Journal of Machine Learning Research 24 (2023) 1-45 Submitted 5/22; Revised 2/23; Published 5/23

Random Forests for Change Point Detection

Malte Londschien MALTE.LONDSCHIEN@AI.ETHZ.CH
Peter Bühlmann PETER.BUEHLMANN@STAT.MATH.ETHZ.CH
Solt Kovács SOLT.KOVACS@STAT.MATH.ETHZ.CH
Seminar for Statistics
ETH Zürich 8092 Zürich, Switzerland

Editor: Zaid Harchaoui

Abstract

We propose a novel multivariate nonparametric multiple change point detection method using classifiers. We construct a classifier log-likelihood ratio that uses class probability predictions to compare different change point configurations. We propose a computationally feasible search method that is particularly well suited for random forests, denoted by `changeforest`. However, the method can be paired with any classifier that yields class probability predictions, which we illustrate by also using a k-nearest neighbor classifier. We prove that it consistently locates change points in single change point settings when paired with a consistent classifier. Our proposed method `changeforest` achieves improved empirical performance in an extensive simulation study compared to existing multivariate nonparametric change point detection methods. An efficient implementation of our method is made available for `R`, Python, and Rust users in the `changeforest` software package.

Keywords: break point detection, classifiers, multivariate time series, nonparametric

1. Introduction

Change point detection considers the localization of abrupt distributional changes in ordered observations, often time series. We focus on offline problems, retrospectively detecting changes after all samples have been observed. Inferring abrupt structural changes has a wide range of applications, including bioinformatics (Olshen et al., 2004; Picard et al., 2005), neuroscience (Kaplan et al., 2001), biochemistry (Hotz et al., 2013), climatology (Reeves et al., 2007), and finance (Kim et al., 2005). A rich literature has developed around statistical methods that recover change points in different scenarios, see Truong et al. (2020) for a recent review.

Parametric change point detection methods typically assume that observations between change points stem from a finite-dimensional family of distributions. Change points can then be estimated by maximizing a regularized log-likelihood over various segmentations. The classical scenario of independent univariate Gaussian variables with constant variance and changes in the mean goes back to the 1950s (Page, 1954, 1955). It has recently been studied

©2023 Malte Londschien, Peter Bühlmann, and Solt Kovács.

License: CC-BY 4.0, see `https://creativecommons.org/licenses/by/4.0/`. Attribution requirements are provided at `http://jmlr.org/papers/v24/22-0512.html`.

by, among others, Frick et al. (2014), Fryzlewicz (2014), and references therein. Pein et al. (2017) consider a relaxed version that allows changes in the variance in addition to the mean shifts. Generalizations exist for multivariate scenarios. Recently, even high-dimensional scenarios have been studied: Wang and Samworth (2018) and Enikeeva and Harchaoui (2019) consider multivariate Gaussian observations with sparse mean shifts, Wang et al. (2021) study the problem of changing covariance matrices of sub-Gaussian random vectors, Roy et al. (2017) study estimation in Markov random field models, and Londschien et al. (2021) consider time-varying Gaussian graphical models.

Nonparametric change point detection methods use measures that do not rely on parametric forms of the distribution or the nature of change. Proposals for univariate nonparametric change point detection methods include Pettitt (1979), Carlstein (1988), Dümbgen (1991), and, more recently, Zou et al. (2014) and Madrid–Padilla et al. (2021a). Multivariate setups are challenging even in parametric scenarios. The few multivariate nonparametric change point detection methods we are aware of are based on ranks (Lung-Yut-Fong et al., 2015), distances (Matteson and James, 2014; Chen and Zhang, 2015; Zhang and Chen, 2021), kernel distances (Arlot et al., 2019; Garreau and Arlot, 2018; Chang et al., 2019), kernel densities (Madrid–Padilla et al., 2021b), and relative density ratio estimation (Liu et al., 2013).

Many of these nonparametric proposals are related to hypothesis testing. For a single change point, they maximize a test statistic that measures the dissimilarity of distributions. In the modern era of statistics and machine learning, many nonparametric methods are available to learn complex conditional class probability distributions, such as random forests (Breiman, 2001) and neural networks (McCulloch and Pitts, 1943). These have proven to often outperform simple rank and distance-based methods. Friedman (2004) proposed to use binary classifiers for two-sample testing. Recently, Lopez-Paz and Oquab (2017) applied this framework in combination with neural networks and Hediger et al. (2022) with random forests. Similar to such two-sample testing approaches, we use classifiers to construct a novel class of multivariate nonparametric multiple change point detection methods.

1.1 Our contribution

We propose a novel classifier-based nonparametric change point detection framework. Motivated by parametric change point detection, we construct a classifier log-likelihood ratio that uses class probability predictions to compare different change point configurations. Theoretical results for the population case motivate the development of our algorithm.

We present a novel search method based on binary segmentation that requires a constant number of classifier fits to find a single change point. We prove that this method consistently recovers a single change point when paired with a classifier that provides consistent class probability predictions. For multiple change points, the number of classifier fits required scales linearly with the number of change points, making the algorithm highly efficient. Our method is implemented for random forests and k-nearest neighbor classifiers in the `changeforest` software package, available for `R`, Python, and Rust users. The algorithm achieves improved empirical performance compared to existing multivariate nonparametric change point detection methods.

2. A nonparametric classifier log-likelihood ratio

We are interested in detecting structural breaks in the probability distribution of a time series. More formally, consider a sequence of independent random variables $(X_i)_{i=1}^n \subset \mathbb{R}^p$ with distributions $\tilde{\mathbb{P}}_1, \ldots, \tilde{\mathbb{P}}_n$. Let

$$\alpha^0 := \{0, n\} \cup \{i : \tilde{\mathbb{P}}_i \neq \tilde{\mathbb{P}}_{i+1}\}$$

be the set of *segment boundaries* and denote with $K^0 := |\alpha^0| - 1$ the total number of segments. We label the elements of α^0 by their natural order starting with zero. Then consecutive elements in α^0 define segments $(\alpha^0_{k\text{-}1}, \alpha^0_k]$ for $k = 1, \ldots, K^0$ within which the map $i \mapsto \tilde{\mathbb{P}}_i$ is constant and the $X_i \sim \mathbb{P}_k := \tilde{\mathbb{P}}_{\alpha_k}$ are i.i.d. We call $\alpha^0_1, \ldots, \alpha^0_{K^0\text{-}1}$ the *change points* of the sequence $X_1, \ldots, X_n$. We aim to estimate the change points (or equivalently α^0) of the time series process $X_1, \ldots, X_n$ upon observing a realization $x_1, \ldots, x_n$. We construct a classifier log-likelihood ratio and use it for change point detection. We motivate this ratio later in Section 2.2, drawing parallels to parametric methods that we introduce in Section 2.1.

For any segmentation α, let $\mathbf{y}_\alpha = (y_i)_{i=1}^n$ be the sequence such that $y_i = k$ whenever $i \in (\alpha_{k\text{-}1}, \alpha_k]$. Consider a classification algorithm $\hat{p}$ that can produce class probability predictions and denote with $\hat{p}_\alpha$ the classifier trained on covariates $\mathbf{X} = (x_i)_{i=1}^n$ and labels $\mathbf{y}_\alpha$. Write $\hat{p}_\alpha(x_i)_k$ for the trained classifier's class k probability prediction for observation x_i. The *classifier log-likelihood ratio* for change point detection is

$$G\left((x_i)_{i=1}^n \mid \alpha, \hat{p}\right) := \sum_{k=1}^{K} \sum_{i=\alpha_{k\text{-}1}+1}^{\alpha_k} \log\left(\frac{n}{\alpha_k - \alpha_{k\text{-}1}} \hat{p}_\alpha(x_i)_k\right). \tag{1}$$

Consider a setup with a single change point. If a split candidate s is close to that change point, we expect the trained classifier $\hat{p}_{\{0,s,n\}}$ to perform better than random at separating $x_1, \ldots x_s$ from $x_{s+1}, \ldots x_n$ and $G\left((x_i)_{i=1}^n \mid \{0, s, n\}, \hat{p}\right)$ will take a positive value. For a segmentation α with multiple change points, the classifier log-likelihood ratio (1) measures the separability of the segments $(\alpha^0, \alpha_1], \ldots, (\alpha_{K\text{-}1}, \alpha_K]$. We expect this separability to be highest for $\alpha = \alpha^0$, which we formalize in Proposition 2.

2.1 The parametric setting

Since first proposals from Page (1954), parametric approaches are typically based on maximizing a parametric log-likelihood, see also Frick et al. (2014) and references therein. For this, one assumes that the $X_i \sim \mathbb{P}_k$ are i.i.d. within segments $i \in (\alpha^0_{k\text{-}1}, \alpha^0_k]$ and that the distributions $\mathbb{P}_k \in \{\mathbb{P}_\vartheta \mid \vartheta \in \Theta\}$ for some finite-dimensional parameter space Θ. The distribution of $(X_i)_{i=1}^n$ can then be parametrized with the tuple $(\alpha^0, (\vartheta^0_k)_{k=1}^{K^0})$ such that for all $k = 1, \ldots, K^0$ and $i = \alpha^0_{k\text{-}1} + 1, \ldots, \alpha^0_k$ the X_i are $\mathbb{P}_{\vartheta^0_k}$-distributed.

Assuming the $\mathbb{P}_\vartheta$ have densities p_ϑ, we can express the log-likelihood of observing some sequence $(x_i)_{i=1}^n$ in a setup parametrized by $(\alpha, (\vartheta_k)_{k=1}^K)$ as

$$\ell\left((x_i)_{i=1}^n \mid \alpha, (\vartheta_k)_{k=1}^K\right) = \sum_{k=1}^{K} \sum_{i=\alpha_{k\text{-}1}+1}^{\alpha_k} \log(p_{\vartheta_k}(x_i)). \tag{2}$$

An estimate of α^0 can be obtained by maximizing (2). Since the log-likelihood is increasing in $K = |\alpha| - 1$, an additional penalty term $\gamma > 0$ has to be subtracted whenever the number of change points is unknown. Choose

$$\hat{\alpha}_\gamma \in \operatorname*{argmax}_{\alpha \in \mathcal{A}} \max_{\vartheta_1, \ldots, \vartheta_K} \ell\left((x_i)_{i=1}^n \mid \alpha, (\vartheta_k)_{k=1}^K\right) - |\alpha|\gamma \ , \tag{3}$$

where $\mathcal{A}$ is a set of possible segmentations, typically restricted such that each segment has a minimum length of δn for some $\delta > 0$.

The most popular parametrization of in-segment distributions are Gaussian distributions with known variance and a changing mean, see for example Yao (1988) and Fryzlewicz (2014). Next to changes in the mean, Killick et al. (2012) also study changes in the variance, where the mean of the Gaussian is unknown but constant over all segments. Kim and Siegmund (1989) study changes in the parameters of a linear regression model. Shen and Zhang (2012) and Cleynen and Lebarbier (2014) consider changes in the rate of a Poisson and changes in the success rate in a negative binomial, with applications in detecting copy number alterations in DNA. Frick et al. (2014) consider the general case where in-segment distributions belong to an exponential family.

2.2 The nonparametric setting

Often, a good parametric model is hard to justify and nonparametric methods can be applied instead. Again, consider a classification algorithm $\hat{p}$ that yields class probability predictions after training and recall that $\mathbf{y}_\alpha = (y_i)_{i=1}^n$ with $y_i = k$ whenever $i \in (\alpha_{k\text{-}1}, \alpha_k]$ and that $\mathbf{X} = (x_i)_{i=1}^n$. For any α, training $\hat{p}$ on $(\mathbf{X}, \mathbf{y}_\alpha)$ yields the function $\hat{p}_\alpha := \hat{p}(\mathbf{X}, \mathbf{y}_\alpha)$ that assigns class k probability estimates $\hat{p}_\alpha(x)_k$ corresponding to segments $(\alpha_{k\text{-}1}, \alpha_k]$ to observations x. Write $\mathbb{P}_{(u,v]} := \frac{1}{v-u}\sum_{i=u+1}^v \tilde{\mathbb{P}}_i$ for the mixture distribution of $X_{u+1}, \ldots, X_v$ and assume that densities $p_{(u,v]}$ exist. Assuming a uniform prior on class probabilities, the trained classifier $\hat{p}_\alpha$ estimates

$$\hat{p}_\alpha(x)_k \approx \mathbb{P}(Y = k \mid X = x) = \frac{\mathrm{d}\mathbb{P}(X = x \mid Y = k)}{\mathrm{d}\mathbb{P}(X = x)}\mathbb{P}(Y = k) = \frac{p_{(\alpha_{k\text{-}1}, \alpha_k]}(x_i)}{p_{(0,n]}(x_i)}\frac{\alpha_k - \alpha_{k\text{-}1}}{n}$$

for $k = 1, \ldots K$. This yields an approximation

$$\frac{p_{(\alpha_{k\text{-}1}, \alpha_k]}(x)}{p_{(0,n]}(x)} \approx \frac{n}{\alpha_k - \alpha_{k\text{-}1}}\hat{p}_\alpha(x)_k \ ,$$

which we use in the definition of the classifier log-likelihood ratio (1):

$$G\left((x_i)_{i=1}^n \mid \alpha, \hat{p}\right) = \sum_{k=1}^K \sum_{i=\alpha_{k\text{-}1}+1}^{\alpha_k} \log\left(\frac{n}{\alpha_k - \alpha_{k\text{-}1}}\hat{p}_\alpha(x_i)_k\right) \approx \sum_{k=1}^K \sum_{i=\alpha_{k\text{-}1}+1}^{\alpha_k} \log\left(\frac{p_{(\alpha_{k\text{-}1}, \alpha_k]}(x_i)}{p_{(0,n]}(x_i)}\right).$$

This resembles the parametric log-likelihood (2), where $p_{(\alpha^0_{k\text{-}1}, \alpha^0_k]} = p_{\vartheta^0_k}$, up to the term $-\sum_{i=1}^n \log\big(p_{(0,n]}(x_i)\big)$, which is independent of α. In Equation (1), the classification algorithm $\hat{p}$ takes the role of the parametrization by $\vartheta \in \Theta$ in Equation (2). Analogously to (3), a nonparametric estimator for α^0 could be given by

$$\hat{\alpha}_\gamma \in \operatorname*{argmax}_{\alpha \in \mathcal{A}} G\left((x_i)_{i=1}^n \mid \alpha, \hat{p}\right) - \gamma|\alpha| \ , \tag{4}$$

where $\mathcal{A}$ is a suitable set of possible segmentations.

Remark 1 *To further motivate the classifier log-likelihood ratio (1), take a generative classifier $\hat{p}$ (as in linear discriminant analysis). Choose $\hat{\vartheta}((u,v]) \in \operatorname{argmax}_\vartheta \sum_{i=u+1}^{v} \log(p_\vartheta(x_i))$ and for any segmentation α let $\pi_k := \frac{\alpha_{k\text{-}1}-\alpha_k}{n}$, such that $\hat{p}_\alpha(x_i)_k = \frac{\pi_k \; p_{\hat{\vartheta}((\alpha_{k\text{-}1},\alpha_k])}(x_i)}{\sum_{j=1}^{K} \pi_j \; p_{\hat{\vartheta}((\alpha_{j\text{-}1},\alpha_j])}(x_i)}$. Then the classifier log-likelihood ratio is equal to the parametric log-likelihood ratio between the segment distributions $p_{\hat{\vartheta}((\alpha_{k\text{-}1},\alpha_k])}$ and the mixture model $\sum_{j=1}^{K} \pi_j \; p_{\hat{\vartheta}((\alpha_{j\text{-}1},\alpha_j])}(\cdot)_j$:*

$$
\begin{aligned}
G\left((x_i)_{i=1}^n \mid \alpha, \hat{p}\right) &= \sum_{k=1}^{K} \sum_{i=\alpha_{k\text{-}1}+1}^{\alpha_k} \log\left(\frac{p_{\hat{\vartheta}((\alpha_{k\text{-}1},\alpha_k])}(x_i)}{\sum_{j=1}^{K} \pi_j p_{\hat{\vartheta}((\alpha_{j\text{-}1},\alpha_j])}(x_i)} \right) \\
&= \ell\left((x_i)_{i=1}^n \mid (\alpha, (\hat{\vartheta}((\alpha_{k\text{-}1},\alpha_k]))_{k=1}^K) \right) - \sum_{n=1}^{n} \log\left(\sum_{j=1}^{K} \pi_j \; p_{\hat{\vartheta}((\alpha_{j\text{-}1},\alpha_j])}(x_i) \right) .
\end{aligned}
$$

The following proposition suggests that for any estimator approximating the Bayes classifier, maximizing G over segmentations α yields a reasonable estimator for α^0.

Proposition 2 *Let*

$$
p_\alpha^\infty(x) := \mathbb{P}\left(Y = k \mid X = x\right) = \left(\frac{\alpha_k - \alpha_{k\text{-}1}}{n} \frac{p_{(\alpha_{k\text{-}1},\alpha_k]}}{p_{(0,n]}} \right)_{k=1}^{|\alpha|-1}
$$

be the infinite sample Bayes classifier corresponding to the segmentation α. If $\mathcal{A}$ is a set of segmentations containing α^0 and $D_{KL}(P \parallel Q)$ is the Kullback-Leibler divergence of Q with respect to P, then

$$
\begin{aligned}
\max_{\alpha \in \mathcal{A}} \mathbb{E}\left[G\left((X_i)_{i=1}^n \mid \alpha, p^\infty\right)\right] &= \mathbb{E}\left[G\left((X_i)_{i=1}^n \mid \alpha^0, p^\infty\right)\right] \\
&= \sum_{k=1}^{K} (\alpha_k^0 - \alpha_{k\text{-}1}^0) D_{KL}(\mathbb{P}_k \parallel \mathbb{P}_{(0,n]})
\end{aligned}
$$

and any maximizer of $\mathbb{E}\left[G\left((X_i)_{i=1}^n \mid \alpha, p^\infty\right)\right]$ is equal to α^0 or is a segmentation containing α^0. In particular, the maximizer α with the smallest cardinality is equal to α^0.

3. The `changeforest` algorithm

Finding a solution to the nonparametric estimator (4) by evaluating the classifier log-likelihood ratio (1) for all possible candidate segmentations $\alpha \in \mathcal{A}$ is infeasible. We present an efficient search procedure based on binary segmentation that finds an approximate solution to (4). We furthermore motivate the use of random forests as classifiers and present a model selection procedure adapted to our search procedure. These building blocks combine to form the `changeforest` algorithm, which we summarize in Section 3.5. Finally, we present a consistency result for our method in the presence of a single change point.

