Selective review of offline change point detection methods

Charles Truong\textsuperscript{a,*}, Laurent Oudre\textsuperscript{b}, Nicolas Vayatis\textsuperscript{a}

\textsuperscript{a}CMLA, CNRS, ENS Paris Saclay
\textsuperscript{b}L2TI, University Paris 13

Abstract

This article presents a selective survey of algorithms for the offline detection of multiple change points in multivariate time series. A general yet structuring methodological strategy is adopted to organize this vast body of work. More precisely, detection algorithms considered in this review are characterized by three elements: a cost function, a search method and a constraint on the number of changes. Each of those elements is described, reviewed and discussed separately. Implementations of the main algorithms described in this article are provided within a Python package called ruptures.

Keywords: change point detection, segmentation, statistical signal processing

Contents

1 Introduction 2

2 Background 3

2.1 Notations . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
2.2 Problem formulation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4
2.3 Selection criteria for the review . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4
2.4 Limitations . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5
2.5 Outline of the article . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

3 Evaluation 6

3.1 Consistency . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6
3.2 Evaluation metrics . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6
3.2.1 AnnotationError . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7
3.2.2 Hausdorff . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7
3.2.3 RandIndex . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7
3.2.4 F1-score . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8

4 Models and cost functions 8

4.1 Parametric models . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9
4.1.1 Maximum likelihood estimation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9
4.1.2 Piecewise linear regression . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11
4.1.3 Mahalanobis-type metric . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13
4.2 Non-parametric models . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14
4.2.1 Non-parametric maximum likelihood . . . . . . . . . . . . . . . . . . . . . . . . . 14
4.2.2 Rank-based detection . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15
4.2.3 Kernel-based detection . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15
4.3 Summary table . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17

*Corresponding author

Accepted to Signal Processing December 8, 2020
1. Introduction

A common task in signal processing is the identification and analysis of complex systems whose underlying state changes, possibly several times. This setting arises when industrial systems, physical phenomena or human activity are continuously monitored with sensors. The objective of practitioners is to extract from the recorded signals a posteriori meaningful information about the different states and transitions of the monitored object for analysis purposes. This setting encompasses a broad range of real-world scenarios and a wide variety of signals.

Change point detection is the task of finding changes in the underlying model of a signal or time series. The first works on change point detection go back to the 50s [1, 2]: the goal was to locate a shift in the mean of independent and identically distributed (iid) Gaussian variables for industrial quality control purposes. Since then, this problem has been actively investigated, and is periodically the subject of in-depth monographs [3–6]. This subject has generated important activity in statistics and signal processing [7–9] but also in various application settings such as speech processing [10–13], financial analysis [7, 14, 15], bio-informatics [16–24], climatology [25–27], network traffic data analysis [28, 29]. Modern applications in bioinformatics, finance, monitoring of complex systems have also motivated recent developments from the machine learning community [18, 30, 31].

Let us take the example of gait analysis, illustrated on the flowchart displayed on Figure 1. In this context, a patient’s movements are monitored with accelerometers and gyroscopes while performing simple activities, for instance walking at preferred speed, running or standing still. The objective is to objectively quantify gait characteristics [32–36]. The resulting signal is described as a succession of non-overlapping segments, each one corresponding to an activity and having its own gait characteristics. Insightful features from homogeneous phases can be extracted if the temporal boundaries of those segments are identified. This analysis therefore needs a preliminary processing of the signals: change point detection.

Change point detection methods are divided into two main branches: online methods, that aim to detect changes as soon as they occur in a real-time setting, and offline methods that retrospectively detect changes when all samples are received. The former task is often referred to as event or anomaly detection, while the latter is sometimes called signal segmentation.

In this article, we propose a survey of algorithms for the detection of multiple change points in multivariate time series. All reviewed methods presented in this paper address the problem of offline (also referred to as retrospective or a posteriori) change point detection, in which segmentation is
performed after the signal has been collected. The objective of this article is to facilitate the search of a suitable detection method for a given application. In particular, focus is made on practical considerations such as implementations and procedures to calibrate the algorithms. This review also presents the mathematical properties of the main approaches, as well as the metrics to evaluate and compare their results. This article is linked with a Python scientific library called ruptures [37], that includes a modular and easy-to-use implementation of all the main methods presented in this paper.

2. Background

This section introduces the main concepts for change point detection, as well as the selection criteria and the outline of this review.

2.1. Notations

In the remainder of this article, we use the following notations. For a given signal $y = \{y_t\}_{t=1}^T$, the $(b-a)$-sample long sub-signal $\{y_t\}_{t=a+1}^b$ ($1 \leq a < b \leq T$) is simply denoted $y_{a:b}$; the complete signal is therefore $y = y_{0:T}$. A set of indexes is denoted by a calligraphic letter: $\mathcal{T} = \{t_1, t_2, \ldots\} \subset \{1, \ldots, T\}$, and its cardinal is $|\mathcal{T}|$. For a set of indexes $\mathcal{T} = \{t_1, \ldots, t_K\}$, the dummy indexes $t_0 := 0$ and $t_{K+1} := T$ are implicitly available.
2.2. Problem formulation

Let us consider a multivariate non-stationary random process \( y = \{y_1, \ldots, y_T\} \) that takes value in \( \mathbb{R}^d \) and has \( T \) samples. The signal \( y \) is assumed to be piecewise stationary, meaning that some characteristics of the process change abruptly at some unknown instants \( t_1^* < t_2^* < \cdots < t_K^* \). Change point detection consists in estimating the indexes \( t_k^* \). Depending on the context, the number \( K^* \) of changes may or may not be known, in which case it has to be estimated too.

Formally, change point detection is cast as a model selection problem, which consists in choosing the best possible segmentation \( \mathcal{T} \) according to a quantitative criterion \( V(\mathcal{T}, y) \) that must be minimized. (The function \( V(\mathcal{T}, y) \) is simply denoted \( V(\mathcal{T}) \) when it is obvious from the context that it refers to the signal \( y \).) The choice of the criterion function \( V(\cdot) \) depends on preliminary knowledge on the task at hand.

In this work, we make the assumption that the criterion function \( V(\mathcal{T}) \) for a particular segmentation is a sum of costs of all the segments that define the segmentation:

\[
V(\mathcal{T}, y) := \sum_{k=0}^{K} c(y_{t_k^*..t_{k+1}}^*)
\]  

where \( c(\cdot) \) is a cost function which measures goodness-of-fit of the sub-signal \( y_{t_k^*..t_{k+1}} = \{y_t\}_{t_{k+1}}^{t_k^*+1} \) to a specific model. The “best segmentation” \( \hat{\mathcal{T}} \) is the minimizer of the criterion \( V(\mathcal{T}) \). In practice, depending on whether the number \( K^* \) of change points is known beforehand, change point detection methods fall into two categories.

- **Problem 1 : known number of changes \( K \).** The change point detection problem with a fixed number \( K \) of change points consists in solving the following discrete optimization problem

  \[
  \min_{|\mathcal{T}|=K} V(\mathcal{T}). \tag{P1}
  \]

- **Problem 2 : unknown number of changes.** The change point detection problem with an unknown number of change points consists in solving the following discrete optimization problem

  \[
  \min_{\mathcal{T}} V(\mathcal{T}) + \text{pen}(\mathcal{T}) \tag{P2}
  \]

  where \( \text{pen}(\mathcal{T}) \) is an appropriate measure of the complexity of a segmentation \( \mathcal{T} \).

All change point detection methods considered in this work yield an exact or an approximate solution to either Problem 1 (P1) or Problem 2 (P2), with the function \( V(\mathcal{T}, y) \) adhering to the format (1).

2.3. Selection criteria for the review

To better understand the strengths and weaknesses of change point detection methods, we propose to classify algorithms according to a comprehensive typology. Precisely, detection methods are expressed as the combination of the following three elements.

- **Cost function.** The cost function \( c(\cdot) \) is a measure of “homogeneity”. Its choice encodes the type of changes that can be detected. Intuitively, \( c(y_{a..b}) \) is expected to be low if the sub-signal \( y_{a..b} \) is “homogeneous” (meaning that it does not contain any change point), and large if the sub-signal \( y_{a..b} \) is “heterogeneous” (meaning that it contains one or several change points).

- **Search method.** The search method is the resolution procedure for the discrete optimization problems associated with Problem 1 (P1) and Problem 2 (P2). The literature contains several methods to efficiently solve those problems, in an exact fashion or in an approximate fashion. Each method strikes a balance between computational complexity and accuracy.
Figure 2: Typology of change point detection methods described in this article. Reviewed algorithms are defined by three elements: a cost function, a search method and a constraint (on the number of change points).

- **Constraint (on the number of change points).** When the number of changes is unknown (P2), a constraint is added, in the form of a complexity penalty \( \text{pen}(\cdot) \) (P2), to balance out the goodness-of-fit term \( V(T, y) \). The choice of the complexity penalty is related to the amplitude of the changes to detect: with too “small” a penalty (compared to the goodness-of-fit) in (P2), many change points are detected, even those that are the result of noise. Conversely, too much penalization only detects the most significant changes, or even none.

This typology of change point detection methods is schematically shown on Figure 2.

### 2.4. Limitations

The described framework, however general, does not encompass all published change point detection methods. In particular, Bayesian approaches are not considered in the remainder of this article, even though they provide state-of-the-art results in several domains, such as speech and sound processing. The most well-known Bayesian algorithm is the Hidden Markov Model (HMM) \([38]\). This model was later extended, for instance with Dirichlet processes \([39, 40]\) or product partition models \([41, 42]\). The interested reader can find reviews of Bayesian approaches in \([4]\) and \([6]\).

Also, several literature reviews with different selection criteria can be found. Recent and important works include \([43]\) which focuses on window-based detection algorithms. In particular, the authors use the quantity of samples needed to detect a change as a basis for comparison. Maximum likelihood and Bayes-type detection are reviewed, from a theoretical standpoint, in \([8]\). Existing asymptotic distributions for change point estimates are described for several statistical models. In \([44]\), detection is formulated as a statistical hypothesis testing problem, and emphasis is put on the algorithmic and theoretical properties of several sequential mean-shift detection procedures.

### 2.5. Outline of the article

Before starting this review, we propose in Section 3 a detailed overview of the main mathematical tools that can be used for evaluating and comparing the change point detection methods. The organization of the remaining of this review article reflects the typology of change point detection methods, which is schematically shown on Figure 2. Precisely, the three defining elements of a detection algorithm are reviewed separately. In Section 4, cost functions from the literature are presented, along with the associated signal model and the type of change that can be detected. Whenever possible, theoretical results on asymptotic consistency are also given. Section 5 lists search methods that efficiently solve the discrete optimizations associated with Problem 1 (P1) and Problem 2 (P2). Both exact and approximate methods are described. Constraints on the number of change points are reviewed in Section 6. A summary table of the literature review can be found in Section 7. The last section 8 is dedicated to the presentation of the Python package that goes with this article and propose a modular implementation of all the main approaches described in this article.
3. Evaluation

Change point detection methods can be evaluated either by proving some mathematical properties of the algorithms (such as consistency) in general case, or empirically by computing several metrics to assess the performances on a given dataset.

