# Geometric-Based Pruning Rules For Change Point Detection in Multiple Independent Time Series

ISSN 2824-7795

Liudmila Pishchagina [1] LaMME, UEVE
Guillem Rigaill INRAE, UEVE
Vincent Runge LaMME, UEVE

Date published: 2023-05-22 Last modified: 2023-05-22

**Abstract**

We consider the problem of detecting multiple changes in multiple independent time series. It can be expressed as finding the segmentation that minimizes a given cost function. We focus on dynamic programming algorithms that solve this minimization problem exactly. When the number of changes is proportional to data length, an inequality-based pruning rule encoded in the PELT algorithm leads to a linear time complexity. Another type of pruning, called functional pruning, gives a close-to-linear time complexity whatever the number of changes, but only for a univariate cost function. We propose a few extensions of functional pruning for multiple independent time series based on the use of simple geometric shapes (balls and hyperrectangles). We focus on the Gaussian case, but some of our rules can be extended to the exponential family. In a simulation study, we compare the computational efficiency of different geometric-based pruning rules and show that for small dimensions (2, 3, 4) some of them ran significantly faster than inequality-based approaches in particular when the underlying number of changes is small compared to the data length.

*Keywords:* Multiple change point detection, dynamic programming, functional pruning, computational geometry

## Contents

[1] Corresponding author: liudmila.pishchagina@univ-evry.fr

## 1 Introduction

A National Research Council report (Data et al. (2013)) has identified change point detection as one of the "inferential giants" in massive data analysis. Detecting change points, either a posteriori or online, is important in areas as diverse as bioinformatics (Olshen et al. (2004), Picard et al. (2005)), econometrics (Bai and Perron (2003), Aue et al. (2006)), medicine (Bosc et al. (2003), Staudacher et al. (2005), Malladi, Kalamangalam, and Aazhang (2013)), climate and oceanography (Reeves et al. (2007), Ducré-Robitaille, Vincent, and Boulet (2003), Killick, Fearnhead, and Eckley (2012), Naoki and Kurths (2010)), finance (Andreou and Ghysels (2002), Fryzlewicz (2014)), autonomous driving (Galceran et al. (2017)), entertainment (Rybach et al. (2009), Radke et al. (2005), Davis, Lee, and Rodriguez-Yam (2006)), computer vision (Ranganathan (2012)) or neuroscience (Jewell, Fearnhead, and Witten (2019)). The most common and prototypical change point detection problem is that of detecting changes in mean of a univariate Gaussian signal and a large number of approaches have been proposed to this problem (see among many others Yao (1984), Lebarbier (2005), Harchaoui and Lévy-Leduc (2010), Frick, Munk, and Sieling (2013), Anastasiou and Fryzlewicz (2022) and the reviews Truong, Oudre, and Vayatis (2020), Aminikhanghahi and Cook (2017)).

*Penalized cost methods.* Some of these methods optimize a penalized cost function (see for example Lebarbier (2005), Auger and Lawrence (1989), Jackson et al. (2005), Killick, Fearnhead, and Eckley (2012), Rigaill (2010), Maidstone et al. (2017)). These methods have good statistical guarantees (Yao (1984), Lavielle and Moulines (2000), Lebarbier (2005)) and have shown good performances in benchmark simulations (Fearnhead, Maidstone, and Letchford (2018)) and on many applications ( Lai et al. (2005), Liehrmann, Rigaill, and Hocking (2021)). From a computational perspective, these methods rely on dynamic programming algorithms that are at worst quadratic in the size of the data, $n$. However using inequality-based and functional pruning techniques (Rigaill (2010), Killick, Fearnhead, and Eckley (2012), Maidstone et al. (2017)) the average run times are typically much smaller allowing to process very large profiles ($n > 10^5$) in a matter of seconds or minutes. In detail, for one time series:

- if the number of change points is proportional to $n$ both PELT (inequality-based pruning) and FPOP (functional pruning) (Killick, Fearnhead, and Eckley (2012), Maidstone et al. (2017)) are on average linear.
- if the number of change points is fixed, FPOP is quasi-linear (on simulations) while PELT is quadratic (Maidstone et al. (2017)).

*Multivariate extensions.* In this paper, we focus on the multivariate problem assuming the cost function or log-likelihood of a segment (denoted $\mathscr{C}$) can be decomposed as a sum over all $p$ dimensions. Informally that is

$$\mathscr{C}(segment) = \sum_{k=1}^{p} \mathscr{C}(segment,\ \text{time series } k)\,.$$

In this context, the PELT algorithm can easily be extended for multiple time series. However, as for the univariate case, it will be algorithmically efficient only if the number of change points is large compared to $n$. In this paper, we study the extension of functional pruning techniques (and more specifically FPOP) to the multivariate case.

At each iteration, FPOP updates the set of parameter values for which a change position $\tau$ is optimal. As soon as this set is empty the change is pruned. For univariate time series, this set is a union of intervals in $\mathbb{R}$. For multi-parametric models, this set is equal to the intersection and difference of convex sets in $\mathbb{R}^p$ (Runge (2020)). It is typically non-convex, hard to update, and deciding whether it is empty or not is not straightforward.

In this work, we present a new algorithm, called Geometric Functional Pruning Optimal Partitioning (GeomFPOP). The idea of our method is to approximate the sets that are updated at each iteration of FPOP using simpler geometric shapes. Their simplicity of description and simple updating allow for a quick emptiness test.

The paper has the following structure. In Section 2 we introduce the penalized optimization problem for segmented multivariate time series. We then review the existing pruned dynamic programming methods for solving this problem. We define the geometric problem that occurs when using functional pruning. The new method, called GeomFPOP, is described in Section Section 3 and based on approximating intersection and exclusion set operators. In Section 4 we introduce two approximation types (sphere-like and rectangle-like) and define the approximation operators for each of them. We then compare in Section 5 the empirical efficiency of GeomFPOP with PELT on simulated data.

## 2 Functional Pruning for Multiple Time Series

### 2.1 Model and Cost

We consider the problem of change point detection in multiple time series of length $n$ and dimension $p$. Our aim is to partition time into segments, such that in each segment the parameter associated to each time series is constant. For a time series $y$ we write $y = y_{1:n} = (y_1, \dots, y_n) \in (\mathbb{R}^p)^n$ with $y_i^k$ the $k$-th component of the $p$-dimensional point $y_i \in \mathbb{R}^p$ in position $i$ in vector $y_{1:n}$. We also use the notation $y_{i:j} = (y_i, \dots, y_j)$ to denote points from index $i$ to $j$. If we assume that there are $M$ change points in a time series, this corresponds to time series splits into $M+1$ distinct segments. Each segment $m \in \{1, \dots, M+1\}$ is generated by independent random variables from a multivariate distribution with the segment-specific parameter $\theta_m = (\theta_m^1, \dots, \theta_m^p) \in \mathbb{R}^p$. A segmentation with $M$ change points is defined by the vector of integers $\tau = (\tau_0 = 0, \tau_1, \dots, \tau_M, \tau_{M+1} = n)$. Segments are given by the sets of indices $\{\tau_i + 1, \dots, \tau_{i+1}\}$ with $i$ in $\{0, 1, \dots, M\}$.