3.1 Binary segmentation

Binary segmentation (Vostrikova, 1981) is a popular greedy algorithm to obtain an approximate solution to the parametric maximum log-likelihood estimator (3). For some $\hat{\vartheta}((u,v]) \in \operatorname{argmax}_\vartheta \sum_{i=u+1}^{v} \log(p_\vartheta(x_i))$, binary segmentation recursively splits segments at the split s maximizing the gain

$$G_{(u,v]}(s) := \sum_{i=u+1}^{s} \log\left(\frac{p_{\hat{\vartheta}((u,s])}(x_i)}{p_{\hat{\vartheta}((u,v])}(x_i)}\right) + \sum_{i=s+1}^{v} \log\left(\frac{p_{\hat{\vartheta}((s,v])}(x_i)}{p_{\hat{\vartheta}((u,v])}(x_i)}\right), \tag{5}$$

the increase in log-likelihood, until a stopping criterion is met. For change in mean, where $\hat{\vartheta}((u,v]) = \frac{1}{v-u}\sum_{i=u+1}^{v} x_i$ and $p_\vartheta(x) = \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}(x-\vartheta)^2\right)$, the normalized gain $\frac{2}{v-u}G_{(u,v]}(s)$ is equal to $\left(\sqrt{\frac{v-s}{s-u}}\sum_{i=u+1}^{s} x_i - \sqrt{\frac{s-u}{v-s}}\sum_{i=s+1}^{v} x_i\right)^2$, the square of the CUSUM statistic, first presented by Page (1954). Binary segmentation typically requires $\mathcal{O}(Kn\log(n))$ evaluations of the gain, where K is the number of change points, and is typically faster than search methods based on dynamic programming such as PELT (Killick et al., 2012).

We replace the parametric log-likelihood ratio $G_{(u,v]}(s)$ in binary segmentation with the nonparametric classifier log-likelihood ratio from (1)

$$\begin{aligned} G\left((x_i)_{i=u+1}^{s} \mid \{u,s,v\}, \hat{p}\right) &= \sum_{i=u+1}^{s} \log\left(\frac{v-u}{s-u}\hat{p}_{\{u,s,v\}}(x_i)_1\right) + \sum_{i=s+1}^{v} \log\left(\frac{v-u}{v-s}\hat{p}_{\{u,s,v\}}(x_i)_2\right) \\ &\approx \sum_{i=u+1}^{s} \log\left(\frac{p_{(u,s]}(x_i)}{p_{(u,v]}(x_i)}\right) + \sum_{i=s+1}^{v} \log\left(\frac{p_{(s,v]}(x_i)}{p_{(u,v]}(x_i)}\right). \end{aligned} \tag{6}$$

In many parametric settings, the expected gain curve $G_{(u,v]}$ will be piecewise convex between the underlying change points (Kovács et al., 2020b). Proposition 3 shows that the same holds for the nonparametric variant (6) in the population case. Figure 1 shows the nonparametric gains for a random forest and a k-nearest neighbor classifier applied to simulated data with two change points.

Proposition 3 *For the Bayes classifier p^∞, the expected classifier log-likelihood ratio*

$$s \mapsto \mathbb{E}\left[G\left((X_i)_{i=u+1}^{v} \mid \{u,s,v\}, p^\infty\right)\right]$$

is piecewise convex between the underlying change points α^0, with strict convexity in the segment $(\alpha_{k\text{-}1}^0, \alpha_k^0]$ if $\mathbb{P}_{(u,\alpha_{k\text{-}1}^0]} \neq \mathbb{P}_{(\alpha_{k\text{-}1}^0,\alpha_k^0]}$ or $\mathbb{P}_{(\alpha_{k\text{-}1}^0,\alpha_k^0]} \neq \mathbb{P}_{(\alpha_k^0,v]}$.

3.2 The two-step search algorithm

Binary segmentation relies on the full grid search, where $G_{(u,v]}(s)$ is evaluated for all $s = u+1,\ldots,v$, to find the maximizer of the gain $G_{(u,v]}(s)$. In many traditional parametric settings, such as change in mean, log-likelihoods of neighboring segments can be recovered using cheap $\mathcal{O}(1)$ updates. This enables change point detection with binary segmentation in $\mathcal{O}(Kn\log(n))$ time, where K is the number of change points.

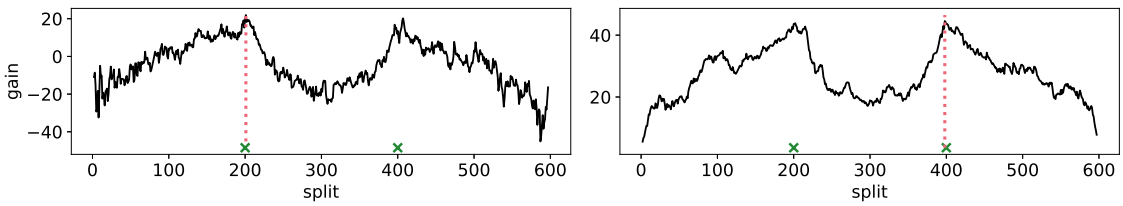

Figure 1: Classifier-based gain curves using a random forest (left) and a k-nearest neighbor classifier (right). The time series with 600 observations of dimension 5 and change points at $t = 200, 400$ (green crosses) were simulated with changing covariances, as described in Section 4.2. The maximizer is marked in red.

For many classifiers, random forests in particular, such updates are unavailable and the classifiers need to be recomputed from scratch, making the computational cost of the grid search in binary segmentation prohibitive. Similar computational issues arise in high-dimensional regression models. Here, Kaul et al. (2019) propose a two-step approach. Starting with an initial guess $s^{(0)}$, they fit a single high-dimensional linear regression for each of the segments $(u, s^{(0)}]$ and $(s^{(0)}, v]$ and generate an improved estimate of the optimal split using the resulting residuals. They repeat the procedure two times and show a result about consistency in the high-dimensional regression setting with a single change point if the change point is sufficiently close to the first guess $s^{(0)}$.

We apply a variant of the two-step search paired with the classifier log-likelihood ratio for multiple change point scenarios. Instead of residuals, we recycle the class probability predictions and the resulting classifier log-likelihood ratios for individual observations from a single classifier fit $\hat{p}_{\{u,s^{(0)},v\}}$, approximating

$$G\left((x_i)_{i=u+1}^{v} \mid \{u,s,v\}, \hat{p}\right) \approx \sum_{i=u+1}^{s} \log\left(\frac{v-u}{s^{(0)}-u}\hat{p}_{\{u,s^{(0)},v\}}(x_i)_1\right) + \sum_{i=s+1}^{v} \log\left(\frac{v-u}{v-s^{(0)}}\hat{p}_{\{u,s^{(0)},v\}}(x_i)_2\right).$$

We call this the *approximate gain* in the following. Note that the classifier and normalization factors are fixed, but the summation varies with the split s. Like Kaul et al. (2019), we compute this twice, using the first maximizer of the approximate gain as a second guess $s^{(1)}$. This allows us to find local maxima of the nonparametric gain (6) in a constant number of classifier fits.

Kaul et al. (2019) only apply the two-step search in scenarios with a single change point. In scenarios with multiple change points, as encountered in binary segmentation, the class probability predictions from the initial classifier fit might contain little information if $\mathbb{P}_{(u,s^{(0)}]} \approx \mathbb{P}_{(s^{(0)},v]}$. To avoid this, we start with multiple initial guesses and select the split point corresponding to the overall highest approximate gain as the second guess. The resulting two-step algorithm, as implemented in `changeforest`, using three initial guesses at the segment's 25%, 50%, and 75%-quantiles is presented in Algorithm 2. Proposition 4 suggests good estimation performance when coupling the two-step search with our classifier-

based approach in single change point scenarios. We present a consistency result of the two-step search when paired with a classifier that yields consistent class probability predictions for single change point scenarios in Section 3.6.

Proposition 4 *For the Bayes classifier p^∞ and any initial guess $u < s^{(0)} < v$, the expected approximate gain*

$$s \mapsto \mathbb{E}\left[\sum_{i=u+1}^{s} \log\left(\frac{v-u}{s^{(0)}-u} p^\infty_{\{u,s^{(0)},v\}}(X_i)_1\right) + \sum_{i=s+1}^{v} \log\left(\frac{v-u}{v-s^{(0)}} p^\infty_{\{u,s^{(0)},v\}}(X_i)_2\right)\right]$$

is piecewise linear between the underlying change points α^0. If there is a single change point $a^0 \in (u, v]$, the expected approximate gain has a unique maximum at $s = a^0$.

Figure 2 shows approximate gain curves and the underlying class probability predictions for a data set simulated with changing covariances, as described in Section 4.2. The approximate gain curves have a piecewise linear shape with kinks at the underlying change points, as predicted by Proposition 4. One can also observe how the distributions of the probability predictions change at the true change points. For the second row, the choice of the initial guess $s^{(0)} = 300$ is precisely such that $\mathbb{P}_{(0,300]} = \mathbb{P}_{(300,600]}$, and the approximate gain curve is flat. Figure 4 shows approximate gain curves in different stages of binary segmentation for a different simulated time series. Again, the approximate gain curves are roughly piecewise linear with kinks at the underlying change points.

3.3 Choice of classifier

We recommend pairing our classifier-based method with random forests (Breiman, 2001). We motivate this choice.

A classifier should satisfy three properties when paired with our change point detection methodology to enjoy good estimation performance and computational feasibility: (i) It needs to have a low dependence on hyperparameter tuning, (ii) it needs to be able to generate unbiased class probability predictions, and (iii) the training cost should be reasonable even for a large number of samples. (i) The hyperparameters optimal for classification can vary between segments encountered in binary segmentation. Retuning them, as is done by Londschien et al. (2021), is infeasible. Thus, the classifier needs to provide suitable class probability predictions even for suboptimal hyperparameters. (ii) If the class probability predictions have a strong bias due to overfitting, the approximate gain curve will have a kink at the initial guess, see Figure 3. For a weak underlying signal, the initial guess might even be the maximizer of the curve, causing the two-step algorithm to fail in finding a true change point. (iii) As for computational cost: Our optimization algorithm based on binary segmentation and the two-step search massively reduces the number of classifier fits required for change point estimation. Still, hundreds of fits might be necessary. For this not to become restrictive, the classifier's training cost should not explode when the number of samples gets large.

Random forests are a perfect fit: They are known to be an off-the-shelf classifier that fits most data without much hyperparameter tuning, they can generate unbiased probability estimates through out-of-bag prediction, and their training complexity is nearly linear in the sample size for a fixed maximal depth.

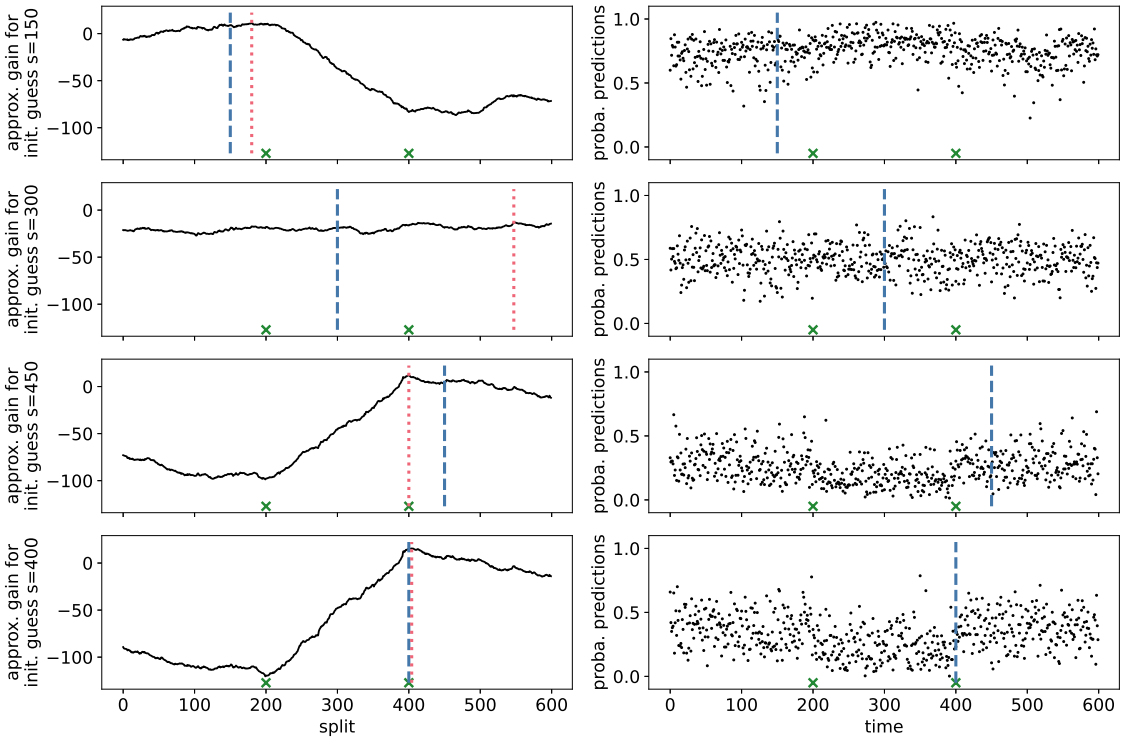

Figure 2: Approximate gain curves (left) from single random forest fits for initial guesses $s^{(0)} = 150$ (top), 300 (second row), and 450 (third row) and the underlying out-of-bag class probability predictions (right). The last row shows the approximate gain curve for the second guess $\hat{s}^{(1)} = 400$. The underlying data is the same as in Figure 1, with true change points at $t = 200, 400$ (green crosses). The initial guess and the curve's maximizer are marked by blue and red vertical dashed lines, respectively.

We highlight another attractive attribute of random forests: They are applicable to a wide range of types of data. The need for nonparametric change point detection arises primarily when little is known about the data to be analyzed. Else, a smartly chosen parametric method might be better suited. In such situations, classifiers with broad applicability are optimal. As a tree-based method, random forests are invariant to feature rescaling and robust to the inclusion of irrelevant features. Due to these properties, random forests are popular as black box or off-the-shelf learning algorithms and are thus well suited for many data sets encountered in nonparametric change point detection.

While random forests are our classifier of choice, our method, including the two-step search, can be paired with any classification algorithm that can produce unbiased class probability predictions. As an illustration, we pair our approach with the Euclidean norm k-nearest neighbor classifier. It has a single hyperparameter k, which we set to the square root of the segment length. Furthermore, the algorithm can be adapted to recover leave-one-out cross-validated probability estimates. Our implementation has a computational complexity of $\mathcal{O}(n^2)$ to compute all pairwise distances once, which we can recycle for all

segments. More efficient implementations exist (Arya et al., 1998). However, the quadratic runtime and memory requirements are not an issue for the reasonably sized data sets we use for benchmarking.

Other choices of classifiers, such as neural networks, would also be possible when paired with a cross-validation procedure to obtain unbiased class probability predictions. However, we do not consider them here due to their high computational cost.

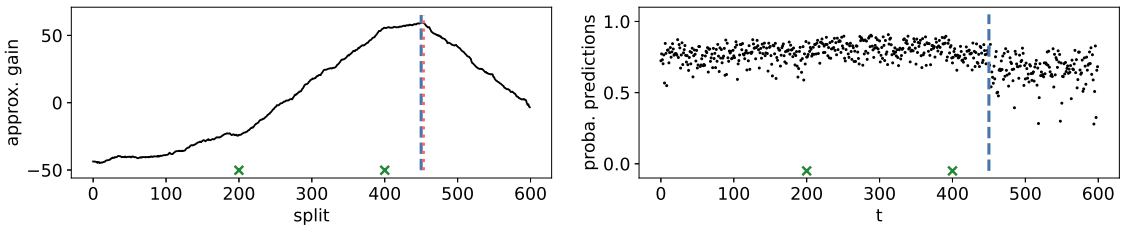

Figure 3: In-sample (not out-of-bag) class probability predictions of a random forest (right) and the corresponding approximate gain curve (left). Predictions are from a single random forest fit for the initial guess $s^{(0)} = 450$ (blue line). The underlying data is the same as in Figures 1 and 2, with true change points at $t = 200, 400$ (green crosses). The maximizer is marked in red. The predictions show a clear bias, dominating the underlying signal (kink around the true change point at $t = 400$), and the approximate gain curve has a peak at the initial guess. Compare this to the third row of Figure 2, where out-of-bag class probability predictions were used, resulting in a maximum around a true change point at $t = 400$.

3.4 Model selection

So far, we have discussed how to accurately and efficiently estimate the location of change points in a signal. We now discuss how to decide whether to keep or discard a candidate split.

Separating true change points from false ones is crucial for good estimation performance and is a task as complex as finding candidate splits. Two approaches for model selection are particularly popular in change point detection: thresholding and permutation tests. For thresholding, a candidate split is kept if its inclusion results in an increase of the log-likelihood at least as large as the threshold γ. Thresholding is often applied in parametric settings, where BIC-type criteria for the minimal gain to split can be derived with asymptotic theory, see for example Yao (1988). Permutation tests are an alternative to thresholding when such asymptotic theory is unavailable. Here, a statistic gathered for a candidate split is compared to values of the statistic gathered after the underlying data was permuted, see for example Matteson and James (2014).

No asymptotic theory is available to suggest a threshold for the (approximate) nonparametric classifier log-likelihood ratio presented in Section 2. Meanwhile, permutation tests require a high number of classifier fits in each splitting step, severely affecting the overall computational cost. We apply a leaner pseudo-permutation test. Instead of permuting the observations and refitting the classifier, we only permute the classifier's predictions gathered in the first step of the two-step search, and thus the classifier log-likelihood ratios. We then

compute the approximate gains after shuffling and take their maxima. In the absence of change points, there is no underlying signal for the classifier to learn other than the relative class sizes. We thus expect the distribution of the classifier log-likelihood ratios to be approximately invariant under permutations. This allows us to compare the maximal gain obtained in the first step of the two-step search to the maximal gains computed after permuting. Algorithm 3 summarizes the procedure.

The significance threshold for the pseudo-permutation test is a tuning parameter rather than a valid significance level. We choose 0.02 by default. This will typically lead to false positives due to multiple testing, as the permutation test is applied to each segment encountered in binary segmentation. We use 199 permutations. We analyze the empirical false positive rate of the pseudo-permutation test in Section 4.7.

3.5 Our proposal

We present `changeforest` in Algorithm 1.

We comment on two details: (i) We use $\log_\eta(x)$ instead of the natural logarithm and (ii) rescale probabilities with $1 \,/\, \pi_i^{\{u,s,v\}} = \frac{v-u-1}{s-u-\mathbb{1}_{\{i \leqslant s\}}}$ instead of $\frac{v-u}{s-u}$. (i) We introduce $\log_\eta(x) \coloneqq \log((1-\eta)x + \eta)$ to use instead of $\log(x)$ in the definition of the classifier log-likelihood ratio. Class probability predictions can take values 0 and 1, prohibiting us from using the logarithm directly to compute individual classifier log-likelihood ratios. With $\eta = \exp(-6)$, we effectively cap individual classifier log-likelihood ratios from below by -6. (ii) Concerning the rescaling: a random forest's out-of-bag probability predictions behave like those from leave-one-out cross-validation. The prediction for observation x_i was generated by a forest that was trained on the $v-u-1$ observations $(x_j)_{j \in \{u+1,\dots,v\} \setminus \{i\}}$. If $i \leqslant s$, then $s-u-1$ of these observations belong to the first class, and if $i > s$, $s-u$ do. Thus, in the absence of any change points, we expect $\hat{p}_{\{u,s,v\}}(x_i) \approx \frac{s-u-\mathbb{1}_{\{i \leqslant s\}}}{v-u-1} = \pi_i^{\{u,s,v\}}$ and thus scale predictions by the inverse of $\pi_i^{\{u,s,v\}}$ before taking the logarithm to recover the classifier log-likelihood ratio estimates.

Our algorithm is visualized in Figure 4, where we display the approximate gain curves of the two-step search for segments encountered in binary segmentation.