3.1. Consistency

A natural question when designing detection algorithms is the consistency of estimated change point indexes, as the number of samples $T$ goes to infinity. In the literature, the “asymptotic setting” is intuitively described as follows: the observed signal $y$ is regarded as a realization of a continuous-time process on an equispaced grid of size $1/T$, and “$T$ goes to infinity” means that the spacing of the sampling grid converges to 0. Precisely, for all $\tau \in [0,1]$, let $Y(\tau)$ denote an $\mathbb{R}^d$-valued random variable such that

$$y_t = Y(t/T) \quad \forall t = 1, \ldots, T. \quad (2)$$

The continuous-time process undergoes $K^*$ changes in the probability distribution at the time instants $\tau^*_k \in (0,1)$. Those $\tau^*_k$ are related to the change point indexes $t^*_k$ through the following relationship:

$$t^*_k = \lfloor T \tau^*_k \rfloor. \quad (3)$$

Generally, for a given change point index $t_k$, the associated quantity $\tau_k = t_k/T \in (0,1)$ is referred to as a change point fraction. In particular, the change point fractions $\tau^*_k$ ($k = 1, \ldots, K^*$) of the time-continuous process $Y$ are change point indexes of the discrete-time signal $y$. Note that in this asymptotic setting, the lengths of each regime of $y$ increase linearly with $T$. The notion of asymptotic consistency of a change point detection method is formally introduced as follows.

**Definition 1 (Asymptotic consistency).** A change point detection algorithm is said to be asymptotically consistent if the estimated segmentation $\hat{T} = \{\hat{t}_1, \hat{t}_2, \ldots\}$ satisfies the following conditions, when $T \longrightarrow +\infty$:

(i) $P(|\hat{T}| = K^*) \longrightarrow 1$,

(ii) $\frac{1}{T} \|\hat{T} - T^*\|_\infty \overset{p}{\longrightarrow} 0$,

where the distance between two change point sets is defined by

$$\|\hat{T} - T^*\|_\infty := \max \{ \max_{\hat{t} \in \hat{T}} \min_{t^* \in T^*} |\hat{t} - t^*|, \min_{t^* \in T^*} \max_{\hat{t} \in \hat{T}} |\hat{t} - t^*| \}. \quad (4)$$

In Definition 1, the first condition is trivially verified when the number $K^*$ of change points is known beforehand. As for the second condition, it implies that the estimated change point fractions are consistent, and not the indexes themselves. In general, distances $|\hat{t} - t^*|$ between true change point indexes and their estimated counterparts do not converge to 0, even for simple models [18, 45–47]. As a result, consistency results in the literature only deal with change point fractions.

3.2. Evaluation metrics

Several metrics from the literature are presented below. Each metric correspond to one of the previously listed criteria by which segmentation performances are assessed. In the following, the set of true change points is denoted by $T^* = \{t^*_1, \ldots, t^*_K\}$, and the set of estimated change points is denoted by $\hat{T} = \{\hat{t}_1, \ldots, \hat{t}_\hat{K}\}$. Note that that the cardinals of each set, $K^*$ and $\hat{K}$, are not necessarily equal.
3.2.1. AnnotationError

The **AnnotationError** is simply the difference between the predicted number of change points $|\hat{T}|$ and the true number of change points $|T^*|$:  

$$\text{AnnotationError} := |\hat{K} - K^*|. \quad (5)$$

This metric can be used to discriminate detection method when the number of changes is unknown.

3.2.2. Hausdorff

The **Hausdorff** metric measures the robustness of detection methods [47, 48]. Formally, it is equal to the greatest temporal distance between a change point and its prediction:  

$$\text{Hausdorff}(T^*, \hat{T}) := \max \left\{ \max_{t \in \hat{T}} \min_{t^* \in T^*} |\hat{t} - t^*|, \max_{t^* \in T^*} \min_{t \in \hat{T}} |\hat{t} - t^*| \right\}. \quad (6)$$

It is the worst error made by the algorithm that produced $\hat{T}$ and is expressed in number of samples. If this metric is equal to zero, both breakpoint sets are equal; it is large when a change point from either $T^*$ or $\hat{T}$ is far from every change point of $\hat{T}$ or $T^*$ respectively. Over-segmentation as well as under-segmentation is penalized. An illustrative example is displayed on Figure 3.

3.2.3. RandIndex

Accuracy can be measured by the **RandIndex**, which is the average similarity between the predicted breakpoint set $\hat{T}$ and the ground truth $T^*$ [30]. Intuitively, it is equal to the number of agreements between two segmentations. An agreement is a pair of indexes which are either in the same segment according to both $\hat{T}$ and $T^*$ or in different segments according to both $\hat{T}$ and $T^*$. Formally, for a breakpoint set $T$, the set of grouped indexes and the set of non-grouped indexes are respectively $\text{gr}(T)$ and $\text{ngr}(T)$:

$$\text{gr}(T) := \{(s, t), 1 \leq s < t \leq T \text{ s.t. } s \text{ and } t \text{ belong to the same segment according to } T \},$$
$$\text{ngr}(T) := \{(s, t), 1 \leq s < t \leq T \text{ s.t. } s \text{ and } t \text{ belong to different segments according to } T \}.$$  

The **RandIndex** is then defined as follows:

$$\text{RandIndex}(T^*, \hat{T}) := \frac{|\text{gr}(\hat{T}) \cap \text{gr}(T^*)| + |\text{ngr}(\hat{T}) \cap \text{ngr}(T^*)|}{T(T-1)}. \quad (6)$$

It is normalized between 0 (total disagreement) and 1 (total agreement). Originally, RandIndex has been introduced to evaluate clustering methods [30, 47]. An illustrative example is displayed on Figure 4.
3.2.4. F1-score

Another measure of accuracy is the F1-Score. Precision is the proportion of predicted change points that are true change points. Recall is the proportion of true change points that are well predicted. A breakpoint is considered detected up to a user-defined margin of error $M > 0$; true positives $T_p$ are true change points for which there is an estimated one at less than $M$ samples, i.e.

$$T_p(T^*, \hat{T}) := \{t^* \in T^* \mid \exists \hat{t} \in \hat{T} \text{ s.t. } |\hat{t} - t^*| < M\}.$$ (7)

Precision $\text{Prec}$ and recall $\text{Rec}$ are then given by

$$\text{Prec}(T^*, \hat{T}) := |T_p(T^*, \hat{T})|/\hat{K} \quad \text{and} \quad \text{Rec}(T^*, \hat{T}) := |T_p(T^*, \hat{T})|/K^*.$$ (8)

Precision and recall are well-defined (i.e., between 0 and 1) if the margin $M$ is smaller than the minimum spacing between two true change point indexes $t_k^*$ and $t_{k+1}^*$. Over-segmentation of a signal causes the precision to be close to zero and the recall close to one. Under-segmentation has the opposite effect. The F1-Score is the harmonic mean of precision $\text{Prec}$ and recall $\text{Rec}$:

$$\text{F1-Score}(T^*, \hat{T}) := 2 \times \frac{\text{Prec}(T^*, \hat{T}) \times \text{Rec}(T^*, \hat{T})}{\text{Prec}(T^*, \hat{T}) + \text{Rec}(T^*, \hat{T})}.$$ (9)

Its best value is 1 and its worse value is 0. An illustrative example is displayed on Figure 5.

4. Models and cost functions

This section presents the first defining element of change detection methods, namely the cost function. In most cases, cost functions are derived from a signal model. In the following, models and their associated cost function are organized in two categories: parametric and non-parametric, as schematically shown in Figure 6. For each model, the most general formulation is first given, then special cases, if any, are described. A summary table of all reviewed costs can be found at the end of this section.
4.1. Parametric models

Parametric detection methods focus on changes in a finite-dimensional parameter vector. Historically, they were the first to be introduced, and remain extensively studied in the literature.

4.1.1. Maximum likelihood estimation

Maximum likelihood procedures are ubiquitous in the change point detection literature. They generalize a large number of models and cost functions, such as mean-shifts and scale shifts in normally distributed data \[2, 49–51\], changes in the rate parameter of Poisson distributed data \[39\], etc. In the general setting of maximum likelihood estimation for change detection, the observed signal \(y = \{y_1, \ldots, y_T\}\) is composed of independent random variables, such that

\[
y_t \sim \sum_{k=0}^{K^*} f(\cdot | \theta_k) \mathbb{1}(t_k^* < t \leq t_{k+1}^*)
\]

(M1)

where the \(t_k^*\) are change point indexes, the \(f(\cdot | \theta)\) are probability density functions parametrized by the vector-valued parameter \(\theta\), and the \(\theta_k\) are parameter values. In other words, the signal \(y\) is modelled by iid variables with piecewise constant distribution. The parameter \(\theta\) represents a quantity of interest whose value changes abruptly at the unknown instants \(t_k^*\), which are to be estimated. Under this setting, change point detection is equivalent to maximum likelihood estimation if the sum of cost \(V(\mathcal{T}, y)\) is equal to the negative log-likelihood. The corresponding cost function, denoted \(c_{\text{i.i.d.}}\), is defined as follows.

Cost function 1 \((c_{\text{i.i.d.}})\). For a given parametric family of distribution densities \(\{f(\cdot | \theta) | \theta \in \Theta\}\) where \(\Theta\) is a compact subset of \(\mathbb{R}^p\) (for a certain \(p\)), the cost function \(c_{\text{i.i.d.}}\) is defined by

\[
c_{\text{i.i.d.}}(y_{a..b}) := -\sup_{\theta} \sum_{t=a+1}^{b} \log f(y_t | \theta).
\]

(C1)

Model M1 and the related cost function \(c_{\text{i.i.d.}}\) encompasses a large number of change point methods. Note that, in this context, the family of distributions must be known before performing the detection, usually thanks to prior knowledge on the data. Historically, the Gaussian distribution was first used, to model mean-shifts \([52-54]\) and scale shifts \([39, 50, 55]\). A large part of the literature then evolved towards other parametric distributions, most notably resorting to distributions from the general exponential family \([15, 25, 56]\).

From a theoretical point of view, asymptotic consistency, as described in Definition 1, has been demonstrated, in the case of a single change point, first with Gaussian distribution (fixed variance), then for
several specific distributions, e.g. Gaussian with mean and scale shifts [3, 6, 51, 57], discrete distributions [49], etc. The case with multiple change points has been tackled later. For certain distributions (e.g. Gaussian), the solutions of the change point detection problems (P1) (known number of change points) and (P2) (unknown number of change points) have been proven to be asymptotically consistent [58]. The general case of multiple change points and a generic distribution family has been addressed decades after the change detection problem has been introduced: the solution of the change point detection problem with a known number of changes and a cost function set to $c_{\text{i.i.d.}}$ is asymptotically consistent [59]. This is true if certain assumptions are satisfied: (i) the signal follows the model (M1) for a distribution family that verifies some regularity assumptions (which are no different from the assumptions needed for generic maximum likelihood estimation, without any change point) and (ii) technical assumptions on the value of the cost function on homogeneous and heterogeneous sub-signals. As an example, distributions from the exponential family satisfy those assumptions.