We define the set $S_t$ of all possible change point locations related to the segmentation of data points between positions 1 to $t$ as

$$S_t = \{\tau = (\tau_0, \tau_1, \dots, \tau_M, \tau_{M+1}) \in \mathbb{N}^{M+2} | 0 = \tau_0 < \tau_1 < \dots < \tau_M < \tau_{M+1} = t\}.$$

Usually the number of changes $M$ is unknown, and has to be estimated. Many approaches to detecting change points define a cost function for segmentation using the opposite log-likelihood (times two). Here the opposite log-likelihood (times two) linked to data point $y_j$ is given by function $\theta \mapsto \Omega(\theta, y_j)$, where $\theta = (\theta^1, \dots, \theta^p) \in \mathbb{R}^p$. Over a segment from $i$ to $t$, the parameter remains the same and the segment cost $\mathcal{C}$ is given by

$$\mathcal{C}(y_{i:t}) = \min_{\theta \in \mathbb{R}^p} \sum_{j=i}^{t} \Omega(\theta, y_j) = \min_{\theta \in \mathbb{R}^p} \sum_{j=i}^{t} \left( \sum_{k=1}^{p} \omega(\theta^k, y_j^k) \right), \tag{1}$$

with $\omega$ the atomic likelihood function associated with $\Omega$ for each univariate time series. This decomposition is made possible by the independence hypothesis between dimensions. Notice that function $\omega$ could have been dimension-dependent with a mixture of different distributions (Gauss, Poisson, negative binomial, etc.). In our study, we consider the same data model for all dimensions.

We consider a penalized version of the cost by a penalty $\beta > 0$, because without a penalty we would end up with $n$ segments. Summing over all segments we end up with a penalty that is linear in the number of segments. Such choice is common in the literature (Yao (1988), Killick, Fearnhead, and Eckley (2012)) although some other penalties have been proposed (Zhang and Siegmund (2007), Lebarbier (2005), Verzelen et al. (2020)). The optimal penalized cost associated with our segmentation problem is then defined by

$$Q_n = \min_{\tau \in S_n} \sum_{i=0}^{M} \{\mathcal{C}(y_{(\tau_i+1):\tau_{i+1}}) + \beta\}. \tag{2}$$

The optimal segmentation $\tau$ is obtained by the argminimum in Equation 2.

## 2.2 Functional Pruning Dynamic Programming Algorithm

The idea of the Optimal Partitioning (OP) method (Jackson et al. (2005)) is to search for the last change point defining the last segment in data $y_{1:t}$ at each iteration (with $Q_0 = 0$), which leads to the recursion:

$$Q_t = \min_{i \in \{0,\dots,t-1\}} \Big(Q_i + \mathcal{C}(y_{(i+1:t)}) + \beta\Big). \tag{3}$$

*Functional description.* In the FPOP method we introduce a last segment parameter $\theta = (\theta^1, \dots, \theta^p) \in \mathbb{R}^p$ and define a functional cost $\theta \mapsto Q_t(\theta)$ depending on $\theta$, that takes the following form:

$$Q_t(\theta) = \min_{\tau \in S_t} \Big( \sum_{i=0}^{M-1} \{\mathcal{C}(y_{(\tau_i+1):\tau_{i+1}}) + \beta\} + \sum_{j=\tau_M+1}^{t} \Omega(\theta, y_j) + \beta\Big).$$

As explained in Maidstone et al. (2017), we can compute the function $Q_{t+1}(\cdot)$ based only on the knowledge of $Q_t(\cdot)$ as for each integer $t$ from 0 to $n-1$. We have:

$$Q_{t+1}(\theta) = \min\{Q_t(\theta), m_t + \beta\} + \Omega(\theta, y_{t+1}), \tag{4}$$

for all $\theta \in \mathbb{R}^p$, with $m_t = \min_\theta Q_t(\theta)$ and the initialization $Q_0(\theta) = 0$, so that $Q_1(\theta) = \Omega(\theta, y_1)$. By looking closely at this relation, we see that each function $Q_t$ is a piece-wise continuous function consisting of at most $t$ different functions on $\mathbb{R}^p$, denoted $q_t^i$:

$$Q_t(\theta) = \min_{i \in \{1,\dots,t\}} \left\{q_t^i(\theta)\right\}, \tag{5}$$

where the $q_t^i$ functions are given by explicit formulas:

$$q_t^i(\theta) = m_{i-1} + \beta + \sum_{j=i}^{t} \Omega(\theta, y_j), \quad \theta \in \mathbb{R}^p, \quad i = 1, \dots, t. \tag{6}$$

and

$$m_{i-1} = \min_{\theta \in \mathbb{R}^p} Q_{i-1}(\theta) = \min_{j \in \{1,\dots,i-1\}} \left\{\min_{\theta \in \mathbb{R}^p} q_{i-1}^j(\theta)\right\}. \tag{7}$$

It is important to notice that each $q_t^i$ function is associated with the last change point $i-1$ and the last segment is given by indices from $i$ to $t$. Consequently, the last change point at step $t$ in $y_{1:t}$ is denoted as $\hat{\tau}_t$ ($\hat{\tau}_t \le t-1$) and is given by

$$\hat{\tau}_t = Arg\,min_{i \in \{1,\dots,t\}} \left\{\min_{\theta \in \mathbb{R}^p} q_t^i(\theta)\right\} - 1. \tag{8}$$

*Backtracking.* Knowing the values of $\hat{\tau}_t$ for all $t = 1, \dots, n$, we can always restore the optimal segmentation at time $n$ for $y_{1:n}$. This procedure is called backtracking. The vector $cp(n)$ of ordered change points in the optimal segmentation of $y_{1:n}$ is determined recursively by the relation $cp(n) = (cp(\hat{\tau}_n), \hat{\tau}_n)$ with stopping rule $cp(0) = \emptyset$.

*Parameter space description.* Applying functional pruning requires a precise analysis of the recursion {eq-Q_tpl1} that depends on the property of the cost function~Ω. In what follows we consider three choices based on a Gaussian, Poisson, and negative binomial distribution assumption on the data. The exact formulas of these cost functions are given in **?@sec-logLikeExamples**.

We denote the set of parameter values for which the function $q_t^i(\cdot)$ is optimal as:

$$Z_t^i = \left\{\theta \in \mathbb{R}^p | Q_t(\theta) = q_t^i(\theta)\right\}, \quad i = 1, \dots, t. \tag{9}$$

The key idea behind functional pruning is that the $Z_t^i$ are nested ($Z_{t+1}^i \subset Z_t^i$) thus as soon as we can prove the emptiness of one set $Z_t^i$, we delete its associated $q_t^i$ function and do not have to consider its minimum anymore at any further iteration (proof in next Section 2.3). In dimension $p = 1$ this is reasonably easy. In this case, the sets $Z_t^i$ ($i = 1, \dots, t$) are unions of intervals and an efficient functional pruning rule is possible by updating a list of these intervals for $Q_t$. This approach is implemented in FPOP (Maidstone et al. (2017)).

In dimension $p \geq 2$ it is not so easy anymore to keep track of the emptiness of the sets $Z_t^i$. We illustrate the dynamics of the $Z_t^i$ sets in Figure 1 in the bi-variate Gaussian case. Each color is associated with a set $Z_t^i$ (corresponding to a possible change at $i-1$) for $t$ equal 1 to 5. This plot shows that sets $Z_t^i$ can be non-convex.