Algorithm 1 `changeforest`

Input: Observations $(x_i)_{i=1}^n$, a classifier $\hat{p}$, and a minimum relative segment length $\delta > 0$.
Output: Change point estimate $\hat{\alpha} \leftarrow \text{BinarySegmentation}((x_i)_{i=1}^n, \hat{p}, \delta)$.

function BinarySegmentation$((x_i)_{i=u+1}^v, \hat{p}, \delta)$
  if $v - u < 2\delta n$ **then**
    return $\varnothing$
  end if

  $\hat{s}, ((\ell_{i,k,j})_{i=u+1,\dots,v}^{k=1,2})_{j=1,2,3} \leftarrow \text{TwoStepSearch}((x_i)_{i=u+1}^v, \hat{p}, \delta)$ ▷ c.f. Algorithm 2
  $q \leftarrow \text{ModelSelection}(((\ell_{i,k,j})_{i=u+1,\dots,v}^{k=1,2})_{j=1,2,3}, \delta)$ ▷ c.f. Algorithm 3

  if $q \leqslant 0.02$ **then**
    $\hat{\alpha}_{\text{left}} \leftarrow \text{BinarySegmentation}((x_i)_{i=u+1}^{\hat{s}}, \hat{p}, \delta)$
    $\hat{\alpha}_{\text{right}} \leftarrow \text{BinarySegmentation}((x_i)_{i=\hat{s}+1}^v, \hat{p}, \delta)$
    return $\hat{\alpha}_{\text{left}} \cup \{\hat{s}\} \cup \hat{\alpha}_{\text{right}}$
  else
    return $\varnothing$
  end if

end function

Algorithm 2 TwoStepSearch

Input: Observations $(x_i)_{i=u+1}^{v}$, a classifier $\hat{p}$, and a minimum relative segment length $\delta > 0$.
Output: A single change point estimate $\hat{s}$ and classifier log-likelihood ratios $((\ell_{i,k,j})_{i=u+1,\dots,v}^{k=1,2})_{j=1,2,3}$ for model selection.

Step 1: Define $(s_1^{(0)}, s_2^{(0)}, s_3^{(0)}) = (\lfloor \frac{3u+v}{4} \rfloor, \lfloor \frac{v+u}{2} \rfloor, \lfloor \frac{u+3v}{4} \rfloor)$ and $\log_\eta(x) = \log((1-\eta)x + \eta)$ for $\eta = \exp(-6)$ and set $\pi_i^{\{u,s,v\}} \leftarrow \frac{s-u-\mathbb{1}_{\{i \leqslant s\}}}{v-u-1}$. For $j = 1, 2, 3$, train binary classifiers $\hat{p}_{\{u,s_j^{(0)},v\}}$ on $(\mathbf{X}, \mathbf{y}_{\{u,s_j^{(0)},v\}}) = ((x_i)_{i=u+1}^{v}, (1 + \mathbb{1}_{\{i > s_j^{(0)}\}})_{i=u+1}^{v})$ and let

$$\ell_{i,1,j} = \log_\eta \left(\hat{p}_{\{u,s_j^{(0)},v\}}(x_i)_1 / \pi_i^{\{u,s_j^{(0)},v\}} \right), \ \ell_{i,2,j} = \log_\eta \left(\hat{p}_{\{u,s_j^{(0)},v\}}(x_i)_2 / (1 - \pi_i^{\{u,s_j^{(0)},v\}}) \right).$$

Select

$$\hat{s}^{(1)} \in \operatorname*{argmax}_{s=u+1+\lceil \delta n \rceil, \dots, v-\lceil \delta n \rceil} \max_{j=1,2,3} \sum_{i=u+1}^{s} \ell_{i,1,j} + \sum_{i=s+1}^{v} \ell_{i,2,j}.$$

Step 2: Train a binary classifier $\hat{p}_{\{u,\hat{s}^{(1)},v\}}$ and let

$$\ell_{i,1} = \log_\eta \left(\hat{p}_{\{u,s^{(1)},v\}}(x_i)_1 / \pi_i^{\{u,s^{(1)},v\}} \right), \ \ell_{i,2} = \log_\eta \left(\hat{p}_{\{u,s^{(1)},v\}}(x_i)_2 / (1 - \pi_i^{\{u,s^{(1)},v\}}) \right).$$

Select

$$\hat{s} = \hat{s}^{(2)} \in \operatorname*{argmax}_{s=u+1+\lceil \delta n \rceil, \dots, v-\lceil \delta n \rceil} \sum_{i=u+1}^{s} \ell_{i,1} + \sum_{i=s+1}^{v} \ell_{i,2}.$$

Return: $\left(\hat{s}, ((\ell_{i,k,j})_{i=u+1,\dots,v}^{k=1,2})_{j=1,2,3} \right)$

Algorithm 3 ModelSelection

Input: Classifier log-likelihood ratios $((\ell_{i,k,j})_{i=u+1,\dots,v}^{k=1,2})_{j=1,2,3}$ from the two-step search and a minimum relative segment length $\delta > 0$.
Output: An approximate p-value from the pseudo-permutation test.

Step 1: Recover the maximal gain encountered in the first step of the two-step search.

$$G_0 = \max_{s=u+\lceil \delta n \rceil+1, \dots, v-\lceil \delta n \rceil} \max_{j=1,2,3} \sum_{i=u+1}^{s} l_{i,1,j} + \sum_{i=s+1}^{v} l_{i,2,j}.$$

Step 2: For random permutations $\sigma_l : \{u+1, \dots, v\} \to \{u+1, \dots, v\}$ for $l = 1 \dots, 199$, compute maximal gains after permuting likelihoods

$$G_l = \max_{j=1,2,3} \max_{s=u+\lceil \delta n \rceil+1, \dots, v-\lceil \delta n \rceil} \sum_{i=u+1}^{s} l_{\sigma_l(i),1,j} + \sum_{i=s+1}^{v} l_{\sigma_l(i),2,j}.$$

Return: $\#\{l = 0, \dots, 199 \mid G_l \geqslant G_0\} / 200$

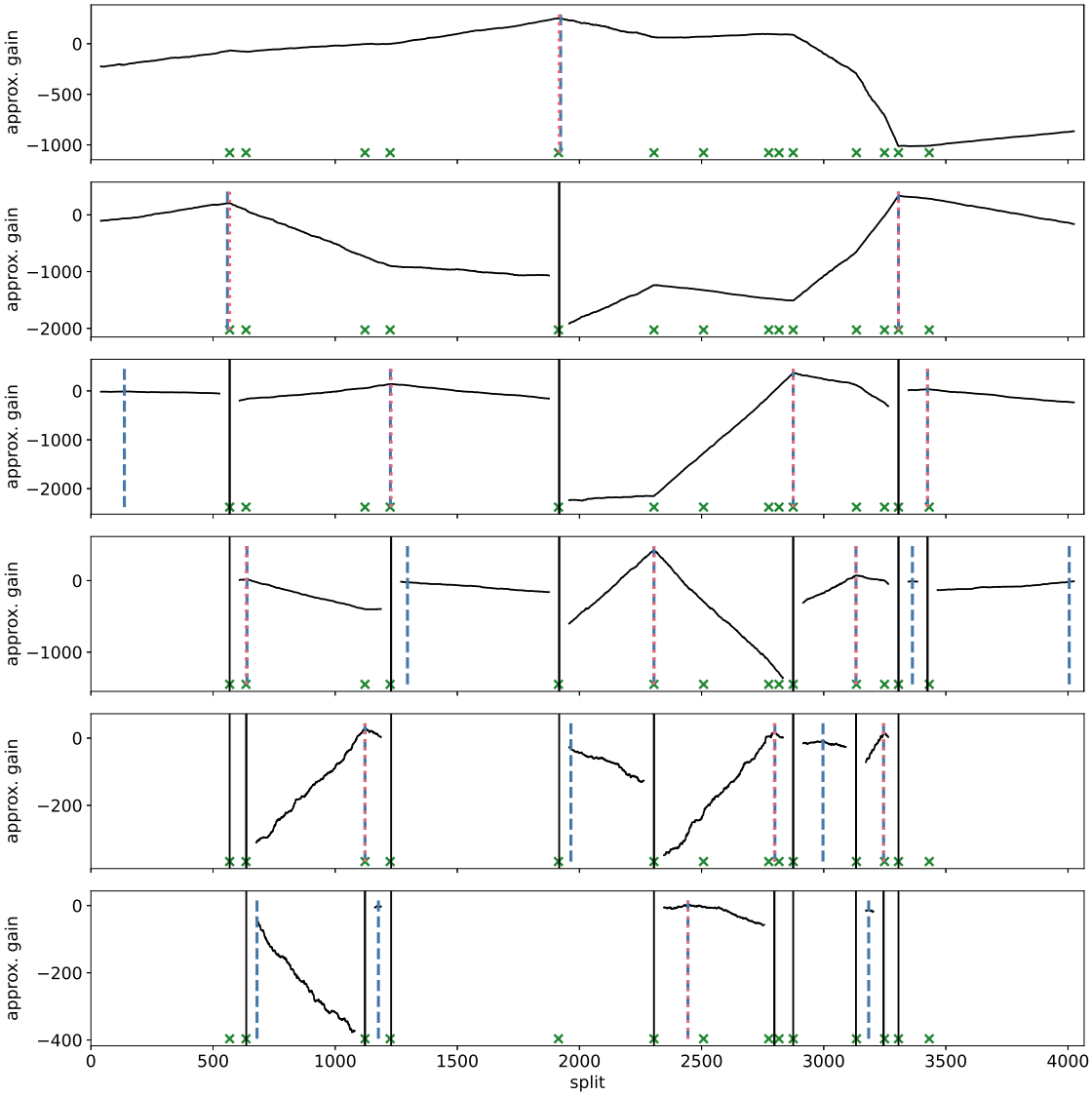

Figure 4: Approximate gain curves from the last step of the two-step search in the different steps of binary segmentation as encountered in `changeforest`. The underlying data set was simulated with the abalone setup, see Section 4.2 for details. The true change points at $t = 568, 635, 1122, 1225, 1914, 2305, 2508, 2775, 2817, 2875, 3134, 3249, 3306, 3432$ are marked with green crosses. Change point estimates, corresponding to maxima of the approximate gain curves that are accepted in our model selection procedure, are marked in red. The second guess $\hat{s}^{(1)}$ for the two-step search is marked in blue. Current segment boundaries are marked with black vertical lines. Here, `changeforest` estimates change points at $t = 568, 636, 1122, 1229, 1917, 2305, 2444, 2798, 2875, 3132, 3245, 3306, 3425$, corresponding to an adjusted Rand index of 0.97.

3.6 Theoretical guarantees

We provide the following finite sample guarantee for the two-step search Algorithm 2 in settings with a single change point.

Theorem 5 *Let $0 < \delta, \eta < 1$ and $\delta \leqslant \tau^0, \rho \leqslant 1 - \delta$. Set $s = \rho n$. Assume a single change point scenario with n observations and a change point at $a^0 = \tau^0 n$. Let $\hat{p}$ satisfy $\max_{i=1,\dots,n} |\hat{p}_s(X_i)_1 - p_s^\infty(X_i)_1| \overset{n\to\infty}{\to} 0$ in probability. Choose*

$$\hat{a} \in \operatorname*{argmax}_{t=1,\dots,n} \sum_{i=1}^{t} \log_\eta \left(\frac{n}{s} \hat{p}_s(X_i)_1 \right) + \sum_{i=t+1}^{n} \log_\eta \left(\frac{n}{n-s} \hat{p}_s(X_i)_2 \right)$$

as in Algorithm 2. Then

$$\mathbb{P}\left[\left| \frac{\hat{a}}{n} - \tau^0 \right| > \frac{C \log(n)}{\delta^2 \mathrm{d}_{\mathrm{TV}}(\mathbb{P}_1, \mathbb{P}_2)^4 n} \right] \overset{n\to\infty}{\longrightarrow} 0,$$

where $\mathrm{d}_{\mathrm{TV}}(\mathbb{P}_1, \mathbb{P}_2)$ is the total variation distance between the two distributions $\mathbb{P}_1$ and $\mathbb{P}_2$ before and after the change point and $C = \frac{1}{(1-\eta)^4} \log\left(\frac{1}{(1-\rho)\rho} \frac{1}{\eta^2} \right)^2$.

In particular, for our choices of $\rho \in \{\frac{1}{4}, \frac{1}{2}, \frac{3}{4}\}$ and $\eta = \exp(-6)$, the estimator $\frac{\hat{a}}{n}$ is consistent at the rate $\frac{\log(n)}{n}$ for τ^0. Interestingly, this convergence rate is independent of the consistency rate of the classifier. Generalizing Theorem 5 to the case of multiple change points would require further assumptions, preventing the possible "canceling-out" effects as seen in Figure 2 (second row). Theorem 5 is based on a uniform consistency assumption for the classifier. Such uniformly consistent classifiers exist (Bierens, 1983), including k-nearest neighbor classifiers (Kudraszow and Vieu, 2013). However, we are not aware of any such results for random forests.

3.7 The `changeforest` software package

A substantial part of this work is the `changeforest` software package, implementing Algorithm 1 for random forests and k-nearest neighbor classifiers. It also implements the parametric change in mean setup, paired with binary segmentation, which we use as a baseline in the simulations.

The `changeforest` package is available for Python users on PyPI and conda-forge (conda-forge community, 2015), `R` users on conda-forge, and Rust users on crates.io. Its backend is implemented in the system programming language Rust (Matsakis and Klock, 2014). For inquiries, installation instructions, a tutorial, and more information, please visit `github.com/mlondschien/changeforest`.

4. Empirical results

We present results from a simulation study comparing the empirical performance of our classifier-based method `changeforest` to available multivariate nonparametric competitors. The source code for the simulations, together with guidance on reproducing tables and figures, is available at `github.com/mlondschien/changeforest-simulations`. The repository can easily be expanded to benchmark new methods provided by users.

4.1 Competing methods

We are aware of the following nonparametric methods for multivariate multiple change point detection with existing reasonably efficient implementations: Matteson and James' (2014) *e-divisive* (ECP), Lung-Yut-Fong et al.'s (2015) *MultiRank*, and kernel change point methods such as Arlot et al.'s (2019) *KCP* and Madrid–Padilla et al.'s (2021b) *MNWBS*.

ECP searches for a single change point minimizing energy distances within segments. The method generalizes to multiple change point scenarios through binary segmentation, and the significance of change points is assessed with a permutation test. An implementation of ECP is available through the `R`-package `ecp` (James and Matteson, 2015), which we use in the simulations.

MultiRank uses a rank-based multivariate homogeneity statistic combined with dynamic programming. The significance of change points is assessed based on asymptotic theory. To run simulations with MultiRank, we used code made available to us by the authors. This is included in the simulations repository.

Kernel change point methods minimize a within-segment average kernelized dissimilarity measure. Arlot et al. (2019) propose KCP, an algorithm based on dynamic programming. An efficient implementation can be found in the Python package `ruptures` (Truong et al., 2020). We used the slope heuristic proposed by Arlot et al. (2019) for model selection.

MNWBS uses a kernel-density-based nonparametric extension of the CUSUM statistic for single change point localization. The method extends to multiple change point scenarios through wild binary segmentation (Fryzlewicz, 2014). Model selection is done by thresholding of the test statistic, where the threshold is chosen based on the Kolmogorov–Smirnov test statistic.

For all packages, we used default parameters, if available. For ECP, this results in $\alpha = 1$ and using 199 permutations at a significance level of 0.05. For KCP, we use a Gaussian kernel with a bandwidth of 0.1 after normalization by the median absolute deviation of absolute consecutive differences, see Section 4.2. This choice of bandwidth was optimal for the simulated scenarios, see also Section 4.6 and Table 9. For MNWBS, we used the bandwidth $5 \cdot (n \log(n)/\delta)^{1/p}$ as proposed by Madrid–Padilla et al. (2021b). We used 50 random intervals to reduce the computational cost. We supplied information about the minimum relative segment length δ, equal to 0.01 in the main simulations, to each method, either by informing about the minimum number of observations per segment $\lceil \delta n \rceil$ or by informing about the maximum number of change points $\lfloor 1/\delta \rfloor$.

Other multivariate nonparametric multiple change point detection methods that we did not include in our simulations include Liu et al.'s (2013) *RuLSIF*, Cabrieto et al.'s (2017) *Decon* with a kernel-based running statistic, and Zhang and Chen's (2021) *gMulti*. For gMulti, no implementation has been made available by the time of writing. The existing implementation for Decon in the R package `kcpRS` does not implement the kernel-based nonparametric running statistic. The implementation for RuLSIF is only available in Matlab and does not include an automatic model selection procedure.

4.2 Simulation setups

We evaluate all methods on three parametric and six nonparametric scenarios. We display their key characteristics below the header in Table 1.

As for parametric setups, we use the *change in mean* (CIM) and *change in covariance* (CIC) setups from Matteson and James (2014, Section 4.3) and the *Dirichlet* setup from Arlot et al. (2019, Section 6.1). For both works, the authors present empirical results of their methods on data sets for three sets of parameters. We use the set of parameters corresponding to medium difficulty.

For the change in mean and change in covariance setups, we generate time series of dimension $d = 5$ with $n = 600$ observations and change points at $t = 200, 400$. Observations in the first and last segment are independently drawn from a standard multivariate Gaussian distribution. In the change in mean setup, entries in the second segment are i.i.d. Gaussian with a mean shift of $\mu = 2$ in each coordinate. In the change in covariance setup, entries in the second segment are i.i.d. Gaussian with mean zero and unit variance, but with a covariance of $\rho = 0.7$ between coordinates. Lastly, the Dirichlet data set consists of $n = 1000$ observations and has dimension $d = 20$. Ten change points are located at $t = 100, 130, 220, 320, 370, 520, 620, 740, 790, 870$. In each segment, the observations are distributed according to the Dirichlet distribution, with the d parameters drawn i.i.d. uniformly from $[0, 0.2]$.

We also generate time series with nonparametric in-segment distributions. For this, we use popular multiclass classification data sets and treat observations corresponding to a single class as i.i.d. draws from a class-specific distribution. We discard classes with fewer than $\delta n = \frac{n}{100}$ observations and shuffle and randomly concatenate the remaining classes. Categorical variables are dummy encoded. Finally, each covariate gets normalized by the median absolute deviation of absolute consecutive differences, as is also used by Fryzlewicz (2014) and further discussed by Kovács et al. (2020a). This normalization does not affect our random forest-based method `changeforest` and rank-based MultiRank but increases performance for distance-based competitors ECP, KCP, and MNWBS. We did not normalize covariates in the parametric simulations setup, as this would decrease performance for ECP, KCP, and MNWBS.

We use the following data sets: The *iris* flower data set (Anderson, 1936) contains 150 samples of four measurements from three species of the iris flower. The *glass* identification data set (Evett and Spiehler, 1989), initially motivated by criminological investigation, contains measurements of chemical compounds for different types of glass. The *wine* data set (Cortez et al., 2009) is the result of a physicochemical analysis of both red and white vinho verde from north-western Portugal. It includes quality scores from blind testing, which we use as class labels. We encode the wine color as binary. The Wisconsin *breast cancer* data set (Street et al., 1993) contains nominal characteristics of cell nuclei computed from a fine needle aspirate of breast tissue labeled as malignant or benign. The resulting simulated time series has a single change point. The *abalone* data set (Waugh, 1995) contains easily obtainable physical measurements of snails and their age, determined by cutting through the shell cone and counting the number of rings. Two categorical variables are dummy encoded. We use the number of rings as class labels. The *dry beans* data set (Koklu and Ozkan, 2020) contains shape measurements derived from images of seven different types of dry beans.

