Detecting possibly frequent change-points: Wild Binary Segmentation 2 and steepest-drop model selection

Piotr Fryzlewicz*

February 24, 2020

Abstract

Many existing procedures for detecting multiple change-points in data sequences fail in frequent-change-point scenarios. This article proposes a new change-point detection methodology designed to work well in both infrequent and frequent change-point settings. It is made up of two ingredients: one is "Wild Binary Segmentation 2" (WBS2), a recursive algorithm for producing what we call a 'complete' solution path to the change-point detection problem, i.e. a sequence of estimated nested models containing $0, \ldots, T-1$ change-points, where T is the data length. The other ingredient is a new model selection procedure, referred to as "Steepest Drop to Low Levels" (SDLL). The SDLL criterion acts on the WBS2 solution path, and, unlike many existing model selection procedures for change-point problems, it is not penalty-based, and only uses thresholding as a certain discrete secondary check. The resulting WBS2.SDLL procedure, combining both ingredients, is shown to be consistent, and to significantly outperform the competition in the frequent change-point scenarios tested. WBS2.SDLL is fast, easy to code and does not require the choice of a window or span parameter.

Key words: Segmentation, break detection, jump detection, randomized algorithms, adaptive algorithms, multiscale methods.

1 Introduction

A recent report (National Research Council, 2013) lists change-point detection as one of the "inferential giants" (common building blocks in knowledge discovery) in massive data analysis. Change-point analysis, be it a posteriori or online, is used in fields as diverse as bioinformatics (Olshen et al., 2004), econometrics and finance (Hansen, 2001; Andreou and Ghysels, 2002; Bai and Perron, 2003), climate and the natural environment (Reeves et al., 2007; Liu et al., 2010; Salarijazi et al., 2012; Pezzatti et al., 2013), autonomous driving (Galceran et al., 2015), computer vision (Ranganathan, 2012), clinical neuroscience

*Department of Statistics, London School of Economics, Houghton Street, London WC2A 2AE, UK. Email: p.fryzlewicz@lse.ac.uk. Work supported by the Engineering and Physical Sciences Research Council grant no. EP/L014246/1.

(Younes et al., 2014) and quality/reliability monitoring (D'Angelo et al., 2011; Huang and Lyu, 2011; Amiri and Allahyari, 2012).

A common testing ground for a posteriori change-point detection methods, and the focus of this paper, is the model

$$X_t = f_t + \varepsilon_t, \quad t = 1, \ldots, T, \tag{1}$$

in which f_t is a deterministic, one-dimensional, piecewise-constant signal with change-points whose number N and locations $\eta_1, \ldots, \eta_N$ are unknown. The sequence ε_t is random and such that $\mathbb{E}(\varepsilon_t)$ is exactly or approximately zero. In the simplest case ε_t are modelled as i.i.d. $N(0, \sigma^2)$, but can also have other marginal distributions and/or exhibit temporal dependence. The task is to estimate N and $\eta_1, \ldots, \eta_N$ under various assumptions on N, the magnitudes of the jumps and the minimum permitted distance between the change-point locations. This article focuses on the i.i.d. $N(0, \sigma^2)$ distribution for ε_t, as even in this simplest case, the above change-point detection problem still poses major challenges for the state of the art, as this article illustrates.

Although performance measures for the above estimation task, naturally, vary across the literature, it can be anticipated that many users will have a preference for methods which

1. accurately estimate the number N and locations η_i of any change-points present in f_t, for both
   (a) signals with few/infrequent change-points, and
   (b) signals with many/frequent change-points;
2. are fast to execute and scale well to long signals.

Depending on the setting, other desirable features may include e.g. the existence of software, ease of implementation and transferability to other, more complex stochastic models. Even though much methodological and algorithmic progress has been made on the above change-point detection problem in recent years, we show in this paper that very few (if any) state-of-the-art techniques simultaneously possess the above desired features 1(a), 1(b) and 2. In particular, many techniques perform surprisingly poorly for signals with frequent change-points. We first briefly describe the history of the multiple change-point detection problem and the state of the art. For reasons of space, our literature review mainly focuses on the parametric case, i.e. models in which the distribution of X_t can be modelled parametrically, with special emphasis on the i.i.d. Gaussian model for ε_t.

A major class of approaches to a posteriori multiple change-point detection is one in which change-points are found by minimising a criterion function consisting of a likelihood-type (or least-squares) term measuring the fit of the estimate to the data plus a penalty term to prevent overfitting. In this category, Yao and Au (1989) consider least-squares estimation of f_t for a fixed N and i.i.d. noise. Braun et al. (2000) extend this work to noise for which the variance is a function of the mean. In the Gaussian case, the SIC (BIC) is used to estimate an unknown but bounded N in Yao (1988), and a more general penalty but also linear in the number of change-points appears in Lee (1995). For an unknown but bounded N, Lavielle and Moulines (2000) consider penalised least-squares estimation, with a penalty linear in the number of change-points, for dependent ε_t's; see also Lavielle (1999) and Lavielle (2005), an article we return to later. For a fixed N, Pan and Chen (2006) propose a likelihood criterion with a penalty depending not only on the number, but

also on the locations of change-points, favouring more uniformly-spread estimated change-points; related ideas appear in Zhang and Siegmund (2007). For an unknown N, Lebarbier (2005) proposes least-squares estimation with a penalty originating from the model selection approach of Birgé and Massart (2001); we revisit this approach below. Boysen et al. (2009) use the least-squares criterion with a linear penalty on the number of change-points. More general forms of Schwarz-like penalties are studied, for example, in Wu (2008), Ciuperca (2011) and Ciuperca (2014). The SMUCE estimator of Frick et al. (2014) can also be cast in the penalised cost function framework (while SMUCE is shown to control the FWER, the related method of Li and Munk (2016) controls the FDR). An empirical Bayes approach to change-point estimation in a marginal likelihood framework appears in Du et al. (2016). The PELT algorithm by Killick et al. (2012) and the pruned dynamic programming by Rigaill (2015) accelerate computation of some penalised change-point estimators to linear time in best-case scenarios (while retaining quadratic speed in worst-case ones), and offer fast implementations. Related ideas appear also in Maidstone et al. (2017).

Other attempts to reduce the computational complexity of the problem include Davis et al. (2006), who (in a time series setting) use a genetic algorithm to minimise a Minimum Description Length criterion, and Harchaoui and Lévy-Leduc (2010) who consider the least-squares criterion with a total variation penalty, which enables them to use the LARS algorithm of Efron et al. (2004) to compute the solution in $O(NT\log(T))$ time. However, the total variation penalty is not an optimal one for change-point detection (in the sense of balancing out type-I and type-II errors, as described in Brodsky and Darkhovsky (1993) and Cho and Fryzlewicz (2011)). The total variation penalty is also considered in the context of peak/trough detection by Davies and Kovac (2001), who propose the 'taut string' approach for fast computation. In the context of multiple change-point detection, it is considered by Rinaldo (2009) (with a subsequent correction published on the author's web page) and Rojas and Wahlberg (2014) as part of the fused lasso penalty, proposed by Tibshirani et al. (2005) and equivalent to taut string in model (1). Lin et al. (2017) propose a generic post-processing procedure for any L_2-consistent, piecewise-constant estimator of f_t, including in particular the total-variation-based one, which yields consistency for the estimators of η_i under certain assumptions on the distances between the change-points and the jump sizes. Similar results for general "trend filtering" (Tibshirani, 2014) estimators appear in Guntuboyina et al. (2020).

Our experience is that many penalty-based methods in which the penalty has been pre-set by the user (rather than having been chosen adaptively from the data) can struggle to offer uniformly good performance across both signals with infrequent change-points, and signals with frequent/numerous ones; in particular, we illustrate in Section 2.1 that the popular BIC (Yao, 1988) and mBIC (Zhang and Siegmund, 2007) penalties can perform poorly for signals with frequent change-points. An interesting approach to data-driven penalty selection is proposed in Birgé and Massart (2001) and Birgé and Massart (2007), who define the so-called minimal penalty; two approaches to estimating this quantity, referred to as 'dimension jump' and 'data-driven slope estimation' are described in Baudry et al. (2012). Both offer impressive performance for short signals with a limited number of jumps, but their performance degrades for signals with frequent jumps (more so for the dimension jump approach). There is also computational price to pay for selecting the penalty in a data-driven way: the 'data-driven slope estimation' approach is particularly slow for long signals. We illustrate these points in Section 4.2. Lavielle (2005), motivated by the problem of selecting a suitable penalty constant from the data, proposes a heuristic algorithm for estimating the

number of change-points, which examines the second differential of the empirical loss of the piecewise-constant model fit as a function of the number of change-points. The approach requires the provision of the maximum number of change-points and a threshold parameter whose value appears critical to the success of the procedure; the theoretical properties of the method are not investigated. We review these and some related approaches to adaptive penalty choice in Section 3.3 in more detail.

Another class of approaches to the multiple change-point detection problem is one in which change-points are estimated one by one, in the order in which they appear in the data. Huskova and Slaby (2001) and Eichinger and Kirch (2018) propose the "moving sum" (MOSUM) technique, which requires the choice of an extra bandwidth parameter, but the latter requirement is circumvented in the **mosum** R package (Meier et al., 2018), which produces estimates combined over a range of automatically chosen bandwidths. The pseudo-sequential procedure of Venkatraman (1992), as well as the method of Ross (2015) are based on an adaptation of online detection algorithms to a posteriori situations and work by bounding the Type I error rate of falsely detecting change-points. The 'Isolate-Detect' method of Anastasiou and Fryzlewicz (2018a) is also pseudo-sequential in nature, avoids to some extent the issue of bandwidth/span selection, and offers particularly fast implementation for signals with frequent change-points. However, from the performance point of view, such signals tend to pose major difficulties for many methods in this category, an issue magnified by the fact that many such techniques rely critically on the accurate estimation of a threshold against which to judge the significance of each change-point candidate, but the magnitude of this threshold can be difficult to estimate well in the presence of many change-points. We demonstrate this phenomenon in Section 2.3.2.

Binary segmentation is one of the most popular approaches to multiple change-point detection. Heuristically, it works by fitting the best model with a single change-point to data X_t, and if the estimated change-point is deemed to be significant, the same procedure is then repeated to the left and to the right of the estimate. Binary segmentation is fast, conceptually simple and easy to implement, but tends not to work well for signals with frequent change-points, the main reason being that for such signals there are no guarantees that the single-change-point model fit at each stage of the procedure will behave well in the presence of multiple change-points in the data segment considered. Possibly the first work to propose binary segmentation in a stochastic process setting is Vostrikova (1981), who shows consistency of binary segmentation for the number and locations of change-points for a fixed N. Venkatraman (1992) offers a proof of the consistency of binary segmentation for N and the change-point locations, even for N increasing with T, albeit with sub-optimal assumptions and rates for the locations. In a setting similar to Vostrikova (1981) (for a fixed N and with ε_t following a linear process), binary segmentation is considered in Bai (1997). Chen et al. (2011) provide a proof of consistency of binary segmentation for the number of change-points in the case of a fixed N and ε_t i.i.d. Gaussian. Binary segmentation is used for univariate time series segmentation in Fryzlewicz and Subba Rao (2014) and Cho and Fryzlewicz (2012), and for multivariate, possibly high-dimensional time series segmentation in Cho and Fryzlewicz (2015). Circular Binary Segmentation (Olshen et al., 2004; Venkatraman and Olshen, 2007), Wild Binary Segmentation (Fryzlewicz, 2014) and the Narrowest-Over-Threshold method (Baranowski et al., 2019) are designed to improve the performance of binary segmentation. We return to Wild Binary Segmentation later in this section.

A number of techniques do not directly fall into any of the above categories. Wang (1995) uses the fast discrete wavelet transform to detect change-points. Muggeo and Adelfio (2011) take the cumulative sum of X_t and then apply the methodology described in Muggeo (2003) in the resulting piecewise-linear framework. An agglomerative, bottom-up, "tail-greedy" method is proposed in Fryzlewicz (2018). Matteson and James (2014) present a heuristic agglomerative algorithm for multiple change-point detection as a computationally attractive alternative to their divisive one. An early review of some multiple change-point detection methods (in the context of DNA segmentation, but applicable more widely) appears in Braun and Mueller (1998).

Wild Binary Segmentation (WBS, Fryzlewicz, 2014) is one of the state-of-the-art procedures for multiple change-point detection in model (1), for signals with a small or moderate number of change-points. Truong et al. (2020) give WBS "three stars" (their highest grade) for scalability to long signals, and Wang et al. (2018) show certain assumption- and error-rate-optimality of WBS. WBS is designed to deal with the shortcomings of binary segmentations (described above) in the following manner. At the start of the procedure, M subsamples $[X_s, \ldots, X_e]$ of the data, living on intervals $[s, e]$, are drawn, with the start- and end-points s and e selected randomly, independently, uniformly and with replacement. The single-change-point model fit is then performed on each subsample, and the first change-point candidate is selected as the one corresponding to the best fit over the M subsamples. The hope is that if M is sufficiently large, then this best fit will come from a subsample containing at most one change-point and will therefore lead to the accurate estimation of the given change-point. If the first detected change-point is considered significant, the same procedure is repeated to the left and to the right of it, re-using (for speed) the previously drawn subsamples.

Notwithstanding this improvement over plain binary segmentation, we show in Section 2.1 that the performance of WBS deteriorates dramatically for signals with frequent change-points (as does that of many other state-of-the-art methods). The roots of its poor performance for such signals are multiple, and can be summarised as follows.

1. The recommended default number M of subsamples drawn by the WBS (the R packages **breakfast** (Fryzlewicz, 2017) and **wbs** (Baranowski and Fryzlewicz, 2015) recommend M not exceeding 20000) can be insufficient to ensure adequate performance of WBS for signals with frequent change-points. Indeed, good performance can only be hoped for if M is large enough for the drawn subsamples to include (with a high probability) ones with start- and end-points immediately to the left and to the right of each change-point; this means that the required M can easily be very large for signals with frequent change-points. Increasing the default M to ensure this is not a viable solution as this would increase the computation time beyond what is acceptable (and is not needed for signals with no or few change-points). In the remainder of this article, we refer to this issue as the *lack of computational adaptivity* of the WBS, in the sense that the procedure does not decide automatically the number M or the locations of the subsamples drawn.

2. A related point is that for WBS to be able to produce an entire solution path to problem (1) (i.e. estimated models with $0, 1, \ldots, T-1$ change-points), we would have to draw all possible subsamples of the data. This would result in a procedure of a cubic computational complexity in T, which is computationally inadmissible.

Any value of M less than this upper limit would not guarantee the computability of the entire solution path. If a solution path algorithm in multiple change-point detection problems does not produce an entire solution path, we refer to this algorithm as *incomplete*. The use of incomplete solution path algorithms is particularly risky in frequent change-point scenarios, as it may lead to obtaining a partial solution path that is shorter than the true number of change-points, in which case any model selection criterion, however good, will not be able to recover all the change-points. We illustrate this important phenomenon in Section 2.3.1.

3. Fryzlewicz (2014) proposes two model selection criteria for the WBS: one based on thresholding, and the other based on the use of the sSIC (a penalty close to the standard BIC/SIC). Both approaches fail in frequent change-point scenarios. Thresholding does so because its success critically depends on the user's ability to accurately estimate the variance of the noise ε_t, but this task can be difficult in frequent change-point settings, which we show in Section 2.3.2. BIC fails because it places too heavy a penalty on frequent-change-point solutions to the estimation problem 1. The failure of BIC in frequent change-point scenarios occurs for other change-point search algorithms too (see Section 2.1), not just WBS.