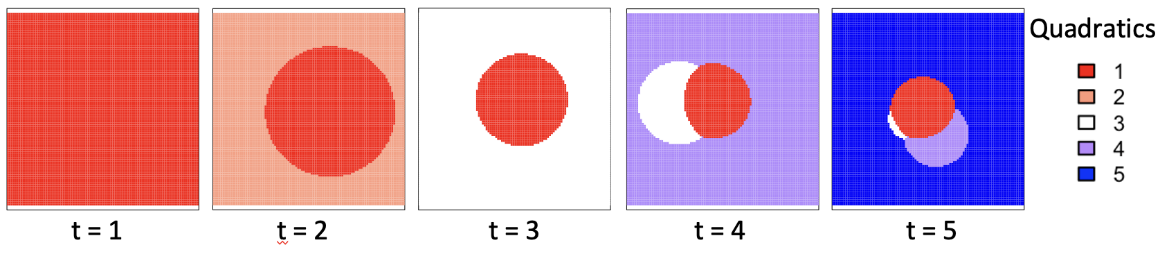

Figure 1: The sets $Z_t^i$ over time for the bi-variate independent Gaussian model on time series without change $y = \{\{0,29; 1,86; 0,9; -1,26; 1,22\}, \{1,93; -0,02; -2,51; 0,91; 1,11\}\}$. From left to right we represent at time $t = 1, 2, 3, 4$, and 5 the parameter space $(\theta^1, \theta^2)$. Each $Z_t^i$ is represented by a color. The change 1 associated with quadratics 2 is pruned at $t = 3$. Notice that each time sequence of $Z_t^i$ with $i$ fixed is a nested sequence of sets.

## 2.3 Geometric Formulation of Functional Pruning

To build an efficient pruning strategy for dimension $p \geq 2$ we need to test the emptiness of the sets $Z_t^i$ at each iteration. Note that to get $Z_t^i$ we need to compare the functional cost $q_t^i$ with any other functional cost $q_t^j$, $j = 1, \dots, t$, $j \neq i$. This leads to the definition of the following sets.

**Definition 2.1** (S-type set)**.** We define $S$-type set $S_j^i$ using the function Ω as

$$S_j^i = \left\{\theta \in \mathbb{R}^p \,|\, \sum_{u=i+1}^{j} \Omega(\theta, y_u) \leq m_j - m_i\right\}, \text{ when } i < j$$

and $S_i^i = \mathbb{R}^p$. We denote the set of all possible S-type sets as **S**.

To ease some of our calculations, we now introduce some additional notations. For $\theta = (\theta^1, \dots, \theta^p) \in \mathbb{R}^p$, $1 \leq i < j \leq n$ we define $p$ univariate functions $\theta^k \mapsto s_{ij}^k(\theta^k)$ associated to the $k$-th time series as

$$s_{ij}^k(\theta^k) \quad = \sum_{u=i+1}^{j} \omega(\theta^k, y_u^k), \quad k = 1, \dots, p. \tag{10}$$

We introduce a constant $\Delta_{ij}$ and a function $\theta \mapsto s_{ij}(\theta)$:

$$\begin{cases} \Delta_{ij} = m_j - m_i\,, \\ s_{ij}(\theta) = \sum_{k=1}^{p} s_{ij}^k(\theta^k) - \Delta_{ij}\,, \end{cases} \tag{11}$$

where $m_i$ and $m_j$ are defined as in Equation 7. The sets $S_j^i$ for $i < j$ are also described by relation

$$S_j^i = s_{ij}^{-1}(-\infty, 0]\,. \tag{12}$$

In Figure 2 we present the level curves for three different parametric models given by $s_{ij}^{-1}(\{w\})$ with $w$ a real number. Each of these curves encloses an S-type set.

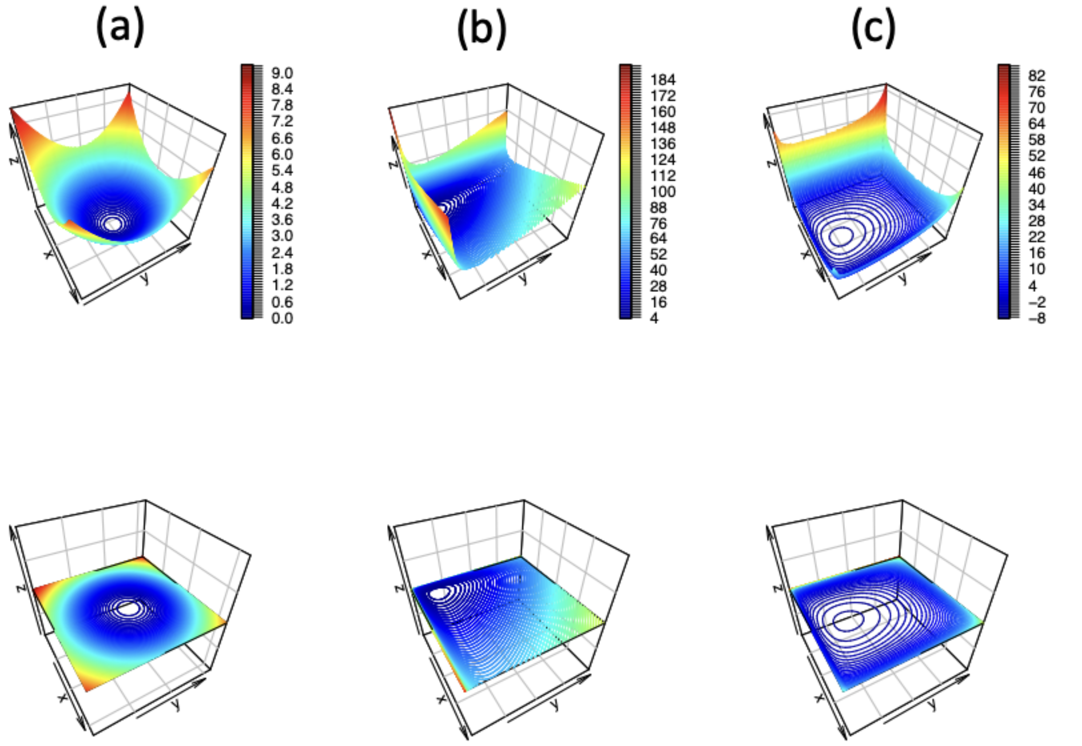

Figure 2: Three examples of the level curves of a function $s_{ij}$ for bi-variate time series $\{x, y\}$. We use the following simulations for univariate time series : (a) $x \sim \mathcal{N}(0,1)$, $y \sim \mathcal{N}(0,1)$, (b) $x \sim \mathcal{P}(1)$, $y \sim \mathcal{P}(3)$, (c) $x \sim \mathcal{NB}(0.5, 1)$, $y \sim \mathcal{NB}(0.8, 1)$.

At time $t = 1, \dots, n$ we define the following sets associated to the last change point index $i - 1$:

past set $\mathcal{P}^i$

$$\mathcal{P}^i = \{S_i^u,\ u = 1, \dots, i-1\}\,. \tag{13}$$

future set $\mathcal{F}^i(t)$

$$\mathcal{F}^i(t) = \{S_v^i,\ v = i, \dots, t\}\,. \tag{14}$$

We denote the cardinal of a set $\mathcal{A}$ as $|\mathcal{A}|$. Using these two sets of sets, the $Z_t^i$ have the following description.

**Proposition 2.1.** *At iteration t, the functional cost $Q_t(\cdot)$ defines the subsets $Z_t^i$ ($i = 1, \dots, t$), each of them being the intersection of the sets in $\mathcal{F}^i(t)$ minus the union of the sets in $\mathcal{P}^i$.*

$$Z_t^i = (\cap_{S \in \mathcal{F}^i(t)} S) \setminus (\cup_{S \in \mathcal{P}^i} S)\,, \quad i = 1, \dots, t. \tag{15}$$

*Proof.* Based on the definition of the set $Z_t^i$, the proof is straightforward. Parameter value $\theta$ is in $Z_t^i$ if and only if $q_t^i(\theta) \leq q_t^u(\theta)$ for all $u \neq i$; these inequalities define the past set (when $u < i$) and the future set (when $u > i$). Notice that, in case $i = t$, $\cap_{S \in \mathcal{F}^i(t)} S = \mathbb{R}^p$.