Related cost functions. The general model (M1) has been applied with different families of distributions. We list below three notable examples and the associated cost functions: change in mean, change in mean and scale, and change in the rate parameter of count data.

- The mean-shift model is the earliest and one of the most studied model in the change point detection literature [2, 53, 60–62]. Here, the distribution is Gaussian, with fixed variance. In other words, the signal $y$ is simply a sequence of independent normal random variables with piecewise constant mean and same variance. In this context, the cost function $c_{\text{i.i.d.}}$ becomes $c_{L_2}$, defined below. This cost function is also referred to as the quadratic error loss and has been applied for instance on DNA array data [15] and geology signals [63].
Cost function 2 ($c_{L_2}$). The cost function $c_{L_2}$ is given by

$$c_{L_2}(y_{a..b}) := \sum_{t=a+1}^{b} \|y_t - \bar{y}_{a..b}\|^2_2$$

where $\bar{y}_{a..b}$ is the empirical mean of the sub-signal $y_{a..b}$.

- A natural extension to the mean-shift model consists in letting the variance abruptly change as well. In this context, the cost function $c_{i.i.d}$ becomes $c_{\Sigma}$, defined below. This cost function can be used to detect changes in the first two moments of random (not necessarily Gaussian) variables, even though it is the Gaussian likelihood that is plugged in $c_{i.i.d}$ [8, 49]. It has been applied for instance on stock market time series [49], biomedical data [63], and electric power consumption monitoring [64].

Cost function 3 ($c_{\Sigma}$). The cost function $c_{\Sigma}$ is given by

$$c_{\Sigma}(y_{a..b}) := (b - a) \log \det \hat{\Sigma}_{a..b} + \sum_{t=a+1}^{b} (y_t - \bar{y}_{a..b})' \hat{\Sigma}_{a..b}^{-1} (y_t - \bar{y}_{a..b})$$

where $\bar{y}_{a..b}$ and $\hat{\Sigma}_{a..b}$ are respectively the empirical mean and the empirical covariance matrix of the sub-signal $y_{a..b}$.

- Change point detection has also been applied on count data modelled by a Poisson distribution [39, 65]. More precisely, the signal $y$ is a sequence of independent Poisson distributed random variables with piecewise constant rate parameter. In this context, the cost function $c_{i.i.d}$ becomes $c_{\text{Poisson}}$, defined below.

Cost function 4 ($c_{\text{Poisson}}$). The cost function $c_{\text{Poisson}}$ is given by

$$c_{\text{Poisson}}(y_{a..b}) := -(b - a) \bar{y}_{a..b} \log \bar{y}_{a..b}$$

where $\bar{y}_{a..b}$ is the empirical mean of the sub-signal $y_{a..b}$.

Remark 1. A model slightly more general than (M1) can be formulated by letting the signal samples to be dependant and the distribution function $f(\cdot|\theta)$ to change over time. This can in particular model the presence of unwanted changes in the statistical properties of the signal (for instance in the statistical structure of the noise [49]). The function $f(\cdot|\theta)$ is replaced in (M1) by a sequence of distribution functions $f_t(\cdot|\theta)$ which are not assumed to be identical for all indexes $t$. Changes in the functions $f_t$ are considered nuisance parameters and only the variations of the parameter $\theta$ must be detected. Properties on the asymptotic consistency of change point estimates can be obtained in this context. We refer the reader to [49, 66] for theoretical results.

4.1.2. Piecewise linear regression

Piecewise linear models are often found, most notably in the econometrics literature, to detect so-called “structural changes” [67–69]. In this context, a linear relationship between a response variable and covariates exists, and this relationship changes abruptly at some unknown instants. Formally, the observed signal $y$ follows a piecewise linear model with change points located at the $t^*_k$:

$$\forall t, \ t^*_k < t \leq t^*_{k+1}, \ y_t = x_t' u_k + z_t' v + \varepsilon_t \quad (k = 0, \ldots, K^*)$$

where $u_k \in \mathbb{R}^p$ and $v \in \mathbb{R}^q$ are unknown regression parameters and $\varepsilon_t$ is noise. Under this setting, the observed signal $y$ is regarded as a univariate response variable (ie $d = 1$) and the signals $x = \{x_t\}_{t=1}^T$
and \( z = \{z_t\}_{t=1}^T \) are observed covariates, respectively \( \mathbb{R}^p \)-valued and \( \mathbb{R}^q \)-valued. In this context, change point detection can be carried out by fitting a linear regression on each segment of the signal. To that end, the sum of costs is made equal to the sum of squared residuals. The corresponding cost function, denoted \( c_{\text{linear}} \), is defined as follows.

**Cost function 5 (\( c_{\text{linear}} \)).** For a signal \( y \) (response variable) and covariates \( x \) and \( z \), the cost function \( c_{\text{linear}} \) is defined by

\[
c_{\text{linear}}(y_{a..b}) := \min_{u \in \mathbb{R}^p, v \in \mathbb{R}^q} \sum_{t=a+1}^{b} (y_t - x'_t u - z'_t v)^2.
\]

In the literature, Model (M2) is also known as a *partial* structural change model because the linear relationship between \( y \) and \( x \) changes abruptly, while the linear relationship between \( y \) and \( z \) remains constant. The *pure* structural change model is obtained by removing the term \( z'_t v \) from (M2). This formulation generalizes several well-known models such as the autoregressive (AR) model [12, 70], multiple regressions [69, 71], etc. A more general formulation of (M2) that can accommodate a multivariate response variable \( y \) exists [72], but is more involved, from a notational standpoint. From a theoretical point of view, piecewise linear models are extensively studied in the context of change point detection by a series of important contributions [14, 67–71, 73–77]. When the number of changes is known, the most general consistency result can be found in [14]. A multivariate extension of this result has been demonstrated in [72]. As for the more difficult situation of an unknown number of changes, statistical tests have been proposed for a single change [78] and multiple changes [74]. All of those results are obtained under various sets of general assumptions on the distributions of the covariates and the noise. The most general of those sets can be found in [79]. Roughly, in addition to some technical assumptions, it imposes the processes \( x \) and \( z \) to be weakly stationary within each regime, and precludes the noise process to have a unit root.

**Related cost functions.** In the rich literature related to piecewise linear models, the cost function \( c_{\text{linear}} \) has been applied and extended in several different settings. Two related cost functions are listed below.

- The first one is \( c_{\text{linear}, L_1} \), which was introduced in order to accommodate certain noise distributions with heavy tails [68, 73] and is defined as follows.

**Cost function 6 (\( c_{\text{linear}, L_1} \)).** For a signal \( y \) (response variable) and covariates \( x \) and \( z \), the cost function \( c_{\text{linear}, L_1} \) is defined by

\[
c_{\text{linear}, L_1}(y_{a..b}) := \min_{u \in \mathbb{R}^p, v \in \mathbb{R}^q} \sum_{t=a+1}^{b} |y_t - x'_t u - z'_t v|.
\]

The difference between \( c_{\text{linear}, L_1} \) and \( c_{\text{linear}} \) lies in the norm used to measure errors: \( c_{\text{linear}, L_1} \) is based on a least absolute deviations criterion, while \( c_{\text{linear}} \) is based on a least squares criterion. As a result, \( c_{\text{linear}, L_1} \) is often applied on data with noise distributions with heavy tails [8, 25]. In practice, the cost function \( c_{\text{linear}, L_1} \) is computationally less efficient than the cost function \( c_{\text{linear}} \) because the associated minimization problem (C6) has no analytical solution. Nevertheless, the cost function \( c_{\text{linear}, L_1} \) is often applied on economic and financial data [67–69]. For instance, changes in several economic parameters of the G-7 growth have been investigated using a piecewise linear model and \( c_{\text{linear}, L_1} \) [80].

- The second cost function related to \( c_{\text{linear}} \) has been introduced to deal with piecewise autoregressive signals. The autoregressive model is a popular representation of random processes, where each variable depends linearly on the previous variables. The associated cost function, denoted \( c_{\text{AR}} \), is defined as follows.
Cost function 7 \((c_{AR})\). For a signal \(y\) and an order \(p \geq 1\), the cost function \(c_{AR}\) is defined by

\[
c_{AR}(y_{a..b}) := \min_{u \in \mathbb{R}^p} \sum_{t=a+1}^{b} \|y_t - x_t'u\|^2
\]

where \(x_t := [y_{t-1}, y_{t-2}, \ldots, y_{t-p}]\) is the vector of lagged samples.

The piecewise autoregressive model is a special case of the generic piecewise linear model, where the term \(z_t'v\) is removed (yielding a pure structural change model) and the covariate signal \(x\) is equal to the signal of lagged samples. The resulting cost function \(c_{AR}\) is able to detect shifts in the autoregressive coefficients of a non-stationary process \([21, 70]\). This model has been applied on EEG/ECG time series \([72]\), functional magnetic resonance imaging (fMRI) time series \([81]\) and speech recognition tasks \([12]\).

4.1.3. Mahalanobis-type metric

The cost function \(c_{L_2}\) (C2), adapted for mean-shift detection, can be extended through the use of Mahalanobis-type pseudo-norm. Formally, for any symmetric positive semi-definite matrix \(M \in \mathbb{R}^{d \times d}\), the associated pseudo-norm \(\|\cdot\|_M\) is given by:

\[
\|y_t\|_M := y_t'My_t
\]

for any sample \(y_t\). The resulting cost function \(c_M\) is defined as follows.

Cost function 8 \((c_M)\). The cost function \(c_M\), parametrized by a symmetric positive semi-definite matrix \(M \in \mathbb{R}^{d \times d}\), is given by

\[
c_M(y_{a..b}) := \sum_{t=a+1}^{b} \|y_t - \bar{y}_{a..b}\|_M^2
\]

where \(\bar{y}_{a..b}\) is the empirical mean of the sub-signal \(y_{a..b}\).

Intuitively, measuring distances with the pseudo-norm \(\|\cdot\|_M\) is equivalent to applying a linear transformation on the data and using the regular (Euclidean) norm \(\|\cdot\|\). Indeed, decomposing the matrix \(M = U'U\) yields:

\[
\|y_t - y_s\|_M^2 = \|Uy_t - Uy_s\|^2.
\]

Originally, the metric matrix \(M\) was set equal to the inverse of the covariance matrix, yielding the Mahalanobis metric \([82]\), ie

\[
M = \hat{\Sigma}^{-1}
\]

where \(\hat{\Sigma}\) is the empirical covariance matrix of the signal \(y\). By using \(c_M\), shifts in the mean of the transformed signal can be detected. In practice, the transformation \(U\) (or equivalently, the matrix \(M\)) is chosen to highlight relevant changes. This cost function generalizes all linear transformations of the data samples. In the context of change point detection, most of the transformations are unsupervised, for instance principal component analysis or linear discriminant analysis \([83]\). Supervised strategies are more rarely found, even though there exist numerous methods to learn a task-specific matrix \(M\) in the context of supervised classification \([83–85]\). Those strategies fall under the umbrella of metric learning algorithms. In the change point detection literature, there is only one work that proposes a supervised procedure to calibrate a metric matrix \(M\) \([30]\). In this contribution, the authors use a training set of annotated signals (meaning that an expert has provided the change point locations) to learn \(M\) iteratively. Roughly, at each step, a new matrix \(M\) is generated in order to improve change point detection accuracy on the training signals. However, using the cost function \(c_M\) is not adapted to certain applications, where a linear treatment of the data is insufficient. In that situation, a well-chosen non-linear transformation of the data samples must be applied beforehand \([30]\).
4.2. Non-parametric models

When the assumptions of parametric models are not adapted to the data at hand, non-parametric change point detection methods can be more robust. Three major approaches are presented here, each based on different non-parametric statistics, such as the empirical cumulative distribution function, rank statistics and kernel estimation.