The objective of this paper is to introduce the "Wild Binary Segmentation 2" (WBS2) solution path algorithm for multiple change-point problems, and the "Steepest Drop to Low Levels" (SDLL) model selection for WBS2. The resulting multiple change-point detection method, labelled WBS2.SDLL, addresses the above shortcomings of WBS in two separate ways.

Firstly, WBS2 produces an entire solution path, i.e. sequences of solutions to the change-point detection problem with $0, 1, \ldots, T-1$ change-points. Among them, WBS2 guarantees (with probability approaching one with T) that the solution with the true number N of change-points is such that the estimated change-point locations $\tilde{\eta}_1, \ldots, \tilde{\eta}_N$ are near-optimally close to the true change-points $\eta_1, \ldots, \eta_N$. Because it produces the entire solution path, the WBS2 is (unlike WBS), by defintion, a complete procedure, and its completeness is achieved in a computationally efficient way as follows. In the first pass through the data, WBS2 draws only a small number $\tilde{M}$ of data subsamples, and marks the first change-point candidate as the arg-max of the resulting $\tilde{M}$ absolute CUSUMs. It then uses this change-point candidate to split its domain of operation into two, and again recursively draws $\tilde{M}$ subsamples to the left and to the right of this change-point candidate, and so on. Therefore, WBS2 adaptively decides where to recursively draw the next subsamples, based on the change-point candidates detected so far (to re-express this is the language of machine learning, WBS2 learns the locations of the subsamples to draw automatically and "on the fly", i.e. as it proceeds through the data and discovers the signal). This is in contrast to the standard WBS, which (non-adaptively) pre-draws all its M subsamples at the start of the procedure. We note that in practice, $\tilde{M}$ is of orders of magnitude smaller than M; in the numerical sections of this paper we will be using $\tilde{M} = 100$. WBS2 draws subsamples on every current subdomain of the data as long as its length is ≥ 2, which ensures completeness. By contrast, WBS (non-adaptively) pre-draws all its M subsamples at the start of the procedure, which leads to its incompleteness except if all possible subsamples are drawn, which is computationally infeasible for even moderately long signals.

Secondly, the proposed SDLL procedure for estimating N from the WBS2 solution path does not use a penalty, and only uses thresholding as a certain secondary check, which

is in contrast to the use of thresholding as a primary model selection tool in WBS. This means that the magnitude of the threshold in WBS2.SDLL does not need to be estimated to a high degree of accuracy (which is difficult to do in frequent change-point scenarios). SDLL works by monitoring the ratios of the the pairs of consecutive maximum absolute CUSUM statistics returned by the WBS2 solution path, arranged in decreasing order. The procedure then selects the largest ratio for which the denominator falls under a pre-specified threshold. This is motivated by the fact that those absolute maximised CUSUMs that carry information about the N change-points have the same order of magnitude between them, which is larger than the order of magnitude of those CUSUMs that correspond to the no-change-points sections of the data. The presence of this "drop" in the magnitude of the absolute maximised CUSUMs gives the SDLL criterion its name.

Our numerical experiments suggest that unlike the standard WBS, the resulting WBS2.SDLL procedure offers good detection accuracy and fast execution times for signals with frequent change-points. We demonstrate that in doing so, WBS2.SDLL significantly outperforms the state of the art for this signal class. Unlike many of its competitors, WBS2.SDLL does not require the choice of a penalty or the specification of the maximum number of change-points, and is easy to calibrate to return zero estimated change-points for constant signals with a required probability. Both the WBS2 and the SDLL components are conceptually simple and easy to code.

The paper is organised as follows. In Section 2, we illustrate the poor performance of the state of the art for signals with frequent change-points, and we investigate the reasons for WBS's poor performance in frequent change-point scenarios. Section 3 motivates and proposes WBS2 and SDLL and investigates their theoretical properties, as well as offering a deeper comparison of WBS2.SDLL to the existing literature. Our practical recommendations and a comparative simulation study are in Section 4. Section 5 describes the application of WBS2.SDLL to a London house price dataset. The proofs of our theoretical results are in the Appendix.

2 Motivation

2.1 Frequent change-points: performance of the state of the art

To motivate the need for our new methodology, we start by illustrating the poor performance of WBS and other state-of-the-art multiple change-point detection methods for signals with frequent change-points, using the example of the `extreme.teeth` signal defined below. We use this signal as a running example throughout the paper to motivate several of our developments, which are then however shown to hold for a wide class of signals both in theory and in practice. Throughout the paper, we work with methods available from the R CRAN repository, and for each method tested, we use the default parameter settings unless stated otherwise. For techniques that require the specification of the maximum number of change-points, we set this to the (rounded) length of the input signal divided by three; in all our examples, the true number of change-points is (well) within the interior of this range. Our R code enabling replication of the results from this paper is available from `https://github.com/pfryz/wild-binary-segmentation-2.0`. We now list the techniques tested in addition to our new WBS2.SDLL methodology.

1. PELT+BIC and PELT+mBIC: the PELT technique from the R package **changepoint** (Killick et al., 2016) with the BIC and mBIC penalties, respectively. The methodology is described in Killick et al. (2012).

2. MOSUM: the multiscale MOSUM technique with localised pruning from the R package **mosum** (Meier et al., 2018). (We do not give results for the multiscale MOSUM technique with bottom-up bandwidth-based merging from the same package as it frequently returned warnings in the examples we tested.)

3. ID: the 'Isolate-Detect' technique from the R package **IDetect** (Anastasiou and Fryzlewicz, 2018b), described in Anastasiou and Fryzlewicz (2018a).

4. FDRSeg: the multiscale FDR-based technique from the R package **FDRSeg** (Li and Sieling, 2017), proposed in Li and Munk (2016).

5. S3IB: the technique from the R package **Segmentor3IsBack** (Cleynen et al., 2016). It implements a fast algorithm for minimising the least-squares cost function for change-point detection, as described in Rigaill (2015). The best model is selected via the "oracle" criterion of Lebarbier (2005), which uses the so-called slope heuristics.

6. SMUCE: the multiscale FWER-based technique from the R package **stepR** (Pein et al., 2018), proposed in Frick et al. (2014).

7. CUMSEG: the linearisation-based technique from the R package **cumSeg** (Muggeo, 2012), described in Muggeo and Adelfio (2011).

8. FPOP: the 'functional pruning and optimal partitioning' technique from the R package **fpop** (Rigaill and Hocking, 2016) with the BIC penalty. The methodology is described in Maidstone et al. (2017).

9. DDSE and DJUMP: the "data-driven slope estimation" and "dimension jump" techniques from the R package **capushe** (Arlot et al., 2016), with computation based also on the R package **jointseg** (Pierre-Jean et al., 2017). The methodology is described in Baudry et al. (2012).

10. TGUH: the tail-greedy, bottom-up agglomerative technique from the R package **breakfast** (Fryzlewicz, 2017). The methodology is described in Fryzlewicz (2018).

11. WBS-C1.0, WBS-C1.3 and WBS-BIC: the Wild Binary Segmentation methods with the model selection determined via thresholding with constants $C = 1.0$ and $C = 1.3$ and the BIC (respectively), as specified in Fryzlewicz (2014) and implemented in the R package **breakfast** (Fryzlewicz, 2017). See Section 2.2 for their definitions.

We do not include the (non-parametric) methodology implemented in the R package **ecp** (James and Matteson, 2014) as this procedure does not appear to be able to return zero estimated change-points. We also do not provide results for the R package **strucchange** (Zeileis et al., 2002) because of the slow speed of the methodology implemented therein.

Our example in this section is the signal $f_t, t = 1, \ldots, T = 1000$, referred to as `extreme.teeth` and defined as follows: $f_t = 0$ if $1 \leq (t \bmod 10) \leq 5$; $f_t = 1$ if $(t \bmod 10) \in \{0, 6, 7, 8, 9\}$. We simulate 100 independent copies of X_t in model (1) with f_t being the `extreme.teeth`

signal and $\varepsilon_t \sim$ i.i.d. $N(0, 0.3^2)$. Note that N, the true number of change-points in f_t, equals 199. Figure 1 shows the first 100 observations of the noiseless and noisy f_t.

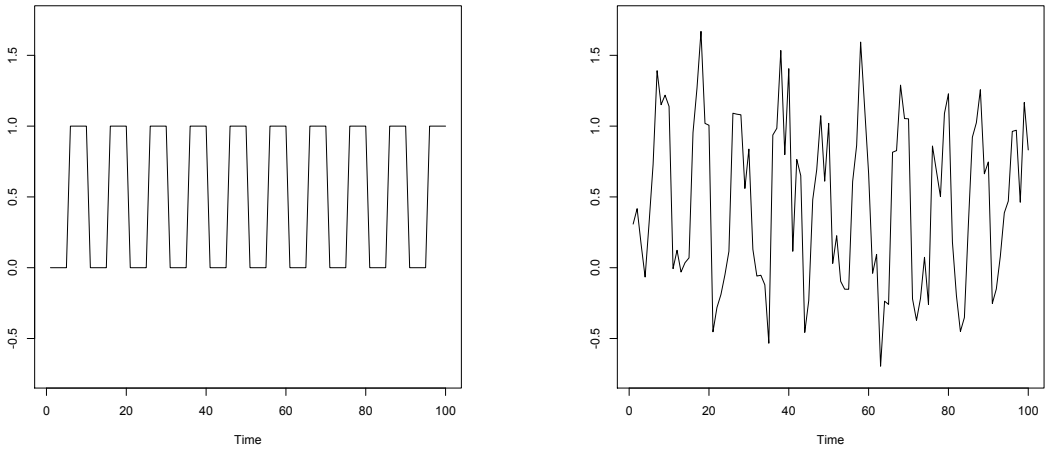

Figure 1: The first 100 observations of the `extreme.teeth` signal (which continues in the same manner for $t = 1, \ldots, 1000$); left: with no noise; right: with additive i.i.d. Gaussian noise with mean zero and $\sigma = 0.3$.

Figure 2 provides evidence that most of the techniques listed above fail somewhat dramatically for this model, with all but three being able to estimate practically none or only a small proportion of the change-points. The three best performers are DDSE, and two versions of our method, labelled WBS2.SDLL(0.9) and WBS2.SDLL(0.95) (we provide the details later). The DDSE, however, is over three times slower than WBS2.SDLL and estimates zero change-points for two out of the 100 sample paths, which results in $\hat{E}_{\mathrm{DDSE}}(\hat{N} - N)^2 \approx 800$ compared to $\hat{E}_{\mathrm{WBS2.SDLL}}(\hat{N} - N)^2 \approx 20$. We return to DDSE and the other techniques in the comparative simulation study in Section 4.2. WBS-C1.0, WBS-C1.3, WBS-BIC are among the failing techniques and we now examine the reasons for their failure.

2.2 Wild Binary Segmentation

We first review WBS as proposed in Fryzlewicz (2014). Throughout the paper, our main "locator" statistic for indicating the locations of change-point candidates is the CUSUM statistic, defined, for the data portion $(X_s, \ldots, X_e)$, as

$$\tilde{X}^b_{s,e} = \sqrt{\frac{e-b}{n(b-s+1)}} \sum_{t=s}^{b} X_t - \sqrt{\frac{b-s+1}{n(e-b)}} \sum_{t=b+1}^{e} X_t, \tag{2}$$

where $s \le b < e$, with $n = e - s + 1$. It is used in different ways in WBS and WBS2. In both these algorithms, we use $\arg\max_b |\tilde{X}^b_{s,e}|$ as the most likely location of a change-point in $(f_s, \ldots f_e)$. This is the same as the location indicated by fitting the best possible piecewise-constant function with one change-point to $(X_s, \ldots, X_e)$ via least squares. Therefore, the use of the CUSUM statistic is equivalent to the use of maximum likelihood in model (1) in which $\varepsilon_t \sim$ i.i.d. $N(0, \sigma^2)$.

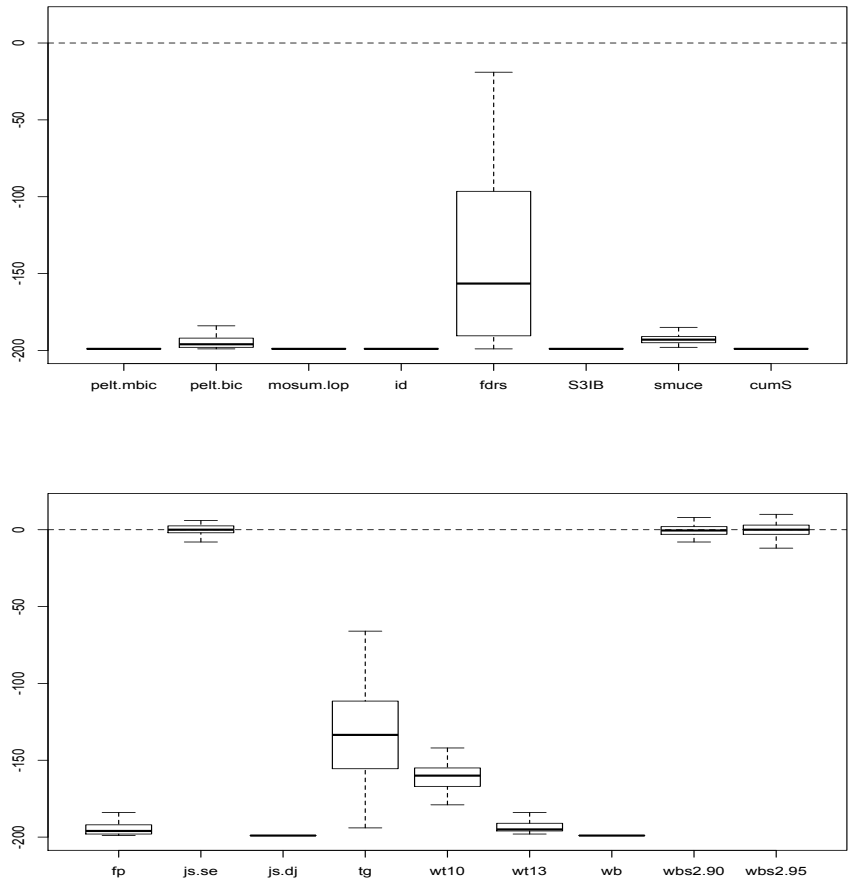

Figure 2: From top to bottom and left to right: boxplots of $\hat{N} - N$ over 100 simulated sample paths, for the noisy `extreme.teeth` signal, for the following techniques: PELT+mBIC, PELT+BIC, MOSUM, ID, FDRSeg, S3IB, SMUCE, CUMSEG, FPOP, DDSE, DJUMP, TGUH, WBS-C1.0, WBS-C1.3, WBS-BIC and two configurations of the WBS2 procedure proposed in this paper – WBS2.SDLL(0.9) and WBS2.SDLL(0.95). See Section 2.1.