$\square$

**Corollary 2.1.** *The sequence $\zeta^i = (Z_t^i)_{t \geq i}$ is a nested sequence of sets.*

Indeed, $Z_{t+1}^i$ is equal to $Z_t^i$ with an additional intersection in the future set. Based on Corollary 2.1, as soon as we prove that the set $Z_t^i$, is empty, we delete its associated $q_t^i$ function and, consequently, we can prune the change point $i - 1$. In this context, functional and inequality-based pruning have a simple geometric interpretation.

*Functional pruning geometry.* The position $i - 1$ is pruned at step $t + 1$, in $Q_{t+1}(\cdot)$, if the intersection set of $\cap_{S \in \mathcal{F}^i(t)} S$ is covered by the union set $\cup_{S \in \mathcal{P}^i} S$.

*Inequality-based pruning geometry.* The inequality-based pruning of PELT is equivalent to the geometric rule: position $i-1$ is pruned at step $t+1$ if the set $S_t^i$ is empty. In that case, the intersection set $\cap_{S \in \mathcal{F}^i(t)} S$ is empty, and therefore $Z_t^i$ is also empty using Equation 15. This shows that if a change is pruned using inequality-based pruning it is also pruned using functional pruning. For the dimension $p = 1$ this claim was theoretically proved in Maidstone et al. (2017).

The construction of set $Z_t^i$ using Proposition 2.1 is illustrated in Figure 3 for a bi-variate independent Gaussian case: we have the intersection of three S-type sets and the subtraction of three S-type sets.

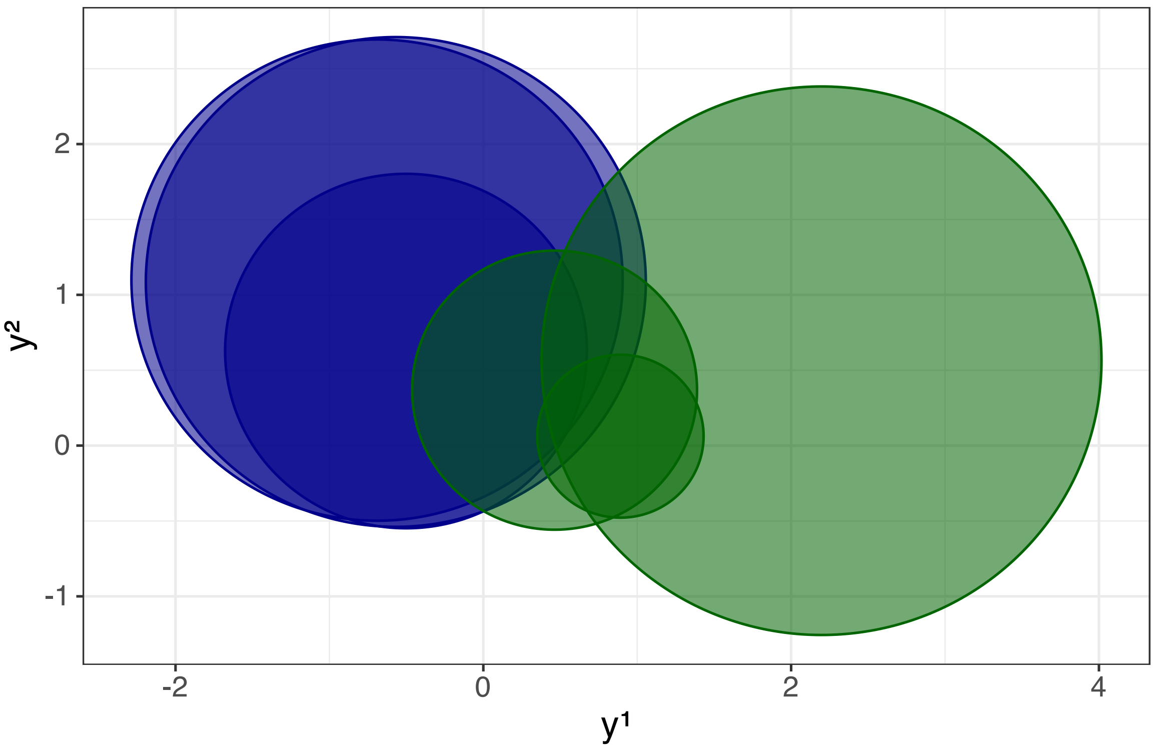

Figure 3: Examples of building a set $Z_t^i$ with $|\mathcal{P}^i| = |\mathcal{F}^i(t)| = 3$ for the Gaussian case in 2-D ($\mu = 0, \sigma = 1$). The green disks are S-type sets of the past set $\mathcal{P}^i$. The blue disks are S-type sets of the future set $\mathcal{F}^i(t)$.

## 3 Geometric Functional Pruning Optimal Partitioning

### 3.1 General Principle of GeomFPOP

Rather than considering an exact representation of the $Z_t^i$, our idea is to consider a hopefully slightly larger set that is easier to update. To be specific for each $Z_t^i$ we introduce $\tilde{Z}_t^i$, called *testing set*, such that $Z_t^i \subset \tilde{Z}_t^i$. If at time $t$ $\tilde{Z}_t^i$ is empty thus is $Z_t^i$ and thus change $i-1$ can be pruned. From proposition (??) we have that starting from $Z = \mathbb{R}^p$ the set $Z_t^i$ is obtained by successively applying two types of operations: intersection with an S-type set $S$ $(Z \cap S)$ or subtraction of an S-type set $S$ $(Z \setminus S)$. Similarly, starting from $\tilde{Z} = \mathbb{R}^p$ we obtain $\tilde{Z}_t^i$ by successively applying approximation of these intersection and subtraction operations. Intuitively, the complexity of the resulting algorithm is a combination of the efficiency of the pruning and the easiness of updating the testing set.

*A Generic Formulation of GeomFPOP.* In what follows we will generically describe GeomFPOP, that is without specifying the precise structure of the testing set $\tilde{Z}_t^i$. We call $\tilde{\mathbf{Z}}$ the set of all possible $\tilde{Z}_t^i$ and assume the existence of two operators $\cap_{\tilde{Z}}$ and $\setminus_{\tilde{Z}}$. We have the following assumptions for these operators.

**Definition 3.1.** The two operators $\cap_{\tilde{Z}}$ and $\setminus_{\tilde{Z}}$ are such that:

1. the left input is a $\tilde{Z}$-type set (that is an element of $\tilde{\mathbf{Z}}$);
2. the right input is a $S$-type set;
3. the output is a $\tilde{Z}$-type set;
4. $\tilde{Z} \cap S \subset \tilde{Z} \cap_{\tilde{Z}} S$ and $\tilde{Z} \setminus S \subset \tilde{Z} \setminus_{\tilde{Z}} S$.

We give a proper description of two types of testing sets and their approximation operators in Section 4.