Signal model. Assume that the observed signal \( y_t = \{y_1, \ldots, y_T\} \) is composed of independent random variables, such that

\[
y_t \sim \sum_{k=0}^{K^*} F_k \, \mathbb{I}(t_k^* < t \leq t_{k+1}^*)
\]

where the \( t_k^* \) are change point indexes and the \( F_k \) are cumulative distribution functions (c.d.f.), not necessarily parametric as in (M1). Under this setting, the sub-signal \( y_{t_k^*:t_{k+1}^*} \), bounded by two change points, is composed of iid variables with c.d.f. \( F_k \). When the \( F_k \) belong to a known parametric distribution family, change point detection is performed with the MLE approach described in Section 4.1.1, which consists in applying the cost function \( c_{i,i,d} \). However, this approach is not possible when the distribution family is either non-parametric or not known beforehand.

4.2.1. Non-parametric maximum likelihood

The first non-parametric cost function example, denoted \( c_{\hat{F}} \), has been introduced for the single change point detection problem in [86] and extended for multiple change points in [87]. It relies on the empirical cumulative distribution function, estimated on sub-signals. Formally, the signal is assumed to be univariate (i.e., \( d = 1 \)) and the empirical cdf on the sub-signal \( y_{a:b} \) is given by

\[
\forall u \in \mathbb{R}, \quad \hat{F}_{a:b}(u) := \frac{1}{b-a} \left[ \sum_{t=a+1}^{b} \mathbb{I}(y_t < u) + 0.5 \times \mathbb{I}(y_t = u) \right]. \tag{13}
\]

In order to derive a log-likelihood function that does not depend on the probability distribution of the data, i.e., the \( f(\cdot|\theta_k) \), the authors use the following fact: for a fixed \( u \in \mathbb{R} \), the empirical cdf \( \hat{F} \) of \( n \) iid random variables, distributed from a certain cdf \( F \) is such that \( n\hat{F}(u) \sim \text{Binomial}(n, F(u)) \) [87]. This observation, combined with careful summation over \( u \), allows a distribution-free maximum likelihood estimation. The resulting cost function \( c_{\hat{F}} \) is defined as follows. Interestingly, this strategy was first introduced to design non-parametric two-sample statistical tests, which were experimentally shown to be more powerful than classical tests such as Kolmogorov-Smirnov and Cramér-von Mises [86, 88].

Cost function 9 \((c_{\hat{F}})\). The cost function \( c_{\hat{F}} \) is given by

\[
c_{\hat{F}}(y_{a:b}) := -(b-a) \sum_{u=1}^{T} \hat{F}_{a:b}(u) \log \hat{F}_{a:b}(u) + (1 - \hat{F}_{a:b}(u)) \log(1 - \hat{F}_{a:b}(u)) \frac{(u - 0.5)(T - u + 0.5)}{(u - 0.5)(T - u + 0.5)} \tag{C9}
\]

where the empirical cdf \( \hat{F}_{a:b} \) is defined by (13).

From a theoretical point of view, asymptotic consistency of change point estimates is verified, when the number of change points is either known or unknown [87]. However, solving either one of the detection problems can be computationally intensive, because calculating the value of the cost function \( c_{\hat{F}} \) on one sub-signal requires to sum \( T \) terms, where \( T \) is the signal length. As a result, the total complexity of change point detection is of the order of \( O(T^3) \) [87]. To cope with this computational burden, several preliminary steps are proposed. For instance, irrelevant change point indexes can be removed before performing the detection, thanks to a screening step [87]. Also, the cost function \( c_{\hat{F}} \) can be approximated, by summing, in (C9), over a few (carefully chosen) terms, instead of \( T \) terms originally [89]. Thanks to those implementation techniques, the cost function \( c_{\hat{F}} \) has been applied on DNA sequences [87] and heart-rate monitoring signals [89].
4.2.2. Rank-based detection

In statistical inference, a popular strategy to derive distribution-free statistics is to replaced the data samples by their ranks within the set of pooled observations [28, 90, 91]. In the context of change point detection, this strategy has first been applied to detect a single change point [28, 29], and then has been extended by [92] to find multiple change points. The associated cost function, denoted $c_{\text{rank}}$, is defined as follows. Formally, it relies on the centered $\mathbb{R}$-valued “rank signal” $r = \{r_t\}_{t=1}^T$, given by

$$r_{t,j} := \sum_{s=1}^T \mathbb{I}(y_{s,j} \leq y_{t,j}) - \frac{T + 1}{2}, \quad \forall 1 \leq t \leq T, \forall 1 \leq j \leq d. \quad (14)$$

In other words, $r_{t,j}$ is the (centered) rank of the $j$th coordinate of the $t$th sample, ie $y_{t,j}$, among the $\{y_{1,j}, y_{2,j}, \ldots, y_{T,j}\}$.

Cost function 10 ($c_{\text{rank}}$). The cost function $c_{\text{rank}}$ is given by

$$c_{\text{rank}}(y_{a..b}) := -(b - a) \hat{r}_{a..b}^T \hat{\Sigma}_r^{-1} \hat{r}_{a..b} \quad (C10)$$

where the signal $r$ is defined in (14) and $\hat{\Sigma}_r \in \mathbb{R}^{d \times d}$ is the following matrix

$$\hat{\Sigma}_r := \frac{1}{T} \sum_{t=1}^T (r_t + 1/2)'(r_t + 1/2). \quad (15)$$

Intuitively, $c_{\text{rank}}$ measures changes in the joint behaviour of the marginal rank statistics of each coordinate, which are contained in $r$. One of the advantages of this cost function is that it is invariant under any monotonic transformation of the data. Several well-known statistical hypothesis testing procedures are based on this scheme, for instance the Wilcoxon-Mann-Whitney test [93], the Friedman test [94], the Kruskal-Wallis test [95], and several others [90, 91]. From a computational point of view, two steps must be performed before the change point detection: the calculation of the rank statistics, in $\mathcal{O}(dT \log T)$ operations, and the calculation of the matrix $\hat{\Sigma}_r$, in $\mathcal{O}(d^2T + d^3)$ operations. The resulting algorithm has been applied on DNA sequences [92] and network traffic data [28, 29].

4.2.3. Kernel-based detection

A kernel-based method has been proposed by [96] to perform change point detection in a non-parametric setting. To that end, the original signal $y$ is mapped onto a reproducing Hilbert space (rkhs) $\mathcal{H}$ associated with a user-defined kernel function $k(\cdot, \cdot) : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$. The mapping function $\phi : \mathbb{R}^d \rightarrow \mathcal{H}$ onto this rkhs is implicitly defined by $\phi(y_t) = k(y_t, \cdot) \in \mathcal{H}$, resulting in the following inner-product and norm:

$$\langle \phi(y_s) | \phi(y_t) \rangle_{\mathcal{H}} = k(y_s, y_t) \quad \text{and} \quad \|\phi(y_t)\|^2_{\mathcal{H}} = k(y_t, y_t) \quad (16)$$

for any samples $y_s, y_t \in \mathbb{R}^d$. The associated cost function, denoted $c_{\text{kernel}}$, is defined as follows. This kernel-based mapping is central to many machine learning developments such as support vector machine or clustering [97, 98].

Cost function 11 ($c_{\text{kernel}}$). For a given kernel function $k(\cdot, \cdot) : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$, the cost function $c_{\text{kernel}}$ is given by

$$c_{\text{kernel}}(y_{a..b}) := \sum_{t=a+1}^b \|\phi(y_t) - \bar{\mu}_{a..b}\|^2_{\mathcal{H}} \quad (C11)$$

where $\bar{\mu}_{a..b} \in \mathcal{H}$ is the empirical mean of the embedded signal $\{\phi(y_t)\}_{t=a+1}^b$ and $\|\cdot\|_{\mathcal{H}}$ is defined in (16).

Remark 2 (Computing the cost function). Thanks to the well-known “kernel trick”, the explicit computation of the mapped data samples $\phi(y_t)$ is not required to calculate the cost function value [99].
Indeed, after simple algebraic manipulations, \( c_{\text{kernel}}(y_{a..b}) \) can be rewritten as follows:

\[
c_{\text{kernel}}(y_{a..b}) = \sum_{t=a+1}^{b} k(y_t, y_t) - \frac{1}{b-a} \sum_{s,t=a+1}^{b} k(y_s, y_t).
\] (17)

**Remark 3 (Intuition behind the cost function).** Intuitively, the cost function \( c_{\text{kernel}} \) is able to detect mean-shifts in the transformed signal \( \{\phi(y_t)\} \). Its use is motivated in the context of Model M3 by the fact that, under certain conditions on the kernel function, changes in the probability distribution coincide with mean-shifts in the transformed signal. This connection has been investigated in several works on kernel methods [97, 98, 100, 101]. Formally, let \( \mathbb{P} \) denote a probability distribution defined over \( \mathbb{R}^d \). Then there exists a unique element \( \mu_\mathbb{P} \in \mathcal{H} [100] \), called the mean embedding (of \( \mathbb{P} \)), such that

\[
\mu_\mathbb{P} = \mathbb{E}_{X \sim \mathbb{P}} [\phi(X)].
\] (18)

In addition, the mapping \( \mathbb{P} \mapsto \mu_\mathbb{P} \) is injective (in which case the kernel is said to be characteristic), meaning that

\[
\mu_\mathbb{P} = \mu_\mathbb{Q} \iff \mathbb{P} = \mathbb{Q},
\] (19)

where \( \mathbb{Q} \) denotes a probability distribution defined over \( \mathbb{R}^d \). In order to determine if a kernel is characteristic (and therefore, useful for change point detection), several conditions can be found in the literature [97, 98, 100]. For instance, if a kernel \( k(\cdot, \cdot) \) is translation invariant, meaning that \( k(y_s, y_t) = \psi(y_s - y_t) \forall s, t \), where \( \psi \) is a bounded continuous positive definite function on \( \mathbb{R}^d \), then it is characteristic [100]. This condition is verified by the commonly used Gaussian kernel. As a consequence, two transformed samples \( \phi(y_s) \) and \( \phi(y_t) \) are distributed around the same mean value if they belong to the same regime, and around different mean-values if they each belong to two consecutive regimes. To put it another way, a signal that follows (M3) is mapped by \( \phi(\cdot) \) to a random signal with piecewise constant mean.

From a theoretical point of view, asymptotic consistency of the change point estimates has been demonstrated for both a known and unknown number of change points in the recent work of [102]. This result, as well as an important oracle inequality on the sum of cost \( V(T) \) [103], also holds in a non-asymptotic setting. In addition, kernel change point detection was experimentally shown to be competitive in many different settings, in an unsupervised manner and with very few parameters to manually calibrate. For instance, the cost function \( c_{\text{kernel}} \) was applied on the Brain-Computer Interface (BCI) data set [96], on a video time series segmentation task [103], DNA sequences [99] and emotion recognition [104].