Denote by F_T^M a set of M random intervals $[s_m, e_m]$, $m = 1, \ldots, M$, whose start- and end-points have been drawn (independently with replacement) uniformly from the set $\{1, \ldots, T\}$. How many intervals M to draw is decided by the user (Fryzlewicz (2014) and version 1.0.0 of the R package **breakfast** (Fryzlewicz, 2017) recommend $M = 5000$ and $M = 20000$, respectively, for signal lengths T not exceeding a few thousand). An appropriate choice of M is key: on the whole, the larger the value of M, the more accurate but slower the procedure. Using pseudocode, the main function of the WBS algorithm, with model selection performed via thresholding with threshold ζ_T, is defined in the following way.

function WBS(s, e, ζ_T)
  if $e - s < 1$ **then**
    STOP
  else

$\mathcal{M}_{s,e}$:= set of those indices m for which $[s_m, e_m] \in F_T^M$ is such that $[s_m, e_m] \subseteq [s, e]$
$(m_0, b_{m_0}) := \arg\max_{m \in \mathcal{M}_{s,e}, b \in \{s_m, \ldots, e_m - 1\}} |\tilde{X}^b_{s_m, e_m}|$
if $|\tilde{X}^{b_{m_0}}_{s_{m_0}, e_{m_0}}| \geq \zeta_T$ **then**
add b_{m_0} to the set of estimated change-points (with the additional side information $(m_0, s_{m_0}, e_{m_0}, |\tilde{X}^{b_{m_0}}_{s_{m_0}, e_{m_0}}|)$)
WBS(s, b_{m_0}, ζ_T)
WBS($b_{m_0} + 1$, e, ζ_T)
else
STOP
end if
end if
end function

The WBS procedure (with model selection via thresholding with threshold ζ_T) is launched by the call WBS(1, T, ζ_T). The WBS-C1.0 and WBS-C1.3 versions, singled out by Fryzlewicz (2014) and used in Section 2.1 above, use, respectively, $\zeta_T = \hat{\sigma}\{2 \log T\}^{1/2}$ and $\zeta_T = 1.3\,\hat{\sigma}\{2 \log T\}^{1/2}$, where $\hat{\sigma}$ is the Median Absolute Deviation (MAD) estimator of $\sigma = \mathrm{Var}^{1/2}(\varepsilon_t)$, computed using $\{2^{-1/2}(X_{t+1} - X_t)\}_{t=1}^{T-1}$ on input. An alternative implementation of WBS with thresholding first computes all possible change-point candidates via the call WBS(1, T, 0) and then tests them, as is done in the definition of the routine WBS($\cdot$, $\cdot$, $\cdot$), against the threshold ζ_T.

As an alternative to thresholding, Fryzlewicz (2014) also proposes model selection via the "strengthened Schwarz Information Criterion" (sSIC). The sSIC penalty is close to the standard SIC (BIC) penalty, and therefore we do not review it here in detail. If an estimated change-point arises via the maximisation of $|\tilde{X}^b_{s_m, e_m}|$ over b, we refer to this estimate as b_m in this paragraph. The WBS-BIC version of the procedure, used in Section 2.1, initially computes all possible change-point candidates via the call WBS(1, T, 0). It then sorts them in the order of decreasing magnitudes of $|\tilde{X}^{b_m}_{s_m, e_m}|$. Let $m_1, m_2, \ldots, m_{\tilde{N}}$ be the indices (i.e. values of the indexing variable m) corresponding to this sorted order. The WBS-BIC returns as the estimated change-points the set $b_{m_1}, \ldots, b_{m_K}$ that yields the minimum of the BIC.

The key difference between the WBS and standard binary segmentation algorithms is that each change-point candidate is found by WBS by examining all intervals $[s_m, e_m]$ contained within $[s, e]$ (rather than examining only $[s, e]$ itself, as in binary segmentation), and finding the overall arg-max of the CUSUM statistics fitted on each such interval $[s_m, e_m]$. The reason why this improves on the standard binary segmentation is that if M is large enough, each true change-point is guaranteed (with probability approaching one in T) to be contained within at least one of the intervals $[s_m, e_m]$ which contains only that single change-point. This alone provides suitable guarantees for the size of the largest maximum of the CUSUM statistics, and is enough to yield substantial theoretical and practical advantages of WBS over binary segmentation, as similar guarantees are unavailable in the latter method. The collection of intervals $[s_m, e_m]$ is only drawn once at the start of the WBS procedure; therefore, previously-drawn intervals are reused as the algorithm progresses.

2.3 Issues affecting Wild Binary Segmentation

2.3.1 Incompleteness of WBS

With the definition of the routine WBS($\cdot$, $\cdot$, $\cdot$) as in Section 2.2, the call WBS(1, T, 0) returns all possible change-point candidates in the sense that with $\zeta_T = 0$, the change-point candidates are not tested against a threshold before they enter the output. Any model selection procedure can then be applied to this output to identify the final set of estimated change-points.

The call WBS(1, T, 0) is defined recursively. The equivalent non-recursive procedure for producing the same set of change-point candidates can be described as follows. For each $[s_m, e_m] \in F_T^M$, let $b_m = \arg\max_{b \in \{s_m, \ldots, e_m - 1\}} |\tilde{X}^b_{s_m, e_m}|$. Initially let $\mathcal{C} := \{(m, s_m, e_m, b_m, |\tilde{X}^{b_m}_{s_m, e_m}|) : m = 1, \ldots, M\}$ and $\mathcal{P} := \emptyset$. While $\mathcal{C} \neq \emptyset$, repeat the following pair of operations.

1. (Identification of next change-point candidate)

$$\begin{aligned} m_0 &:= \operatorname{argmax}_{m \,:\, (m, s_m, e_m, b_m, |\tilde{X}^{b_m}_{s_m, e_m}|) \in \mathcal{C}} |\tilde{X}^{b_m}_{s_m, e_m}|, \\ \mathcal{P} &:= \mathcal{P} \cup (m_0, s_{m_0}, e_{m_0}, b_{m_0}, |\tilde{X}^{b_{m_0}}_{s_{m_0}, e_{m_0}}|). \end{aligned}$$

2. (Removal of all intervals containing this change-point candidate)

$$\mathcal{C} := \mathcal{C} \setminus \{(m, s_m, e_m, b_m, |\tilde{X}^{b_m}_{s_m, e_m}|) \;:\; s_m \leq b_{m_0} < e_m\}.$$

At the end of this process, the set $\mathcal{P}$ contains the same set of change-point candidates as the output of WBS(1, T, 0). To see the equivalence between these two observe that the two recursive steps WBS(s, b_{m_0}, ζ_T) and WBS($b_{m_0} + 1$, e, ζ_T) in the definition of the WBS routine are equivalent to: after the recording of each change-point candidate b_{m_0}, remove from consideration all those intervals that contain b_{m_0}, and only operate on those intervals that fall entirely within $[s, b_{m_0}]$ and $[b_{m_0} + 1, e]$. In the above non-recursive definition of $\mathcal{P}$, this removal is explictly performed in stage 2. The removal is essential for the procedure to avoid repeated estimation of the same change-point multiple times.

Let $\tilde{N} = |\mathcal{P}|$ (where $|\cdot|$ returns the cardinality of its argument when it is a set). $\mathcal{P}$ can be viewed as an ordered set in the following sense. Let $(m_k)_{k=1}^{\tilde{N}}$ be the order in which the 5-tuples $(m_k, s_{m_k}, e_{m_k}, b_{m_k}, |\tilde{X}^{b_{m_k}}_{s_{m_k}, e_{m_k}}|)$ enter $\mathcal{P}$. The sequence $(|\tilde{X}^{b_{m_k}}_{s_{m_k}, e_{m_k}}|)_{k=1}^{\tilde{N}}$ is non-increasing, as its each next element is a maximum over the progressively shrinking set $\mathcal{C}$. In the remainder of this paper, we treat $\mathcal{P}$ as an ordered set with this particular ordering. Therefore, the ordered set $\mathcal{P}$ can be viewed as a WBS "solution path": the change-point candidates contained in $\mathcal{P}$ are arranged in order of decreasing importance (i.e. decreasing magnitude of $|\tilde{X}^{b_{m_k}}_{s_{m_k}, e_{m_k}}|$), and therefore any reasonable WBS model selection procedure would typically choose a certain number of the first components of $\mathcal{P}$ as the final set of estimated change-points.

As a result of the repeated removal of intervals from the set $\mathcal{C}$, the cardinality $\tilde{N}$ of the solution path $\mathcal{P}$ is typically much smaller than the cardinality M of the initial set $\mathcal{C}$. This is illustrated in the following example.

Example 2.1 Consider a sample path simulated from the model $X_t = f_t + \varepsilon_t$, $t = 1, \ldots, T = 1000$, where f_t is the `extreme.teeth` signal defined in Section 2.1 and $\varepsilon_t \sim N(0, 0.3^2)$ and has been simulated in R on setting the random seed to 1. We simulate $M = 5000$ (a value recommended in Fryzlewicz (2014)) intervals $[s_m, e_m]$. On completion of the computation of $\mathcal{P}$ as described in this section, we obtain $\tilde{N} = 119$. As the length $\tilde{N}$ of the solution path $\mathcal{P}$ is less than 199, the true number of change-points in f_t, no model selection procedure is able to recover all the change-points in this example: indeed, even if we were to accept all change-point candidates in $\mathcal{P}$, we would only estimate 119 change-points. Therefore, WBS as proposed in Fryzlewicz (2014) fails in this case. (End of example.)

The above example illustrates that too small a value of M can prevent WBS from working well for signals with frequent change-points, regardless even of the model selection procedure used. Increasing the default value of M in the hope of obtaining a longer solution path $\mathcal{P}$ is not a viable option as it can easily slow down the WBS procedure beyond what is acceptable (in particular, drawing all possible sub-intervals of $[1, \ldots, T]$ leads to WBS having computational speed $O(T^3)$) and is unnecessary for signals with no or few change-points, which require far fewer intervals. In other words, WBS does not adapt to the data in terms of the number M or the locations $[s_m, e_m]$ of the intervals it draws. From the user perspective, this lack of adaptivity creates two related major practical issues: (a) in the absence of prior information about the number and frequency of change-points, it is unclear whether any particular value of M is large enough in practice, and (b) setting M to be very large to be "on the safe side" is not an option from the computational point of view.

We formalise the issue of the length of solution paths in the language of "completeness", which we now introduce.

Definition 2.1 *Let A be a solution path algorithm in a multiple change-point detection problem which produces a sequence of candidate models with $0, 1, \ldots, \tilde{N}$ estimated change-points. We refer to A as* complete *if $\tilde{N} = T - 1$, where T is the length of the input data, and* incomplete *otherwise.*

Note that $\tilde{N} = T - 1$ means that every time point is a change-point candidate, so that the candidate model with $\tilde{N} = T - 1$ is maximal. The definition of completeness leads to the following proposition for WBS. The proof is straightforward, so we omit it.

Proposition 2.1 *The solution path $\mathcal{P}$ obtained via WBS as described in this section is guaranteed to be complete in the sense of Definition 2.1 if and only if $\{[s_m, e_m]\}_{m=1}^M$ represent all possible sub-intervals of $[1, \ldots, T]$, in which case $M = T(T-1)/2$. Furthermore, if $M = O(T^2)$, then the WBS solution path procedure is of computational order $O(T^3)$.*

We argue that completeness is a desirable feature of the solution path component of any change-point detection procedure if it is to be applied in frequent change-point scenarios, as the presence of completeness guarantees that we avoid situations (such as the one illustrated in this section) in which the solution path is too short to enable estimation of all the change-points. Completeness embodies the idea that no change-point candidate should be dismissed as inadmissible at the "solution path" stage: this should only happen at the model selection stage. For both scalability and good performance in frequent change-point scenarios, it is important for a solution path procedure to be both fast and complete, and we have shown in this section that WBS does not simultaneously possess these two features.

2.3.2 Shortcomings of thresholding and BIC as model selection for WBS

Fryzlewicz (2014) proposes two model selection criteria for WBS: one based on thresholding, and the other based on the sSIC, an information criterion for all practical purposes equivalent to the standard SIC (a.k.a. BIC). In this section, we argue that both can easily be inadequate in frequent change-point scenarios (independently of the issue of completeness described in Section 2.3.1).

The success of thresholding as a model selection criterion for WBS critically relies on the user's ability to accurately estimate $\sigma = \mathrm{Var}^{1/2}(\varepsilon_t)$. The following example illustrates that this task can be difficult to achieve in frequent change-point models.

Example 2.2 Consider 100 sample paths simulated from the model $X_t = f_t + \varepsilon_t$, $t = 1, \ldots, T = 1000$, where f_t is the `extreme.teeth` signal defined in Section 2.1. and $\varepsilon_t \sim N(0, 0.3^2)$. For each simulated sample path, we evaluate the Median Absolute Deviation (MAD) and Inter-Quartile Range (IQR) estimators (two commonly used robust estimators) of $\sigma = 0.3$ calibrated for the Gaussian distribution, computed with $\{2^{-1/2}(X_{t+1} - X_t)\}_{t=1}^{T-1}$ on input. Figure 3 shows that both estimators are heavily biased upwards on this example and overestimate σ by around 25%. The degree of this overestimation prevents thresholding from being an effective model selection tool in this model. (End of example.)

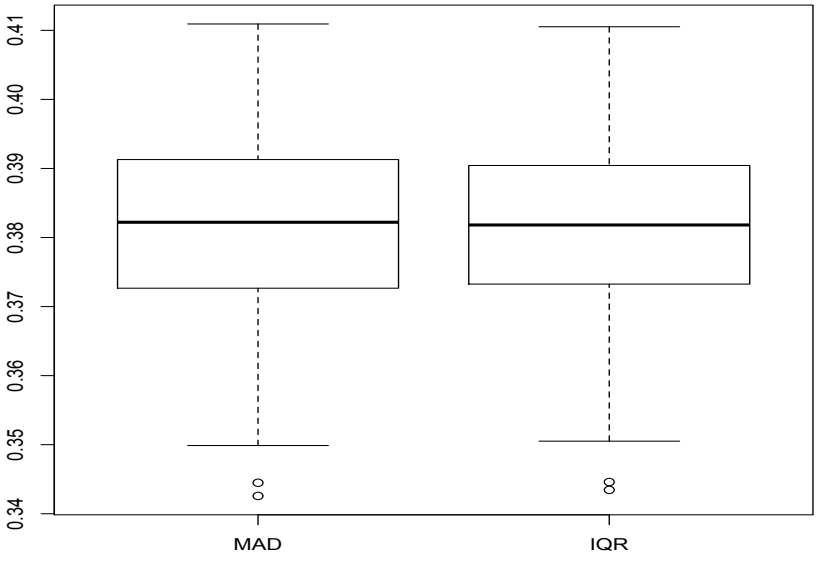

Figure 3: Boxplots of the empirical distribution of the MAD (left) and IQR (right) estimators of σ in the `extreme.teeth` model described in Section 2.3.2, over 100 simulated sample paths. The true value of σ is 0.3.

In addition, there is ample empirical evidence that the BIC is a poor model selector in frequent change-point scenarios, also for search algorithms other than WBS. For instance, we showed in Section 2.1 that the PELT method (Killick et al., 2012) equipped with the BIC stopping rule failed on the `extreme.teeth` example. In brief, this is due to the fact that the BIC term places too heavy a penalty on frequent-change-point solutions.

It is tempting to ask at this point whether choosing a penalty adaptively from the data (so that possibly different penalties are applied in infrequent and frequent change-point scenarios) may provide a successful remedy to this issue. Based on our empirical experience, the best such approach that we have come across is the "data-driven slope estimation" technique implemented in the R package **capushe** (Arlot et al., 2016) and described in Baudry et al. (2012). However, we show in Section 4.2 that this approach scales poorly to long signals and is outperformed by WBS2.SDLL in the frequent change-point examples we tested.