At each iteration $t$ GeomFPOP will construct $\tilde{Z}_{t+1}^i$ from $\tilde{Z}_t^i$, $\mathcal{P}^i$ and, $\mathcal{F}^i(t)$ iteratively using the two operators $\cap_{\tilde{Z}}$ and $\setminus_{\tilde{Z}}$. To be specific, we define $S_j^F$ the j-th element of $\mathcal{F}^i(t)$ and $S_P^j$ the j-th element of $\mathcal{P}^i$, we use the following iteration:

$$\begin{cases} A_0 = \tilde{Z}_t^i\,, & A_j = A_{j-1} \cap_{\tilde{Z}} S_j^F\,, \quad j = 1, \dots, |\mathcal{F}^i(t)|\,, \\ B_0 = A_{|\mathcal{F}^i(t)|}\,, & B_j = B_{j-1} \setminus_{\tilde{Z}} S_P^j\,, \quad j = 1, \dots, |\mathcal{P}^i|\,, \end{cases}$$

and define $\tilde{Z}_{t+1}^i = B_{|\mathcal{P}^i|}$. Using the fourth property of Definition 3.1 and Proposition 2.1, we get that at any time of the algorithm $\tilde{Z}_t^i$ contains $Z_t^i$.

The pseudo-code of this procedure is described in Figure 4. The `select`($\mathcal{A}$) step in Figure 4, where $\mathcal{A} \subset \mathbf{S}$, returns a subset of $\mathcal{A}$ in $\mathbf{S}$. By default, `select`($\mathcal{A}$) `:=` $\mathcal{A}$.

We denote the set of change points candidates at time $t$ as $\tau_t$. Note that for any $(i-1) \in \tau_t$ the sum of $|\mathcal{P}^i|$ and $|\mathcal{F}^i(t)|$ is $|\tau_t|$. With the default `select()` procedure we do $\mathcal{O}(p|\tau_t|)$ operations in Figure 4. By limiting the number of elements returned by `select()` we can reduce the complexity.

*Remark.* For example, if the operator $\mathcal{A} \mapsto$ `select`($\mathcal{A}$), regardless of $|\mathcal{A}|$, always returns a subset of constant size, then the overall complexity of GeomFPOP is at worst equal to that of PELT with $\sum_{t=1}^n \mathcal{O}(p|\tau_t|)$ time complexity.

Using this `updateZone()` procedure we can now informally describe the GeomFPOP algorithm. At each iteration the algorithm will

1. find the minimum value for $Q_t$, $m_t$; and the best position for last change point $\hat{\tau}_t$ (note that this step is standard: as in the PELT algorithm we need to minimize the cost of the last segment defined in Equation 1);

**Algorithm 1** Geometric update rule of $\tilde{Z}_t^i$

```
1: procedure updateZone($\tilde{Z}_{t-1}^i, \mathcal{P}^i, \mathcal{F}^i(t), i, t$)
2: $\tilde{Z}_t^i \leftarrow \tilde{Z}_{t-1}^i$
3: for $S \in$ select($\mathcal{F}^i(t)$) do
4:     $\tilde{Z}_t^i \leftarrow \tilde{Z}_t^i \cap_{\tilde{\mathbf{Z}}} S$
5: end for
6: for $S \in$ select($\mathcal{P}^i$) do
7:     $\tilde{Z}_t^i \leftarrow \tilde{Z}_t^i \setminus_{\tilde{\mathbf{Z}}} S$
8: end for
9: return $\tilde{Z}_t^i$
```

Figure 4: Geometric update rule of $\tilde{Z}_t^i$

2. compute all sets $\tilde{Z}_t^i$ using $\tilde{Z}_{t+1}^i$, $\mathscr{P}^i$, and $\mathscr{F}^i(t)$ with the `updateZone`() procedure;
3. remove changes such that $\tilde{Z}_{t+1}^i$ is empty.

To simplify the pseudo-code of GeomFPOP, we also define the following operators:

1. `bestCost&Tau`($t$) operator returns two values: the minimum value of $Q_t$, $m_t$, and the best position for last change point $\hat{\tau}_t$ at time $t$ (see {sec-UpdateRule);
2. `getPastFutureSets`($i, t$) operator returns a pair of sets ($\mathscr{F}^i(t)$, $\mathscr{P}^i$) for change point candidate $i - 1$ at time $t$;
3. `backtracking`($\hat{\tau}, n$) operator returns the optimal segmentation for $y_{1:n}$.

The pseudo-code of GeomFPOP is presented in Figure 5.

## 4 Approximation Operators $\cap_{\tilde{Z}}$ and $\setminus_{\tilde{Z}}$

The choice of the geometric structure and the way it is constructed directly affects the computational cost of the algorithm. We consider two types of testing set $\tilde{Z} \in \tilde{\mathbf{Z}}$, a S-type set $\tilde{S} \in \mathbf{S}$ (see Definition 2.1) and a hyperrectangle $\tilde{R} \in \mathbf{R}$ defined below.

**Definition 4.1** (Hyperrectangle)**.** Given two vectors in $\mathbb{R}^p$, $\tilde{l}$ and $\tilde{r}$ we define the set $\tilde{R}$, called *hyperrectangle*, as:

$$\tilde{R} = [\tilde{l}_1, \tilde{r}_1] \times \cdots \times [\tilde{l}_p, \tilde{r}_p]\,. \tag{16}$$

We denote the set of all possible sets $\tilde{R}$ as $\mathbf{R}$.

To update the testing sets we need to give the strict definition of the operators $\cap_{\tilde{Z}}$ and $\setminus_{\tilde{Z}}$ for each type of testing set. To facilitate the following discussion, we rename them. For the first type of geometric structure, we rename the testing set $\tilde{Z}$ as $\tilde{S}$, the operators $\cap_{\tilde{Z}}$ and $\setminus_{\tilde{Z}}$ as $\cap_S$ and $\setminus_S$ and $\tilde{Z}$-type approximation as S-type approximation. And, likewise, we rename the testing set $\tilde{Z}$ as $\tilde{R}$, the operators $\cap_{\tilde{Z}}$ and $\setminus_{\tilde{Z}}$ as $\cap_R$ and $\setminus_R$ and $\tilde{Z}$-type approximation as R-type approximation for the second type of geometric structure.

**Algorithm 2** GeomFPOP algorithm

```
 1: procedure GeomFPOP($y, \Omega(\cdot,\cdot), \beta$)
 2: $m_0 \leftarrow 0, \quad Q_0(\theta) \leftarrow 0\,, \quad \tau_0 \leftarrow \emptyset, \quad \{\tilde{Z}^i_{i-1}\}_{i\in\{1,\dots,n\}} \leftarrow \mathbb{R}^p$
 3: for $t = 1, \dots, n$ do
 4:     $Q_t(\theta) \leftarrow \min\{Q_{t-1}(\theta), m_{t-1} + \beta\} + \Omega(\theta, y_t)$
 5:     $(m_t, \hat{\tau}_t) \leftarrow$ bestCost&Tau($t$)
 6:     for $i - 1 \in \tau_t$ do
 7:         $(\mathcal{P}^i, \mathcal{F}^i(t)) \leftarrow$ getPastFutureSets($i, t$)
 8:         $\tilde{Z}^i_t \leftarrow$ updateZone($\tilde{Z}^i_{t-1}, \mathcal{P}^i, \mathcal{F}^i(t), i, t$)
 9:         if $\tilde{Z}^i_t = \emptyset$ then
                $\tau_t \leftarrow \tau_t \backslash \{i-1\}$
10:         end if
11:     end for
12:     $\tau_t \leftarrow (\tau_{t-1}, t-1)$
13: end for
14: return  $cp(n) \leftarrow$ backtracking($\hat{\tau}, n$)
```

Figure 5: GeomFPOP algorithm

## 4.1 S-type Approximation

With this approach, our goal is to keep track of the fact that at time $t = 1, \dots, n$ there is a pair of changes $(u_1, u_2)$, with $u_1 < i < u_2 \leq t$ such that $S^i_{u_2} \subset S^{u_1}_i$ or there is a pair of changes $(v_1, v_2)$, with $i < v_1 < v_2 \leq t$ such that $S^i_{v_1} \cap S^i_{v_2}$ is empty. If at time $t$ at least one of these conditions is met, we can guarantee that the set $\tilde{S}$ is empty, otherwise, we propose to keep as the result of approximation the last future S-type set $S^i_t$, because it always includes the set $Z^i_t$. This allows us to quickly check and prove (if $\tilde{S} = \emptyset$) the emptiness of set $Z^i_t$.

