, and M. Reimherr, "Break detection in the covariance structure of multivariate time series models," *The Annals of Statistics*, vol. 37, no. 6B, pp. 4046 – 4087, 2009.

[32] H. Cho and P. Fryzlewicz, "Multiple-change-point detection for high dimensional time series via sparsified binary segmentation," *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, vol. 77, no. 2, pp. 475–507, 2015.

[33] Y. Kawahara and M. Sugiyama, "Sequential change-point detection based on direct density-ratio estimation," *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 5, no. 2, pp. 114–127, 2012.

[34] T. Kanamori, T. Suzuki, and M. Sugiyama, "Theoretical analysis of density ratio estimation," *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, vol. 93-A, pp. 787–798, 2010.

[35] M. Sugiyama, T. Suzuki, S. Nakajima, H. Kashima, P. von Bünau, and M. Kawanabe, "Direct importance estimation for covariate shift adaptation," *Annals of the Institute of Statistical Mathematics*, vol. 60, pp. 699–746, Dec 2008.

[36] N. Keriven, D. Garreau, and I. Poli, "NEWMA: A new method for scalable model-free online change-point detection," *IEEE Transactions on Signal Processing*, vol. 68, pp. 3515–3528, 2020.

[37] D. Chatterjee and S. Bhattacharya, "Posterior convergence of gaussian and general stochastic process regression under possible misspecifications," *arXiv preprint arXiv:1810.10495*, 2018.

[38] N. Arora, M. Martolia, and E. Arora, "A novel ternary search algorithm," *International Journal of Computer Applications*, vol. 144, pp. 35–36, 06 2016.

[39] D. Kifer, S. Ben-David, and J. Gehrke, "Detecting change in data streams," in *Proceedings of the 30th International Conference on Very Large Data Bases - Volume 30*, VLDB '04, p. 180–191, VLDB Endowment, 2004.

[40] S. Li, Y. Xie, H. Dai, and L. Song, "Scan B-statistic for kernel change-point detection," *Sequential Analysis*, vol. 38, no. 4, pp. 503–544, 2019.

[41] C. Williams, E. V. Bonilla, and K. M. Chai, "Multi-task Gaussian process prediction," *Advances in neural information processing systems*, pp. 153–160, 2007.

[42] J. R. Gardner, G. Pleiss, D. Bindel, K. Q. Weinberger, and A. G. Wilson, "GPyTorch: Blackbox matrix-matrix gaussian process inference with GPU acceleration," in *Advances in Neural Information Processing Systems*, 2018.

[43] N. Stevenson, K. Tapani, L. Lauronen, and S. Vanhatalo, "A dataset of neonatal EEG recordings with seizures annotations. Zenodo," 06 2018.

[44] E. Snelson and Z. Ghahramani, "Sparse gaussian processes using pseudo-inputs," in *Advances in Neural Information Processing Systems* (Y. Weiss, B. Schölkopf, and J. Platt, eds.), vol. 18, MIT Press, 2006.

[45] T. D. Bui, C. Nguyen, and R. E. Turner, "Streaming sparse gaussian process approximations," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

Appendix A

Synthetic datasets

The change point locations for the synthetic datasets were randomly chosen with a minimum gap between change points of $\Delta t = 50$. Between each consecutive pair of change points, the observations were sampled from independent GPs. The locations were set to $\mathcal{C} = [0, 60, 150, 240, 450, 650, 800, 890]$. Regarding the hyperparameter values, we considered a initial value, and then for each segment we multiplied it by a factor. The initial values were $\ell = 1.0, \sigma_f = 0.5, \mu = 1, \sigma_n = 0.01$. The sequence of factors were chosen so that the changes were visually perceptible.

Table A.1: **Hyperparameters for each synthetic time series.**

Dataset	**Factors**
Change in lengthscale ℓ	$[10, 2, 10, 1, 5, 1/5, 1, 20]$
Change in output scale σ_f	$[1/10, 10, 1/20, 1, 10, 1/10, 3, 1/8]$
Change in mean μ	$[0, 2, -1, 3, 0, -1.4, 3.5, 0.2]$
Change in noise variance σ_n	$[1/5, 10, 1/5, 10, 1/5, 5, 1/5, 5]$

Appendix B

Implementation details

B.1. Data preprocessing and hyperparameters

In our experiments, all datasets were standardized. In the case of the multivariate datasets, each channel was independently standardized. Table B.1 summarizes the selected index interval of the dataset and the downsampling rate, when applicable. Also, GOCPD waits for a period of size T_{ini} after each detection, to ensure that the likelihoods does not change abruptly due to the lack of enough data after dropping the observations previous to the detected change point.

Table B.1: **Data preprocessing and detection hyperparameters.**

Dataset	**Interval**	**Downsampling rate**	**Period** T_{ini}
Well-log	complete	N/A	80
EEG	$[18, 29] \times 10^4$	10	300
Activity (uni)	complete	5	300
Activity (multi)	$[19, 31] \times 10^3$	5	80
Respiration	$[10, 14] \times 10^3$	N/A	300

For baseline methods, we use the same downsampling rate for all datasets.

B.2. Hardware setup

Our experiments were implemented in GPyTorch [42], thus processed can be accelerated with GPUs. All the experiments were carried out on a NVIDIA GTX 1660Ti 6GB GPU.

Appendix C

Effectively evaluated timestamps

Table C.1 summarizes the effect of saving the candidate change point at each iteration and using ternary search to optimize the change point location metric. Importantly, combining both components reduced the number of evaluations in two orders of magnitude.

Table C.1: **Average size of the original interval since the last detected change point, average effective interval size of evaluation and average number of evaluations over all iterations among univariate datasets.**

Dataset	Well-log	EEG	Activity	Respiration
Total interval size	305.4±173.7	634.8±388.7	414.8±154.1	433.1±172.3
Effective interval size	42.3±31.6	137.7±228.7	81.6±62.3	36.8±24.2
#Evaluations	4.0±2.2	6.7±2.9	5.6±3.2	5.1±2.8