3 Wild Binary Segmentation 2 and Steepest Drop to Low Levels

3.1 WBS2: data-adaptive computation of complete solution path

3.1.1 Computation of the WBS2 solution path

This section describes how the WBS2 solution path is computed. This is the first of the two ingredients of the WBS2.SDLL procedure; the other is the application of the new "Steepest Drop to Low Levels" model selection criterion, described in Section 3.2.

We start with a heuristic description, followed by a formal algorithmic definition. In the first pass through the data, WBS2 draws a small number $\tilde{M}$ of data subsamples, with start- and end-points chosen randomly, uniformly and independently with replacement from the set $\{1, \ldots, T\}$. WBS2 marks the first change-point candidate as the arg-max of the resulting $\tilde{M}$ absolute CUSUMs. It then uses this change-point candidate to split its domain of operation $\{1, \ldots, T\}$ into two, and again recursively draws $\tilde{M}$ subsamples to the left and to the right of this change-point candidate, and so on. Eventually, on short sub-domains, the number of all possible subsamples will be less than $\tilde{M}$, in which case WBS2 will draw all possible subsamples of each such sub-domain. WBS2 stops and takes no action on a given current sub-domain only if its length is 1. Having completed this step, WBS2 then sorts all thus-obtained change-point candidates in the decreasing order of their corresponding absolute CUSUMs. The resulting object is referred to as the WBS2 solution path. Algorithmically, the main ingredient of the computation of the WBS2 solution path is the routine WBS2.Sol.Path, defined recursively as follows.

function WBS2.Sol.Path(s, e, $\tilde{M}$)
  if $e - s < 1$ **then**
    STOP
  end if
  if $\tilde{M} \geq \frac{1}{2}(e - s + 1)(e - s)$ **then**
    $\tilde{M} := \frac{1}{2}(e - s + 1)(e - s)$
    Draw all intervals $[s_m, e_m] \subseteq [s, s + 1, \ldots, e]$, $m = 1, \ldots, \tilde{M}$, s.t. $e_m - s_m > 1$.
  else
    Randomly draw intervals $[s_m, e_m] \subseteq [s, s+1, \ldots, e]$, $m = 1, \ldots, \tilde{M}$, s.t. $e_m - s_m > 1$ and s_m, e_m drawn uniformly and independently with replacement.
  end if
  $(m_0, b_{m_0}) := \arg\max_{m=1,\ldots,\tilde{M},\, b \in \{s_m, \ldots, e_m - 1\}} |\tilde{X}^b_{s_m, e_m}|$

Add $(s_{m_0}, e_{m_0}, b_{m_0}, |\tilde{X}^{b_{m_0}}_{s_{m_0},e_{m_0}}|)$ to the (unsorted) solution path $\tilde{\mathcal{P}}$.
WBS2.SOL.PATH(s, b_{m_0}, $\tilde{M}$)
WBS2.SOL.PATH($b_{m_0}+1$, e, $\tilde{M}$)
end function

The WBS2 solution path $\tilde{\mathcal{P}}$ is computed in two stages, defined below.

1. (Identification of change-point candidates)

   Initialise $\tilde{\mathcal{P}} := \emptyset$.

   Add elements to the (unsorted) WBS2 solution path $\tilde{\mathcal{P}}$ by executing the command WBS2.SOL.PATH(1, T, $\tilde{M}$). We state in Section 3.1.2 that the length of the thus-obtained sequence $\tilde{\mathcal{P}}$ is $T-1$.

2. (Sorting of the WBS2 solution path)

   Rearrange the elements of $\tilde{\mathcal{P}}$ so that the sequence of its 4th components, denoted by $(|\tilde{X}^{b_k}_{s_k,e_k}|)_{k=1}^{T-1}$, is non-increasing, breaking any ties at random. Denote the elements of the resulting (sorted) sequence $\tilde{\mathcal{P}}$ by $(s_k, e_k, b_k, |\tilde{X}^{b_k}_{s_k,e_k}|)_{k=1}^{T-1}$.

The philosophy of WBS2 is fundamentally different from WBS. In WBS, all M intervals are pre-drawn before the launch of the procedure, and there needs to be a sufficient number of them for *all* change-points to be detectable from this single draw. By contrast, in WBS2, with each draw of $\tilde{M}$ intervals, we only hope to detect *one* of the remaining undetected change-points (if there are any). For this reason, $\tilde{M}$ will typically be much smaller than M. We comment in Section 3.1.2 on the theoretical orders of magnitude of $\tilde{M}$; for now suffice it to say that we use $\tilde{M} = 100$ in all numerical examples of this paper.

In comparison to WBS, the computation of the WBS2 solution path is data-adaptive. WBS2 does not use the data in drawing its M intervals; by contrast, in WBS2, the location of the domain of each next batch of $\tilde{M}$ intervals is determined by the locations of the previously detected change-point candidates. One consequence of this computational adaptivity of WBS2 is that on average, the intervals drawn by WBS2 rapidly become shorter as the procedure progresses, as the procedure operates on shorter and shorter sub-domains of $[1, \ldots, T]$. This makes WBS2 fast; we comment on the theoretical computational complexity of the WBS2 solution path in Section 3.1.2 and actual computation times are given in Section 4.2.

Finally, we note that the computation of the WBS2 solution path requires no other parameters besides $\tilde{M}$, and that it is natural for it to be defined (and indeed implemented) recursively.

3.1.2 Theoretical properties of the WBS2 solution path

We introduce the following technical assumption.

Assumption 3.1

(a) The sequence $(\varepsilon_t)_{t=1}^T$ *is i.i.d.* $N(0, \sigma^2)$.

(b) With the additional notation $\eta_0 = 1$, $\eta_{N+1} = T + 1$, *the minimum spacing between change-point satisfies* $\min_{i=1,\ldots,N+1}(\eta_i - \eta_{i-1}) \geq \delta_T$, *where* $\delta_T = \delta T$ *with* $\delta > 0$.

(c) $|f_t| \leq \bar{f} < \infty$ *for* $t = 1, \ldots, T$.

(d) The magnitudes $f_i' = |f_{\eta_i} - f_{\eta_i - 1}|$ *of the jumps satisfy* $\min_{i=1,\ldots,N} f_i' \geq \underline{f}_T \geq CT^{\kappa}$ *for some* $C > 0$ *and* $\kappa > -1/2$.

Assumptions 3.1(a) or (b) are not strictly necessary for the procedure to work, and both have been made for technical simplicity and to simplify the exposition as much as possible, while still enabling the demonstration of the full mechanics of the procedure; various extensions are in principle possible. In frequent change-point scenarios, the reader is invited to think of situations in which δ, the smallest possible spacing between neighbouring change-points as a fraction of the sample size, is small. We note, also, that the minimum jump size in Assumption 3.1(d) satisfies the condition spelled out in Assumption 3.3 of Fryzlewicz (2014). Finally, we emphasise that it is permitted for the jump sizes f_i' to have different orders of magnitude in T, as long as their minimum satisfies the lower bound in Assumption 3.1(d).

To facilitate the statement of the theorem below, we introduce the concept of 'scale' at which the WBS2 procedure operates as follows. Initially, the procedure operates on the domain $[1, \ldots, T]$, and we refer to this stage as scale 1. It then subdivides this domain into two and operates on the two sub-domains; this is scale 2. The scale parameter then increases by one with each subdivision.

We have the following result.

Theorem 3.1 *Let* X_t *follow model (1) and let Assumption 3.1 hold. Let* N *and* $\eta_1, \ldots, \eta_N$ *denote, respectively, the number and the locations of the change-points. Let* $\tilde{\mathcal{P}}$ *be the solution path of the WBS2 algorithm, defined in Section 3.1.1. The following statements are true.*

(i) The length of $\tilde{\mathcal{P}}$ *is* $T - 1$.

(ii) Let Γ_j *be the length of the longest sub-domain on which the WBS2 solution path algorithm operates at scale* $j = 1, 2, \ldots$. *Let* $J = \min\{j : \Gamma_j = 1\} - 1$. *The computational complexity of the WBS2 solution path algorithm is* $O(\tilde{M}JT)$.

(iii) Let $\{b_k\}_{k=1}^{T-1}$ *be the 3rd components of the elements of* $\tilde{\mathcal{P}}$. *Sort* $\{b_k\}_{k=1}^{N}$ *in increasing order, and denote the thus-ordered sequence by* $(\tilde{\eta}_i)_{i=1}^{N}$. *Define the event*

$$\begin{aligned}\mathcal{A}_T \;=\; & \{\max_{i=1,\ldots,N} |\eta_i - \tilde{\eta}_i| \leq C_1(\Delta)(\underline{f}_T)^{-2} \log T \quad \cap \quad |\tilde{X}^{b_k}_{s_k,e_k}| \gtrsim \underline{f}_T T^{1/2} \text{ for } k = 1, \ldots, N \\ & \cap \quad |\tilde{X}^{b_k}_{s_k,e_k}| \leq C_2(\Delta) \log^{1/2} T \text{ for } k = N+1, \ldots, T-1\},\end{aligned}$$

where $C_1(\Delta), C_2(\Delta)$ *are positive constants that only depend on the positive* Δ *parameter below, and the* $\gtrsim$ *symbol means "of the order of or larger". For* T *large enough, we have*

$$P(\mathcal{A}_T) \geq 1 - \alpha_T - \frac{1}{2}N(N+1)(2C_1(\Delta)(\underline{f}_T)^{-2} \log T + 1)^2 (1 - \delta^2/9)^{\tilde{M}}, \qquad (3)$$

where

$$\alpha_T = 1 - P(\mathcal{B}_{\Delta,T})$$

with

$$\mathcal{B}_{\Delta,T} = \left\{ \forall\ 1 \le l \le m \le T \qquad |m-l+1|^{-1/2} \left| \sum_{i=l}^{m} \varepsilon_i \right| \le \sigma \{2(1+\Delta)\log T\}^{1/2} \right\}.$$

We now comment on several aspects of Theorem 3.1. Part (i) means that the WBS2 solution path procedure is complete in the sense of Definition 2.1. In part (ii), we investigate the magnitudes of $\tilde{M}$ and J to gain a better idea of the computational complexity of the WBS2 solution path procedure. Beginning with $\tilde{M}$, we now discuss the minimum number of draws $\tilde{M}$ required to ensure that $P(\mathcal{A}_T)$ in part (iii) converges to 1 suitably fast. To match the equivalent rate for WBS (see Theorem 3.2 in Fryzlewicz (2014)), we require

$$\frac{1}{2}N(N+1)(2C_1(\Delta)(\underline{f}_T)^{-2}\log T + 1)^2(1-\delta^2/9)^{\tilde{M}} \le T^{-1},$$

which is equivalent to

$$\tilde{M} \ge \log^{-1}\{(1-\delta^2/9)^{-1}\}(\log T + \log\{N(N+1)/2\} + 2\log(2C_1(\Delta)(\underline{f}_T)^{-2}\log T + 1)),$$

and therefore $\tilde{M} = O(\log T)$. The maximum magnitude of J depends on the location of the splits b_{m_0} within each each interval $[s,e]$ on which the routine WBS2.Sol.Path$(s,e,\tilde{M})$ is executed. In particular, we have the following result (the proof is straightforward, so we omit it).

Proposition 3.1 *If, in each execution of the routine* WBS2.Sol.Path$(s,e,\tilde{M})$*, the split location b_{m_0} is such that*

$$\gamma \le \frac{b_{m_0}-s+1}{e-s+1} \le 1-\gamma, \tag{4}$$

for a certain fixed $\gamma \in (0,1)$, then $J = O(\log T)$.

It is immediate that condition (4) is automatically satisfied if $e-s \le 1/\gamma - 1$, i.e. for short intervals. From the proof of Theorem 3.1, it is also automatically satisfied (for γ small enough, with probability tending to 1 with T) for those intervals $[s,e]$ which contain previously undetected change-points. Since the locations of the splits b_{m_0} in those intervals $[s,e]$ that do not contain previously undetected change-points is immaterial for the performance of the procedure, this in turn implies that an alternative WBS2 solution path procedure, in which the existing selection of b_{m_0} were replaced by constrained maximisation of the form

$$(m_0, b_{m_0}) := \arg \max_{m=1,\ldots,\tilde{M},\, b\in\{s_m,\ldots,e_m-1\}} |\tilde{X}^b_{s_m,e_m}| \quad \text{s.t. condition (4)}$$

would enjoy exactly the same properties as the original WBS2 solution path procedure, plus would come with a guarantee that $J = O(\log T)$. However, we refrain from discussing such a modified procedure further as it would require the choice of γ, an additional parameter.

Proposition 3.1 and the above discussion regarding the magnitude of $\tilde{M}$ imply the following corollary.

Corollary 3.1 *Under the assumptions of Theorem 3.1 and Proposition 3.1, the computational complexity of the WBS2 solution path algorithm is $O(T\log^2 T)$.*

Regarding part (iii) of Theorem 3.1, it is the statement of Lemma 1 in Yao (1988) that $P(\mathcal{B}_{\Delta,T}) \to 1$ for all $\Delta > 0$, at a rate that depends on the magnitude of Δ. The uniform bound on the magnitudes of the errors $|\eta_i - \tilde{\eta}_i|$ is the same as in WBS and is 'near-optimal' in the sense described in Fryzlewicz (2014).

We emphasise that part (iii) does not yet constitute a complete consistency result for the WBS2.SDLL procedure, as it does not specify a model selection process; it merely says that the model with the true number N of estimated change-points is such that the locations of all N change-points are estimated near-optimally. In Section 3.2, we will describe how to estimate N consistently via SDLL and this will lead to a complete consistency result (for the number and locations of estimated change-points) for the WBS2.SDLL method.

3.2 "Steepest Drop to Low Levels" model selection

3.2.1 Motivating example and methodology of SDLL model selection

This section introduces our new model selection criterion, labelled "Steepest Drop to Low Levels", which will be used to select a model along the WBS2 solution path $\tilde{\mathcal{P}}$. Given $\tilde{\mathcal{P}}$, the model selection task is equivalent to indicating an estimate $\hat{N}$ of N (the number of change-points), as the estimated change-point locations will then be given by $\{b_k\}_{k=1}^{\hat{N}}$, if $\hat{N} > 0$. This section will define our estimator $\hat{N}$. We start with a motivating example.

Example 3.1 As in Section 2.1 and Example 2.1, we consider the model $X_t = f_t + \varepsilon_t$, $t = 1, \ldots, 1000$, where f_t is the `extreme.teeth` signal. However, the set-up in the current example is even more challenging in that we consider ε_t i.i.d. $\sim N(0, 0.4^2)$ as compared to $\varepsilon_t \sim N(0, 0.3^2)$ in Section 2.1 and Example 2.1. For a particular realisation of this model, we compute the WBS2 solution path $\tilde{\mathcal{P}}$ with $\tilde{M} = 100$ and plot the resulting sequence $\{|\tilde{X}^{b_k}_{s_k,e_k}|\}_{k=1}^{T-1}$ (Figure 4, top plot). Recall that by the construction of $\tilde{\mathcal{P}}$, this sequence is non-increasing.