We consider two generic S-type sets, $S$ and $\tilde{S}$ from $\mathbf{S}$, described as in Equation 12 by the functions $s$ and $\tilde{s}$:

$$s(\theta) = \sum_{k=1}^{p} s^k(\theta^k) - \Delta\,, \qquad \tilde{s}(\theta) = \sum_{k=1}^{p} \tilde{s}^k(\theta^k) - \tilde{\Delta}\,. \tag{17}$$

**Definition 4.2.** For all $S$ and $\tilde{S}$ in $\mathbf{S}$ we define the operators $\cap_S$ and $\setminus_S$ as:

$$\begin{aligned} \tilde{S} \cap_S S &= \begin{cases} \emptyset\,, & \text{if } \tilde{S} \cap S = \emptyset\,, \\ \tilde{S}\,, & \text{otherwise}\,. \end{cases} \\ \tilde{S} \setminus_S S &= \begin{cases} \emptyset\,, & \text{if } \tilde{S} \subset S\,, \\ \tilde{S}\,, & \text{otherwise}\,. \end{cases} \end{aligned} \tag{18}$$

As a consequence, we only need an easy way to detect any of these two geometric configurations: $\tilde{S} \cap S$ and $\tilde{S} \subset S$.

In the Gaussian case, the S-type sets are $p$-balls and an easy solution exists based on comparing radii (see **?@sec-InterandExclBalls** for details). In the case of other models (as Poisson or negative binomial), intersection and inclusion tests are performed through an iterative algorithm solving convex problems (see (**append:IntersectionSsets?**)). This iterative approach is not constant in time, which is why we also considered another type of testing set.

## 4.2 R-type Approximation

Here, we approximate the sets $Z^i_t$ by hyperrectangles $\tilde{R}^i_t \in \mathbf{R}$. A key insight of this approximation is that given a hyperrectangle $R$ and an S-type set $S$ we can efficiently (in $\mathcal{O}(p)$ using Proposition 4.2) recover the best hyperrectangle approximation of $R \cup S$ and $R \setminus S$. Formally we define these operators as follows.

**Definition 4.3** (Hyperrectangles Operators $\cap_R$, $\setminus_R$ )**.** For all $R, \tilde{R} \in \mathbf{R}$ and $S \in \mathbf{S}$ we define the operators $\cap_R$ and $\setminus_R$ as:

$$\begin{aligned} R \cap_R S &= \cap_{\{\tilde{R} | R \cap S \subset \mathbf{R}\}} \tilde{R}\,, \\ R \setminus_R S &= \cap_{\{\tilde{R} | R \setminus S \subset \mathbf{R}\}} \tilde{R}\,. \end{aligned}$$

We now explain how we compute these two operators. First, we note that they can be recovered by solving a $2p$ one-dimensional optimization problem.

**Proposition 4.1.** *The k-th minimum coordinates $\tilde{l}_k$ and maximum coordinates $\tilde{r}_k$ of $\tilde{R} = R \cap_R S$ (resp. $\tilde{R} = R \setminus_R S$) is obtained as*

$$\tilde{l}_k \text{ or } \tilde{r}_k = \begin{cases} \min\limits_{\theta_k \in \mathbb{R}} \text{ or } \max\limits_{\theta_k \in \mathbb{R}} \theta_k\,, \\ \text{subject to } \varepsilon s(\theta) \leq 0\,, \\ \qquad\qquad l_j \leq \theta_j \leq r_j\,, \quad j = 1, \dots, p\,, \end{cases} \tag{19}$$

*with $\varepsilon = 1$ (resp. $\varepsilon = -1$).*

To solve the previous problems ($\varepsilon = 1$ or $-1$), we define the following characteristic points.

**Definition 4.4** (Minimal, closest and farthest points)**.** Let $S \in \mathbf{S}$, described by function $s(\theta) = \sum_{k=1}^{p} s^k(\theta^k) - \Delta$ from the family of functions (Equation 11), with $\theta \in \mathbb{R}^p$. We define the minimal point $\mathbf{c} \in \mathbb{R}^p$ of $S$ as:

$$\mathbf{c} = \left\{\mathbf{c}^k\right\}_{k=1,\dots,p}, \quad \text{with} \quad \mathbf{c}^k = \underset{\theta^k \in \mathbb{R}}{Arg\min}\{s^k(\theta^k)\}. \tag{20}$$

Moreover, with $R \in \mathbf{R}$ defined through vectors $l, r \in \mathbb{R}^p$, we define two points of $R$, the closest point $\mathbf{m} \in \mathbb{R}^p$ and the farthest point $\mathbf{M} \in \mathbb{R}^p$ relative to $S$ as

$$\begin{aligned} \mathbf{m} &= \left\{\mathbf{m}^k\right\}_{k=1,\dots,p}, \quad \text{with} \quad \mathbf{m}^k = \underset{l^k \le \theta^k \le r^k}{Arg\min}\left\{s^k(\theta^k)\right\}, \\ \mathbf{M} &= \left\{\mathbf{M}^k\right\}_{k=1,\dots,p}, \quad \text{with} \quad \mathbf{M}^k = \underset{l^k \le \theta^k \le r^k}{Arg\max}\left\{s^k(\theta^k)\right\}. \end{aligned} \tag{21}$$

*Remark.* In the Gaussian case, $S$ is a ball in $\mathbb{R}^p$ and

- $\mathbf{c}$ is the center of the ball;
- $\mathbf{m}$ is the closest point to $\mathbf{c}$ inside $R$;
- $\mathbf{M}$ is the farthest point to $\mathbf{c}$ in $R$.

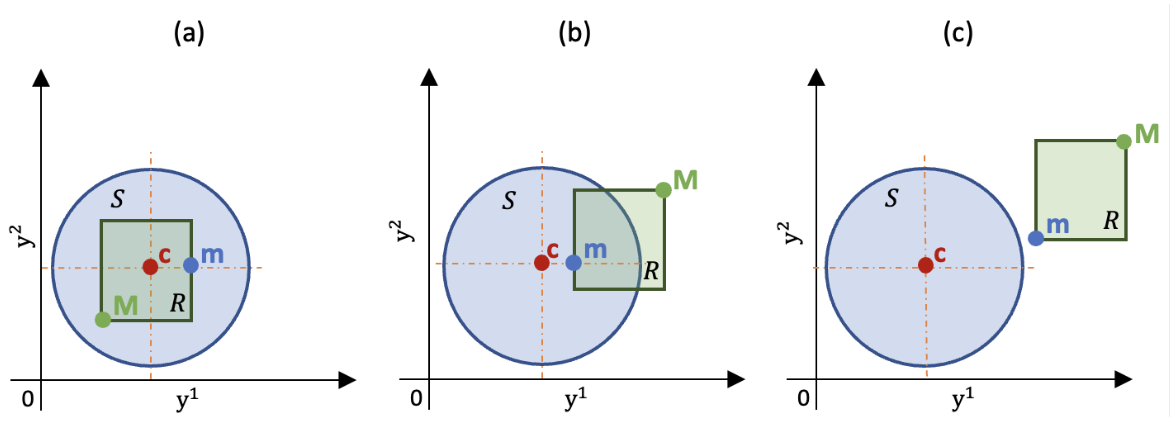

Figure 6: Three examples of minimal point $\mathbf{c}$, closest point $\mathbf{m}$ and farthest point $\mathbf{M}$ for bi-variate Gaussian case: (a) $R \subset S$; (b) $R \cap S \neq \emptyset$; (c) $R \cap S = \emptyset$.