The change in mean and change in covariance and possibly also the abalone and wine data sets feature similar or equal distributions in non-neighboring segments. We expect these scenarios to be challenging for our methodology, see Section 3.2 and Figure 2.

4.3 Performance measures

We mainly use the adjusted Rand index (Hubert and Arabie, 1985), a standard measure to compare clusterings, to evaluate the goodness of fit of estimated segmentations. Given two partitionings of n observations, the Rand index (Rand, 1971) is the number of agreements, pairs of observations either in the same or in different subsets for both partitionings, divided by the total number of pairs $\binom{n}{2}$. The adjusted Rand index is the normalized difference between the Rand index and its expectation when choosing partitions randomly. Consequently, the adjusted Rand index can take negative values but no values greater than one. Its expected value is zero for random class assignments, but the expected value for random segmentations, namely class assignments consistent with the time structure, is significantly higher. We present a set of segmentations together with their adjusted Rand indices in Table 4 in the Appendix.

We believe that an adjusted Rand index above 0.95 corresponds to a segmentation that is close to perfect. An adjusted Rand index of 0.9 corresponds to a good segmentation, possibly oversegmenting with a false positive close to a true change point or missing one of two close change points. On the other hand, we expect a segmentation with an adjusted Rand index significantly below 0.8 not to provide much scientific value.

We additionally use the Hausdorff distance to evaluate change point estimates. For two segmentations α, α', set $d(\alpha, \alpha') := \frac{1}{n} \max_{a \in \alpha} \min_{a' \in \alpha'} |a - a'|$. Then $d(\alpha^0, \hat{\alpha})$ is the largest relative distance of a true change point in α^0 to its closest counterpart in $\hat{\alpha}$, and $d(\hat{\alpha}, \alpha^0)$ is the largest relative distance of a found change point in $\hat{\alpha}$ to the closest entry in α^0. Thus $d(\alpha^0, \hat{\alpha})$ especially penalizes undersegmentation while $d(\hat{\alpha}, \alpha)$ penalizes oversegmentation. Define the Hausdorff distance as $d_H(\alpha, \alpha') := \max(d(\alpha, \alpha'), d(\alpha', \alpha))$. Table 4 also includes the Hausdorff distance for the selected segmentations.

4.4 Main simulation results

We present the results of our main simulation study in Table 1, where we display average adjusted Rand indices over 500 simulations.

Our method, `changeforest`, scores above 0.9 on average for each simulation scenario studied. All other methods have an average score lower than 0.75 for at least one scenario. `changeforest` is the best-performing method for four setups: Dirichlet, glass, abalone, and wine. Furthermore, in four further setups, change in mean, iris, breast cancer, and dry beans, our method `changeforest` scores above 0.975 on average, corresponding to an almost perfect segmentation. The only setup for which `changeforest` does not perform best or above 0.975 is change in covariance. Here, `changeforest` scores second-best behind KCP and the only other method to score above 0.5 is MNWBS. We compute the average score for each method to combine the results for the nine simulation scenarios. `changeforest` scores highest. Overall, `changeforest` is widely applicable, with robust performance on all parametric and nonparametric setups.

We include Table 5 with median Hausdorff distances in the Appendix. The results are similar, with `changeforest` being among the methods with the best score, except for change in covariance, where it achieves a median Hausdorff distance of 0.013, behind KCP with a median Hausdorff distance of 0.007.

Table 1: Average adjusted Rand indices over 500 simulations, with standard deviations in parentheses. Optimal scores are marked in bold. The MultiRank method was unable to produce results for the dry beans simulation setup. For the abalone, wine, and dry beans simulation setups, the computational cost of MNWBS was prohibitive. Average and worst denote the average and worst performances across all setups.

	CIM	CIC	Dirichlet	iris	glass
n, d, K	$600, 5, 3$	$600, 5, 3$	$1000, 20, 11$	$150, 4, 3$	$214, 8, 6$
change in mean	1.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.93 (0.08)	0.49 (0.17)
changekNN	0.99 (0.02)	0.01 (0.08)	0.69 (0.25)	0.99 (0.03)	0.88 (0.09)
ECP	0.99 (0.03)	0.40 (0.32)	0.84 (0.08)	0.99 (0.03)	0.61 (0.24)
KCP	1.00 (0.01)	**0.95** (0.06)	0.87 (0.08)	**0.99** (0.02)	0.74 (0.23)
MultiRank	**1.00** (0.00)	0.00 (0.04)	0.97 (0.02)	0.57 (0.26)	0.60 (0.09)
MNWBS	0.99 (0.06)	0.69 (0.18)	0.67 (0.10)	0.99 (0.04)	0.71 (0.18)
`changeforest`	0.99 (0.03)	0.93 (0.12)	**0.99** (0.02)	0.98 (0.04)	**0.92** (0.07)

	breast cancer	abalone	wine	dry beans	average	worst
n, d, K	$699, 9, 2$	$4066, 9, 15$	$6462, 12, 5$	$13611, 16, 7$		
change in mean	0.79 (0.03)	0.88 (0.06)	0.98 (0.02)	0.99 (0.01)	0.68 (0.07)	0.00
changekNN	0.91 (0.17)	0.88 (0.10)	0.97 (0.04)	1.00 (0.01)	0.81 (0.11)	0.01
ECP	1.00 (0.02)	0.87 (0.06)	0.98 (0.03)	**1.00** (0.00)	0.86 (0.14)	0.43
KCP	**1.00** (0.03)	0.92 (0.04)	0.99 (0.01)	**1.00** (0.00)	0.94 (0.09)	0.74
MultiRank	0.92 (0.12)	0.81 (0.12)	0.96 (0.03)		0.73 (0.11)	0.00
MNWBS	0.99 (0.05)				0.84 (0.12)	0.67
`changeforest`	0.98 (0.07)	**0.93** (0.05)	**0.99** (0.02)	1.00 (0.01)	**0.97** (0.06)	**0.92**

Table 6 in the Appendix shows the average number of estimated change points for each method and simulation setup. `changeforest` slightly oversegments for most setups, but not for the glass and wine setups.

Figure 7 in the Appendix shows histograms of the nonparametric methods' change point estimates for the Dirichlet setup, where the underlying change points are constant across simulations. Corresponding to high adjusted Rand indices of 0.99 and 0.97, change point estimates by `changeforest` and MultiRank are sharply concentrated around the true change point locations. Meanwhile, estimates by changekNN, ECP, KCP, and MNWBS are visually more spread out, corresponding to worse adjusted Rand indices smaller than 0.9.

We present average computation times for each method in Table 2. Simulations were run on eight Intel Xeon 2.3 GHz cores with 4 GB of RAM available per core (32 GB in total). The simple change in mean method is blazingly fast. On the largest simulation setup dry beans with $n = 13611$ and $d = 16$, the average computation time was around 0.002 seconds. `changeforest` is the fastest nonparametric method on the dry beans simulation setup, requiring around 2.6 seconds on average. On the other hand, MNWBS and ECP are prohibitively slow. MNWBS takes around four hours on average to estimate change points on the Dirichlet simulation setup. It did not finish in a reasonable time on the larger

dry beans, wine, and abalone simulation setups. ECP is faster but still requires more than one hour to estimate change points on the dry beans simulation setup on average. The code for MultiRank is experimental and broke for the dry beans simulation setup due to the multiple dummy encoded columns. We further analyze the computational efficiency of available implementations in Section 4.5.

Table 2: Average computation times in seconds on 8 Intel Xeon 2.3 GHz cores.

	CIM	CIC	Dirichlet	iris	glass	breast cancer	abalone	wine	dry beans
n, d	600, 5	600, 5	1000, 20	150, 4	214, 8	699, 9	4066, 9	6462, 12	13611, 16
change in mean	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01
changekNN	0.04	0.04	0.15	<0.01	0.01	0.05	1.8	4.7	24
ECP	4.2	4.5	18	0.37	0.72	4.3	386	741	5356
KCP	0.08	0.09	0.18	0.02	0.04	0.10	2.5	6.6	31
MultiRank	0.77	0.91	2.1	0.10	0.17	1.0	34	90	
MNWBS	2132	2143	14261	115	207	3211			
`changeforest`	0.08	0.10	0.43	0.03	0.06	0.07	0.86	0.89	2.6

4.5 Performance for varying numbers of segments and samples

The performance of change point estimation methods depends on an interplay between the number of observations, the size of distributional shifts between segments, and the number of change points. In Table 1 we see that for some data sets, such as CIM, iris, and dry beans, all methods achieve average adjusted Rand indices above 0.975. Here, we believe that the signal-to-noise ratio is so high that change point estimation is easy, and the difference between the average adjusted Rand index and the perfect score of 1.0 is primarily a result of random false positives. On the other hand, the change in covariance and glass setups appear truly difficult, clearly separating methods that perform well (above 0.9) and those performing not much better than random guessing (below 0.8).

We introduce two simulation settings that let us generate time series of any length n with any number of change points K, allowing us to control the underlying signal-to-noise ratio. We draw $\tilde{N}_k \sim \mathrm{Exp}(1)$ i.i.d. for $k = 1, \ldots, K$ and define $N_i = \frac{1}{10K} + \frac{0.9}{\sum_{j=1}^{K} \tilde{N}_j} \tilde{N}_k$. We use $\mathrm{round}(nN_k)$ as segment lengths, where we round such that $\sum_{k=1}^{K} \mathrm{round}(nN_k) = n$. This results in a minimum relative segment length of $\delta = \frac{1}{10K}$.

We present two simulation setups: One based on the Dirichlet distribution and one based on the dry beans data set. We construct the former as in Section 4.2. For each segment, we draw i.i.d. observations from a Dirichlet distribution of dimension $d = 20$, after sampling its parameters uniformly from $[0, 0.2]$. To generate time series of arbitrary length based on the dry beans data set, we first normalize each covariate to have an average within-class variance of one. Then, for each segment, we draw a class label different to that of the previous segment and draw $\mathrm{round}(nN_k)$ observations with replacement from the corresponding class. We add i.i.d. standard Gaussian noise to each observation. As in Section 4.2, we finally normalize each covariate by the median absolute deviation of absolute consecutive differences. We simulate 500 data sets with $K = 20, 80$ and vary $n = 250, 354, 500, 707, 1000, \ldots, 64000$

for each setup. The mean adjusted Rand indices of change point estimates from our and competing methods are displayed in Figure 5. Each method was supplied with the minimum relative segment length $\delta = \frac{1}{10K}$.

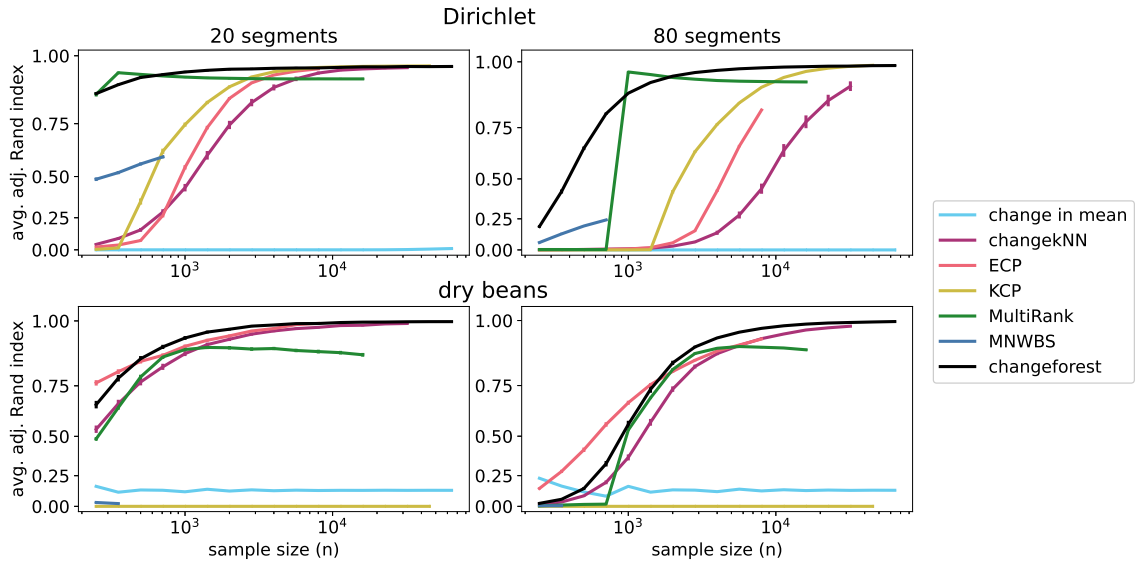

Figure 5: Evolution of the average adjusted Rand index for methods by the number of observations for 20 and 80 segments. The y-axis was scaled to better pronounce values in $[0.6, 1]$. Two times the standard deviation of the mean score is marked with vertical bars. All methods were run until computational time got prohibitive (ECP and MNWBS) or 32 GB of memory was insufficient for computation (KCP, MultiRank, and changekNN).

The change in mean method does not perform well for either setup. Our method `changeforest` performs best, except for very small sample sizes. For the dry beans simulation setup when $n/K < 20$, ECP performs best, but not much better than 0.8. For the Dirichlet simulation setup for $K = 20, n = 354$ and $K = 80, n = 1000$, MultiRank performs best. Surprisingly, KCP performs very badly for the dry beans-based simulation, not selecting any change points at all. This might be due to the higher number of change points compared to the main simulations or the added Gaussian noise.

We display the average computational time of the different methods in Figure 6, together with dashed grey lines corresponding to linear and quadratic time complexity. Again, the parametric change in mean method is very fast. For both change in mean and `changeforest`, time complexity appears to scale approximately linear in the number of observations. Meanwhile, changekNN, MultiRank, KCP, and MNWBS scale approximately quadratic, while ECP appears to have worse than quadratic time complexity. `changeforest` is the quickest nonparametric method for $n \gtrsim 5000$.

4.6 Dependence on hyperparameters

We investigate how the hyperparameters of random forests affect `changeforest`'s performance. For this, we vary the three most important parameters for random forests: the

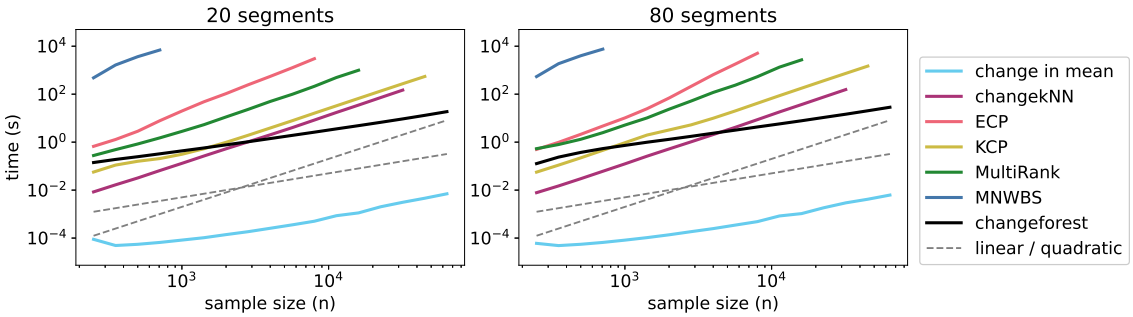

Figure 6: Average time for change point detection on 8 Intel Xeon 2.3 GHz cores with 4 GB of RAM available per core for the Dirichlet simulation setup. All methods were run until computational time got prohibitive (ECP and MNWBS) or 32 GB of memory was insufficient (KCP, MultiRank, and changekNN).

number of trees from 20 to 100 (default) to 500, the maximal depth of individual trees from 2 to 8 (default) to infinity, and the number of features considered at each split (mtry) from 1 to $\sqrt{d}$ (default, rounded down) to d. The average adjusted Rand indices for each set of parameters for the simulation setups introduced in Section 4.2 are in Table 7 in the Appendix. The average computational times are in Table 8. We summarize the results here.

Choosing even very extreme parameters, such as number of trees = 20, maximal depth = 2, and mtry = 1, yields acceptable change point estimation performance for simple simulation setups. With these parameters, `changeforest` performs on average better than ECP, MultiRank, MNWBS, and changekNN, our method paired with a k-nearest neighbor classifier. It even outperforms the default configuration (100, 8, $\sqrt{d}$) for the iris, breast cancer, and wine simulation setups, possibly due to selecting fewer false positives because of higher noise in the predictions. On the other hand, increasing the number of trees from 100 to 500 and splitting individual trees to purity does not significantly increase the performance compared to the default values while increasing computational cost fivefold.

This low dependence on the choice of tuning parameters is in contrast to KCP, where the bandwidth of the Gaussian kernel has to be fine-tuned for optimal performance. Table 9 in the Appendix shows the estimation performance of KCP for different kernels and bandwidths.

4.7 False positive rates

We empirically analyze our model selection procedure. For this, we select the largest class of each data set from Section 4.2, shuffle the observations before applying the change point detection methods, and collect the percentage out of 2500 simulations where at least one change point was detected. See Table 3 for the results.

As our model selection procedure is not a true permutation test, we observe more false positives than expected, given the approximate significance threshold of 0.02 for both `changeforest` and the k-nearest neighbor method changekNN. However, `changeforest` has a similar false positive rate across all simulation setups. The methods based on heuristics

Table 3: Percentage of simulations where at least one change point was detected in a homogeneous data set. For each data set and method, 2500 simulations were performed.

data set	CIM	CIC	Dirichlet	iris	glass	breast cancer	abalone	wine	dry beans	worst
change in mean	0.12	0.12	100.0	27.04	91.44	3.44	7.68	1.20	1.40	100.00
changekNN	2.44	2.44	1.80	3.64	2.40	29.80	3.28	6.80	2.04	29.80
ECP	5.76	6.20	5.08	6.80	5.72	4.76	4.76	5.16	5.16	6.80
KCP	0.00	0.00	0.00	3.04	0.32	5.24	11.32	0.08	0.04	11.32
MultiRank	0.20	0.20	12.08	93.64	90.44	100.0	100.0	100.0		100.00
MNWBS	95.08	95.08	79.36	95.16	77.68	98.20				98.20
`changeforest`	3.36	3.36	3.76	5.00	3.80	3.80	3.00	3.52	2.48	5.00

(KCP and MNWBS) or asymptotic theory (MultiRank, change in mean) have varying degrees of false positive rates by data set. ChangekNN, which uses the same model selection procedure as `changeforest`, has a false positive rate of almost 30% for the breast cancer data set.

5. Future work

We comment on some possible avenues for future work.

Time series featuring serial correlations The setting outlined in Section 2.2 requires independent observations. In practice, time series data often exhibit serial correlation and the change in distribution may only manifest through a change in the structure of this serial correlation. Even though the model assumptions are violated, our algorithm can be applied to such data sets by augmenting the data with lagged observations. It would be interesting to compare the performance of `changeforest` when applied to such augmented time series with the performance of change point detection methods specifically designed to detect changes in the autocorrelation structure.

Extending the theoretical guarantees to multiple change points As discussed in Section 3.6, the two-step search is not guaranteed to find a true change point in a multiple change point scenario due to possible canceling out effects (see also Figure 2). One possible remedy would be to pair the two-step search with wild binary segmentation (Fryzlewicz, 2014) or seeded binary segmentation (Kovács et al., 2023). This would help in scenarios with very short segments and equal or similar distributions in surrounding segments but would require changes to the model selection procedure presented to avoid overfitting. As we already observe very good empirical performance with binary segmentation, we did not pursue this further.