**Related cost functions.** The cost function \( c_{\text{kernel}} \) can be combined with any kernel to accommodate various types of data (not just \( \mathbb{R}^2 \)-valued signals). Notable examples of kernel functions include [101]:

- The linear kernel \( k(x, y) = \langle x | y \rangle \) with \( x, y \in \mathbb{R}^d \).
- The polynomial kernel \( k(x, y) = (\langle x | y \rangle + C)^{\text{deg}} \) with \( x, y \in \mathbb{R}^d \), and \( C \) and \( \text{deg} \) are parameters.
- The Gaussian kernel \( k(x, y) = \exp(-\gamma \|x - y\|^2) \) with \( x, y \in \mathbb{R}^d \) and \( \gamma > 0 \) is the so-called bandwidth parameter.
- The \( \chi^2 \)-kernel \( k(x, y) = \exp(-\gamma \sum_i [(x_i - y_i)^2/(x_i + y_i)]) \) with \( \gamma \in \mathbb{R} \) a parameter. It is often used for histogram data.

Arguably, the most commonly used kernels for numerical data are the linear kernel and the Gaussian kernel. When combined with the linear kernel, the cost function \( c_{\text{kernel}} \) is formally equivalent to \( c_{L_2} \). As for the Gaussian kernel, the associated cost function, denoted \( c_{\text{rbf}} \), is defined as follows.
Cost function 12 ($c_{rbf}$). The cost function $c_{rbf}$ is given by

$$c_{rbf}(y_{a..b}) := (b - a) - \frac{1}{b - a} \sum_{s,t=a+1}^{b} \exp(-\gamma \|y_s - y_t\|^2)$$  \hspace{1cm} (C12)

where $\gamma > 0$ is the so-called bandwidth parameter.

The parametric cost function $c_M$ (based on a Mahalanobis-type norm) can be extended to the non-parametric setting through the use of a kernel. Formally, the Mahalanobis-type norm $\|\cdot\|_{\mathcal{H},M}$ in the feature space $\mathcal{H}$ is defined by

$$\|\phi(y_s) - \phi(y_t)\|_{\mathcal{H},M}^2 = (\phi(y_s) - \phi(y_t))' M (\phi(y_s) - \phi(y_t))$$  \hspace{1cm} (20)

where $M$ is a (possibly infinite dimensional) symmetric positive semi-definite matrix defined on $\mathcal{H}$. The associated cost function, denoted $c_{\mathcal{H},M}$, is defined below. Intuitively, using $c_{\mathcal{H},M}$ instead of $c_M$ introduces a non-linear treatment of the data samples.

Cost function 13 ($c_{\mathcal{H},M}$). For a given kernel function $k(\cdot,\cdot) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ and $M$ a symmetric positive semi-definite matrix defined on the associated rkhs $\mathcal{H}$, the cost function $c_{\mathcal{H},M}$ is given by

$$c_{\mathcal{H},M}(y_{a..b}) := \sum_{t=a+1}^{b} \|\phi(y_t) - \bar{\mu}_{a..b}\|_{\mathcal{H},M}^2$$  \hspace{1cm} (C13)

where $\mu_{a..b}$ is the empirical mean of the transformed sub-signal $\{\phi(y_t)\}_{t=a+1}^{b}$ and $\|\cdot\|_{\mathcal{H},M}$ is defined in (20).

4.3. Summary table

Reviewed cost functions (parametric and non-parametric) are summarized in Table 1. For each cost, the name, expression and parameters of interest are given.
\begin{tabular}{|l|l|l|}
\hline
Name & \(c(y_{a..b})\) & Parameters \\
\hline
\(c_{\text{l.i.d.}}\) \((C1)\) & \(-\sup_b \sum_{t=a+1}^b \log f(y_t|\theta)\) & \(\theta: \) changing parameter; density function: \(f(\cdot|\theta)\) \\
\(c_{\text{L2}}\) \((C2)\) & \(\sum_{t=a+1}^b \|y_t - \bar{y}_{a..b}\|^2\) & \(\bar{y}_{a..b}: \) empirical mean of \(y_{a..b}\) \\
\(c_\Sigma\) \((C3)\) & \((b - a) \log \det \bar{\Sigma}_{a..b} + \sum_{t=a+1}^b (y_t - \bar{y}_{a..b})\bar{\Sigma}_{a..b}^{-1}(y_t - \bar{y}_{a..b})\) & \(\bar{\Sigma}_{a..b}: \) empirical covariance of \(y_{a..b}\) \\
\(c_{\text{Poisson}}\) \((C4)\) & \(-(b - a)\bar{y}_{a..b} \log y_{a..b}\) & \(y_{a..b}: \) empirical mean of \(y_{a..b}\) \\
\(c_{\text{linear}}\) \((C5)\) & \(\min_{u \in \mathbb{R}^p, x \in \mathbb{R}^q} \sum_{t=a+1}^b (y_t - x'_t u - z'_t v)^2\) & \(x_t \in \mathbb{R}^p, z_t \in \mathbb{R}^q: \) covariates \\
\(c_{\text{linear},1}\) \((C6)\) & \(\min_{u \in \mathbb{R}^p, x \in \mathbb{R}^q} \sum_{t=a+1}^b |y_t - x'_t u - z'_t v|\) & \(x_t \in \mathbb{R}^p, z_t \in \mathbb{R}^q: \) covariates \\
\(c_{\text{AR}}\) \((C7)\) & \(\sum_{t=a+1}^b \|y_t - \bar{y}_{a..b}\|^2\) & \(x_t = [y_{t-1}, y_{t-2}, \ldots, y_{t-p}]: \) lagged samples \\
\(c_M\) \((C8)\) & \(-\sum_{t=a+1}^b \|y_t - \bar{y}_{a..b}\|^2_M\) & \(M \in \mathbb{R}^d \times d: \) positive semi-definite matrix \\
\(c_P\) \((C9)\) & \(-(b - a) \sum_{t=1}^T \bar{F}_{\Delta_{a..b}(s)}(\log \hat{F}_{\Delta_{a..b}(s)} + (1 - \hat{F}_{\Delta_{a..b}(s)}) \log(1 - \hat{F}_{\Delta_{a..b}(s)}) )\) & \(\bar{F}: \) empirical c.d.f. \((13)\) \\
\(c_{\text{rank}}\) \((C10)\) & \(-(b - a) \bar{r}_{a..b} \hat{S}_{r.t}^{-1} \bar{r}_{a..b}\) & \(r: \) rank signal \((14): \hat{S}_r: \) empirical covariance of \(r \ (15)\) \\
\(c_{\text{kernel}}\) \((C11)\) & \(\sum_{t=a+1}^b k(y_t, y_t) - \frac{1}{n^2} \sum_{t,s=a+1}^b k(y_s, y_t)\) & \(k(\cdot, \cdot): \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}: \) kernel function \\
\(c_{\text{rbf}}\) \((C12)\) & \(\frac{1}{\gamma} \sum_{t=1}^T \exp(-\gamma \|y_t - y_{t-1}\|^2)\) & \(\gamma > 0: \) bandwidth parameter \\
\(c_{\text{H},M}\) \((C13)\) & \(\sum_{t=a+1}^b \|y_t - \bar{y}_{a..b}\|^2_{\|\cdot\|_M}\) & \(M: \) positive semi-definite matrix \\
\hline
\end{tabular}

Table 1: Summary of cost reviewed functions

5. Search methods

This section presents the second defining element of change detection methods, namely the search method. Reviewed search methods are organized in two general categories, as shown on Figure 7: optimal methods, that yield the exact solution to the discrete optimization of \((P1)\) and \((P2)\), and the approximate methods, that yield an approximate solution. Described algorithms can be combined with cost functions from Section 4. Note that, depending on the chosen cost function, the computational complexity of the complete algorithm changes. As a consequence, in the following, complexity analysis is done with the assumption that applying the cost function on a sub-signal requires \(\mathcal{O}(1)\) operations. Also, the practical implementations of the most important algorithms are given in pseudo-code.

5.1. Optimal detection

Optimal detection methods find the exact solutions of Problem 1 \((P1)\) and Problem 2 \((P2)\). A naive approach consists in enumerating all possible segmentations of a signal, and returning the one that minimizes the objective function. However, for \((P1)\), minimization is carried out over the set \(\{T \ s.t. \ |T| = K\}\) (which contains \((T-1)_K\) elements), and for \((P2)\), over the set \(\{T \ s.t. \ 1 \leq |T| < T\}\) (which contains \(\sum_{K=1}^{T-1} (T-1)_K\) elements). This makes exhaustive enumeration impractical, in both situations. We describe in this section two major approaches to efficiently find the exact solutions of \((P1)\) and \((P2)\).

5.1.1. Solution to Problem 1 \((P1)\): Opt

In \((P1)\), the number of change points to detect is fixed to a certain \(K \geq 1\). The optimal solution to this problem can be computed efficiently, thanks to a method based on dynamic programming. The
algorithm, denoted $\text{Opt}$, relies on the additive nature of the objective function $V(\cdot)$ to recursively solve sub-problems. Precisely, $\text{Opt}$ is based on the following observation:

\[
\min_{|T|=K} V(T, y = y_0..T) = \min_{0=t_0<t_1<\cdots<t_K<T} \sum_{k=0}^{K} c(y_{t_k..t_{k+1}}) \\
= \min_{t \leq T-K} \left[ c(y_{0..t}) + \min_{t=t_0<t_1<\cdots<t_{K-1}<t_K=T} \sum_{k=0}^{K-1} c(y_{t_k..t_{k+1}}) \right] \\
= \min_{t \leq T-K} \left[ c(y_{0..t}) + \min_{|T|=K-1} V(T, y_{t..T}) \right]
\]  \hspace{1cm} (21)

Intuitively, Equation 21 means that the first change point of the optimal segmentation is easily computed if the optimal partitions with $K - 1$ elements of all sub-signals $y_{t..T}$ are known. The complete segmentation is then computed by recursively applying this observation. This strategy, described in detail in Algorithm 1, has a complexity of the order of $O(KT^2)$ [105, 106]. Historically, $\text{Opt}$ was introduced for a non-related problem [107] and later applied to change point detection, in many different contexts, such as EEG recordings [66, 108], DNA sequences [99, 109], tree growth monitoring [20], financial time-series [49, 76], radar waveforms [110], etc.
Algorithm 1 Algorithm Opt

**Input:** signal \( \{y_t\}_{t=1}^T \), cost function \( c(\cdot) \), number of regimes \( K \geq 2 \).

**for all** \((u, v), 1 \leq u < v \leq T\) **do**

Initialize \( C_1(u, v) \leftarrow c(\{y_t\}_{t=u}^v) \).

**end for**

**for** \( k = 2, \ldots, K - 1 \) **do**

**for all** \( u, v \in \{1, \ldots, T\}, v - u \geq k \) **do**

\[
C_k(u, v) \leftarrow \min_{u+k-1 \leq t < v} C_{k-1}(u, t) + C_1(t + 1, v)
\]

**end for**

**end for**

Initialize \( L \), a list with \( K \) elements.