The true number of change-points in f_t is $N = 199$. A visual inspection of the top plot in Figure 4 reveals an increasing (negative) gradient in $\{|\tilde{X}^{b_k}_{s_k,e_k}|\}_{k=1}^{T-1}$ at around $k = 199$. This appears to align well with the assertion of part (iii) of Theorem 3.1, according to which, on a set of high probability, the elements of $\{|\tilde{X}^{b_k}_{s_k,e_k}|\}_{k=1}^{T-1}$ are of a higher order of magnitude ($\gtrsim \underline{f}_T T^{1/2}$) for $k = 1, \ldots, N$ than they are for $k = N+1, \ldots, T-1$ ($O(\log^{1/2} T)$). This observation provides a basis for our procedure for estimating N: our $\hat{N}$ will be an estimator of the location of this change in orders of magnitude, or equivalently the location of this large negative gradient.

In order to perform the estimation of this location, it is natural to work on the log scale: we denote $Y_k = \log |\tilde{X}^{b_k}_{s_k,e_k}|$. The reason is that taking the differences $Z_k = Y_k - Y_{k+1}$ leads to the cancellation of the log T terms for $k = 1, \ldots, N-1$ and the cancellation of the log log T terms for $k = N+1, \ldots, K$ (where K is the largest k for which $|\tilde{X}^{b_{k+1}}_{s_{k+1},e_{k+1}}| \geq C \log^{1/2} T$ for a certain constant C). Therefore, the terms Z_k for $k = 1, \ldots, N-1, N+1, \ldots, K$ will be bounded in T, whereas the single term Z_N will tend to infinity with T at the speed of log T. As a consequence, it is tempting to consider setting $\hat{N}$ to the value of k for which Z_k is the largest. Our new model selection criterion does exactly that, but with a certain finite-sample correction, which we motivate next.

The bottom plot of Figure 4 shows the sequence Z_k for $k = 1, \ldots, K = 333$. In light of

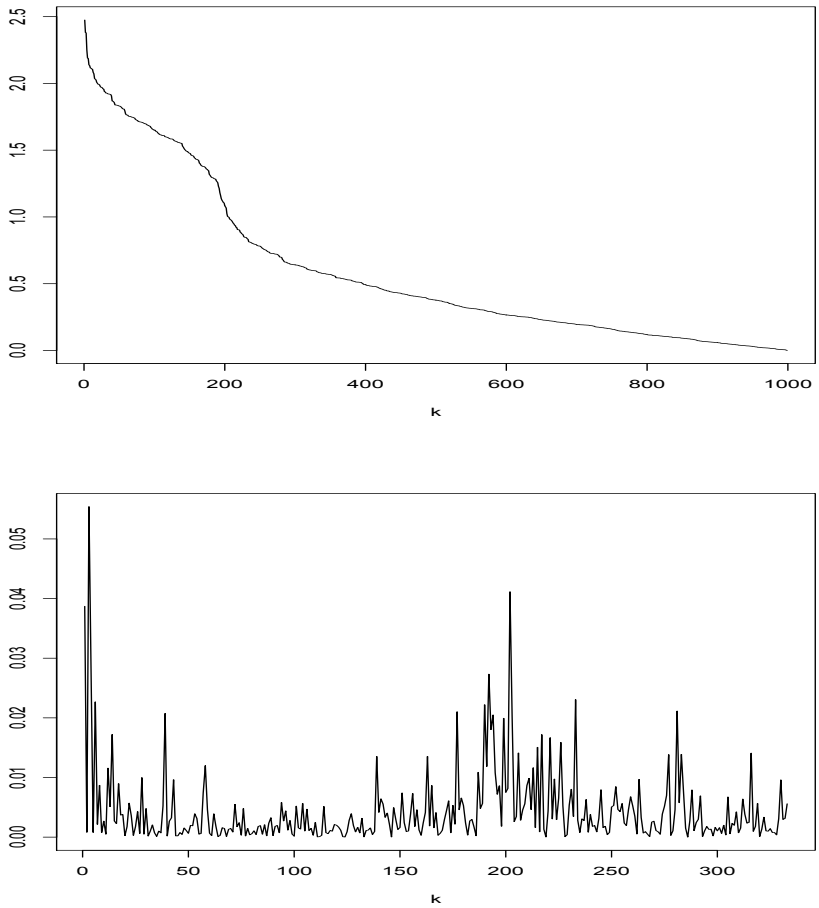

Figure 4: Top: the sequence $\{|\tilde{X}^{b_k}_{s_k,e_k}|\}_{k=1}^{T-1}$ of Example 3.1; bottom: the sequence $\{Z_k\}_{k=1}^{333}$ of Example 3.1.

the above discussion, it is unsurprising to see a prominent peak in Z_k around $k = N = 199$ (in this particular example, the location of this local peak is $k = 202$). Yet, setting $\hat{N} = \arg\max_{k=1,\ldots,K} Z_k$ would have failed in this particular case, as the global maximum of Z_k is achieved for $k = 3$. However, one way of checking that $\hat{N} = 3$ leads to an infeasible model in this example is to examine the size of Y_4, the smaller term entering the difference $Z_3 = Y_3 - Y_4$. If $k = 3$ were to be the true model size, then by part (iii) of Theorem 3.1 we would expect $|\tilde{X}^{b_4}_{s_4,e_4}| = \exp(Y_4) = O(\log^{1/2} T)$. We use a threshold to decide whether $|\tilde{X}^{b_4}_{s_4,e_4}|$ is indeed of this magnitude, and the form of this "universal" threshold is $1.2\,\hat{\sigma}_T\{2\log T\}^{1/2}$ (where $\hat{\sigma}_T$ is the MAD estimate of σ with $\{2^{-1/2}(X_{t+1} - X_t)\}_{t=1}^{T-1}$ on input); this formula will be motivated in Sections 3.2.2 and 4.1. In this particular example, we have $|\tilde{X}^{b_4}_{s_4,e_4}| \approx 2.25 > 2.0 \approx 1.2\,\hat{\sigma}_T\{2\log T\}^{1/2}$, and therefore $k = 3$ is rejected as a feasible model. Our procedure then goes on to examine the next largest Z_k, which in this case is Z_{202}; this leads to examination of $|\tilde{X}^{b_{203}}_{s_{203},e_{203}}|$, whose magnitude is well under $1.2\,\hat{\sigma}_T\{2\log T\}^{1/2} \approx 2.0$. As a consequence, the model with 202 change-points is accepted and we set $\hat{N} = 202$ (over-estimating the true number of change-points by 3). (End of example.)

We now comment heuristically on some aspects of Example 3.1 before introducing a formal algorithmic description of our new "Steepest Drop to Low Levels" (SDLL) model selection procedure.

The name "Steepest Drop to Low Levels" has its origins in the fact that our model selection procedure does not merely look for the "steepest drop" (= largest negative gradient) in $\log|\tilde{X}^{b_k}_{s_k,e_k}|$ (this occurs at $k = 3$ in the above example) but it searches for the steepest drop followed by "low levels" (= values of $|\tilde{X}^{b_k}_{s_k,e_k}|$ that fall under the threshold); this occurs at $k = 202$ in the example.

We emphasise that the use of thresholding in the SDLL model selection is new and different from the classical use of thresholding in change-point detection problems (as in e.g. Fryzlewicz (2014)). In classical thresholding, a threshold is used as the main model selection device in the sense that its value is used to separate significant from insignificant change-point candidates. This happens in a "continuous" manner, in the sense that even small changes in the threshold value can lead to the exclusion or inclusion of additional candidate change-points and therefore to changes in the selected model. A satisfactory threshold should in theory be linear with σ, so due to this sensitivity of the selected model to the threshold value, the user should have the ability to estimate σ accurately, which is not always possible in frequent change-point settings, as illustrated earlier.

By contrast, SDLL uses thresholding only as a secondary model selection device, the main criterion being the size of the negative gradient in $\log|\tilde{X}^{b_k}_{s_k,e_k}|$. In SDLL, thresholding is used "discretely" in the sense that the consecutive models being tested via thresholding are often very remote from each other in terms of their numbers of change-points; in the example above, we first tested a model with 3 change-points followed by a model with 202 change-points. In other words, we used thresholding not so much to "choose" a model, but to test the feasibility of models pre-ordered according to their respective negative gradients in $\log|\tilde{X}^{b_k}_{s_k,e_k}|$ (rather than according to their numbers of change-points). The model with 3 change-points was rejected as easily as the model with 202 change-points was accepted. This suggests that SDLL does not require the threshold to be selected as precisely as is the case in classical thresholding. This is potentially a useful feature of SDLL as it facilitates accurate model selection in the presence of frequent change-points, when the user is not always able to estimate σ (and hence the threshold) well.

We are now in a position to introduce a complete algorithm describing our SDLL model selection procedure. It is defined by the WBS2.SDLL routine below, which takes three parameters on input: $\tilde{\mathcal{P}}$ is a WBS2 solution path, whose computation is outlined in Section 3.1.1; $\tilde{\zeta}_T$ is a threshold; and $\beta \in (0,1)$ is a constant. In the remainder of this paper, we use $\tilde{\zeta}_T = \tilde{C}\hat{\sigma}_T\{2\log T\}^{1/2}$, and this choice will be motivated in Section 3.2.2; the choice of the constants $\tilde{C}$ and β will be described in Section 4.1.

function WBS2.SDLL($\tilde{\mathcal{P}}$, $\tilde{\zeta}_T$, β)
  if $|\tilde{\mathcal{P}}| = 0$ **then**
    STOP
  end if
  if $|\tilde{X}^{b_1}_{s_1,e_1}| < \tilde{\zeta}_T$ **then**
    $\hat{N} := 0$
  else
    $K := \max\{k \ : \ |\tilde{X}^{b_{k+1}}_{s_{k+1},e_{k+1}}| \geq \beta\,\tilde{\zeta}_T\}$

if $K = 0$ **then**
$\hat{N} := 1$
else
$Z_k := \log|\tilde{X}^{b_k}_{s_k,e_k}| - \log|\tilde{X}^{b_{k+1}}_{s_{k+1},e_{k+1}}|$, $k = 1, \ldots, K$
if $\exists\ k = 1, \ldots, K$ s.t. $|\tilde{X}^{b_{k+1}}_{s_{k+1},e_{k+1}}| \leq \tilde{\zeta}_T$ **then**
$\hat{N} := \arg\{\max_{k=1,\ldots,K} Z_k$ s.t. $|\tilde{X}^{b_{k+1}}_{s_{k+1},e_{k+1}}| \leq \tilde{\zeta}_T\}$
else
$\hat{N} := K + 1$
end if
end if
$(\tilde{\eta}_1, \ldots, \tilde{\eta}_{\hat{N}}) := \textsc{Sort}((b_1, \ldots, b_{\hat{N}}), \textsc{Increasing})$
end if
end function

If the magnitudes of $|\tilde{X}^{b_k}_{s_k,e_k}|$ all fall under $\tilde{\zeta}_T$, the procedure returns zero change-points. Otherwise, the procedure extracts all those indices $k = 1, \ldots, K$ for which $|\tilde{X}^{b_{k+1}}_{s_{k+1},e_{k+1}}| \geq \beta\tilde{\zeta}_T$; this is to ensure that we only work with those k for which $|\tilde{X}^{b_{k+1}}_{s_{k+1},e_{k+1}}| \geq C\ \log^{1/2} T$ as motivated in Example 3.1. Within this range of k, it then looks for the maximum negative gradient in $\log|\tilde{X}^{b_k}_{s_k,e_k}|$, subject to the constraint that $|\tilde{X}^{b_{k+1}}_{s_{k+1},e_{k+1}}| \leq \tilde{\zeta}_T$, and assigns $\hat{N}$ to be that location; if the constraint $|\tilde{X}^{b_{k+1}}_{s_{k+1},e_{k+1}}| \leq \tilde{\zeta}_T$ is never satisfied, this necessarily means that the end of the range has been reached, in which case $\hat{N} = K + 1$.

We now comment on a few important aspects of the SDLL model selection. As described earlier in the discussion of Example 3.1, SDLL is not a classical thresholding model selection procedure, but also does not used a penalty-based approach. Its execution, not including the call to the Sort routine, is of computational order $O(T)$. Under the assumptions of Theorem 3.2, the execution of the Sort routine is of computational order $O(1)$ with high probability. Unlike some penalty-based approached, SDLL does not need to know the maximum number of change-points present in the data, in either theory or practice.

3.2.2 Theoretical properties of SDLL model selection

In this section, we formulate a complete consistency theorem for the WBS2 method equipped with the SDLL model selection criterion, as regards the estimation of both the number N and the locations $\eta_1, \ldots, \eta_N$ of the change-points. We have the following result.

Theorem 3.2 *Let X_t follow model (1) and let Assumption 3.1 hold. Let N and $\eta_1, \ldots, \eta_N$ denote, respectively, the number and the locations of the change-points. Let $\tilde{\mathcal{P}}$ be the solution path of the WBS2 algorithm, defined in Section 3.1.1. Let $\hat{N}$ and $\tilde{\eta}_1, \ldots, \tilde{\eta}_{\hat{N}}$ be the estimates of the number and the locations (respectively) of the change-points, returned by the SDLL model selection criterion as defined by the WBS2.SDLL routine with $\tilde{\mathcal{P}}$, $\tilde{\zeta}_T$ and β on input, with $\beta \in (0, 1)$ and $\tilde{\zeta}_T = \tilde{C}\hat{\sigma}_T\{2\log T\}^{1/2}$ for a large enough constant $\tilde{C}$. Define the events*

$$\begin{aligned} \Theta_{\theta,T} &= \{(1+\theta)^{-1} < \hat{\sigma}_T/\sigma < 1 + \theta\}, \\ \mathcal{E}_T &= \{\hat{N} = N \quad \cap \quad \max_{i=1,\ldots,\hat{N}} |\eta_i - \tilde{\eta}_i| \leq C_1(\Delta)(\underline{f}_T)^{-2}\log T\}, \end{aligned}$$

where θ is a positive constant and $C_1(\Delta)$ is defined in the statement of Theorem 3.1. For

T large enough, we have

$$P(\mathcal{E}_T) \geq 1 - \alpha_T - \frac{1}{2}N(N+1)(2C_1(\Delta)(\underline{f}_T)^{-2}\log T + 1)^2(1-\delta^2/9)^{\tilde{M}} - P(\Theta^c_{\theta,T}), \quad (5)$$

where α_T is defined in the statement of Theorem 3.1.

The remarks below complement and extend the discussion of Theorem 3.1 in Section 3.1.2. In that discussion, we already clarified that $\alpha_T \to 1$ as $T \to \infty$ and that $\frac{1}{2}N(N+1)(2C_1(\Delta)(\underline{f}_T)^{-2}\log T + 1)^2(1-\delta^2/9)^{\tilde{M}} = O(T^{-1})$ if $\tilde{M} \geq C \log T$ for a large enough constant C. We note that for $P(\Theta^c_{\theta,T}) \to 0$, we do not even require that $\hat{\sigma}_T$ should be a consistent estimator of σ, but only that the ratio $\hat{\sigma}_T/\sigma$ be bounded with probability tending to 1 with T. This is emphasised in order to reflect the empirical fact that σ may be difficult to estimate well in frequent change-point scenarios.