6. Conclusion

Our proposed `changeforest` algorithm uses random forests to efficiently and accurately perform multiple change point detection across a wide range of data set types. Motivated by parametric change point detection methodology, the algorithm optimizes a nonparametric

classifier-based log-likelihood ratio. When coupled with random forests, it performs very well or better than competitors for all simulation setups studied. The computation time of `changeforest` scales almost linearly with the number of observations, enabling nonparametric change point detection even for very large data sets. The `changeforest` software package is available to `R`, Python, and Rust users. See `github.com/mlondschien/changeforest` for installation instructions.

Acknowledgments

Malte Londschien is supported by the ETH Foundations of Data Science and the ETH AI Center. Solt Kovács and Peter Bühlmann are supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 786461 CausalStats - ERC-2017-ADG).

Authors' contributions

S.K. conceived and co-supervised the project. M.L. and S.K. developed the methodology. M.L. designed and implemented the changeforest software package, designed, implemented, and ran the simulations, developed the theory, and wrote the manuscript. P.B. co-supervised the project and provided feedback and ideas during the development. S.K. and P.B. provided feedback on the manuscript.

Appendix A. Proofs

Proposition 2 *Let*

$$p_\alpha^\infty(x) := \mathbb{P}\left(Y = k \mid X = x\right) = \left(\frac{\alpha_k - \alpha_{k\text{-}1}}{n} \frac{p_{(\alpha_{k\text{-}1},\alpha_k]}}{p_{(0,n]}}\right)_{k=1}^{|\alpha|-1}$$

be the infinite sample Bayes classifier corresponding to the segmentation α. If $\mathcal{A}$ is a set of segmentations containing α^0 and $D_{KL}(P \parallel Q)$ is the Kullback-Leibler divergence of Q with respect to P, then

$$\begin{aligned}\max_{\alpha\in\mathcal{A}} \mathbb{E}\left[G\left((X_i)_{i=1}^n \mid \alpha, p^\infty\right)\right] &= \mathbb{E}\left[G\left((X_i)_{i=1}^n \mid \alpha^0, p^\infty\right)\right] \\ &= \sum_{k=1}^K (\alpha_k^0 - \alpha_{k\text{-}1}^0) D_{KL}(\mathbb{P}_k \parallel \mathbb{P}_{(0,n]})\end{aligned}$$

and any maximizer of $\mathbb{E}\left[G\left((X_i)_{i=1}^n \mid \alpha, p^\infty\right)\right]$ is equal to α^0 or is a segmentation containing α^0. In particular, the maximizer α with the smallest cardinality is equal to α^0.

Proof Let $u < v$ and define $\kappa((u,v]) := \{k \in \{1,\dots,K\} \mid (\alpha_{k\text{-}1}^0, \alpha_k^0] \cap (u,v] \neq \varnothing\}$. If there exists some k such that $\alpha_{k\text{-}1}^0 \leqslant u < v \leqslant \alpha_k^0$, then $\kappa((u,v]) = \{k\}$ and $p_{(u,v]} = p_{(\alpha_{k\text{-}1}^0,\alpha_k^0]} = p_k$. Consequently, as the $X_{u+1},\dots,X_{v-1}$ are $\mathbb{P}_k$-distributed, we have

$$\mathbb{E}\left[\sum_{i=u+1}^v \log\left(\frac{p_{(u,v]}(X_i)}{p_{(0,n]}(X_i)}\right)\right] = (v-u) D_{KL}(\mathbb{P}_k \parallel \mathbb{P}_{(0,n]}).$$

This implies that, for any segmentation α containing α^0 and thus $|\kappa((\alpha_{k\text{-}1},\alpha_k])| = 1$ for all $k = 1,\dots,K$,

$$\mathbb{E}\left[G\left((X_i)_{i=1}^n \mid \alpha, p^\infty\right)\right] = \sum_{k=1}^K (\alpha_k^0 - \alpha_{k\text{-}1}^0) D_{KL}(\mathbb{P}_k \parallel \mathbb{P}_{(0,n]}).$$

Now assume that $\kappa((u,v])$ contains more than a single element. Set $\pi_k((u,v]) := \frac{|(\alpha_{k\text{-}1}^0,\alpha_k^0] \cap (u,v]|}{v-u}$. Then $\mathbb{P}_{(u,v]} = \sum_{k\in\kappa((u,v])} \pi_k((u,v])\mathbb{P}_k$ and

$$\begin{aligned}\mathbb{E}\left[\sum_{i=u+1}^v \log\left(\frac{p_{(u,v]}(X_i)}{p_{(0,n]}(X_i)}\right)\right] &= (v-u)\mathbb{E}_{X\sim\mathbb{P}_{(u,v]}}\left[\log\left(\frac{p_{(u,v]}(X)}{p_{(0,n]}(X)}\right)\right] \\ &= (v-u)\mathbb{E}_{X\sim\mathbb{P}_{(u,v]}}\left[\log\left(\frac{\sum_{k\in\kappa((u,v])} \pi_k((u,v]) p_k(X)}{p_{(0,n]}(X)}\right)\right] \\ &< (v-u) \sum_{k\in\kappa((u,v])} \pi_k((u,v]) \mathbb{E}_{X\sim\mathbb{P}_k}\left[\log\left(\frac{p_k(X)}{p_{(0,n]}(X)}\right)\right] \\ &= (v-u) \sum_{k\in\kappa((u,v])} \pi_k((u,v]) D_{KL}(\mathbb{P}_k \parallel \mathbb{P}_{(0,n]}).\end{aligned}$$

In the third line, we used the fact that $x \mapsto x\log(x)$ is strictly convex. Consider any segmentation α that does not contain α^0. Then there exists at least one segment $(\alpha_{k'\text{-}1}, \alpha_{k'}]$ such that $\kappa((\alpha_{k'\text{-}1}, \alpha_{k'}])$ contains more than one element and thus

$$\begin{aligned}
\mathbb{E}\left[G\left((X_i)_{i=1}^n \mid \alpha, p^\infty\right)\right] &= \sum_{k=1}^{K} \mathbb{E}\left[\sum_{i=\alpha_{k\text{-}1}+1}^{\alpha_k} \log\left(\frac{p_{(\alpha_{k\text{-}1},\alpha_k]}(X_i)}{p_{(0,n]}(X_i)}\right)\right] \\
&< \sum_{k=1}^{K} (\alpha_k - \alpha_{k\text{-}1}) \sum_{l \in \kappa((\alpha_{k\text{-}1},\alpha_k])} \pi_l((\alpha_{k\text{-}1}, \alpha_k]) D_{KL}(\mathbb{P}_l \parallel \mathbb{P}_{(0,n]}) \\
&= \sum_{k=1}^{K} (\alpha_k^0 - \alpha_{k\text{-}1}^0) D_{KL}(\mathbb{P}_k \parallel \mathbb{P}_{(0,n]}),
\end{aligned}$$

applying the above inequality with $u = \alpha_{k'\text{-}1}$ and $v = \alpha_{k'}$. ■

Proposition 3 *For the Bayes classifier p^∞, the expected classifier log-likelihood ratio*

$$s \mapsto \mathbb{E}\left[G\left((X_i)_{i=u+1}^v \mid \{u, s, v\}, p^\infty\right)\right]$$

is piecewise convex between the underlying change points α^0, with strict convexity in the segment $(\alpha_{k\text{-}1}^0, \alpha_k^0]$ if $\mathbb{P}_{(u,\alpha_{k\text{-}1}^0]} \neq \mathbb{P}_{(\alpha_{k\text{-}1}^0,\alpha_k^0]}$ or $\mathbb{P}_{(\alpha_{k\text{-}1}^0,\alpha_k^0]} \neq \mathbb{P}_{(\alpha_k^0,v]}$.

Proof Let $s \in (u, v-1] \backslash \alpha^0$. We show that the expected classifier log-likelihood ratio

$$G(s) := \mathbb{E}\left[G\left((X_i)_{i=u+1}^v \mid \{u, s, v\}, p^\infty\right)\right]$$

is convex at s, that is, $G(s+1) - 2G(s) + G(s-1) > 0$. By definition

$$G(s) = \mathbb{E}\left[\sum_{i=u+1}^{s} \log\left(\frac{p_{(u,s]}(X_i)}{p_{(u,v]}(X_i)}\right) + \sum_{i=s+1}^{v} \log\left(\frac{p_{(s,v]}(X_i)}{p_{(u,v]}(X_i)}\right)\right]$$

and

$$\begin{aligned}
G(s+1) - G(s) &= \mathbb{E}\left[\sum_{i=u+1}^{s+1} \log\left(\frac{p_{(u,s+1]}(X_i)}{p_{(u,s]}(X_i)}\right) + \sum_{i=s+2}^{v} \log\left(\frac{p_{(s+1,v]}(X_i)}{p_{(s,v]}(X_i)}\right) + \log\left(\frac{p_{(u,s]}(X_{s+1})}{p_{(s,v]}(X_{s+1})}\right)\right] \\
&= (s-u+1) D_{KL}(\mathbb{P}_{(u,s+1]} \parallel \mathbb{P}_{(u,s]}) + (v-s-1) D_{KL}(\mathbb{P}_{(s+1,v]} \parallel \mathbb{P}_{(s,v]}) + \\
&\qquad \mathbb{E}\left[\log\left(\frac{p_{(u,s]}(X_{s+1})}{p_{(s,v]}(X_{s+1})}\right)\right]
\end{aligned}$$

Similarly,

$$\begin{aligned}
G(s-1) - G(s) &= \mathbb{E}\left[\sum_{i=u+1}^{s-1} \log\left(\frac{p_{(u,s-1]}(X_i)}{p_{(u,s]}(X_i)}\right) + \sum_{i=s}^{v} \log\left(\frac{p_{(s-1,v]}(X_i)}{p_{(s,v]}(X_i)}\right) + \log\left(\frac{p_{(s,v]}(X_s)}{p_{(u,s]}(X_s)}\right)\right] \\
&= (s-u-1) D_{KL}(\mathbb{P}_{(u,s-1]} \parallel \mathbb{P}_{(u,s]}) + (v-s+1) D_{KL}(\mathbb{P}_{(s-1,v]} \parallel \mathbb{P}_{(s,v]}) + \\
&\qquad \mathbb{E}\left[\log\left(\frac{p_{(s,v]}(X_s)}{p_{(u,s]}(X_s)}\right)\right]
\end{aligned}$$

As $s \notin \alpha^0$, the X_s, X_{s+1} are i.i.d. and thus

$$\mathbb{E}\left[\log\left(\frac{p_{(u,s]}(X_{s+1})}{p_{(s,v]}(X_{s+1})}\right) + \log\left(\frac{p_{(s,v]}(X_s)}{p_{(u,s]}(X_s)}\right)\right] = \mathbb{E}\left[\log\left(\frac{p_{(u,s]}(X_{s+1})}{p_{(u,s]}(X_s)}\right) + \log\left(\frac{p_{(s,v]}(X_s)}{p_{(s,v]}(X_{s+1})}\right)\right] = 0.$$

Combining everything,

$$\begin{aligned} G(s+1) - 2G(s) + G(s-1) &= (G(s+1) - G(s)) + (G(s-1) - G(s)) \\ &= (s-u+1)D_{KL}(\mathbb{P}_{(u,s+1]} \parallel \mathbb{P}_{(u,s]}) \ + (v-s-1)D_{KL}(\mathbb{P}_{(s+1,v]} \parallel \mathbb{P}_{(s,v]}) + \\ &\quad (s-u-1)D_{KL}(\mathbb{P}_{(u,s-1]} \parallel \mathbb{P}_{(u,s]}) \ + (v-s+1)D_{KL}(\mathbb{P}_{(s-1,v]} \parallel \mathbb{P}_{(s,v]}). \end{aligned}$$

This is nonnegative, as the Kullback-Leibler divergence between two distributions is nonnegative and as $u+1 \leqslant s \leqslant v-1$. Furthermore, as $s-u+1 \geqslant 1$ and $v-s+1 \geqslant 1$, this term is positive if $\mathbb{P}_{(u,s+1]} \neq \mathbb{P}_{(u,s]}$ or $\mathbb{P}_{(s-1,v]} \neq \mathbb{P}_{(s,v]}$. Choose k such that $s \in (\alpha^0_{k\text{-}1}, \alpha^0_k]$. Then, as $s \notin \alpha^0$, also $s+1 \in (\alpha^0_{k\text{-}1}, \alpha^0_k]$. Then $\mathbb{P}_{(u,s+1]} \neq \mathbb{P}_{(u,s]}$ or $\mathbb{P}_{(s-1,v]} \neq \mathbb{P}_{(s,v]}$ if and only if $\mathbb{P}_{(u,\alpha^0_{k\text{-}1}]} \neq \mathbb{P}_{(\alpha^0_{k\text{-}1},\alpha^0_k]}$ or $\mathbb{P}_{(\alpha^0_{k\text{-}1},\alpha^0_k]} \neq \mathbb{P}_{(\alpha^0_k,v]}$. ■

Proposition 4 *For the Bayes classifier p^∞ and any initial guess $u < s^{(0)} < v$, the expected approximate gain*

$$s \mapsto \mathbb{E}\left[\sum_{i=u+1}^{s} \log\left(\frac{v-u}{s^{(0)}-u} p^\infty_{\{u,s^{(0)},v\}}(X_i)_1\right) + \sum_{i=s+1}^{v} \log\left(\frac{v-u}{v-s^{(0)}} p^\infty_{\{u,s^{(0)},v\}}(X_i)_2\right)\right]$$

is piecewise linear between the underlying change points α^0. If there is a single change point $a^0 \in (u,v]$, the expected approximate gain has a unique maximum at $s = a^0$.

Proof Write

$$\begin{aligned} G(s) &\coloneqq \sum_{i=u+1}^{s} \log\left(\frac{v-u}{s^{(0)}-u} p^\infty_{\{u,s^{(0)},v\}}(X_i)_1\right) + \sum_{i=s+1}^{v} \log\left(\frac{v-u}{v-s^{(0)}} p^\infty_{\{u,s^{(0)},v\}}(X_i)_2\right) \\ &= \sum_{i=u+1}^{s} \log\left(\frac{p_{(u,s^{(0)}]}(X_i)}{p_{(u,v]}(X_i)}\right) + \sum_{i=s+1}^{v} \log\left(\frac{p_{(s^{(0)},v]}(X_i)}{p_{(u,v]}(X_i)}\right) \end{aligned}$$

and define

$$U_j \coloneqq G(j) - G(j-1) = \log\left(\frac{p_{(u,s^{(0)}]}(X_j)}{p_{(s^{(0)},v]}(X_j)}\right),$$

such that $G(s) = G(u) + \sum_{j=u+1}^{s} U_j$. For any segment $(\alpha^0_{k\text{-}1}, \alpha_k]$ intersecting the segment $(u,v]$ and any $s \in (\alpha^0_{k\text{-}1}, \alpha_k]$, the $U_{\alpha^0_{k\text{-}1}+1}, \ldots, U_{\alpha_k}$ are i.i.d. Consequently, $\mathbb{E}[G(s)] = \mathbb{E}[G(\alpha^0_{k\text{-}1})] + (s - \alpha^0_{k\text{-}1})\mathbb{E}[U_{\alpha^0_{k\text{-}1}+1}]$, proving that $\mathbb{E}[G(s)]$ is piecewise linear between the change points in α^0.

Now assume that there is a single change point a^0 in $(u,v]$. Assume without loss of generality that $s^{(0)} \geqslant a^0$ such that $p_{(u,s^{(0)}]} = \frac{a^0-u}{s^{(0)}-u} p_{(u,a^0]} + \frac{s^{(0)}-a^0}{s^{(0)}-u} p_{(a^0,v]}$ and $p_{(s^{(0)},v]} = p_{(a^0,v]}$.

Then, for $j \leqslant a^0$,

$$\begin{aligned}
\mathbb{E}[U_j] &= \mathbb{E}_{\mathbb{P}_{(u,a^0]}}\left[\log\left(\frac{p_{(u,s^{(0)}]}(X_j)}{p_{(s^{(0)},v]}(X_j)}\right)\right] \\
&= \frac{s^{(0)}-u}{a^0-u}\mathbb{E}_{\mathbb{P}_{(u,s^{(0)}]}}\left[\log\left(\frac{p_{(u,s^{(0)}]}(X_j)}{p_{(s^{(0)},v]}(X_j)}\right)\right] - \frac{s^{(0)}-a^0}{a^0-u}\mathbb{E}_{\mathbb{P}_{(a^0,v]}}\left[\log\left(\frac{p_{(u,s^{(0)}]}(X_j)}{p_{(s^{(0)},v]}(X_j)}\right)\right] \\
&= \frac{s^{(0)}-u}{a^0-u}D_{KL}(\mathbb{P}_{(u,s^{(0)}]} \parallel \mathbb{P}_{(a^0,v]}) + \frac{s^{(0)}-a^0}{a^0-u}D_{KL}(\mathbb{P}_{(s^{(0)},v]} \parallel \mathbb{P}_{(u,s^{(0)}]}) > 0.
\end{aligned}$$

Similarly, for $j > a^0$

$$\mathbb{E}[U_s] = \mathbb{E}_{\mathbb{P}_{(s^{(0)},v]}}\left[\log\left(\frac{p_{(u,s^{(0)}]}(X_j)}{p_{(s^{(0)},v]}(X_j)}\right)\right] = -D_{KL}(\mathbb{P}_{(a^{(0)},v]} \parallel \mathbb{P}_{(u,s^{(0)}]}) < 0,$$

which shows that $s \mapsto \mathbb{E}[G(s)] = \mathbb{E}[G(u)] + \sum_{i=u+1}^{s}\mathbb{E}[U_i]$ has a unique maximum at $s = a^0$. ■

Lemma 6 *Let $\mathbb{P}_1, \mathbb{P}_2$ be probability measures with corresponding densities p_1, p_2. Let $\delta_1, \delta_2 \in [0,1]$ with $\delta_1 \neq \delta_2$. Define $\mathbb{Q}(\delta) := (1-\delta)\mathbb{P}_1 + \delta\mathbb{P}_2$ with density $q(\delta) = (1-\delta)p_1 + \delta p_2$. Then*

$$\mathbb{E}_{\mathbb{P}_1}\left[\log\left(\frac{q(\delta_1)}{q(\delta_2)}\right)\right] = \frac{\delta_2}{\delta_2-\delta_1}D_{KL}(\mathbb{Q}(\delta_1) \parallel \mathbb{Q}(\delta_2)) + \tag{7}$$

$$\frac{\delta_1}{\delta_2-\delta_1}D_{KL}(\mathbb{Q}(\delta_2) \parallel \mathbb{Q}(\delta_1)). \tag{8}$$