Initialize the last element: \( L[K] \leftarrow T \).

Initialize \( k \leftarrow K \).

**while** \( k > 1 \) **do**

\( s \leftarrow L(k) \)

\( t^* \leftarrow \arg\min_{k-1 \leq t < s} C_{k-1}(1, t) + C_1(t + 1, s) \)

\( L(k - 1) \leftarrow t^* \)

\( k \leftarrow k - 1 \)

**end while**

Remove \( T \) from \( L \)

**Output:** set \( L \) of estimated breakpoint indexes.

**Related search methods.** Several extensions of Opt have been proposed in the literature. The proposed methods still find the exact solution to (P1).

- The first extension is the “forward dynamic programming” algorithm [20]. Contrary to Opt, which returns a single partition, the “forward dynamic programming” algorithm computes the top \( L (L \geq 1) \) most probable partitions (ie with lowest sum of costs). The resulting computational complexity is \( O(LKT^2) \) where \( L \) is the number of computed partitions. This method is designed as a diagnostic tool: change points present in many of the top partitions are considered very likely, while change points present in only a few of the top partitions might not be as relevant. Thanks to “forward dynamic programming”, insignificant change points are trimmed and overestimation of the number of change point is corrected [20], at the expense of a higher computational burden. It is applied on tree growth monitoring time series [20] that are relatively short with around a hundred samples.

- The “pruned optimal dynamic programming” procedure [109] is an extension of Opt that relies on a pruning rule to discard indexes that can never be change points. Thanks to this trick, the set of potential change point indexes is reduced. All described cost functions can be plugged into this method. As a result, longer signals can be handled, for instance long array-based DNA copy number data (up to \( 10^6 \) samples, with the quadratic error cost function) [109]. However, worst case complexity remains of the order of \( O(KT^2) \).

5.1.2. Solution to Problem 2 (P2): Pelt

In (P2), the number of changes point is unknown, and the objective function to minimize is the penalized sum of costs. A naive approach consists in applying Opt for \( K = 1, \ldots, K_{\text{max}} \) for a sufficiently large \( K_{\text{max}} \), then choosing among the computed segmentations the one that minimizes the penalized problem. This would prove computational cumbersome because of the quadratic complexity of the resolution method Opt. Fortunately a faster method exists for a general class of penalty functions, namely linear penalties. Formally, linear penalties are linear functions of the number of change points, meaning that

\[
\text{pen}(T) = \beta |T| \tag{22}
\]
where $\beta > 0$ is a smoothing parameter. (More details on such penalties can be found in Section 6.1.)

The algorithm \textbf{Pelt} (for “Pruned Exact Linear Time”) \cite{91} was introduced to find the exact solution of \((P2)\), when the penalty is linear \((22)\). This approach considers each sample sequentially and, thanks to an explicit pruning rule, may or may not discard it from the set of potential change points. Precisely, for two indexes $t$ and $s$ $(t < s < T)$, the pruning rule is given by:

\[
\text{if } \left[ \min_T V(T, y_{0..t}) + \beta |T| \right] + c(y_{t..s}) \geq \left[ \min_T V(T, y_{0..s}) + \beta |T| \right] \text{ holds,}
\]

then $t$ cannot be the last change point prior to $T$. \hspace{1cm} (23)

This results in a considerable speed-up: under the assumption that regime lengths are randomly drawn from a uniform distribution, the complexity of \textbf{Pelt} is of the order $O(T)$. The detailed algorithm can be found in Algorithm 2. An extension of \textbf{Pelt} is described in \cite{9} to solve the linearly penalized change point detection for a range of smoothing parameter values $[\beta_{\min}, \beta_{\max}]$. \textbf{Pelt} has been applied on DNA sequences \cite{16, 17}, physiological signals \cite{89}, and oceanographic data \cite{91}.

\begin{algorithm}
\caption{Algorithm \textbf{Pelt}}
\begin{algorithmic}
\State \textbf{Input:} signal $\{y_t\}_{t=1}^{T}$, cost function $c(\cdot)$, penalty value $\beta$.
\State Initialize $Z$ a $(T + 1)$-long array; $Z[0] \leftarrow -\beta$.
\State Initialize $L[0] \leftarrow \emptyset$.
\State Initialize $\chi \leftarrow \{0\}$.
\For{$t = 1, \ldots, T$}
\State $i \leftarrow \arg\min_{s \in \chi} \left[ Z[s] + c(y_{s..t}) + \beta \right]$.
\State $Z[t] \leftarrow Z[i] + c(y_{i..t}) + \beta$.
\State $L[t] \leftarrow L[i] \cup \{t\}$.
\State $\chi \leftarrow \{s \in \chi : Z[s] + c(y_{s..t}) \leq Z[t]\} \cup \{t\}$.
\EndFor
\State \textbf{Output:} set $L[T]$ of estimated breakpoint indexes.
\end{algorithmic}
\end{algorithm}

\subsection{5.2. Approximate detection}

When the computational complexity of optimal methods is too great for the application at hand, one can resort to approximate methods. In this section, we describe three major types of approximate segmentation algorithms, namely window-based methods, binary segmentation and bottom-up segmentation. All described procedures fall into the category of sequential detection approaches, meaning that they return a single change point estimate $\hat{t}^{(k)}$ $(1 \leq \hat{t}^{(k)} < T)$ at the $k$-th iteration. (In the following, the subscript $^{(k)}$ refers to the $k$-th iteration of a sequential algorithm.) Such methods can be used to solve (approximately) either \((P1)\) or \((P2)\). Indeed, if the number $K^*$ of changes is known, $K^*$ iterations of a sequential algorithm are enough to retrieve a segmentation with the correct number of changes. If $K^*$ is unknown, the sequential algorithm is run until an appropriate stopping criterion is met.

\subsubsection{5.2.1. Window sliding}

The window-sliding algorithm, denoted \textbf{Win}, is a fast approximate alternative to optimal methods. It consists in computing the discrepancy between two adjacent windows that slide along the signal $y$. For a given cost function $c(\cdot)$, this discrepancy between two sub-signals is given by

\[
\text{d}(y_{a..t}, y_{t..b}) = c(y_{a..b}) - c(y_{a..t}) - c(y_{t..b}) \quad (1 \leq a < t < b \leq T).
\]

(24)

When the two windows cover dissimilar segments, the discrepancy reaches large values, resulting in a peak. In other words, for each index $t$, \textbf{Win} measures the discrepancy between the immediate past
Figure 8: Schematic view of Win

(“left window”) and the immediate future (“right window”). Once the complete discrepancy curve has been computed, a peak search procedure is performed to find change point indexes. The complete Win algorithm is given in Algorithm 3 and a schematic view is displayed on Figure 8. The main benefits of Win are its low complexity (linear in the number of samples) and ease of implementation.

**Algorithm 3 Algorithm Win**

**Input:** signal \( \{y_t\}_{t=1}^T \), cost function \( c(\cdot) \), half-window width \( w \), peak search procedure \( \text{PKSearch} \).

Initialize \( Z \leftarrow [0,0,\ldots] \) a \( T \)-long array filled with 0.

for \( t = w, \ldots, T - w \) do

\( p \leftarrow (t - w) \ldots t \).

\( q \leftarrow t \ldots (t + w) \).

\( r \leftarrow (t - w) \ldots (t + w) \).

\( Z[t] \leftarrow c(y_r) - [c(y_p) + c(y_q)] \).

end for

\( L \leftarrow \text{PKSearch}(Z) \)

**Output:** set \( L \) of estimated breakpoint indexes.

In the literature, the discrepancy measure \( d(\cdot, \cdot) \) is often derived from a two-sample statistical test (see Remark 4), and not from a cost function, as in (24). However, the two standpoints are generally equivalent: for instance, using \( c_{L2}, c_{i.i.d.} \) or \( c_{\text{kernel}} \) is respectively equivalent to applying a Student t-test [3], a generalized likelihood ratio (GLR) [112] test and a kernel Maximum Mean Discrepancy (MMD) test [98]. As a consequence, practitioners can capitalize on the vast body of work in the field of statistical tests to obtain asymptotic distributions for the discrepancy measure [28, 29, 98, 113], and sensible calibration strategies for important parameters of Win (such as the window size or the peak search procedure). Win has been applied in numerous contexts: for instance, on biological signals [11, 114–117], on network data [28, 29], on speech time series [10, 11, 118] and on financial time series [3, 119, 120]. It should be noted that certain window-based detection methods in the literature rely on a discrepancy measure which is not related to a cost function, as in (24) [11, 121–123]. As a result, those methods, initially introduced in the online detection setting, cannot be extended to work
with optimal algorithms (Opt, Pelt).

**Remark 4 (Two-sample test).** A two-sample test (or homogeneity test) is a statistical hypothesis testing procedure designed to assess whether two populations of samples are identical in distribution. Formally, consider two sets of iid $\mathbb{R}^d$-valued random samples $\{x_t\}_t$ and $\{z_t\}_t$. Denote by $P_x$ the distribution function of the $x_t$ and by $P_z$, the distribution function of the $z_t$. A two-sample test procedure compares the two following hypotheses:

\[
H_0 : \quad P_x = P_z \\
H_1 : \quad P_x \neq P_z.
\]

(25)

A general approach is to consider a probability (pseudo)-metric $d(\cdot, \cdot)$ on the space of probability distributions on $\mathbb{R}^d$. Well-known examples of such a metric include the Kullback-Leibler divergence, the Kolmogorov-Smirnov distance, the Maximum Mean Discrepancy (MMD), etc. Observe that, under the null hypothesis, $d(P_x, P_z) = 0$. The testing procedure consists in computing the empirical estimates $\hat{P}_x$ and $\hat{P}_z$ and rejecting $H_0$ for “large” values of the statistics $d(\hat{P}_x, \hat{P}_z)$. This general formulation relies on a consistent estimation of arbitrary distributions from a finite number of samples. In the parametric setting, additional assumptions are made on the distribution functions: for instance, Gaussian assumption [3, 63, 113], exponential family assumption [15, 124], etc. In the non-parametric setting, the distributions are only assumed to be continuous. They are not directly estimated; instead, the statistics $d(\hat{P}_x, \hat{P}_z)$ are computed [11, 90, 98, 122].

In the context of single change point detection, the two-sample test setting is adapted to assess whether a distribution change has occurred at some instant in the input signal. Practically, for a given index $t$, the homogeneity test is performed on the two populations $\{y_s\}_{s \leq t}$ and $\{y_s\}_{s > t}$. The estimated change point location is given by

\[
\hat{t} = \arg\max_t d(\hat{P}_{\leq t}, \hat{P}_{> t})
\]

(26)

where $\hat{P}_{\leq t}$ and $\hat{P}_{> t}$ are the empirical distributions of respectively $\{y_s\}_{s \leq t}$ and $\{y_s\}_{s > t}$.