Proving such a boundedness condition is typically not problematic for a variety of commonly used estimators of σ. As an example, we now sketch how boundedness from above can be shown for a simplified Median Absolute Deviation estimator, defined by

$$\hat{\sigma}^{\mathrm{MAD}}_T = C\,\mathrm{med}\{|X_{t+1} - X_t|\}_{t=1}^{T-1},$$

where C is a certain universal normalising constant and med is the median operator. Let avg denote the sample mean operator, and let $a_t = C|X_{t+1} - X_t|$. Similarly to Mallows (1991), recalling that the median measures the centre of the data in the l_1 sense, we have

$$\begin{aligned} |\mathrm{avg}\{a_t\} - \mathrm{med}\{a_t\}| &= |\mathrm{avg}\{a_t - \mathrm{med}\{a_t\}\}| \leq \mathrm{avg}|a_t - \mathrm{med}\{a_t\}| \\ &\leq \mathrm{avg}|a_t| = \mathrm{avg}\{a_t\}, \end{aligned}$$

which gives $\mathrm{med}\{a_t\} \leq 2\,\mathrm{avg}\{a_t\}$. Therefore the boundedness from above of $\hat{\sigma}^{\mathrm{MAD}}_T$ will follow from that of $\mathrm{avg}\{a_t\}$. For the latter, note that its expectation, $\mathbb{E}(\mathrm{avg}\{a_t\})$, is necessarily bounded, regardless of the number of change-points in the signal, due to Assumption 3.1(c). Therefore, it suffices to establish a concentration bound on $\mathrm{avg}\{a_t\}$ around its expectation, but this is a standard result, available, e.g., in Theorem 1.4 in Bosq (1998), which applies to zero-mean weakly-dependent processes satisfying Cramer's conditions, such as $a_t - \mathbb{E}(a_t)$. This completes the sketch of this argument.

Finally, note that the rate of $O(\log^{1/2} T)$ for the threshold $\tilde{\zeta_T}$ is standard and also appears, for example, in the standard Wild Binary Segmentation algorithm of Fryzlewicz (2014) and in the Tail-Greedy bottom-up method of Fryzlewicz (2018).

3.3 WBS2.SDLL vs other adaptive change-point model selection methods

In this section, we compare and contrast WBS2.SDLL, in a more thorough way than was done in Section 1, to some existing methods in the literature which select the change-point model (i.e. estimate N) by choosing the value of an appropriate tuning parameter in a data-driven way; for brevity, we refer to such techniques as adaptive.

In this sense, WBS2.SDLL can also be interpreted as an adaptive technique: given a sorted WBS2 solution path $\tilde{\mathcal{P}}$, constructed as described in Section 3.1.1, the SDLL model selection procedure regards as significant the $\hat{N}$ change-points that correspond to the first $\hat{N}$ CUSUMs $|\tilde{X}^{b_k}_{s_k,e_k}|$, for $k = 1, \ldots, \hat{N}$, which are also the largest ones. Therefore SDLL can

be interpreted as an "adaptive thresholding" technique, which retains a change-point candidate b_k if and only if $|\tilde{X}^{b_k}_{s_k,e_k}| \geq \lambda_{SDLL}(X)$ and discards it otherwise, where $\lambda_{SDLL}(X)$ is a certain data-driven threshold, chosen by the SDLL procedure from the shape of the solution path $\tilde{\mathcal{P}}$ as described in the definition of SDLL in Section 3.2.1.

Birgé and Massart (2001) and Birgé and Massart (2007) propose the adaptive choice of what they refer to as the minimal penalty, in the context of penalised Gaussian model selection in which the cost function is the L_2 risk, via an algorithm referred to in later works as "dimension jump". The main idea of dimension jump is as follows. Consider a family of penalties of the form CD_m/T, where D_m is the dimensionality of the space of candidate models (in our context, this would be the postulated number of change-points). Heuristically, too small a choice of the constant C when minimising the corresponding penalised L_2 risk criterion will lead to models with a lot of change-points, whereas a large (or indeed too large) a choice will reduce this considerably. Therefore the dimension jump approach looks at the graph of the function $C/\sigma^2 \mapsto D_{\hat{m}}$ and chooses the constant C that corresponds to the largest jump in this function, hoping that this will locate the "sweet spot" between penalties that are too small and those that are too large.

In the approach termed "slope estimation" (Lebarbier, 2005) the observation is that the optimal penalty for Gaussian model selection under the L_2 risk, in the sense of Birgé and Massart (2007), depends on the unknown parameter σ^2. For any postulated number of change-points N', let $\hat{f}_t^{N'}$ be the best fit to the data in the L_2 sense that contains exactly N' change-points. For N' large enough, the function $N' \mapsto \sum_{t=1}^{T}(X_t - \hat{f}_t^{N'})^2$ is approximately linear with slope $-\sigma^2/T$, which can in principle be used to estimate σ^2, which can then be plugged into the optimal penalty.

Specific algorithmic ideas regarding the implementation of dimension jump and slope estimation in the context of the multiple change-point problem studied in this paper appear, amongst others, in Lebarbier (2005) and Baudry et al. (2012). Our comparative simulation study in Section 4.2 includes both these approaches and describes the details of the R packages used. Arlot (2019) is an excellent and comprehensive review article on minimal penalties and the slope heuristics, which describes these two approaches and their variants in detail, including an unpublished two-stage refinement by Rozenholc, branded "statistical base jumping".

We note that SDLL is fundamentally different from both dimension jump and slope estimation. To start with, SDLL makes its model selection decision based on the shape of the function $k \mapsto \log\{|\tilde{X}^{b_k}_{s_k,e_k}|\}$, where $|\tilde{X}^{b_k}_{s_k,e_k}|$ are the WBS2 CUSUMs sorted in decreasing order. As described above, dimension jumps considers the shape of an entirely different function, which does not appear obviously related to that considered by SDLL. Slope estimation considers the tail behaviour (which is very different from the steepest drop to low levels as considered by SDLL) of the function $N' \mapsto \sum_{t=1}^{T}(X_t - \hat{f}_t^{N'})^2$, which for each N' measures the impact of the N' most important change-points in the data. Therefore if we were to take the difference $\sum_{t=1}^{T}(X_t - \hat{f}_t^{N'})^2 - \sum_{t=1}^{T}(X_t - \hat{f}_t^{N'+1})^2$, this would provide a measure of the importance of the $N'+1$st change-point. Even this is different from the measure of importance of each change-point used by SDLL, which is the WBS2 CUSUM. The attractive feature of considering WBS2 CUSUMs in the context of SDLL, rather than, say, the differences $\sum_{t=1}^{T}(X_t - \hat{f}_t^{N'})^2 - \sum_{t=1}^{T}(X_t - \hat{f}_t^{N'+1})^2$, is that the WBS2 procedure targets and returns the *maximum* available CUSUM at each stage. As a result, it is hoped that WBS2 maximises the measure of each significant change-point, without unduly inflat-

ing the CUSUMs corresponding to the insignificant change-points, as these are uniformly bounded. This aggressive separation makes it easier for SDLL to identify the drop in the sorted sequence of WBS2 CUSUMs. Indeed, in experiments not reported in this paper, we tried using the SDLL criterion applied to measures of change-point significance closely related to $\sum_{t=1}^{T}(X_t - \hat{f}_t^{N'})^2 - \sum_{t=1}^{T}(X_t - \hat{f}_t^{N'+1})^2$ and the results were much less encouraging than those for WBS2.SDLL. For the same reason, SDLL does not appear to work well when coupled with the standard binary segmentation.

While dimension jump and slope estimation are approaches to choosing a suitable penalty constant (which is then used for model choice), SDLL is a direct model selector. In this latter category is also the proposal of Lavielle (2005), who (motivated by the problem of selecting a suitable penalty constant from the data), proposes a heuristic algorithm for estimating the number of change-points. The proposed estimator of N is the largest value of N' for which the second derivative of the (normalised) empirical risk $\sum_{t=1}^{T}(X_t - \hat{f}_t^{N'})^2$ is greater than a certain threshold. Other that the difference in how this approach and WBS2.SDLL measure the importance of each change-point (see the discussion in the previous paragraph), the two model selectors: SDLL and Lavielle (2005) use fundamentally different functionals of their corresponding solution paths, with Lavielle (2005) (in contrast to SDLL) requiring the provision of the maximum number of change-points and a threshold parameter whose value appears critical to the success of the procedure; the theoretical properties of the method in Lavielle (2005) are not investigated.

4 Practicalities and comparative simulation study

4.1 Practicalities

This section discusses our default choices and recommendations for the parameters of the WBS2 solution path procedure with the SDLL model selection; as before, we refer to this combination as WBS2.SDLL.

The number $\tilde{M}$ of interval draws. $\tilde{M}$, the maximum number of random interval draws in each recursive execution of the WBS2.Sol.Path routine defined in Section 3.1.1, is the only parameter of the WBS2 solution path. In all numerical examples in this paper, we use $\tilde{M} = 100$. This value has not been optimised and it is likely that other values may lead to even better performance.

The estimator $\hat{\sigma}_T$ of σ. We use the MAD estimator for Gaussian data, with $\{2^{-1/2}(X_{t+1} - X_t)\}_{t=1}^{T+1}$ on input. In our experience, this estimator generally behaves well in infrequent and moderate number of change-point scenarios. Section 2.3.2 shows that it may not be a very accurate estimator of σ in frequent change-point settings, but this is of little relevance as Theorem 3.2 shows that we only require an estimator for which $\hat{\sigma}_T/\sigma$ is bounded (which even allows for inconsistent estimators).

The threshold constant $\tilde{C}$. Recall that the threshold $\tilde{\zeta_T}$, used in the SDLL model selection procedure, is of the form $\tilde{C}\hat{\sigma}_T\{2\log T\}^{1/2}$. We numerically calibrate the multiplicative threshold constant $\tilde{C}$ over a range of sample sizes T, so that the WBS2.SDLL procedure correctly estimates zero change-points for signals that have none, in either 90% or 95% of the simulated sample paths, based on 1000 simulations. For the remaining sample sizes within the range of T considered, we extrapolate $\tilde{C}$ linearly; for any sample sizes outside

the range of T considered, we extrapolate $\tilde{C}$ as continuous and constant. For the 90% level, the values of $\tilde{C}$ range from 1.42 for $T = 10$ and under, to 1.135 for $T = 10000$ and above. For the 90% level, they range from 1.55 for $T = 10$ and under to 1.17 for $T = 10000$ and above. We refer to the two versions of the procedure as, respectively, WBS2.SDLL(0.9) and WBS2.SDLL(0.95). To obtain $\tilde{C}$ for any T, we refer the reader to our R code at `https://github.com/pfryz/wild-binary-segmentation-2.0`.

Constant β. Recall from the definition of the WBS2.SDLL routine in Section 3.2 that SDLL performs model selection over those model sizes for which the corresponding WBS2 absolute CUSUM statistics are of magnitude $\beta\tilde{\zeta}_T$ or larger. We have tested β is the region 0.3–0.6 and have found that the SDLL model choice is relatively robust to the values of β within this range. We use $\beta = 0.3$ in all examples of this paper and this is also the default value in our code.

Reducing the randomness of the output of WBS2.SDLL. WBS2 is a random procedure, in the sense that it relies on the randomly drawn intervals $[s_m, e_m]$. As a result, the output of the WBS2.SDLL procedure can vary across executions on the same dataset. This variation can be particularly pronounced for datasets for which the signal to noise ratio is very low. Two naive ways to reduce or eliminate the randomness of the output of WBS2.SDLL might be: to set a fixed random seed before the execution of the procedure, or to use a fixed grid for the interval draws. Fixing the seed is risky as the user has no knowledge of what particular seeds may lead to favourable interval draws. On the other hand, using a fixed uniform grid makes it impossible to request any given number $\tilde{M}$ of intervals as this is constrained by the spacing of the grid. Also, this normally excludes short intervals (which can be good for detecting finer-scale change-points) unless $\tilde{M}$ is very large, which tends not to be the case in WBS2.

Instead, we propose the following solution:

1. run the WBS2.SDLL procedure R times;
2. choose the run that leads to the median number of change-points across the R runs;
3. return this "median run" along with the ensemble of estimated change-point locations pooled across all R runs.

Our experiments (with $R = 9$), not reported in this paper, suggest that the median run is remarkably stable with respect to the number and the locations of estimated change-points, and tends to show only very little, if any, randomness. Also, the pooled ensemble of change-point estimates can be used to construct the histogram of all change-point locations detected across the R runs, which provides an interesting informal visualisation of the significance of the estimated change-point locations. We provide an implementation of this procedure in our R code at `https://github.com/pfryz/wild-binary-segmentation-2.0`. However, to save computation time, in the remainder of the paper we consider the standard WBS2.SDLL procedure, without this extra randomness-reduction feature.

4.2 Comparative simulation study

In the first part of the simulation study, we exhibit the finite-sample performance of the WBS2.SDLL method on signals with small/moderate numbers of change-points, and compare it with that of the competitors defined in Section 2.1. Our test models are defined in

Appendix B of Fryzlewicz (2014) and are as follows: (1) the `blocks` model (length 2048, 11 change-points), (2) the `fms` model (length 497, 6 change-points), (3) the `mix` model (length 560, 13 change-points), (4) the `teeth10` model (length 140, 13 change-points), (5) the `stairs10` model (length 150, 14 change-points). For techniques that require the specification of the maximum number of change-points, we set this to the (rounded) length of the input signal divided by three; in all our examples, the true number of change-points is (well) within the interior of this range. We set the random seed to 1 before running 100 simulations for each of the test signals and each of the competitors.

The average absolute error in estimating the number N of change-points, denoted by $\hat{E}|\hat{N} - N|$, is an easy-to-interpret error measure which, in our experience, tends to correlate strongly with many measures of accuracy of the estimated change-point locations. In particular, for each method, let $\hat{f}_t$ be the piecewise-constant function whose value between each pair of consecutive estimated change-points is the mean of the data over the interval delimited by this pair of change-points, and let $\hat{E}(\hat{f} - f)^2$ be the mean-square error in estimating f_t, averaged across the simulations. We have found that the empirical across-method correlation between $\log\{\hat{E}|\hat{N} - N|\}$ and $\log\{\hat{E}(\hat{f} - f)^2\}$, averaged over the five models tested, is 0.799.

method	(1)	(2)	(3)	(4)	(5)
PELT+mBIC	0	1	0	0	1
PELT+BIC	1	1	0	0	1
MOSUM	1	1	0	0	1
ID	1	1	0	1	1
FDRSeg	1	1	0	0	1
S3IB	0	0	0	0	0
SMUCE	0	1	0	0	0
CUMSEG	0	0	0	0	1
FPOP	1	1	0	0	1
DDSE	1	1	1	1	1
DJUMP	1	1	1	0	1
TGUH	1	1	0	1	1
WBS-C1.0	1	0	1	1	1
WBS-C1.3	0	1	0	0	1
WBS-BIC	1	1	0	1	1
WBS2.SDLL(0.9)	1	1	0	1	1
WBS2.SDLL(0.95)	1	1	0	1	1

Table 1: Logical flags indicating the performance of each method on each of the models (1)–(5): 1 means $\hat{E}|\hat{N} - N| < 1$ (the true number of change-points is mis-estimated by less than 1 on average across 100 simulated sample paths); 0 means $\hat{E}|\hat{N} - N| \geq 1$.

Table 1 shows the (method, model) combinations for which $\hat{E}|\hat{N} - N| < 1$, i.e. the true number of change-points is mis-estimated by less than 1 on average. The WBS2.SDLL method is one of the better performers, achieving $\hat{E}|\hat{N} - N| < 1$ for four out of the five test models. The only exception is the `mix` model, the most challenging out of the five models tested, for which WBS2.SDLL(0.9) and WBS2.SDLL(0.95) achieve $\hat{E}|\hat{N} - N| = 1.41$ and 1.4, respectively.