Proof The Kullback Leibler divergence between distribution Q_1, Q_2 with densities q_1, q_2 is $D_{KL}(Q_1 \parallel Q_2) = \mathbb{E}_{Q_1}[\log\left(\frac{q_1(X)}{q_2(X)}\right)]$. Thus

$$D_{KL}(\mathbb{Q}(\delta_1) \parallel \mathbb{Q}(\delta_2)) = (1-\delta_1)\mathbb{E}_{\mathbb{P}_1}\left[\log\left(\frac{q(\delta_1)}{q(\delta_2)}\right)\right] + \delta_1\mathbb{E}_{\mathbb{P}_2}\left[\log\left(\frac{q(\delta_1)}{q(\delta_2)}\right)\right]$$

and

$$D_{KL}(\mathbb{Q}(\delta_2) \parallel \mathbb{Q}(\delta_1)) = (1-\delta_2)\mathbb{E}_{\mathbb{P}_1}\left[\log\left(\frac{q(\delta_2)}{q(\delta_1)}\right)\right] + \delta_2\mathbb{E}_{\mathbb{P}_2}\left[\log\left(\frac{q(\delta_2)}{q(\delta_1)}\right)\right]$$

such that

$$\begin{aligned}
&\frac{\delta_2}{\delta_2-\delta_1}D_{KL}(\mathbb{Q}(\delta_1) \parallel \mathbb{Q}(\delta_2)) + \frac{\delta_1}{\delta_2-\delta_1}D_{KL}(\mathbb{Q}(\delta_2) \parallel \mathbb{Q}(\delta_1)) \\
&\quad = \left(\frac{\delta_2}{\delta_2-\delta_1}(1-\delta_1) - \frac{\delta_1}{\delta_2-\delta_1}(1-\delta_2)\right)\mathbb{E}_{\mathbb{P}_1}\left[\log\left(\frac{q(\delta_1)}{q(\delta_2)}\right)\right] + \\
&\qquad \left(\frac{\delta_2}{\delta_2-\delta_1}\delta_1 - \frac{\delta_1}{\delta_2-\delta_1}\delta_2\right)\mathbb{E}_{\mathbb{P}_2}\left[\log\left(\frac{q(\delta_1)}{q(\delta_2)}\right)\right] = \mathbb{E}_{\mathbb{P}_1}\left[\log\left(\frac{q(\delta_1)}{q(\delta_2)}\right)\right].
\end{aligned}$$

■

Lemma 7 *Let $Z_1, \dots, Z_n$ be independent random variables bounded from below by a and from above by b. Let $1 \leqslant K \leqslant n$. Then,*

$$\mathbb{P}\left[\bigcup_{k=1,\dots,K}\left\{\sum_{i=1}^{k}(Z_i - \mathbb{E}[Z_i]) \geqslant (b-a)\sqrt{k\log(n)}\right\}\right] \leqslant \frac{K}{n^2}.$$

Proof Define $\tilde{Z}_i := Z_i - \mathbb{E}[Z_i]$ such that $a_i := a - \mathbb{E}[Z_i] \leqslant Z_i \leqslant b - \mathbb{E}[Z_i] := b_i$, with $b_i - a_i = b - a$ for all $i = 1, \dots, n$. By Hoeffding's theorem, for each $k = 1, \dots, K$ and $\lambda > 0$, we have

$$\mathbb{P}\left[\sum_{i=1}^{k}\tilde{Z}_i \geqslant \lambda\right] \leqslant \exp\left(-\frac{2\lambda^2}{k(b_i - a_i)^2}\right).$$

Setting $\lambda = (b-a)\sqrt{k\log(n)}$ and taking a union bound yields

$$\mathbb{P}\left[\bigcup_{k=1,\dots,K}\left\{\sum_{i=1}^{k}(Z_i - \mathbb{E}[Z_i]) \geqslant (b-a)\sqrt{k\log(n)}\right\}\right] \leqslant K\exp\left(-\frac{2k(b-a)^2\log(n)}{k(b-a)^2}\right) = \frac{K}{n^2}.$$

■

Theorem 5 *Let $0 < \delta, \eta < 1$ and $\delta \leqslant \tau^0, \rho \leqslant 1 - \delta$. Set $s = \rho n$. Assume a single change point scenario with n observations and a change point at $a^0 = \tau^0 n$. Let $\hat{p}$ satisfy $\max_{i=1,\dots,n} |\hat{p}_s(X_i)_1 - p_s^\infty(X_i)_1| \overset{n\to\infty}{\to} 0$ in probability. Choose*

$$\hat{a} \in \operatorname*{argmax}_{t=1,\dots,n} \sum_{i=1}^{t}\log_\eta\left(\frac{n}{s}\hat{p}_s(X_i)_1\right) + \sum_{i=t+1}^{n}\log_\eta\left(\frac{n}{n-s}\hat{p}_s(X_i)_2\right)$$

as in Algorithm 2. Then

$$\mathbb{P}\left[\left|\frac{\hat{a}}{n} - \tau^0\right| > \frac{C\log(n)}{\delta^2 \mathrm{d}_{\mathrm{TV}}(\mathbb{P}_1, \mathbb{P}_2)^4 n}\right] \overset{n\to\infty}{\longrightarrow} 0,$$

where $\mathrm{d}_{\mathrm{TV}}(\mathbb{P}_1, \mathbb{P}_2)$ is the total variation distance between the two distributions $\mathbb{P}_1$ and $\mathbb{P}_2$ before and after the change point and $C = \frac{1}{(1-\eta)^4}\log\left(\frac{1}{(1-\rho)\rho}\frac{1}{\eta^2}\right)^2$.

Proof Recall that $\log_\eta(x) = \log(\eta + (1-\eta)x)$ and write

$$G^{s,\eta}(t,p) = \sum_{i=1}^{t}\log_\eta\left(\frac{n}{s}p_s(X_i)_1\right) + \sum_{i=t+1}^{n}\log_\eta\left(\frac{n}{n-s}p_s(X_i)_2\right)$$

for $t = 0, \dots, n$ and any classifier p. Recall that p^∞ is the Bayes classifier with $p_s^\infty = \left(\frac{s}{n}\frac{p_{(0,s]}}{p_{(0,n]}}, \frac{n-s}{n}\frac{p_{(s,n]}}{p_{(0,n]}}\right)^T$. We proceed in four steps.

(i) First, we show that, for any $t = 1, \dots, n$,

$$\mathbb{E}[G^{s,\eta}(a^0, p^\infty) - G^{s,\eta}(t, p^\infty)] \geqslant 2|t - a^0|\delta(1-\eta)^2 \mathrm{d}_{\mathrm{TV}}(\mathbb{P}_1, \mathbb{P}_2)^2.$$

(ii) Second, we show that

$$\mathbb{P}\Bigg[\bigcap_{t=1,\dots n} \Bigg\{ \left[G^{s,\eta}(a^0, p^\infty) - G^{s,\eta}(t, p^\infty)\right] - \mathbb{E}\left[G^{s,\eta}(a^0, p^\infty) - G^{s,\eta}(t, p^\infty)\right] \\ > -2\log\left(\frac{1}{\rho(1-\rho)}\frac{1}{\eta^2}\right)\sqrt{|t-a^0|\log(n)}\Bigg\}\Bigg] \geqslant 1 - \frac{1}{n} \overset{n\to\infty}{\longrightarrow} 1.$$

(iii) Next, we argue by $\frac{1}{\eta}$-Lipschitz continuity of $\log_\eta$ and consistency of $\hat{p}$ that, for any $\varepsilon > 0$,

$$\mathbb{P}\left[\bigcap_{t=1,\dots,n} \left\{[G^{s,\eta}(a^0, \hat{p}) - G^{s,\eta}(t, \hat{p})] - [G^{s,\eta}(a^0, p^\infty) - G^{s,\eta}(t, p^\infty)] > -|t-a^0|\varepsilon\right\}\right] \overset{n\to\infty}{\longrightarrow} 1.$$

(iv) Combined, these steps imply that, if $|t - a^0| \geqslant \frac{C\log(n)}{\delta^2 \mathrm{d}_{\mathrm{TV}}(\mathbb{P}_1,\mathbb{P}_2)^4}$, then, with high probability, $G^{s,\eta}(a^0, \hat{p}) > G^{s,\eta}(t, \hat{p})$ and thus $|\hat{a} - a^0| < \frac{C\log(n)}{\delta^2 \mathrm{d}_{\mathrm{TV}}(\mathbb{P}_1,\mathbb{P}_2)^4}$.

Step 1: For $i = 1, \dots, n$, define

$$U^{s,\eta}(p)_i := G^{s,\eta}(i, p) - G^{s,\eta}(i-1, p) = \log_\eta\left(\frac{n}{s}p_s(X_i)_1\right) - \log_\eta\left(\frac{n}{n-s}p_s(X_i)_2\right).$$

We show that

$$|\mathbb{E}[U^{s,\eta}(p^\infty)_i]| \geqslant 2(1-\eta)\delta d_{\mathrm{TV}}(\mathbb{P}_1, \mathbb{P}_2)^2,$$

with a positive sign for $i \leqslant a^0$ and a negative sign for $i > a^0$.

By symmetry, we can assume, without loss of generality, that $s \geqslant a^0$. Then, $\mathbb{P}_{(0,s]} = \frac{a^0}{s}\mathbb{P}_1 + \frac{s-a^0}{s}\mathbb{P}_2$ and $\mathbb{P}_{(s,n]} = \mathbb{P}_2$ with densities $p_{(0,s]} = \frac{a^0}{s}p_1 + \frac{s-a^0}{s}p_2$ and $p_{(s,n]} = p_2$. Furthermore, $\mathbb{P}_{(0,n]} = \frac{a^0}{n}\mathbb{P}_1 + \frac{n-a^0}{n}\mathbb{P}_2$ with density $p_{(0,n]} = \frac{a^0}{n}p_1 + \frac{n-a^0}{n}p_2$. For each $i = 1, \dots n$, we have

$$\begin{aligned}
U^{s,\eta}(p^\infty)_i &= \log\left(\eta + (1-\eta)\frac{p_{(0,s]}(X_i)}{p_{(0,n]}(X_i)}\right) - \log\left(\eta + (1-\eta)\frac{p_{(s,n]}(X_i)}{p_{(0,n]}(X_i)}\right) \\
&= \log\left(\frac{\eta p_{(0,n]}(X_i) + (1-\eta)p_{(0,s]}(X_i)}{p_{(0,n]}(X_i)}\right) - \log\left(\frac{\eta p_{(0,n]}(X_i) + (1-\eta)p_{(s,n]}(X_i)}{p_{(0,n]}(X_i)}\right) \\
&= \log\left(\frac{\eta(\frac{a^0}{n}p_1(X_i) + \frac{n-a^0}{n}p_2(X_i)) + (1-\eta)(\frac{a^0}{s}p_1 + \frac{s-a^0}{s}p_2)(X_i)}{\eta(\frac{a^0}{n}p_1(X_i) + \frac{n-a^0}{n}p_2(X_i)) + (1-\eta)p_2(X_i)}\right) \\
&= \log\left(\frac{\frac{a^0}{s}(1-\eta\frac{n-s}{n})p_1(X_i) + (1-\frac{a^0}{s}(1-\eta\frac{n-s}{n}))p_2(X_i)}{\eta\frac{a^0}{n}p_1(X_i) + (1-\eta\frac{a^0}{n})p_2(X_i)}\right).
\end{aligned}$$

Let $\mathbb{Q}(\delta) = (1-\delta)\mathbb{P}_1 + \delta\mathbb{P}_2$. Then, by Lemma 6 with $\delta_1 = 1 - \frac{a^0}{s}(1-\eta\frac{n-s}{n})$, $\delta_2 = 1 - \eta\frac{a^0}{n}$, and thus $\delta_2 - \delta_1 = (1-\eta)\frac{a^0}{s}$,

$$\mathbb{E}_{X_i\sim\mathbb{P}_1}[U^{s,\eta}(p^\infty)_i] = \frac{1-\eta\frac{a^0}{n}}{(1-\eta)\frac{a^0}{s}} D_{KL}(\mathbb{Q}(\delta_1) \parallel \mathbb{Q}(\delta_2)) + \frac{1-\frac{a^0}{s}(1-\eta\frac{n-s}{n})}{(1-\eta)\frac{a^0}{s}} D_{KL}(\mathbb{Q}(\delta_2) \parallel \mathbb{Q}(\delta_1)).$$

By Pinsker's inequality, $D_{KL}(\mathbb{Q}(\delta_1) \parallel \mathbb{Q}(\delta_2)) \geqslant 2\mathrm{d}_{\mathrm{TV}}(\mathbb{Q}(\delta_1), \mathbb{Q}(\delta_2))^2$. Furthermore, it holds that $\mathrm{d}_{\mathrm{TV}}(\mathbb{Q}(\delta_1), \mathbb{Q}(\delta_2)) = |\delta_2 - \delta_1|\mathrm{d}_{\mathrm{TV}}(\mathbb{P}_1, \mathbb{P}_2)$. Thus,

$$\begin{aligned}
\mathbb{E}_{X_i\sim\mathbb{P}_1}[U^{s,\eta}(p^\infty)_i] &\geqslant 2(1-\eta)\frac{a^0}{s}\left(1-\eta\frac{a^0}{n} + 1 - \frac{a^0}{s}(1-\eta\frac{n-s}{n})\right)\mathrm{d}_{\mathrm{TV}}(\mathbb{P}_1,\mathbb{P}_2)^2 \\
&= 2(1-\eta)\frac{a^0}{s}\left(2 - \frac{a^0}{s}(1-\eta\frac{n-2s}{n})\right)\mathrm{d}_{\mathrm{TV}}(\mathbb{P}_1,\mathbb{P}_2)^2 \\
&\geqslant 2(1-\eta)^2\delta\mathrm{d}_{\mathrm{TV}}(\mathbb{P}_1,\mathbb{P}_2)^2.
\end{aligned}$$

Similarly,

$$-\mathbb{E}_{X_i\sim\mathbb{P}_2}[U^{s,\eta}(p^\infty)_i] \geqslant 2(1-\eta)^2\delta\mathrm{d}_{\mathrm{TV}}(\mathbb{P}_1,\mathbb{P}_2)^2,$$

which proves the claim.

Step 2: Note that $\log_\eta(x)$ is bounded from below by $\log(\eta)$ and from above by $\log(x)$ for all $x \geqslant 1$. Consequently, the $U^{s,\eta}(p)_i = \log_\eta(\frac{n}{s}p_s(X_i)_1) - \log_\eta(\frac{n}{n-s}p_s(X_i)_2)$ are bounded from below by $a = -\log\left(\frac{1}{1-\rho}\frac{1}{\eta}\right)$ and from above by $b = \log\left(\frac{1}{\rho}\frac{1}{\eta}\right)$ with $b - a = \log\left(\frac{1}{\eta^2}\frac{1}{\rho(1-\rho)}\right)$. Write

$$\Delta^{s,\eta}(t) := \left[G^{s,\eta}(a^0,p^\infty) - G^{s,\eta}(t,p^\infty)\right] - \mathbb{E}\left[G^{s,\eta}(a^0,p^\infty) - G^{s,\eta}(t,p^\infty)\right].$$

Then, for $t \leqslant a^0$,

$$\Delta^{s,\eta}(t) = \sum_{i=t+1}^{a^0}(U^{s,\eta}(p^\infty)_i - \mathbb{E}[U^{s,\eta}(p^\infty)_i]).$$

We apply Lemma 7 with $K = a^0$ and $Z_i = -U^{s,\eta}(p^\infty)_{a^0-i+1}$ for $i = 1,\dots,K$. This yields

$$\begin{aligned}
&\mathbb{P}\left[\bigcup_{t=1,\dots,a^0}\left\{\Delta^{s,\eta}(t) \leqslant -\log\left(\frac{1}{\eta^2}\frac{1}{\rho(1-\rho)}\right)\sqrt{|t-a^0|\log(n)}\right\}\right] \\
&= \mathbb{P}\left[\bigcup_{k=1,\dots,K}\left\{\sum_{i=1}^k Z_i - \mathbb{E}[Z_i] \geqslant \log\left(\frac{1}{\eta^2}\frac{1}{\rho(1-\rho)}\right)\sqrt{|k|\log(n)}\right\}\right] \leqslant \frac{K}{n^2} = \frac{a^0}{n^2}.
\end{aligned}$$

Similarly, for $t \geqslant a^0 + 1$,

$$\Delta^{s,\eta}(t) = \sum_{i=a^0+1}^{t}(U^{s,\eta}(p^\infty)_i - \mathbb{E}[U^{s,\eta}(p^\infty)_i])$$

and we can apply Lemma 7 with $K = n - a^0$ and $Z_i = U_{a^0+i}$ for $i = 1, \dots, K$. This yields

$$\begin{aligned}
&\mathbb{P}\left[\bigcup_{t=a^0+1,\dots,n} \left\{\Delta^{s,\eta}(t) \leqslant -\log\left(\frac{1}{\eta^2}\frac{1}{\rho(1-\rho)}\right)\sqrt{|t-a^0|\log(n)}\right\}\right] \\
&= \mathbb{P}\left[\bigcup_{k=1,\dots,K} \left\{\sum_{i=1}^{k} Z_i - \mathbb{E}[Z_i] \geqslant \log\left(\frac{1}{\eta^2}\frac{1}{\rho(1-\rho)}\right)\sqrt{|k|\log(n)}\right\}\right] \leqslant \frac{K}{n^2} = \frac{n-a^0}{n^2}.
\end{aligned}$$

Combining the two bounds, we obtain

$$\mathbb{P}\left[\bigcup_{t=1,\dots,n} \left\{\Delta(t) \leqslant -\log\left(\frac{1}{\eta^2}\frac{1}{\rho(1-\rho)}\right)\sqrt{|t-a^0|\log(n)}\right\}\right] \leqslant \frac{n-a^0}{n^2} + \frac{a^0}{n^2} = \frac{1}{n}$$

and thus

$$\mathbb{P}\left[\bigcap_{t=1,\dots,n} \left\{\Delta(t) > -\log\left(\frac{1}{\eta^2}\frac{1}{\rho(1-\rho)}\right)\sqrt{|t-a^0|\log(n)}\right\}\right] \geqslant 1 - \frac{1}{n}$$

as desired.

Step 3: Let $\varepsilon > 0$. As $\log_\eta$ is $\frac{1}{\eta}$-Lipschitz, whenever

$$\max_{t=1,\dots,n} |p_s^\infty(X_i)_1 - \hat{p}_s(X_i)_1| \leqslant \eta\rho(1-\rho)\varepsilon$$

and thus also

$$\max_{i=1,\dots,n} |p_s^\infty(X_i)_2 - \hat{p}_s(X_i)_2| = \max_{i=1,\dots,n} |(1 - p_s^\infty(X_i)_1) - (1 - \hat{p}_s(X_i)_1)| \leqslant \eta\rho(1-\rho)\varepsilon,$$

then

$$\max_{i=1,\dots,n} |U^{s,\eta}(p^\infty)_i - U^{s,\eta}(\hat{p})_i| \leqslant \varepsilon,$$

and, for all $t = 1, \dots, n$,

$$\left|[G^{s,\eta}(a^0, \hat{p}) - G^{s,\eta}(t, \hat{p})] - [G^{s,\eta}(a^0, p^\infty) - G^{s,\eta}(t, p^\infty)]\right| \leqslant |t - a^0|\varepsilon.$$

Thus, in particular,

$$\begin{aligned}
&\mathbb{P}\left[\bigcap_{t=1,\dots,n} \left\{[G^{s,\eta}(a^0, \hat{p}) - G^{s,\eta}(t, \hat{p})] - [G^{s,\eta}(a^0, p^\infty) - G^{s,\eta}(t, p^\infty)] \geqslant -|t-a^0|\varepsilon\right\}\right] \\
&\geqslant \mathbb{P}\left[\bigcap_{t=1,\dots,n} \left\{\left|[G^{s,\eta}(a^0, \hat{p}) - G^{s,\eta}(t, \hat{p})] - [G^{s,\eta}(a^0, p^\infty) - G^{s,\eta}(t, p^\infty)]\right| \leqslant |t-a^0|\varepsilon\right\}\right] \\
&\geqslant \mathbb{P}\left[\max_{t=1,\dots,n} |p_s^\infty(X_i)_1 - \hat{p}_s(X_i)_1| \leqslant \eta\rho(1-\rho)\varepsilon\right] \xrightarrow{n\to\infty} 1
\end{aligned}$$

by the consistency assumption of $\hat{p}$, proving the claim.