5.2.2. Binary segmentation

Binary segmentation, denoted BinSeg, is a well-known alternative to optimal methods [53], because it is conceptually simple and easy to implement [63, 111, 125]. BinSeg is a greedy sequential algorithm, outlined as follows. The first change point estimate $\hat{t}^{(1)}$ is given by

\[
\hat{t}^{(1)} := \arg\min_{1 \leq t < T-1} \frac{c(y_{0:t}) + c(y_{t:T})}{V(T=(t))}.
\]

(27)

This operation is “greedy”, in the sense that it searches the change point that lowers the most the sum of costs. The signal is then split in two at the position of $\hat{t}^{(1)}$; the same operation is repeated on the resulting sub-signals until a stopping criterion is met. A schematic view of the algorithm is displayed on Figure 9 and an implementation is given in Algorithm 4. The complexity of BinSeg is of the order of $O(T \log T)$. This low complexity comes at the expense of optimality: in general, BinSeg’s output is only an approximation of the optimal solution. As argued in [111, 126], the issue is that the estimated change points $\hat{t}^{(k)}$ are not estimated from homogeneous segments and each estimate depends on the previous ones. Change points that are close are imprecisely detected especially [8]. Applications of BinSeg range from financial time series [7, 63, 113, 126, 127] to context recognition for mobile devices [128] and array-based DNA copy number data [19, 125, 129].

**Related search methods.** Several extensions of BinSeg have been proposed to improve detection accuracy.

- Circular binary segmentation [125] is a well-known extension of BinSeg. This method is also a sequential detection algorithm that splits the original at each step. Instead of searching for a
single change point in each sub-signal, circular binary segmentation searches two change points. Within each treated sub-segment, it assumes a so-called “epidemic change model”: the parameter of interest shifts from one value to another at the first change point and returns to the original value at the second change point. The algorithm is dubbed “circular” because, under this model, the sub-segment has its two ends (figuratively) joining to form a circle. Practically, this method has been combined with $c_{L_2}$ $C_2$, to detect changes in the mean of array-based DNA copy number data [125, 130, 131]. A faster version of the original algorithm is described in [132].

- Another extension of BinSeg is the wild binary segmentation algorithm [127]. In a nutshell, a single point detection is performed on multiple intervals with start and end points that are drawn uniformly. Small segments are likely to contain at most one change but have lower statistical power, while the opposite is true for long segments. After a proper weighting of the change score to account for the differences on sub-signals’ length, the algorithm returns the most “pronounced” ones, i.e., those that lower the most the sum of costs. An important parameter of this method is the number of random sub-segments to draw. Wild binary search is combined with $c_{L_2}$ $C_2$ to detect mean-shifts of univariate piecewise constant signals (up to 2000 samples) [127].

5.2.3. Bottom-up segmentation

Bottom-up segmentation, denoted BotUp, is the natural counterpart of BinSeg. Contrary to BinSeg, BotUp starts by splitting the original signal in many small sub-signals and sequentially merges them until there remain only $K$ change points. At every step, all potential change points (indexes separating adjacent sub-segments) are ranked by the discrepancy measure $d(\cdot, \cdot)$, defined in 24, between the segments they separate. Change points with the lowest discrepancy are then deleted, meaning that the segments they separate are merged. BotUp is often dubbed a “generous” method, by opposition to BinSeg, which is “greedy” [133]. A schematic view of the algorithm is displayed on Figure 10 and an implementation is provided in Algorithm 5. Its benefits are its linear computational complexity and conceptual simplicity. However, if a true change point does not belong to the original set of indexes, BotUp never considers it. Moreover, in the first iterations, the merging procedure can be unstable because it is performed on small segments, for which statistical significance is smaller. In the literature, BotUp is somewhat less studied than its counterpart, BinSeg: no theoretical convergence study
Algorithm 4 Algorithm BinSeg

Input: signal \( \{y_t\}_{t=1}^T \), cost function \( c(\cdot) \), stopping criterion.

Initialize \( L \leftarrow \{\} \). \( \triangleright \) Estimated breakpoints.

repeat

\( k \leftarrow |L| \). \( \triangleright \) Number of breakpoints

\( t_0 \leftarrow 0 \) and \( t_{k+1} \leftarrow T \) \( \triangleright \) Dummy variables.

if \( k > 0 \) then

Denote by \( t_i \) \( (i = 1, \ldots, k) \) the elements (in ascending order) of \( L \), ie \( L = \{t_1, \ldots, t_k\} \).

end if

Initialize \( G \) a \((k+1)\)-long array. \( \triangleright \) list of gains

for \( i = 0, \ldots, k \) do

\( G[i] \leftarrow c(y_{t_i..t_{i+1}}) - \min_{t_i < t < t_{i+1}} [c(y_{t_i..t}) + c(y_{t..t_{i+1}})] \).

end for

\( \hat{i} \leftarrow \text{argmax}_i G[i] \)

\( \hat{t} \leftarrow \text{argmin}_{t_i < t < t_{i+1}} [c(y_{t_i..t}) + c(y_{t..t_{i+1}})] \).

\( L \leftarrow L \cup \{\hat{t}\} \)

until stopping criterion is met.

Output: set \( L \) of estimated breakpoint indexes.

is available. It has been applied on speech time series to detect mean and scale shifts [119]. Besides, the authors of [133] have found that BotUp outperforms BinSeg on ten different data sets such as physiological signals (ECG), financial time-series (exchange rate), industrial monitoring (water levels), etc.

6. Estimating the number of changes

This section presents the third defining element of change detection methods, namely the constraint on the number of change points. Here, the number of change points is assumed to be unknown (P2). Existing procedures are organized by the penalty function that they are based on. Common heuristics are also described. The organization of this section is schematically shown in Figure 11.

6.1. Linear penalty

Arguably the most popular choice of penalty [111], the linear penalty (also known as \( l_0 \) penalty) generalizes several well-known criteria from the literature such as the Bayesian Information Criterion (BIC) and the Akaike Information Criterion (AIC) [134, 135]. The linear penalty, denoted \( \text{pen}_{l_0} \), is formally defined as follows.

Penalty 1 (\( \text{pen}_{l_0} \)). The penalty function \( \text{pen}_{l_0} \) is given by

\[
\text{pen}_{l_0}(T) := \beta |T| \tag{28}
\]

where \( \beta > 0 \) is the smoothing parameter.

Intuitively, the smoothing parameter controls the trade-off between complexity and goodness-of-fit (measured by the sum of costs): low values of \( \beta \) favour segmentations with many regimes and high values of \( \beta \) discard most change points.

Calibration. From a practical standpoint, once the cost function has been chosen, the only parameter to calibrate is the smoothing parameter. Several approaches, based on model selection, can be found in the literature: they assume a model on the data, for instance (M1), (M2), (M3), and choose a value of \( \beta \) that optimizes a certain statistical criterion. The best-known example of such an approach is
BIC, which aims at maximizing the constrained log-likelihood of the model. The exact formulas of several linear penalties, derived from model selection procedures, are given in the following paragraph. Conversely, when no model is assumed, different heuristics are applied to tune the smoothing parameter. For instance, one can use a procedure based on cross-validation [136] or the slope heuristics [137]. In [138, 139], supervised algorithms are proposed: the chosen $\beta$ is the one that minimizes an approximation of the segmentation error on an annotated set of signals.

Related penalties. A number of model selection criteria are special cases of the linear penalty $\text{pen}_{l_0}$. For instance, under Model (M1) (iid with piecewise constant distribution), the constrained likelihood that is derived from the BIC and the penalized sum of costs are formally equivalent, upon setting $c = c_{\text{i.i.d.}}$ and $\text{pen} = \text{pen}_{\text{BIC}}$, where $\text{pen}_{\text{BIC}}$ is defined as follows.

**Penalty 2 (pen}_{\text{BIC}}).** The penalty function $\text{pen}_{\text{BIC}}$ is given by

$$\text{pen}_{\text{BIC}}(T) := \frac{p}{2} \log T |T|$$

(29)

where $p \geq 1$ is the dimension of the parameter space in (M1).

In the extensively studied model of an univariate Gaussian signal, with fixed variance $\sigma^2$ and piecewise constant mean, the penalty $\text{pen}_{\text{BIC}}$ becomes $\text{pen}_{l_2}$, defined below. Historically, it was one of the first penalties introduced for change point detection [134, 140].

**Penalty 3 (pen}_{\text{BIC},l_2).** The penalty function $\text{pen}_{\text{BIC},l_2}$ is given by

$$\text{pen}_{\text{BIC},l_2}(T) := \sigma^2 \log T |T|,$$

(30)

where $\sigma$ is the standard deviation and $T$ is the number of samples.

In the same setting, AIC, which is a generalization of Mallows’ $C_p$ [62], also yields a linear penalty, namely $\text{pen}_{\text{AIC},l_2}$, defined as follows.
\begin{algorithm}[h]
\hspace*{0.02\textwidth} \textbf{Algorithm 5: Algorithm BotUp} \\
\textbf{Input:} signal \( \{y_t\}_{t=1}^T \), cost function \( c(\cdot) \), stopping criterion, grid size \( \delta > 2 \). \\
\text{Initialize} \( L \leftarrow \{\delta, 2\delta, \ldots, (\lfloor T/\delta \rfloor - 1)\delta\} \quad \triangleright \text{Estimated breakpoints.} \\
\text{repeat} \\
\hspace{0.5cm} k \leftarrow |L| \quad \triangleright \text{Number of breakpoints} \\
\hspace{0.5cm} t_0 \leftarrow 0 \text{ and } t_{k+1} \leftarrow T \quad \triangleright \text{Dummy variables.} \\
\hspace{0.5cm} \text{Denote by } t_i \ (i = 1, \ldots, k) \text{ the elements (in ascending order) of } L, \text{ ie } L = \{t_1, \ldots, t_k\}. \\
\hspace{0.5cm} \text{Initialize } G \text{ a } (k - 1)\text{-long array.} \\
\hspace{0.5cm} \text{for } i = 1, \ldots, k - 1 \text{ do} \\
\hspace{1cm} \hat{i} \leftarrow \text{argmin}_i G[i] \\
\hspace{1cm} \text{Remove } t_{i+1} \text{ from } L. \\
\hspace*{0.5\textwidth} \text{end for} \\
\text{until} \text{ stopping criterion is met.} \\
\textbf{Output:} set \( L \) of estimated breakpoint indexes.
\end{algorithm}

\textbf{Penalty 4 (pen}_{\text{AIC, } L_2} \). The penalty function \( \text{pen}_{\text{AIC, } L_2} \) is given by
\[ \text{pen}_{\text{AIC, } L_2}(T) := \sigma^2 |T|. \] (31)
where \( \sigma \) is the standard deviation.

6.2. Fused lasso

For the special case where the cost function is \( c_{L_2} \), a faster alternative to \( \text{pen}_{l_0} \) can be used. To that end, the \( l_0 \) penalty is relaxed to a \( l_1 \) penalty [18, 48]. The resulting penalty function, denoted \( \text{pen}_{l_1} \), is defined as follows.