The other techniques achieving $\hat{E}|\hat{N} - N| < 1$ for four out of the five test models are ID,

DJUMP, TGUH, WBS-C1.0 and WBS-BIC, with DDSE being the only method achieving $\hat{E}|\hat{N} - N| < 1$ for all five test models.

However, in the second part of the simulation study, we show that the impressive performance of DDSE does not appear to extend to signals with frequent change-points. We first provide, in Table 2, the full simulation results for the `extreme.teeth` example from Section 2.1, for which $N = 199$. The two WBS2.SDLL methods are clear winners; relative to them, the large value of $\hat{E}(\hat{N} - N)^2$ for DDSE originates from the fact that DDSE estimates $\hat{N} = 0$ for two out of 100 sample paths. All other methods perform consistently poorly.

method	$\hat{E}(\hat{N}) - N$	$\hat{E}\lvert\hat{N} - N\rvert$	$\hat{E}(\hat{N} - N)^2$	$\hat{E}(\hat{f} - f)^2$	time
PELT+mBIC	-198.97	198.97	39589.09	0.250	0.001
PELT+BIC	-193.23	193.23	37392.13	0.247	0.001
MOSUM	-199.00	199.00	39601.00	0.250	0.007
ID	-186.00	186.00	35842.88	0.239	0.039
FDRSeg	-139.32	139.32	22568.66	0.197	1.329
S3IB	-199.00	199.00	39601.00	0.250	1.738
SMUCE	-192.47	192.47	37057.27	0.251	0.006
CUMSEG	-199.00	199.00	39601.00	0.250	1.282
FPOP	-193.23	193.23	37392.13	0.247	0.001
DDSE	-3.71	6.49	803.19	0.051	0.721
DJUMP	-199.00	199.00	39601.00	0.250	0.069
TGUH	-132.38	132.38	18543.98	0.194	0.061
WBS-C1.0	-160.84	160.84	25935.10	0.242	0.083
WBS-C1.3	-193.70	193.70	37531.66	0.250	0.083
WBS-BIC	-199.00	199.00	39601.00	0.250	0.517
WBS2.SDLL(0.9)	-0.52	3.52	26.42	0.049	0.193
WBS2.SDLL(0.95)	-0.08	3.22	17.20	0.049	0.191

Table 2: Various measures of performance for the competing methods on the `extreme.teeth` example with $\sigma = 0.3$, averaged over 100 simulated sample paths. "Time" denotes the average execution time in seconds (in R, on a 2015 iMac); the other column headings are self-explanatory.

We further consider a slightly altered set-up in which the signal under consideration has more frequent change-points, but less noise. The signal, labelled `extreme.extreme.teeth`, is one in which the pattern $0, 0, 0, 0, 1, 1, 1$ repeats itself 100 times, so that we have $T = 700$ and $N = 199$. We use $\varepsilon_t \sim N(0, 0.2^2)$ (rather than the $\varepsilon_t \sim N(0, 0.3^2)$ used in the `extreme.teeth` example). See Figure 5 for an illustration. The full simulation results are in Table 3. This time the two WBS2.SDLL methods are the only ones with acceptable performance and all the other methods, along with DDSE, fail somewhat dramatically.

We close with a few further comparative remarks regarding DDSE and WBS2.SDLL. Unlike WBS2.SDLL, the DDSE method requires the specification of the maximum number of change-points and this was set to $[T/3]$, but we also tried $T - 1$ and the results did not improve. Importantly, DDSE appears to scale poorly: the execution on a signal of length 10^4 took 22 seconds, was interrupted by us on a signal of length 5×10^4 after several minutes, and we were unable to execute DDSE on a signal of length 10^5 because of what we believe were memory-related issues. By contrast, the corresponding execution times for

WBS2.SDLL were: 3.7 sec ($T = 10^4$), 15 sec ($T = 5 \times 10^4$), 30 sec ($T = 10^5$), which suggests empirical computation times close to $O(T)$.

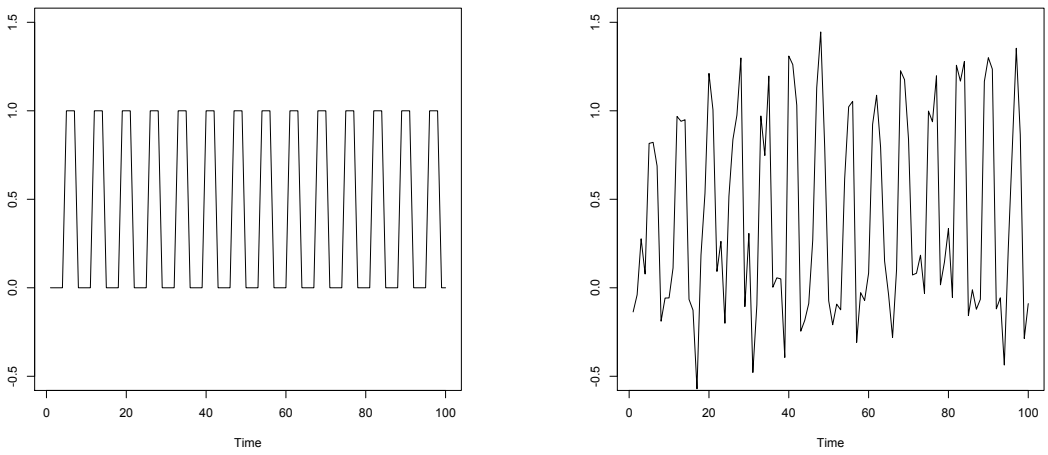

Figure 5: The first 100 observations of the `extreme.extreme.teeth` signal (which continues in the same manner for $t = 1, \ldots, 700$); left: with no noise; right: with additive i.i.d. Gaussian noise with mean zero and $\sigma = 0.2$.

4.3 Performance for heavy-tailed noise

In this section, we illustrate the performance of WBS2.SDLL in the presence of heavy-tailed noise. The noise distribution is not known to the algorithm, and it proceeds as it would if the noise were Gaussian. We consider the `extreme.teeth` signal and use the following noise distributions: $t_{2.5}$, t_5, t_7, t_{10} and Gaussian, all scaled, as in Section 2.1, to have the standard deviation of 0.3.

Figure 6 shows the results. We are encouraged by the fact that the upward bias in the estimation of N appears to be rather small even in the case of $t_{2.5}$, which is strongly heavy-tailed.

5 London House Price Index

We consider monthly percentage changes in the UK House Price Index (UKHPI), for all property types, in the 32 London boroughs (the 32 local authority districts that make up the Greater London county). We exclude the City of London as its resident population is much smaller than in any of the 32 boroughs and, as a result, the monthly fluctuations in the UKHPI for the City of London are much more variable. As of December 2018, the data are available from `http://landregistry.data.gov.uk/app/ukhpi`. The 32-dimensional time series $X_t = (X_t^{(1)}, \ldots, X_t^{(32)})'$ runs from February 1995 to September 2018, so that $t = 1, \ldots, T = 284$. In the order of the indexing ($i = 1, \ldots, 32$), the boroughs are: Barking and Dagenham, Barnet, Bexley, Brent, Bromley, Camden, City of Westminster, Croydon, Ealing, Enfield, Greenwich, Hackney, Hammersmith and Fulham, Haringey, Harrow, Havering, Hillingdon, Hounslow, Islington, Kensington and Chelsea, Kingston upon Thames,

method	$\hat{E}(\hat{N}) - N$	$\hat{E}\lvert\hat{N} - N\rvert$	$\hat{E}(\hat{N} - N)^2$	$\hat{E}(\hat{f} - f)^2$	time
PELT+mBIC	-198.97	198.97	39589.09	0.245	0.001
PELT+BIC	-189.95	189.95	36238.03	0.237	0.001
MOSUM	-199.00	199.00	39601.00	0.245	0.007
ID	-173.64	173.64	32457.76	0.220	0.026
FDRSeg	-140.77	140.77	24744.07	0.181	0.460
S3IB	-199.00	199.00	39601.00	0.245	0.865
SMUCE	-195.42	195.42	38197.30	0.245	0.006
CUMSEG	-199.00	199.00	39601.00	0.245	0.554
FPOP	-189.95	189.95	36238.03	0.237	0.001
DDSE	-167.16	167.28	33265.02	0.208	0.439
DJUMP	-199.00	199.00	39601.00	0.245	0.033
TGUH	-84.92	84.92	8692.78	0.124	0.048
WBS-C1.0	-154.47	154.47	23939.55	0.230	0.068
WBS-C1.3	-192.37	192.37	37027.93	0.244	0.068
WBS-BIC	-199.00	199.00	39601.00	0.245	0.438
WBS2.SDLL(0.9)	0.26	0.76	1.92	0.017	0.117
WBS2.SDLL(0.95)	0.31	0.71	1.71	0.017	0.116

Table 3: Various measures of performance for the competing methods on the `extreme.extreme.teeth` example with $\sigma = 0.2$, averaged over 100 simulated sample paths. "Time" denotes the average execution time in seconds (in R, on a 2015 iMac); the other column headings are self-explanatory.

Lambeth, Lewisham, Merton, Newham, Redbridge, Richmond upon Thames, Southwark, Sutton, Tower Hamlets, Waltham Forest, and Wandsworth.

As a preliminary dimension-reduction step, we run the PCA on a version of X_t scaled so that the empirical second moment of each of the 32 components is one; the scaling appears necessary due to the fact that the empirical variances of $X_t^{(i)}$ range from 0.97 (for Bromley) to 9.02 (for Kensington and Chelsea). The scree plot is shown in the top left plot of Figure 8 and appears to offer visual evidence that the 2nd principal component may be of importance in explaining some of the contemporaneous relationship between the components of X_t beyond the contemporaneous mean (the 1st PC is close to the contemporaneous average of $X_t^{(i)}$ and we do not discuss it here).

Let $\omega_i^{(2)}$ denote the loading of $X_t^{(i)}$ in the 2nd principal component of X_t. The 2nd PC appears to furnish an interesting and interpretable separation of X_t according to the sign of $\omega_i^{(2)}$. Let $\mathcal{U}, \mathcal{V}$ be two complementary subsets of $\mathcal{J} = \{1, 2, \ldots, 32\}$ corresponding to the opposing signs of $\omega_i^{(2)}$; that is, $\mathcal{U} \cup \mathcal{V} = \mathcal{J}$; $\mathcal{U} \cap \mathcal{V} = \emptyset$ and $\forall\, i \in \mathcal{U}\ \forall\, j \in \mathcal{V}\ \omega_i^{(2)}\omega_j^{(2)} < 0$. The assignment of boroughs to sets $\mathcal{U}$ and $\mathcal{V}$ is in Table 4, and it is visualised in Figure 9. It is interesting to observe that the 16 boroughs forming part of set $\mathcal{U}$, which we refer to as "Inner(PC_2)" below, overlap very strongly with London's most expensive boroughs by average house price as of September 2018 (source: `https://www.gov.uk/government/publications/uk-house-price-index-england-september-2018/uk-house-price-index-england-september-2018`). We refer to the boroughs in set $\mathcal{V}$ as "Outer(PC_2)". The Inner(PC_2) form a spatially contiguous area; the Outer(PC_2) boroughs are made up of two such areas. In summary, the 2nd PC appears to act as a separator between the relatively expensive and inexpensive London

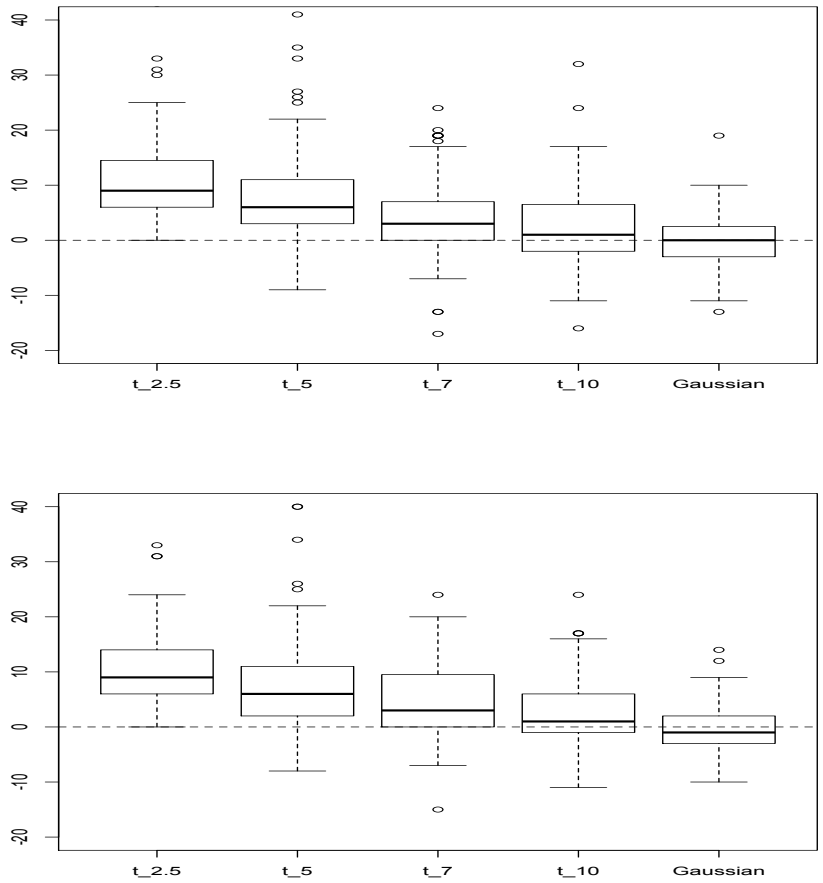

Figure 6: From top to bottom and left to right: boxplots of $\hat{N} - N$ over 100 simulated sample paths, for the noisy `extreme.teeth` signal, for WBS2.SDLL(0.9) [top] and WBS2.SDLL(0.95) [bottom] and the following noise distributions, each with standard deviation 0.3: $t_{2.5}$, t_5, t_7, t_{10} and Gaussian.

boroughs.