Step 4: We combine the results of steps 1, 2, and 3 to prove the claim.

Let $t = 1, \dots, n$. We bound $G(a^0, \hat{p}) - G(t, \hat{p})$ from below with high probability. For this, we decompose

$$\begin{aligned} G^{s,\eta}(a^0, \hat{p}) - G^{s,\eta}(t, \hat{p}) = \underbrace{\mathbb{E}[G^{s,\eta}(a^0, p^\infty) - G^{s,\eta}(t, p^\infty)]}_{A(t)} + \\ \underbrace{[G^{s,\eta}(a^0, p^\infty) - G^{s,\eta}(t, p^\infty)] - \mathbb{E}[G^{s,\eta}(a^0, p^\infty) - G^{s,\eta}(t, p^\infty)]}_{B(t)} + \\ \underbrace{[G^{s,\eta}(a^0, \hat{p}) - G^{s,\eta}(t, \hat{p})] - [G^{s,\eta}(a^0, p^\infty) - G^{s,\eta}(t, p^\infty)]}_{C(t)}. \end{aligned}$$

By step 1,

$$\forall t = 1, \dots, n : A(t) \geqslant 2|t - a^0|\delta(1-\eta)^2 \mathrm{d}_{\mathrm{TV}}(\mathbb{P}_1, \mathbb{P}_2)^2.$$

By step 2,

$$\mathbb{P}\left[\forall t = 1, \dots, n : B(t) > -\log\left(\frac{1}{\rho(1-\rho)}\frac{1}{\eta^2}\right)\sqrt{|t - a^0|\log(n)}\right] \geqslant 1 - \frac{1}{n} \overset{n\to\infty}{\longrightarrow} 1.$$

By step 3, choosing $\varepsilon = \delta(1-\eta)^2 \mathrm{d}_{\mathrm{TV}}(\mathbb{P}_1, \mathbb{P}_2)^2$,

$$\mathbb{P}\left[\forall t = 1, \dots, n : C(t) \geqslant -|t - a^0|(1-\eta)^2\delta \mathrm{d}_{\mathrm{TV}}(\mathbb{P}_1, \mathbb{P}_2)^2\right] \overset{n\to\infty}{\longrightarrow} 1.$$

Thus

$$\begin{aligned} \mathbb{P}\Bigg[\forall t = 1, \dots, n : G^{s,\eta}(a^0, \hat{p}) - G^{s,\eta}(t, \hat{p}) = A(t) + B(t) + C(t) > \\ |t - a^0|\delta(1-\eta)^2 \mathrm{d}_{\mathrm{TV}}(\mathbb{P}_1, \mathbb{P}_2)^2 - \log\left(\frac{1}{\rho(1-\rho)}\frac{1}{\eta^2}\right)\sqrt{\log(n)|t - a^0|}\Bigg] \overset{n\to\infty}{\longrightarrow} 1 \end{aligned}$$

and

$$\mathbb{P}\left[\forall t = 1, \dots n \text{ s.t. } |t - a^0| \leqslant \frac{C\log(n)}{\delta^2 \mathrm{d}_{\mathrm{TV}}(\mathbb{P}_1, \mathbb{P}_2)^4} : G^{s,\eta}(a^0, \hat{p}) > G^{s,\eta}(t, \hat{p})\right] \overset{n\to\infty}{\longrightarrow} 1,$$

where $C = \frac{1}{(1-\eta)^4}\log\left(\frac{n^2}{n-s}\frac{1}{\eta^2}\right)^2$, implying

$$\mathbb{P}\left[\left|\hat{a} - a^0\right| > \frac{C\log(n)}{\delta^2 \mathrm{d}_{\mathrm{TV}}(\mathbb{P}_1, \mathbb{P}_2)^4}\right] \overset{n\to\infty}{\longrightarrow} 0,$$

as $\hat{a} \in \operatorname{argmax}_{t=1,\dots,n} G^{s,\eta}(t, \hat{p})$.

■

Appendix B. Figures

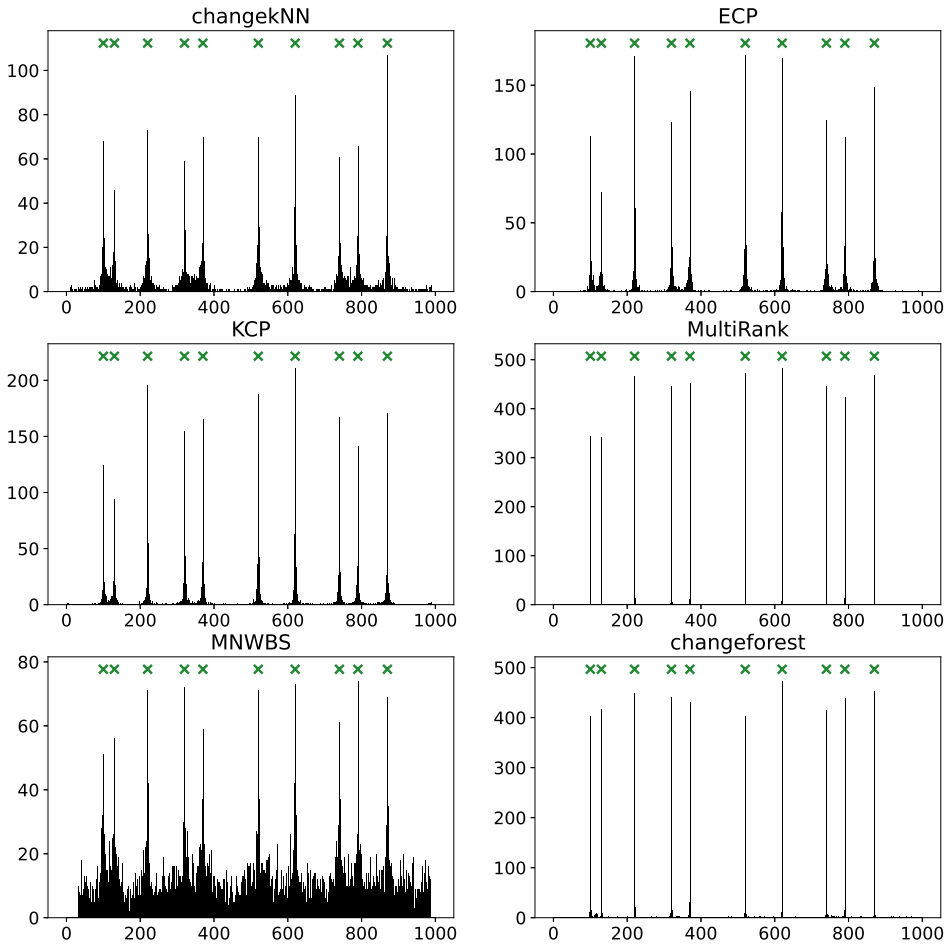

Figure 7: Histograms of cumulative change point estimates for 500 simulation runs on the Dirichlet simulation setup. The change in mean method is excluded as it did not produce any change point estimates for the Dirichlet simulation setup. The true change point locations are marked with green crosses. Note the different scales on the y-axis.

Appendix C. Tables

Table 4: Adjusted Rand indices (ARI) and Hausdorff distances (d_H) for different segmentations. The true segmentation comes from the parametric change in mean (CIC, $\alpha^0 = (0, 50, 100, 150)$) and the nonparametric glass ($\alpha^0 = (0, 17, 46, 55, 68, 144, 214)$) setup.

true segmentation	estimated segmentation	ARI	d_H	comment
0, 50, 100, 150	0, 50, 100, 150	1.00	0.000	perfect fit
	0, 52, 99, 150	0.94	0.013	almost perfect fit
	0, 23, 50, 100, 150	0.87	0.153	one extra change point
	0, 43, 87, 97, 150	0.75	0.087	two extra change points
	0, 50, 150	0.57	0.333	one missing change point
	0, 20, 70, 150	0.37	0.200	random segmentation
0, 17, 46, 55, 68, 144, 214	0, 17, 46, 55, 68, 144, 214	1.00	0.000	perfect fit
	0, 15, 45, 55, 68, 142, 214	0.95	0.009	almost perfect fit
	0, 17, 46, 55, 68, 80, 144, 214	0.91	0.056	one extra change point
	0, 17, 46, 55, 68, 100, 144, 214	0.83	0.150	one extra change point
	0, 46, 55, 68, 144, 214	0.95	0.079	one missing change point
	0, 17, 46, 55, 144, 214	0.89	0.061	one missing change point
	0, 50, 100, 150, 214	0.61	0.150	random segmentation
	0, 214	0.00	0.327	no segmentation

Table 5: Median relative Hausdorff distance over 500 simulations (in percent), with median absolute deviations in parentheses. Optimal scores are marked in bold.

	CIM	CIC	Dirichlet	iris	glass
change in mean	**0.0** (0.1)	33.3 (0.0)	48.0 (0.0)	1.3 (4.9)	16.8 (1.5)
changekNN	**0.0** (1.0)	33.3 (0.6)	6.6 (7.7)	**0.0** (1.5)	7.9 (3.4)
ECP	**0.0** (1.0)	32.0 (9.3)	5.2 (2.8)	**0.0** (1.2)	31.1 (13.2)
KCP	**0.0** (0.3)	**0.7** (2.4)	5.0 (2.5)	**0.0** (0.8)	7.9 (10.1)
MultiRank	**0.0** (0.0)	33.3 (0.3)	3.0 (1.3)	15.3 (4.0)	15.9 (2.1)
MNWBS	**0.0** (0.7)	15.3 (12.5)	5.9 (0.9)	**0.0** (0.5)	8.9 (5.9)
`changeforest`	**0.0** (1.8)	1.3 (3.7)	**0.2** (1.4)	**0.0** (2.1)	**2.8** (3.7)

	breast cancer	abalone	wine	dry beans	average
change in mean	16.7 (1.2)	5.0 (1.8)	0.4 (1.4)	1.3 (1.3)	13.7
changekNN	**0.0** (7.5)	5.0 (3.5)	1.5 (1.7)	**0.0** (0.6)	6.0
ECP	**0.0** (0.8)	6.2 (2.1)	0.3 (1.7)	**0.0** (0.1)	8.1
KCP	**0.0** (0.4)	4.2 (1.6)	0.2 (1.4)	**0.0** (0.0)	2.0
MultiRank	**0.0** (7.3)	9.1 (4.4)	3.0 (1.1)		10.0
MNWBS	**0.0** (0.7)				5.0
`changeforest`	**0.0** (2.1)	**3.1** (1.9)	**0.0** (0.9)	**0.0** (0.6)	**0.9**

Table 6: The average number of change points estimated over 500 simulations, with standard deviations in parentheses.

	CIM	CIC	Dirichlet	iris	glass
ground truth	2	2	10	2	5
change in mean	2.00 (0.00)	0.00 (0.00)	0.00 (0.00)	2.99 (1.40)	22.06 (7.19)
changekNN	2.12 (0.41)	0.58 (1.24)	8.01 (2.68)	2.11 (0.35)	4.16 (0.97)
ECP	2.07 (0.34)	1.21 (1.05)	7.70 (1.07)	2.09 (0.36)	2.58 (1.10)
KCP	2.03 (0.18)	2.39 (0.66)	8.39 (1.16)	2.05 (0.31)	3.44 (1.12)
MultiRank	2.00 (0.00)	0.41 (3.72)	9.18 (0.46)	13.31 (8.00)	21.74 (3.48)
MNWBS	2.14 (1.05)	1.81 (1.25)	21.96 (4.82)	2.05 (0.60)	5.26 (3.05)
`changeforest`	2.13 (0.38)	2.38 (0.67)	10.54 (0.77)	2.15 (0.42)	4.93 (0.78)

data set	breast cancer	abalone	wine	dry beans
ground truth	1	14	4	6
change in mean	16.62 (3.94)	11.07 (1.38)	3.65 (0.52)	7.04 (0.98)
changekNN	1.49 (0.90)	11.74 (1.76)	4.52 (1.00)	6.21 (0.49)
ECP	1.07 (0.36)	9.73 (1.10)	3.71 (0.57)	6.05 (0.25)
KCP	1.01 (0.14)	10.62 (1.07)	3.60 (0.49)	6.00 (0.04)
MultiRank	8.28 (10.91)	11.77 (1.93)	3.00 (0.71)	
MNWBS	1.05 (0.57)			
`changeforest`	1.08 (0.34)	12.09 (1.37)	4.21 (0.49)	6.23 (0.47)

Table 7: Average adjusted Rand indices of `changeforest` for different tuning parameters over 500 simulations, with standard deviations in parentheses. Optimal scores and those within two standard deviations are marked in bold. When necessary, mtry, the number of features evaluated at each split, was rounded down to the next integer.

trees	max. depth	mtry	CIM	CIC	Dirichlet	iris	glass
20	2	1	0.99 (0.03)	0.13 (0.29)	0.97 (0.02)	0.98 (0.04)	0.89 (0.08)
		$\sqrt{d}$	0.99 (0.03)	0.19 (0.34)	0.98 (0.02)	0.98 (0.04)	0.90 (0.08)
		d	0.99 (0.03)	0.19 (0.34)	0.98 (0.02)	0.99 (0.03)	0.91 (0.08)
	8	1	**0.99** (0.02)	0.29 (0.41)	0.98 (0.01)	**0.99** (0.02)	0.89 (0.08)
		$\sqrt{d}$	**0.99** (0.02)	0.27 (0.40)	**0.99** (0.01)	**0.99** (0.02)	0.89 (0.10)
		d	0.99 (0.05)	0.21 (0.37)	**0.99** (0.01)	**0.99** (0.02)	0.87 (0.13)
	∞	1	0.87 (0.27)	0.00 (0.00)	0.98 (0.02)	**0.99** (0.02)	0.86 (0.15)
		$\sqrt{d}$	0.85 (0.30)	0.00 (0.00)	**0.99** (0.01)	**0.99** (0.02)	0.87 (0.12)
		d	0.78 (0.36)	0.00 (0.00)	**0.99** (0.01)	**0.99** (0.02)	0.85 (0.16)
100	2	1	0.98 (0.04)	0.50 (0.44)	0.98 (0.02)	0.98 (0.04)	0.90 (0.08)
		$\sqrt{d}$	0.98 (0.04)	0.52 (0.44)	0.98 (0.02)	0.98 (0.04)	0.91 (0.07)
		d	0.98 (0.04)	0.50 (0.43)	0.98 (0.02)	0.98 (0.04)	**0.92** (0.07)
	8	1	0.99 (0.03)	0.93 (0.13)	0.99 (0.02)	0.98 (0.04)	**0.92** (0.07)
		$\sqrt{d}$	0.99 (0.03)	0.92 (0.12)	0.99 (0.02)	0.98 (0.04)	**0.92** (0.07)
		d	**0.99** (0.03)	0.89 (0.20)	0.98 (0.02)	0.99 (0.03)	0.92 (0.08)
	∞	1	0.98 (0.04)	0.88 (0.22)	0.99 (0.02)	0.98 (0.04)	0.92 (0.08)
		$\sqrt{d}$	0.99 (0.04)	0.88 (0.22)	**0.99** (0.02)	0.98 (0.04)	**0.92** (0.07)
		d	0.99 (0.03)	0.82 (0.31)	0.98 (0.02)	0.99 (0.04)	0.92 (0.08)
500	2	1	0.98 (0.05)	0.53 (0.43)	0.98 (0.02)	0.98 (0.05)	0.91 (0.07)
		$\sqrt{d}$	0.98 (0.04)	0.57 (0.43)	0.98 (0.03)	0.98 (0.04)	0.92 (0.07)
		d	0.98 (0.04)	0.52 (0.43)	0.97 (0.02)	0.98 (0.04)	0.92 (0.07)
	8	1	0.98 (0.04)	**0.95** (0.05)	0.99 (0.02)	0.98 (0.05)	**0.92** (0.07)
		$\sqrt{d}$	0.98 (0.04)	0.94 (0.07)	0.99 (0.02)	0.98 (0.04)	**0.92** (0.07)
		d	0.99 (0.04)	0.93 (0.11)	0.98 (0.02)	0.99 (0.04)	**0.92** (0.08)
	∞	1	0.98 (0.04)	0.94 (0.08)	**0.99** (0.02)	0.98 (0.04)	**0.92** (0.07)
		$\sqrt{d}$	0.98 (0.04)	**0.94** (0.07)	0.99 (0.02)	0.98 (0.04)	**0.92** (0.07)
		d	0.98 (0.04)	0.93 (0.11)	0.98 (0.02)	0.99 (0.04)	0.92 (0.08)

Continued on the next page.