**Proposition 4.2.** *Let $\tilde{R} = R \cap_R S$ (resp. $R \setminus_R S$), with $R \in \mathbf{R}$ and $S \in \mathbf{S}$. We compute the boundaries $(\tilde{l}, \tilde{r})$ of $\tilde{R}$ using the following rule:*

*i) We define the point $\tilde{\theta} \in \mathbb{R}^p$ as the closest point $\mathbf{m}$ (resp. farthest $\mathbf{M}$). For all $k = 1, \dots p$ we find the roots $\theta^{k_1}$ and $\theta^{k_2}$ of the one-variable $(\theta^k)$ equation*

$$s^k(\theta^k) + \sum_{j \neq k} s^j(\tilde{\theta}^j) - \Delta = 0. \tag{22}$$

*If the roots are real-valued we consider that $\theta^{k_1} \le \theta^{k_2}$, otherwise we write $\left[\theta^{k_1}, \theta^{k_2}\right] = \emptyset$.*

*ii) We compute the boundary values $\tilde{l}^k$ and $\tilde{r}^k$ of $\tilde{R}$ as:*

- *For $R \cap_R S$ ($k = 1, \dots, p$):*

$$\left[\tilde{l}^k, \tilde{r}^k\right] = \left[\theta^{k_1}, \theta^{k_2}\right] \cap \left[l^k, r^k\right]. \tag{23}$$

- *For $R \setminus_R S$ ($k = 1, \dots, p$):*

$$\left[\tilde{l}^k, \tilde{r}^k\right] = \begin{cases} \left[l^k, r^k\right] \setminus \left[\theta^{k_1}, \theta^{k_2}\right], & \text{if} \quad \left[\theta^{k_1}, \theta^{k_2}\right] \not\subset \left[l^k, r^k\right], \\ \left[l^k, r^k\right], & \text{otherwise}. \end{cases}$$

*If there is a dimension $k$ for which $\left[\tilde{l}^k, \tilde{r}^k\right] = \emptyset$, then the set $\tilde{R}$ is empty.*

The proof of Proposition 4.2 is presented in **?@sec-proof_prop_solution_rect**.

## 5 Simulation Study of GeomFPOP

In this section, we study the efficiency of GeomFPOP using simulations of multivariate independent time series. For this, we implemented GeomFPOP (with S and R types) and PELT for the Multivariate Independent Gaussian Model in the R-package 'GeomFPOP' https://github.com/lpishchagina/GeomFPOP written in R/C++. By default, the value of penalty $\beta$ for each simulation was defined by the Schwarz Information Criterion proposed in Yao (1984) ($\beta = 2p \log n$).

*Overview of our simulations.* First, as a quality control we made sure that the output of PELT and GeomFPOP were identical on a number of simulated profiles. Second, we studied cases where the PELT approach is not efficient, that is when the data has no or few changes relative to $n$. Indeed, it was shown in Killick, Fearnhead, and Eckley (2012) and Maidstone et al. (2017) that the run time of PELT is close to $\mathcal{O}(n^2)$ in such cases. So we considered simulations of multivariate time series without change (only one segment). By these simulations we evaluated the pruning efficiency of GeomFPOP (using S and R types) for dimension $2 \leq p \leq 10$ (see **?@fig-Figure5** in **?@sec-NC**). For small dimensions ($2 \leq p \leq 4$) we also evaluated the run time of GeomFPOP and PELT and compare them (see **?@fig-Figure6** in **?@sec-TCsmall**). In addition, we considered another approximation of the $Z_t^i$ where we applied our $\cap_R$ and $\setminus_R$ operators only for a randomly selected subset of the past and future balls. In practice, this strategy turned out to be faster computationally than the full/original GeomFPOP and PELT (see **?@fig-Figure7** in **?@sec-GeomFPOP_random**). For this strategy we also generated time series of a fixed size ($10^6$ data points) and varying number of segments and evaluated how the run time vary with the number of segments for small dimensions ($2 \leq p \leq 4$). Our empirical results confirmed that the GeomFPOP (R-type: `random/random`) approach is computationally comparable to PELT when the number of changes is large (see **?@fig-Figure9** in **?@sec-Run_time_segment_nb**).

## References

Aminikhanghahi, Samaneh, and Diane J Cook. 2017. "A Survey of Methods for Time Series Change Point Detection." *Knowledge and Information Systems* 51 (2): 339–67.

Anastasiou, Andreas, and Piotr Fryzlewicz. 2022. "Detecting Multiple Generalized Change-Points by Isolating Single Ones." *Metrika* 85 (February). https://doi.org/10.1007/s00184-021-00821-6.

Andreou, Elena, and Eric Ghysels. 2002. "Detecting Multiple Breaks in Financial Market Volatility Dynamics." *Journal of Applied Econometrics* 17 (5): 579–600. http://www.jstor.org/stable/4129273.

Aue, Alexander, Lajos Horváth, Marie Hušková, and Piotr Kokoszka. 2006. "Change-Point Monitoring in Linear Models." *The Econometrics Journal* 9 (3): 373–403. http://www.jstor.org/stable/23114925.

Auger, Ivan E., and Charles E. Lawrence. 1989. "Algorithms for the Optimal Identification of Segment Neighborhoods." *Bulletin of Mathematical Biology* 51 (1): 39–54. https://doi.org/10.1007/BF02458835.

Bai, Jushan, and Pierre Perron. 2003. "Computation and Analysis of Multiple Structural-Change." *Journal of Applied Econometrics* 18 (January).

Bosc, Marcel, Fabrice Heitz, Jean-Paul Armspach, Izzie Namer, Daniel Gounot, and Lucien Rumbach. 2003. "Automatic Change Detection in Multimodal Serial MRI: Application to Multiple Sclerosis Lesion Evolution." *NeuroImage 20(2)*, 643–56. https://doi.org/https://doi.org/10.1016/S1053-8119(03)00406-3.

Data, Committee, Committee Statistics, Board Applications, Division Sciences, and National Council. 2013. *Frontiers in Massive Data Analysis. Frontiers in Massive Data Analysis*. https://doi.org/10.17226/18374.

Davis, Richard A., Thomas C. M. Lee, and Gabriel A. Rodriguez-Yam. 2006. "Structural Break Estimation for Nonstationary Time Series Models." *Journal of the American Statistical Association* 101: 223–39. https://EconPapers.repec.org/RePEc:bes:jnlasa:v:101:y:2006:p:223-239.

Ducré-Robitaille, Jean-François, Lucie A. Vincent, and Gilles Boulet. 2003. "Comparison of Techniques for Detection of Discontinuities in Temperature Series." *International Journal of Climatology* 23.

Fearnhead, Paul, Robert Maidstone, and Adam Letchford. 2018. "Detecting Changes in Slope with an L0 Penalty." *Journal of Computational and Graphical Statistics*, 1–11.

Frick, Klaus, Axel Munk, and Hannes Sieling. 2013. "Multiscale Change-Point Inference." arXiv. https://doi.org/10.48550/ARXIV.1301.7212.

Fryzlewicz, Piotr. 2014. "Wild Binary Segmentation for Multiple Change-Point Detection." *The Annals of Statistics* 42 (6). https://doi.org/10.1214/14-aos1245.