\textbf{Penalty 5 (pen}_{l_1} \). The penalty function \( \text{pen}_{l_1} \) is given by
\[ \text{pen}_{l_1}(T) := \beta \sum_{k=1}^{\lfloor |T| \rfloor} \|\bar{y}_{t_{k-1}}^{t_k} - \bar{y}_{t_k}^{t_{k+1}}\|_1 \] (32)
where \( \beta > 0 \) is the smoothing parameter, the \( t_k \) are the elements of \( T \) and \( \bar{y}_{t_{k-1}}^{t_k} \) is the empirical mean of sub-signal \( y_{t_{k-1}}^{t_k} \).

This relaxation strategy (from \( l_0 \) to \( l_1 \)) is shared with many developments in machine learning, for instance sparse regression, compressive sensing, sparse PCA, dictionary learning [83], where \( \text{pen}_{l_1} \) is also referred to as the fused lasso penalty. In numerical analysis and image denoising, it is also known as the total variation regularizer [13, 18, 48]. Thanks to this relaxation, the optimization of the penalized sum of costs (1) in (P2) is transformed into a convex optimization problem, which can be solved efficiently using Lars (for “least absolute shrinkage and selection operator”) [18, 48]. The resulting complexity is of this order of \( \mathcal{O}(T \log T) \) [83, 141]. From a theoretical standpoint, under the mean-shift model (piecewise constant signal with Gaussian white noise), the estimated change point fractions are asymptotically consistent [48]. This result is demonstrated for an appropriately converging sequence of values of \( \beta \). This consistency property is obtained even though classical assumptions from the Lasso regression framework (such as the irrepresentable condition) are not satisfied [48]. In the literature, \( \text{pen}_{l_1} \), combined with \( c_{L_2} \), is applied on DNA sequences [16, 18], speech signals [12] and climatological data [142].

6.3. Complex penalties

Several other penalty functions can be found in the literature. However they are more complex, in the sense that the optimization of the penalized sum of cost is not tractable. In practice, the solution
is found by computing the optimal segmentations with \( K \) change points, with \( K = 1, 2, \ldots, K_{\text{max}} \) for a sufficiently large \( K_{\text{max}} \), and returning the one that minimizes the penalized sum of costs. When possible, the penalty can also be approximated by a linear penalty, in which case, \texttt{Pelt} can be used. In this section, we describe two examples of complex penalties. Both originate from theoretical considerations, under the univariate mean-shift model, with the cost function \( c_{L_2} \). The first example is the modified BIC criterion (mBIC) \cite{143}, which consists in maximizing the asymptotic posterior probability of the data. The resulting penalty function, denoted \( \text{pen}_{\text{mBIC}} \), depends on the number and repartition of the change point indexes: intuitively, it favours evenly spaced change points.

**Penalty 6 (pen_{mBIC}).** The penalty function \( \text{pen}_{\text{mBIC}} \) is given by

\[
\text{pen}_{\text{mBIC}}(T) := 3|T| \log T + \sum_{k=0}^{|T|+1} \log \left( \frac{t_{k+1} - t_k}{T} \right) \tag{33}
\]

where the \( t_k \) are the elements of \( T \).

In \cite{144}, a model selection procedure leads to another complex penalty function, namely \( \text{pen}_{\text{Leb}} \). Upon using this penalty function, the penalized sum of costs satisfied a so-called oracle inequality, which holds in a non-asymptotic setting, contrary to the other penalties previously described.

**Penalty 7 (pen_{Leb}).** The cost function \( \text{pen}_{\text{Leb}} \) is given by

\[
\text{pen}_{\text{Leb}}(T) := |T| + 1 - \sigma^2(a_1 \log \frac{|T| + 1}{T} + a_2) \tag{34}
\]

where \( a_1 > 0 \) and \( a_2 > 0 \) are positive parameters and \( \sigma^2 \) is the noise variance.

### 7. Summary table

This literature review is summarized in Table 2. When applicable, each publication is associated with a search method (such as \texttt{Opt}, \texttt{Pelt}, \texttt{BinSeg} or \texttt{Win}): this is a rough categorization rather than an
exact implementation. Note that \texttt{Pelt} (introduced in 2012) is sometimes associated with publications prior to 2012. It is because some linear penalties [62, 143] were introduced long before \texttt{Pelt} was, and authors then resorted to quadratic (at best) algorithms. Nowadays, the same results can be obtained faster with \texttt{Pelt}. A guide of computational complexity is also provided. Quadratic methods are the slowest and have only one star while linear methods are given three stars. Algorithms for which the number of change points is an explicit input parameter work under the “known \textit{K}” assumption. Algorithms that can be used even if the number of change points is unknown work under the “unknown \textit{K}” assumption. (Certain methods can accommodate both situations.)
<table>
<thead>
<tr>
<th>Publication</th>
<th>Search method</th>
<th>Cost function</th>
<th>Known K</th>
<th>Scalability (wrt T)</th>
<th>Package</th>
<th>Additional information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sen and Srivastava (1975), Vostrikova (1981)</td>
<td>BinSeg</td>
<td>$cL_2$</td>
<td>✓</td>
<td>-</td>
<td>***</td>
<td>✓</td>
</tr>
<tr>
<td>Yao (1988)</td>
<td>Opt</td>
<td>$cL_2$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Basseville and Nikiforov (1993)</td>
<td>Opt</td>
<td>$c_{i.i.d.} \cdot cL_2$</td>
<td>-</td>
<td>-</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Bai (1994), Bai and Perron (2003)</td>
<td>Opt</td>
<td>$c_{linear} \cdot cL_2$</td>
<td>-</td>
<td>-</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Bai (1995)</td>
<td>Opt</td>
<td>$c_{linear} \cdot L_1$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Larivière (1998)</td>
<td>Opt</td>
<td>$cAR$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Bai (2000)</td>
<td>Opt</td>
<td>$cAR$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Bai and Perron (2003)</td>
<td>Opt</td>
<td>$cL_2$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Olshen et al. (2004), Venkatraman and Olshen (2007)</td>
<td>BinSeg</td>
<td>$cL_2$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>✓</td>
</tr>
<tr>
<td>Lebarbier (2005)</td>
<td>Opt</td>
<td>$cL_2$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Douc et al. (2005)</td>
<td>Win</td>
<td>$cKernel$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Harchaoui and Cappé (2007)</td>
<td>Opt</td>
<td>$cKernel \cdot cbf$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Zhang and Siegmund (2007)</td>
<td>Naive</td>
<td>$cL_2$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Harchaoui et al. (2009)</td>
<td>Win</td>
<td>$cAR$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Lévy-Leduc and Ronef (2009), Lung-Yut-Fong et al. (2012)</td>
<td>Win</td>
<td>$cRank$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>✓</td>
</tr>
<tr>
<td>Bai (2010)</td>
<td>Opt</td>
<td>$cL_2 \cdot c\Sigma$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Vert and Bleakley (2010)</td>
<td>Fused Lasso</td>
<td>$cL_2$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Harchaoui and Lévy-Leduc (2010)</td>
<td>Fused Lasso</td>
<td>$cL_2$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Arlot et al. (2012)</td>
<td>Opt</td>
<td>$cKernel \cdot cbf$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Killick et al. (2012)</td>
<td>Opt</td>
<td>any (·)</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Angelosante and Giannakis (2012)</td>
<td>Fused Lasso</td>
<td>$cAR$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Liu et al. (2013)</td>
<td>Win</td>
<td>$cAR$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Hocking et al. (2013)</td>
<td>Naive</td>
<td>$cL_2$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Fryzlewicz (2014)</td>
<td>BinSeg</td>
<td>$cL_2$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>✓</td>
</tr>
<tr>
<td>Lajugie et al. (2014)</td>
<td>Opt</td>
<td>$cM$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Frick et al. (2014)</td>
<td>BinSeg</td>
<td>$c\text{i.i.d.}$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>✓</td>
</tr>
<tr>
<td>Lung-Yut-Fong et al. (2015)</td>
<td>Opt</td>
<td>$cRank$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>✓</td>
</tr>
<tr>
<td>Garreau and Arlot (2017)</td>
<td>Naive</td>
<td>$cKernel \cdot cbf$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>-</td>
</tr>
<tr>
<td>Haynes et al. (2017)</td>
<td>Naive</td>
<td>any (·)</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>✓</td>
</tr>
<tr>
<td>Chakar et al. (2017)</td>
<td>Naive</td>
<td>$cAR$</td>
<td>✓</td>
<td>✓</td>
<td>***</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 2: Summary table of literature review.
8. Presentation of the Python package

Most of the approaches presented in this article are included in a Python scientific library for multiple change point detection in multivariate signals called ruptures [37]. The ruptures library is written in pure Python and available on Mac OS X, Linux and Windows platforms. Source code is available from [37] under the BSD license and deployed with a complete documentation that includes installation instructions and explanations with code snippets on advance use.

A schematic view is displayed on Figure 12. Each block of this diagram is described in the following brief overview of ruptures’ features.

- **Search methods** Our package includes the main algorithms from the literature, namely dynamic programming, detection with a \( l_0 \) constraint, binary segmentation, bottom-up segmentation and window-based segmentation. This choice is the result of a trade-off between exhaustiveness and adaptiveness. Rather than providing as many methods as possible, only algorithms which have been used in several different settings are included. In particular, numerous “mean-shift only” detection procedures were not considered. Implemented algorithms have sensible default parameters that can be changed easily through the functions’ interface.

- **Cost functions** Cost functions are related to the type of change to detect. Within ruptures, one has access to parametric cost functions that can detect shifts in standard statistical quantities (mean, scale, linear relationship between dimensions, autoregressive coefficients, etc.) and non-parametric cost functions (kernel-based or Mahalanobis-type metric) that can, for instance, detect distribution changes [30, 96].

- **Constraints** All methods can be used whether the number of change points is known or not. In particular, ruptures implements change point detection under a cost budget and with a linear penalty term [17, 111].

- **Evaluation** Evaluation metrics are available to quantitatively compare segmentations, as well as a display module to visually inspect algorithms’ performances.

- **Input** Change point detection can be performed on any univariate or multivariate signal that fits into a Numpy array. A few standard non-stationary signal generators are included.

- **Consistent interface and modularity** Discrete optimization methods and cost functions are the two main ingredients of change point detection. Practically, each is related to a specific object in the code, making the code highly modular: available optimization methods and cost functions can be connected and composed. An appreciable by-product of this approach is that a new contribution, provided its interface follows a few guidelines, can be integrated seamlessly into ruptures.
• **Scalability** Data exploration often requires to run several times the same methods with different sets of parameters. To that end, a cache is implemented to keep intermediate results in memory, so that the computational cost of running the same algorithm several times on the same signal is greatly reduced. We also add the possibility for a user with speed constraints to sub-sample their signals and set a minimum distance between change points.

9. Conclusion

In this article, we have reviewed numerous methods to perform change point detection, organized within a common framework. Precisely, all methods are described as a collection of three elements: a cost function, a search method and a constraint on the number of changes to detect. This approach is intended to facilitate prototyping of change point detection methods: for a given segmentation task, one can pick among the described elements to design an algorithm that fits its use-case. Most detection procedures described above are available within the Python language from the package ruptures [37], which is the most comprehensive change point detection library. Its consistent interface and modularity allow painless comparison between methods and easy integration of new contributions. In addition, a thorough documentation is available for novice users. Thanks to the rich Python ecosystem, ruptures can be used in coordination with numerous other scientific libraries.
References