Table 7 (Continued): Average adjusted Rand indices of `changeforest` for different tuning parameters over 500 simulations, with standard deviations in parentheses. Optimal scores and those within two standard deviations are marked in bold. When necessary, mtry, the number of features evaluated at each split, was rounded down to the next integer.

trees	max. depth	mtry	breast cancer	abalone	wine	dry beans	average
20	2	1	0.99 (0.05)	0.87 (0.08)	0.98 (0.02)	1.00 (0.01)	0.87 (0.11)
		$\sqrt{d}$	0.99 (0.04)	0.91 (0.06)	0.99 (0.01)	1.00 (0.01)	0.88 (0.12)
		d	0.99 (0.05)	0.92 (0.05)	0.99 (0.02)	1.00 (0.01)	0.88 (0.12)
	8	1	1.00 (0.03)	0.84 (0.11)	1.00 (0.01)	**1.00** (0.00)	0.89 (0.14)
		$\sqrt{d}$	0.99 (0.04)	0.85 (0.11)	1.00 (0.01)	**1.00** (0.00)	0.89 (0.14)
		d	0.99 (0.04)	0.81 (0.13)	**1.00** (0.00)	**1.00** (0.00)	0.87 (0.14)
	∞	1	**1.00** (0.01)	0.08 (0.14)	0.99 (0.01)	**1.00** (0.00)	0.75 (0.11)
		$\sqrt{d}$	**1.00** (0.01)	0.09 (0.14)	0.99 (0.01)	**1.00** (0.00)	0.75 (0.12)
		d	**1.00** (0.01)	0.07 (0.12)	0.99 (0.01)	**1.00** (0.00)	0.74 (0.14)
100	2	1	0.98 (0.09)	0.91 (0.05)	0.98 (0.04)	0.99 (0.02)	0.91 (0.15)
		$\sqrt{d}$	0.98 (0.08)	0.93 (0.04)	0.98 (0.04)	0.99 (0.02)	0.92 (0.15)
		d	0.97 (0.08)	0.94 (0.04)	0.98 (0.04)	0.99 (0.03)	0.92 (0.15)
	8	1	0.98 (0.08)	0.93 (0.05)	0.99 (0.03)	0.99 (0.01)	**0.97** (0.06)
		$\sqrt{d}$	0.98 (0.07)	0.93 (0.05)	0.99 (0.02)	1.00 (0.01)	**0.97** (0.06)
		d	0.98 (0.07)	0.93 (0.06)	0.99 (0.02)	1.00 (0.01)	0.96 (0.08)
	∞	1	1.00 (0.02)	0.88 (0.10)	**1.00** (0.00)	**1.00** (0.01)	0.96 (0.09)
		$\sqrt{d}$	0.99 (0.03)	0.90 (0.07)	**1.00** (0.01)	1.00 (0.01)	0.96 (0.08)
		d	1.00 (0.02)	0.89 (0.08)	**1.00** (0.00)	0.99 (0.02)	0.95 (0.11)
500	2	1	0.98 (0.09)	0.92 (0.05)	0.97 (0.06)	0.98 (0.03)	0.91 (0.15)
		$\sqrt{d}$	0.97 (0.09)	0.93 (0.04)	0.97 (0.06)	0.98 (0.03)	0.92 (0.15)
		d	0.97 (0.09)	**0.94** (0.04)	0.97 (0.06)	0.98 (0.03)	0.92 (0.15)
	8	1	0.97 (0.10)	**0.93** (0.05)	0.98 (0.04)	0.99 (0.02)	**0.97** (0.05)
		$\sqrt{d}$	0.98 (0.08)	**0.94** (0.04)	0.99 (0.04)	0.99 (0.02)	**0.97** (0.05)
		d	0.98 (0.08)	0.93 (0.05)	0.99 (0.02)	1.00 (0.01)	**0.97** (0.06)
	∞	1	0.99 (0.03)	0.91 (0.06)	**1.00** (0.01)	1.00 (0.01)	**0.97** (0.05)
		$\sqrt{d}$	0.99 (0.03)	0.93 (0.05)	**1.00** (0.01)	0.99 (0.02)	**0.97** (0.04)
		d	0.99 (0.03)	0.92 (0.06)	1.00 (0.01)	0.99 (0.02)	**0.97** (0.05)

Table 8: Average computational times of `changeforest` in seconds on 8 Intel Xeon 2.3 GHz cores. When necessary, mtry, the number of features evaluated at each split, was rounded down to the next integer.

trees	max. depth	mtry	CIM	CIC	Dirichlet	iris	glass	wine	breast cancer	abalone	dry beans
20	2	1	0.03	0.01	0.10	0.01	0.02	0.02	0.23	0.24	0.64
		$\sqrt{d}$	0.02	0.01	0.11	0.01	0.02	0.02	0.24	0.25	0.68
		d	0.03	0.01	0.14	0.01	0.02	0.02	0.27	0.27	0.86
	8	1	0.03	0.02	0.16	0.01	0.02	0.03	0.32	0.34	0.84
		$\sqrt{d}$	0.03	0.02	0.16	0.01	0.03	0.03	0.35	0.36	0.97
		d	0.04	0.03	0.21	0.01	0.03	0.03	0.42	0.45	1.5
	∞	1	0.04	0.02	0.18	0.01	0.03	0.03	0.23	0.57	1.2
		$\sqrt{d}$	0.04	0.02	0.17	0.01	0.03	0.03	0.27	0.57	1.4
		d	0.05	0.02	0.22	0.01	0.03	0.03	0.31	0.72	2.1
100	2	1	0.04	0.03	0.22	0.02	0.03	0.04	0.43	0.45	1.3
		$\sqrt{d}$	0.04	0.03	0.24	0.02	0.04	0.04	0.46	0.48	1.4
		d	0.05	0.04	0.33	0.02	0.04	0.04	0.56	0.56	2.1
	8	1	0.07	0.08	0.43	0.02	0.06	0.07	0.77	0.79	2.1
		$\sqrt{d}$	0.08	0.09	0.43	0.02	0.06	0.07	0.85	0.87	2.6
		d	0.10	0.10	0.57	0.02	0.07	0.07	1.1	1.2	4.7
	∞	1	0.09	0.10	0.50	0.03	0.06	0.09	1.3	1.6	3.6
		$\sqrt{d}$	0.10	0.11	0.46	0.02	0.06	0.09	1.4	1.6	4.2
		d	0.11	0.12	0.60	0.03	0.07	0.09	1.7	2.2	7.4
500	2	1	0.11	0.08	0.67	0.04	0.10	0.10	1.2	1.4	4.3
		$\sqrt{d}$	0.12	0.09	0.75	0.04	0.10	0.11	1.3	1.5	5.0
		d	0.14	0.11	1.1	0.05	0.12	0.12	1.7	1.9	8.2
	8	1	0.25	0.27	1.6	0.08	0.20	0.22	2.7	3.0	8.4
		$\sqrt{d}$	0.27	0.30	1.6	0.08	0.20	0.22	3.1	3.4	11
		d	0.33	0.37	2.2	0.08	0.22	0.25	4.2	5.0	21
	∞	1	0.32	0.36	1.9	0.08	0.22	0.30	4.9	6.8	15
		$\sqrt{d}$	0.33	0.38	1.7	0.08	0.21	0.29	5.3	7.0	18
		d	0.38	0.44	2.3	0.08	0.22	0.31	6.7	9.7	33

Table 9: Average adjusted Rand indices of KCP (Arlot et al., 2019) for different tuning parameters over 500 simulations, with standard deviations in parentheses. Optimal scores and those within two standard deviations are marked in bold. Note that for all nonparametric simulation setups, the covariates have been normalized by the median absolute deviation of absolute consecutive differences, such that the reported bandwidths are relative to the scale of the covariates. For the bandwidth *median*, the bandwidth was chosen according to the median heuristic (Garreau et al., 2017). Here, differently from the normalization based on consecutive differences, the median is taken over all possible pairwise distances and thus mostly contains differences across different distributions. As a consequence, the bandwidths selected by the median heuristic tend to be large (greater than 1). The oracle is the best performing hyperparameter combination. We used a Gaussian kernel with bandwidth 0.1 in the main simulations.

kernel	bandwidth	CIM	CIC	Dirichlet	iris	glass
cosine		0.80 (0.17)	0.36 (0.25)	0.86 (0.08)	0.03 (0.00)	0.03 (0.00)
linear		0.99 (0.03)	0.34 (0.28)	**0.87** (0.08)	0.85 (0.16)	0.37 (0.15)
Gaussian	0.025	0.99 (0.03)	0.38 (0.36)	0.80 (0.11)	0.94 (0.11)	**0.83** (0.16)
Gaussian	0.05	1.00 (0.02)	0.73 (0.33)	**0.87** (0.08)	0.98 (0.06)	**0.83** (0.17)
Gaussian	0.1	1.00 (0.01)	0.95 (0.06)	**0.87** (0.08)	0.99 (0.02)	0.74 (0.23)
Gaussian	0.2	1.00 (0.00)	0.98 (0.03)	**0.87** (0.08)	**1.00** (0.01)	0.59 (0.26)
Gaussian	0.4	1.00 (0.00)	**0.98** (0.02)	**0.87** (0.08)	0.99 (0.02)	0.46 (0.28)
Gaussian	0.8	**1.00** (0.00)	0.97 (0.09)	**0.87** (0.08)	0.83 (0.22)	0.15 (0.25)
Gaussian	median	0.00 (0.00)	0.00 (0.00)	**0.87** (0.08)	0.00 (0.00)	0.00 (0.00)
oracle		1.00 (0.00)	0.98 (0.02)	0.87 (0.08)	1.00 (0.01)	0.83 (0.17)

kernel	bandwidth	breast cancer	abalone	wine	dry beans	average
cosine		0.04 (0.01)	0.39 (0.04)	0.16 (0.02)	0.79 (0.08)	0.38 (0.11)
linear		0.97 (0.05)	0.84 (0.06)	0.63 (0.16)	0.96 (0.02)	0.76 (0.14)
Gaussian	0.025	**1.00** (0.00)	0.89 (0.05)	0.99 (0.02)	**1.00** (0.00)	0.87 (0.14)
Gaussian	0.05	**1.00** (0.00)	**0.90** (0.05)	**0.99** (0.01)	**1.00** (0.00)	0.92 (0.13)
Gaussian	0.1	1.00 (0.03)	**0.92** (0.04)	**0.99** (0.01)	**1.00** (0.00)	**0.94** (0.09)
Gaussian	0.2	0.99 (0.06)	0.92 (0.04)	**0.99** (0.01)	**1.00** (0.00)	0.93 (0.09)
Gaussian	0.4	0.99 (0.05)	0.88 (0.08)	0.97 (0.03)	**1.00** (0.00)	0.91 (0.10)
Gaussian	0.8	0.99 (0.05)	0.82 (0.10)	0.92 (0.03)	0.97 (0.04)	0.84 (0.12)
Gaussian	median	0.98 (0.06)	0.61 (0.13)	0.00 (0.03)	0.00 (0.00)	0.27 (0.06)
oracle		1.00 (0.00)	0.92 (0.04)	0.99 (0.01)	1.00 (0.00)	0.95 (0.07)

References

Edgar Anderson. The species problem in Iris. *Annals of the Missouri Botanical Garden*, 23 (3):457–509, 1936.

Sylvain Arlot, Alain Celisse, and Zaid Harchaoui. A kernel multiple change-point algorithm via model selection. *Journal of Machine Learning Research*, 20(162):1–56, 2019.

Sunil Arya, David M. Mount, Nathan S. Netanyahu, Ruth Silverman, and Angela Y. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM*, 45(6):891–923, 1998.

Herman J. Bierens. Uniform consistency of kernel estimators of a regression function under generalized conditions. *Journal of the American Statistical Association*, 78(383):699–707, 1983.

Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

Jedelyn Cabrieto, Francis Tuerlinckx, Peter Kuppens, Mariel Grassmann, and Eva Ceulemans. Detecting correlation changes in multivariate time series: A comparison of four non-parametric change point detection methods. *Behavior Research Methods*, 49(3):988–1005, 2017.

Edward Carlstein. Nonparametric change-point estimation. *The Annals of Statistics*, 16(1): 188–197, 1988.

Wei-Cheng Chang, Chun-Liang Li, Yiming Yang, and Barnabás Póczos. Kernel change-point detection with auxiliary deep generative models. In *International Conference on Learning Representations*, 2019.

Hao Chen and Nancy Zhang. Graph-based change-point detection. *The Annals of Statistics*, 43(1):139–176, 2015.

Alice Cleynen and Émilie Lebarbier. Segmentation of the Poisson and negative binomial rate models: a penalized estimator. *ESAIM: Probability and Statistics*, 18:750–769, 2014.

conda-forge community. The conda-forge Project: Community-based Software Distribution Built on the conda Package Format and Ecosystem, July 2015.

Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos, and José Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision support systems*, 47(4):547–553, 2009.

Lutz Dümbgen. The asymptotic behavior of some nonparametric change-point estimators. *The Annals of Statistics*, 19(3):1471–1495, 1991.

Farida Enikeeva and Zaid Harchaoui. High-dimensional change-point detection under sparse alternatives. *The Annals of Statistics*, 47(4):2051–2079, 2019.

Ian W. Evett and Ernest J. Spiehler. Rule induction in forensic science. In *Knowledge Based Systems*, pages 152–160. Online Publications, 1989.

Klaus Frick, Axel Munk, and Hannes Sieling. Multiscale change point inference. *Journal of the Royal Statistical Society: Series B*, 76(3):495–580, 2014.

Jerome Friedman. On multivariate goodness-of-fit and two-sample testing. Technical report, Stanford University, 2004.

Piotr Fryzlewicz. Wild binary segmentation for multiple change-point detection. *The Annals of Statistics*, 42(6):2243–2281, 2014.

Damien Garreau and Sylvain Arlot. Consistent change-point detection with kernels. *Electronic Journal of Statistics*, 12(2):4440–4486, 2018.

Damien Garreau, Wittawat Jitkrittum, and Motonobu Kanagawa. Large sample analysis of the median heuristic. *arXiv preprint arXiv:1707.07269*, 2017.

Simon Hediger, Loris Michel, and Jeffrey Näf. On the use of random forest for two-sample testing. *Computational Statistics & Data Analysis*, 170:107435, 2022.

Thomas Hotz, Ole Mathis Schütte, Hannes Sieling, Tatjana Polupanow, Ulf Diederichsen, Claudia Steinem, and Axel Munk. Idealizing ion channel recordings by a jump segmentation multiresolution filter. *IEEE Transactions on NanoBioscience*, 12(4):376–386, 2013.

Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2(1): 193–218, 1985.

Nicholas James and David Matteson. ecp: An R package for nonparametric multiple change point analysis of multivariate data. *Journal of Statistical Software*, 62(7):1–25, 2015.

Alexander Kaplan, Joachim Röschke, Boris Darkhovsky, and Juergen Fell. Macrostructural EEG characterization based on nonparametric change point segmentation: Application to sleep analysis. *Journal of Neuroscience Methods*, 106(1):81–90, 2001.

Abhishek Kaul, Venkata K. Jandhyala, and Stergios B. Fotopoulos. An efficient two step algorithm for high dimensional change point regression models without grid search. *Journal of Machine Learning Research*, 20:1–40, 2019.

Rebecca Killick, Paul Fearnhead, and Idris A. Eckley. Optimal detection of changepoints with a linear computational cost. *Journal of the American Statistical Association*, 107 (500):1590–1598, 2012.

Chang-Jin Kim, James C. Morley, and Charles R. Nelson. The structural break in the equity premium. *Journal of Business & Economic Statistics*, 23(2):181–191, 2005.

Hyune-Ju Kim and David Siegmund. The likelihood ratio test for a change-point in simple linear regression. *Biometrika*, 76(3):409–423, 1989.

Murat Koklu and Ilker Ali Ozkan. Multiclass classification of dry beans using computer vision and machine learning techniques. *Computers and Electronics in Agriculture*, 174: 105507, 2020.

Solt Kovács, Housen Li, and Peter Bühlmann. Seeded intervals and noise level estimation in change point detection: A discussion of Fryzlewicz (2020). *Journal of the Korean Statistical Society*, 49(4):1081–1089, 2020a.

Solt Kovács, Housen Li, Lorenz Haubner, Axel Munk, and Peter Bühlmann. Optimistic search: Change point estimation for large-scale data via adaptive logarithmic queries. *arXiv:2010.10194*, 2020b.

Solt Kovács, Peter Bühlmann, Housen Li, and Axel Munk. Seeded binary segmentation: a general methodology for fast and optimal changepoint detection. *Biometrika*, 110(1): 249–256, 2023.

Nadia L Kudraszow and Philippe Vieu. Uniform consistency of kNN regressors for functional variables. *Statistics & Probability Letters*, 83(8):1863–1870, 2013.

Song Liu, Makoto Yamada, Nigel Collier, and Masashi Sugiyama. Change-point detection in time-series data by relative density-ratio estimation. *Neural Networks*, 43:72–83, 2013.

Malte Londschien, Solt Kovács, and Peter Bühlmann. Change-point detection for graphical models in the presence of missing values. *Journal of Computational and Graphical Statistics*, 30(3):768–779, 2021.

David Lopez-Paz and Maxime Oquab. Revisiting classifier two-sample tests. In *International Conference on Learning Representations*, 2017.

Alexandre Lung-Yut-Fong, Céline Lévy-Leduc, and Olivier Cappé. Homogeneity and change-point detection tests for multivariate data using rank statistics. *Journal de la Société Française de Statistique*, 156(4):133–162, 2015.

Oscar–Hernan Madrid–Padilla, Yi Yu, Daren Wang, and Alessandro Rinaldo. Optimal nonparametric change point analysis. *Electronic Journal of Statistics*, 15(1):1154–1201, 2021a.

Oscar–Hernan Madrid–Padilla, Yi Yu, Daren Wang, and Alessandro Rinaldo. Optimal nonparametric multivariate change point detection and localization. *IEEE Transactions on Information Theory*, 68(3):1922–1944, 2021b.

Nicholas D. Matsakis and Felix S. Klock. The rust language. In *ACM SIGAda Ada Letters*, volume 34, pages 103–104. ACM, 2014.

David S. Matteson and Nicholas A. James. A nonparametric approach for multiple change point analysis of multivariate data. *Journal of the American Statistical Association*, 109 (505):334–345, 2014.

Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4):115–133, 1943.

Adam B. Olshen, E. Seshan Venkatraman, Robert Lucito, and Michael Wigler. Circular binary segmentation for the analysis of array-based DNA copy number data. *Biostatistics*, 5(4):557–572, 2004.

Ewan S. Page. Continuous inspection schemes. *Biometrika*, 41(1/2):100–115, 1954.

Ewan S. Page. A test for a change in a parameter occurring at an unknown point. *Biometrika*, 42(3/4):523–527, 1955.

Florian Pein, Hannes Sieling, and Axel Munk. Heterogeneous change point inference. *Journal of the Royal Statistical Society: Series B*, 79(4):1207–1227, 2017.

Anthony N. Pettitt. A non-parametric approach to the change-point problem. *Journal of the Royal Statistical Society: Series C*, 28(2):126–135, 1979.

Franck Picard, Stephane Robin, Marc Lavielle, Christian Vaisse, and Jean-Jacques Daudin. A statistical approach for array CGH data analysis. *BMC Bioinformatics*, 6(1):1–14, 2005.

William M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

Jaxk Reeves, Jien Chen, Xiaolan L. Wang, Robert Lund, and Qi Qi Lu. A review and comparison of changepoint detection techniques for climate data. *Journal of Applied Meteorology and Climatology*, 46(6):900–915, 2007.

Sandipan Roy, Yves Atchadé, and George Michailidis. Change point estimation in high dimensional Markov random-field models. *Journal of the Royal Statistical Society: Series B*, 79(4):1187–1206, 2017.

Jeremy J. Shen and Nancy R. Zhang. Change-point model on nonhomogeneous Poisson processes with application in copy number profiling by next-generation DNA sequencing. *The Annals of Applied Statistics*, 6(2):476–496, 2012.

Nick Street, William H. Wolberg, and Olvi L. Mangasarian. Nuclear feature extraction for breast tumor diagnosis. In *Biomedical Image Processing and Biomedical Visualization*, volume 1905. SPIE, 1993.

Charles Truong, Laurent Oudre, and Nicolas Vayatis. Selective review of offline change point detection methods. *Signal Processing*, 167:107299, 2020.

Lyudmila Y. Vostrikova. Detecting 'disorder' in multidimensional random processes. *Soviet Mathematics Doklady*, 24:55–59, 1981.

Daren Wang, Yi Yu, and Alessandro Rinaldo. Optimal covariance change point localization in high dimensions. *Bernoulli*, 27(1):554–575, 2021.

Tengyao Wang and Richard J. Samworth. High dimensional change point estimation via sparse projection. *Journal of the Royal Statistical Society: Series B*, 80(1):57–83, 2018.

Samuel G. Waugh. *Extending and benchmarking Cascade-Correlation: extensions to the Cascade-Correlation architecture and benchmarking of feed-forward supervised artificial neural networks*. PhD thesis, University of Tasmania, 1995.

Yi-Ching Yao. Estimating the number of change-points via Schwarz' criterion. *Statistics & Probability Letters*, 6(3):181–189, 1988.

Yuxuan Zhang and Hao Chen. Graph-based multiple change-point detection. *arXiv:2110.01170*, 2021.

Changliang Zou, Guosheng Yin, Long Feng, and Zhaojun Wang. Nonparametric maximum likelihood approach to multiple change-point problems. *The Annals of Statistics*, 42(3): 970–1002, 2014.