Galceran, Enric, Alexander Cunningham, Ryan Eustice, and Edwin Olson. 2017. "Multipolicy Decision-Making for Autonomous Driving via Changepoint-Based Behavior Prediction: Theory and Experiment." *Autonomous Robots* 41 (August). https://doi.org/10.1007/s10514-017-9619-z.

Harchaoui, Z., and C. Lévy-Leduc. 2010. "Multiple Change-Point Estimation with a Total Variation Penalty." *Journal of the American Statistical Association* 105 (492): 1480–93. http://www.jstor.org/stable/27920180.

Jackson, Brad, Jeffrey D Scargle, David Barnes, Sundararajan Arabhi, Alina Alt, Peter Gioumousis, Elyus Gwin, Paungkaew Sangtrakulcharoen, Linda Tan, and Tun Tao Tsai. 2005. "An Algorithm for Optimal Partitioning of Data on an Interval." *IEEE Signal Processing Letters* 12 (2): 105–8.

Jewell, Sean, Paul Fearnhead, and Daniela Witten. 2019. "Testing for a Change in Mean After Changepoint Detection." arXiv. https://doi.org/10.48550/ARXIV.1910.04291.

Killick, R., P. Fearnhead, and I. A. Eckley. 2012. "Optimal Detection of Changepoints with a Linear Computational Cost." *Journal of the American Statistical Association* 107 (500): 1590–98.

Lai, Weil R, Mark D Johnson, Raju Kucherlapati, and Peter J Park. 2005. "Comparative Analysis of Algorithms for Identifying Amplifications and Deletions in Array CGH Data." *Bioinformatics* 21 (19): 3763–70.

Lavielle, Marc, and Eric Moulines. 2000. "Least-Squares Estimation of an Unknown Number of Shifts in a Time Series." *Journal of Time Series Analysis* 21 (1): 33–59.

Lebarbier, Emilie. 2005. "Detecting Multiple Change-Points in the Mean of Gaussian Process by Model Selection." *Signal Processing* 85 (April): 717–36. https://doi.org/10.1016/j.sigpro.2004.11.012.

Liehrmann, Arnaud, Guillem Rigaill, and Toby Dylan Hocking. 2021. "Increased Peak Detection Accuracy in over-Dispersed ChIP-Seq Data with Supervised Segmentation Models." *BMC Bioinformatics* 22 (1): 1–18.

Maidstone, Robert, Toby Hocking, Guillem Rigaill, and Paul Fearnhead. 2017. "On Optimal Multiple Changepoint Algorithms for Large Data." *Statistics and Computing* 27 (2): 519–33.

Malladi, Rakesh, Giridhar P. Kalamangalam, and Behnaam Aazhang. 2013. "Online Bayesian Change Point Detection Algorithms for Segmentation of Epileptic Activity." *2013 Asilomar Conference on Signals, Systems and Computers*, 1833–37.

Naoki, Itoh, and Juergen Kurths. 2010. "Change-Point Detection of Climate Time Series by Nonparametric Method." *Lecture Notes in Engineering and Computer Science* 2186 (October).

Olshen, Adam, E. S. Venkatraman, Robert Lucito, and Michael Wigler. 2004. "Circular Binary Segmentation for the Analysis of Array-Based DNA Copy Number Data." *Biostatistics (Oxford,* *England)* 5 (November): 557–72. https://doi.org/10.1093/biostatistics/kxh008.

Picard, Franck, Stephane Robin, Marc Lavielle, Christian Vaisse, and Jean-Jacques Daudin. 2005. "A statistical approach for array CGH data analysis." *BMC Bioinformatics* 6: np. https://doi.org/10.1186/1471-2105-6-27.

Radke, R. J., S. Andra, O. Al-Kofahi, and B. Roysam. 2005. "Image Change Detection Algorithms: A Systematic Survey." *IEEE Transactions on Image Processing* 14 (3): 294–307. https://doi.org/10.1109/TIP.2004.838698.

Ranganathan, Ananth. 2012. "PLISS: Labeling Places Using Online Changepoint Detection." *Auton. Robots* 32 (4): 351–68. https://doi.org/10.1007/s10514-012-9273-4.

Reeves, Jaxk, Jien Chen, Xiaolan L. Wang, Robert Lund, and Qi Qi Lu. 2007. "A Review and Comparison of Changepoint Detection Techniques for Climate Data." *Journal of Applied Meteorology and Climatology* 46 (6): 900–915. https://doi.org/10.1175/JAM2493.1.

Rigaill, Guillem. 2010. "A Pruned Dynamic Programming Algorithm to Recover the Best Segmentations with 1 to $K_{max}$ Change-Points." https://doi.org/10.48550/ARXIV.1004.0887.

Runge, Vincent. 2020. "Is a Finite Intersection of Balls Covered by a Finite Union of Balls in Euclidean Spaces?" *Journal of Optimization Theory and Applications* 187 (2): 431–47.

Rybach, David, Christian Gollan, Ralf Schluter, and Hermann Ney. 2009. "Audio Segmentation for Speech Recognition Using Segment Features." In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, 4197–4200. https://doi.org/10.1109/ICASSP.2009.4960554.

Staudacher, Martin, Stefan Telser, Anton Amann, Hartmann Hinterhuber, and Monika Ritsch-Marte. 2005. "A New Method for Change-Point Detection Developed for on-Line Analysis of the Heart Beat Variability During Sleep." *Physica A-Statistical Mechanics and Its Applications* 349: 582–96.

Truong, Charles, Laurent Oudre, and Nicolas Vayatis. 2020. "Selective Review of Offline Change Point Detection Methods." *Signal Processing* 167: 107299.

Verzelen, Nicolas, Magalie Fromont, Matthieu Lerasle, and Patricia Reynaud-Bouret. 2020. "Optimal Change-Point Detection and Localization." arXiv. https://doi.org/10.48550/ARXIV.2010.11470.

Yao, Yi-Ching. 1984. "Estimation of a Noisy Discrete-Time Step Function: Bayes and Empirical Bayes Approaches." *Ann. Statist.* 12 (4): 1434–47. https://doi.org/10.1214/aos/1176346802.

———. 1988. "Estimating the Number of Change-Points via Schwarz'criterion." *Statistics & Probability Letters* 6 (3): 181–89.

Zhang, Nancy, and David Siegmund. 2007. "A Modified Bayes Information Criterion with Applications to the Analysis of Comparative Genomic Hybridization Data." *Biometrics* 63 (April): 22–32. https://doi.org/10.1111/j.1541-0420.2006.00662.x.

## Session information

```r
sessionInfo()
```

```
R version 4.2.2 (2022-10-31)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 22.04.2 LTS

Matrix products: default
BLAS:   /usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3
LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/libopenblasp-r0.3.20.so

locale:
 [1] LC_CTYPE=C.UTF-8       LC_NUMERIC=C           LC_TIME=C.UTF-8
 [4] LC_COLLATE=C.UTF-8     LC_MONETARY=C.UTF-8    LC_MESSAGES=C.UTF-8
 [7] LC_PAPER=C.UTF-8       LC_NAME=C              LC_ADDRESS=C
[10] LC_TELEPHONE=C         LC_MEASUREMENT=C.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] stats     graphics  grDevices datasets  utils     methods   base

loaded via a namespace (and not attached):
 [1] compiler_4.2.2  fastmap_1.1.0   cli_3.6.0       htmltools_0.5.4
 [5] tools_4.2.2     yaml_2.3.7      rmarkdown_2.20  knitr_1.42
 [9] jsonlite_1.8.4  xfun_0.37       digest_0.6.31   rlang_1.0.6
[13] renv_0.16.0     evaluate_0.20
```