We form separate contemporaneous averages of X_t over the sets $\mathcal{U}$ and $\mathcal{V}$, i.e. corresponding to the Inner(PC_2) and Outer(PC_2) boroughs, respectively. We denote

$$U_t = \frac{1}{16} \sum_{i \in \mathcal{U}} X_t^{(i)}, \qquad V_t = \frac{1}{16} \sum_{i \in \mathcal{V}} X_t^{(i)}.$$

The scatter plot of U_t versus V_t is shown in the top right plot of Figure 8. We model both U_t and V_t as piecewise-constant signals observed with noise. We are interested in a comparative analysis of the estimated change-point locations in $E(U_t)$ and $E(V_t)$ and the behaviour of the estimates of $E(U_t)$ and $E(V_t)$ between the estimated change-points. To this end, we use the WBS2.SDLL(0.9) method. The resulting piecewise-constant estimates of $E(U_t)$ and $E(V_t)$, denoted, respectively, by $\hat{E}(U_t)$ and $\hat{E}(V_t)$, are shown in Figure 7 (together with the data U_t and V_t, respectively) and in the middle left plot of Figure 8 (without the data). Both $\hat{E}(U_t)$ and $\hat{E}(V_t)$ contain frequent change-points (38 and 33, respectively, in a sample of

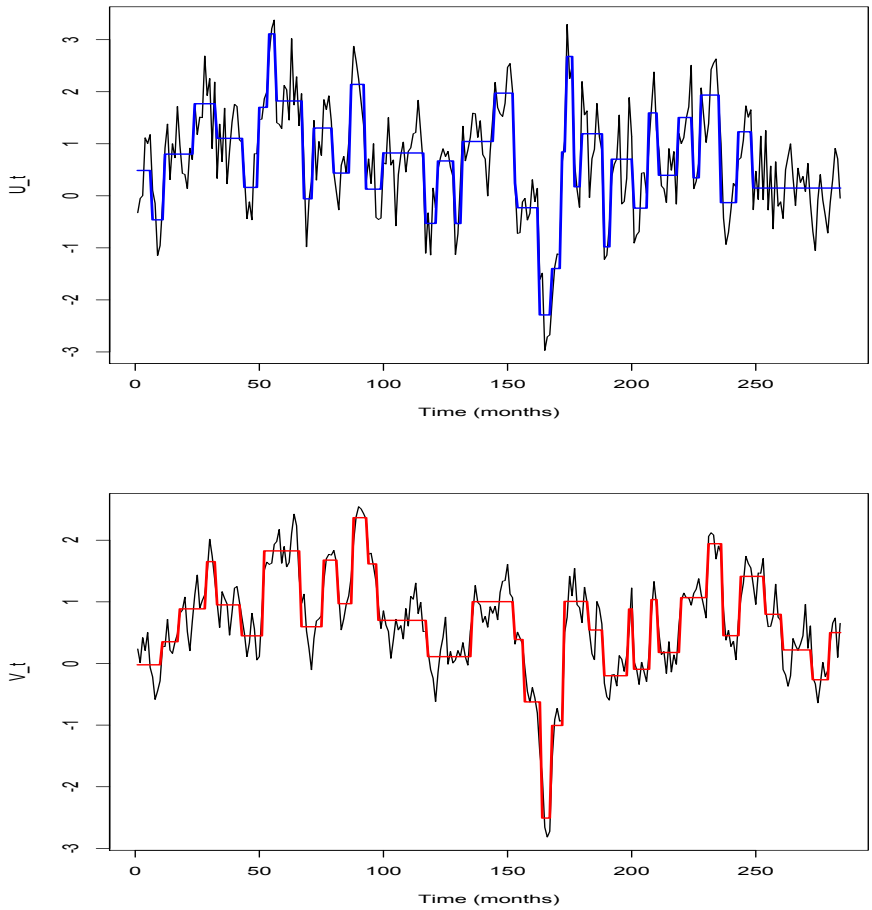

Figure 7: Top: U_t and $\hat{E}(U_t)$ (blue); bottom: V_t and $\hat{E}(V_t)$ (red).

length 284), which justifies the use of WBS2.SDLL. The bottom plots in Figure 8 show the sample autocorrelation functions of the empirical residuals $U_t - \hat{E}(U_t)$ and $V_t - \hat{E}(V_t)$ from the respective piecewise-constant fits. The estimated residuals display only a limited degree of autocorrelation. The piecewise-constant fits also remove practically all contemporaneous correlation between U_t and V_t: the sample correlation between U_t and V_t is 0.7, while the sample correlation between $U_t - \hat{E}(U_t)$ and $V_t - \hat{E}(V_t)$ is 0.06. It is tempting to conclude that the piecewise-constant modelling of $E(U_t)$ and $E(V_t)$ with frequent change-points leads to reasonable goodness-of-fit.

The British mainstream press and professional bodies regularly comment on the perceived link between the outcome of the 2016 UK European Union membership referendum and the evolution of house prices in London and elsewhere in the UK, see e.g. `https://www.theguardian.com/business/2018/dec/13/uk-property-market-at-weakest-since-2012-as-brexit-takes-toll-rics`, amongst a large amount of material on this topic. We investigate the behaviour of $\hat{E}(U_t)$ and $\hat{E}(V_t)$ between June 2016, the month in which the referendum was held, and September 2018; this corresponds to the time range $t = 257, \ldots, 284$ in our data. The corresponding illustration is in the middle right plot of Figure 8. It is interesting to observe that $\hat{E}(U_t)$, the estimated average monthly percentage change in price for the Inner(PC_2) boroughs,

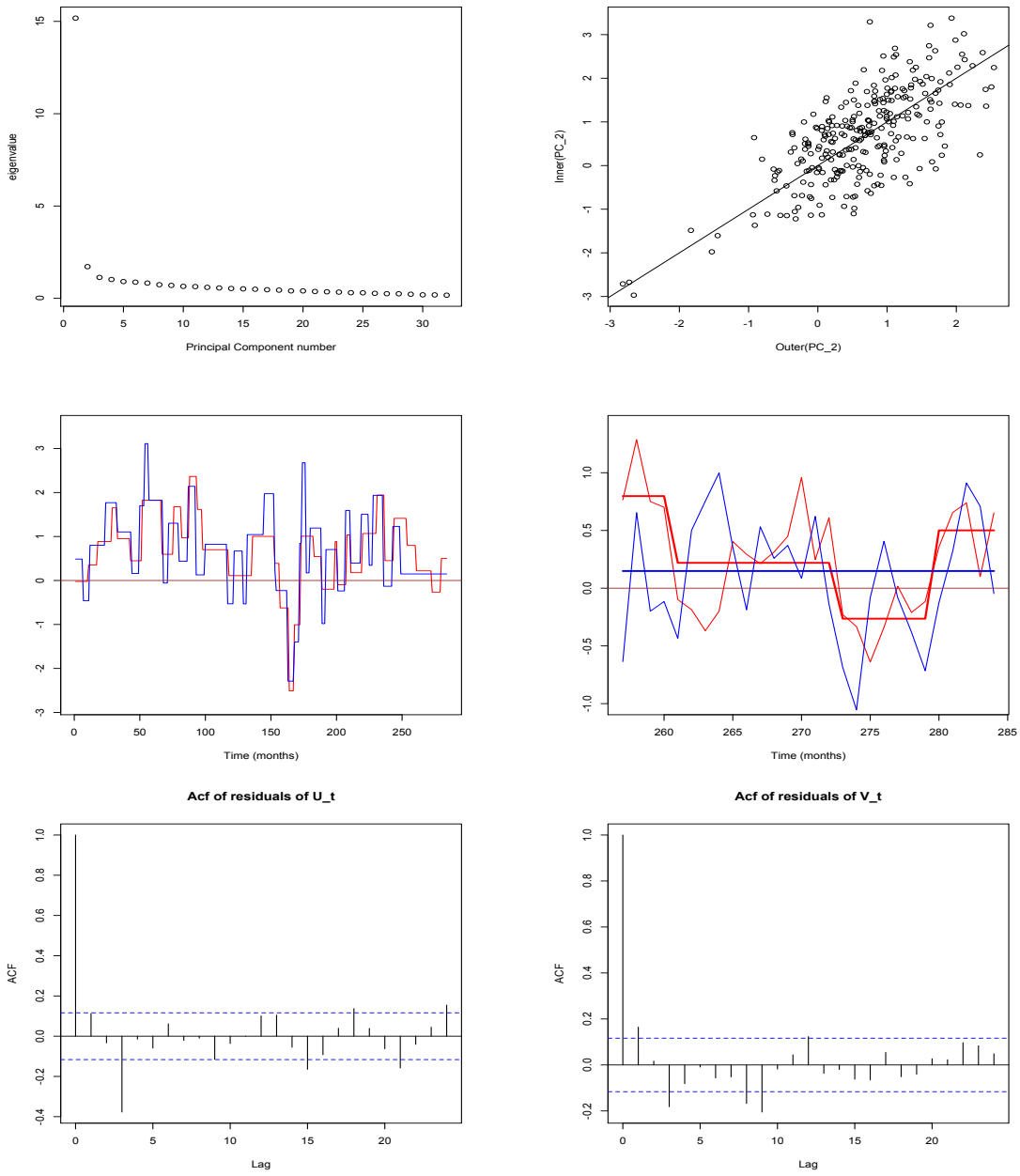

Figure 8: Top left: scree plot for the PCA of the (scaled) X_t; top right: scatter plot of U_t versus V_t; middle left: $\hat{E}(U_t)$ (blue) and $\hat{E}(V_t)$ (red); middle right: U_t (thin blue), $\hat{E}(U_t)$ (thick blue), V_t (thin red), $\hat{E}(V_t)$ (thick red) [note: the time ranges in the two middle row plots are different]; bottom left: sample acf of $U_t - \hat{E}(U_t)$; bottom right: sample acf of $V_t - \hat{E}(V_t)$. See Section 5 for details.

Indexing set	List of boroughs	Notation
$\mathcal{U}$	Barnet, Camden, City of Westminster, Ealing, Hackney, Hammersmith and Fulham, Haringey, Hounslow, Islington, Kensington and Chelsea, Kingston upon Thames, Lambeth, Merton, Richmond upon Thames, Southwark, Wandsworth	Inner(PC_2)
$\mathcal{V}$	Barking and Dagenham, Bexley, Brent, Bromley, Croydon, Enfield, Greenwich, Harrow, Havering, Hillingdon, Lewisham, Newham, Redbridge, Sutton, Tower Hamlets, Waltham Forest	Outer(PC_2)

Table 4: The memberships of the sets $\mathcal{U}$ (Inner(PC_2) boroughs) and $\mathcal{V}$ (Outer(PC_2) boroughs). See Section 5 for details.

contains no change-points over this time period (and has the numerical value of 0.15), while $\hat{E}(V_t)$, the same quantity for the Outer(PC_2) boroughs, shows three change-points which take it from the positive value of 0.8 to the negative value of -0.26, before a rebound which takes it to the level of 0.5. These three most recent change-points in $\hat{E}(V_t)$ occur in October 2016 (the level changes from 0.8 to 0.22), October 2017 (from 0.22 to -0.26) and May 2018 (from -0.26 to 0.5). We mention that the larger number of change-points in $\hat{E}(V_t)$ than in $\hat{E}(U_t)$ over this particular time period is in contrast to the fact that it is $\hat{E}(U_t)$ that has more change-points in total (38 to $\hat{E}(V_t)$'s 33). We conclude that the average house price growth in the Inner(PC_2) and Outer(PC_2) boroughs appears to have followed different paths in the time period since the referendum: the average growth in the Inner(PC_2) boroughs has remained relatively slow but constant, while the average growth in the Outer(PC_2) boroughs had moved from a relatively high value into the negative territory before recovering considerably in the recent months.

6 Discussion

We emphasise the modular character of the proposed WBS2.SDLL change-point detection methodology: one ingredient is the WBS2 solution path, and the other is the SDLL model selection criterion. The two parts have been designed with each other in mind, but it is in principle possible to use them separately; that is, to use WBS2 with an alternative model selection criterion, and to attempt to adapt the SDLL model selection principle to other settings, including ones unrelated to change-point detection.

This paper introduces WBS2 and SDLL in the possibly simplest change-point detection setting, in which a one-dimensional signal is observed in i.i.d. Gaussian noise; part of this paper's message that even this simplest framework was traditionally challenging for the state of the art. We believe that WBS2.SDLL fills an important gap in the literature by providing a solution to this canonical problem in frequent-change-point scenarios. Just as the WBS procedure was extended from the canonical signal + i.i.d. Gaussian noise setting

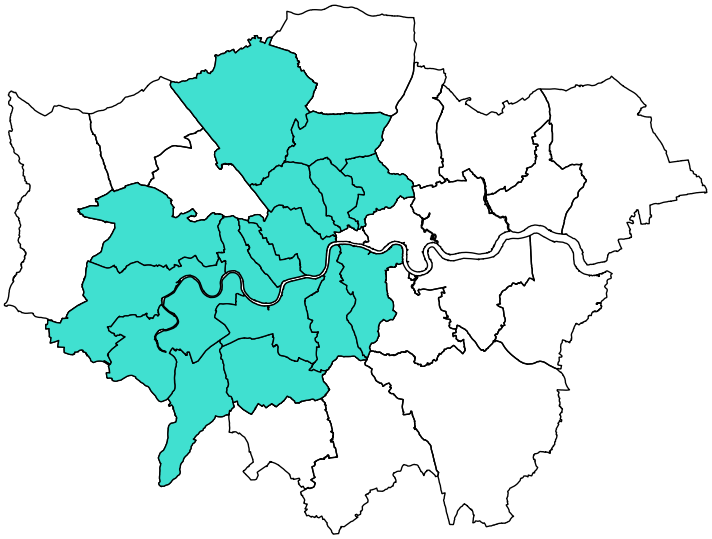

Figure 9: London boroughs. Turquoise: the Inner(PC_2) boroughs (included in set $\mathcal{U}$); white: the Outer(PC_2) boroughs (included in set $\mathcal{V}$). See Table 4 for a list, and Section 5 for details.

to univariate time series (Korkas and Fryzlewicz, 2017) and high-dimensional panel data (Wang and Samworth, 2018), we envisage that WBS2's and SDLL's simplicity means that they are both extendible to these and other stochastic models.

A Proofs

Proof of Theorem 3.1.

Part (i). The statement is trivially true if $T = 1$. Suppose, inductively, that it is true for data lengths $1, \ldots, T-1$. For an input data sequence of length T, the first addition to the solution path breaks the domain of operation into two, of lengths b_{m_0} and $T - b_{m_0}$. Therefore by the inductive hypothesis, the length of the solution path for the input data will be $1 + (b_{m-0} - 1) + (T - b_{m_0} - 1) = T - 1$, which completes the proof of part (i).

Part (ii). The computation of the CUSUM statistic for all b in formula (2) is of computational order $O(e - s)$. Therefore the addition of elements to the solution path from all sub-domains within a single scale of operation is of computational order $O(\tilde{M}T)$. If the

scale has reached J, there is nothing to do and the procedure has stopped. Therefore the overall computational complexity before the sorting step is of order $O(\tilde{M}JT)$. The sorting can be performed in computational time $O(T \log T)$, which never dominates $O(\tilde{M}JT)$ as the smallest possible J is of computational order $O(\log T)$, which is straightforward to see from its definition. This completes the proof of part (ii).

Part (iii). We first set the probabilistic framework in which we analyse the behaviour of the WBS2 solution path algorithm. With the change-point locations denoted by $\eta_1, \ldots, \eta_N$ (with the additional notation $\eta_0 = 1$, $\eta_{N+1} = T + 1$), we define the intervals

$$I_{i,j}^{k,l} = [\max(1, \eta_i + k), \min(T, \eta_j + l)],$$

where $k, l \in \mathbb{Z} \cap [-C_1(\Delta)(\underline{f}_T)^{-2} \log T, C_1(\Delta)(\underline{f}_T)^{-2} \log T]$ for each $1 \leq i + 1 < j \leq N + 1$. Suppose that on each interval $I_{i,j}^{k,l}$, $\tilde{M}$ intervals $\{[s_m, e_m]\}_{m=1}^{\tilde{M}}$ have been drawn, with the start- and end-points having been drawn uniformly with replacement from $I_{i,j}^{k,l}$ (if $\tilde{M} > |I_{i,j}^{k,l}|(|I_{i,j}^{k,l}| - 1)/2$, then we understand $\{[s_m, e_m]\}_{m=1}^{\tilde{M}}$ to contain all possible subintervals of $I_{i,j}^{k,l}$). Note we do not reflect the (stochastic) dependence of s_m, e_m on i, j, k, l so as not to over-complicate the notation. As in the proof of Theorem 3.2 in Fryzlewicz (2014), we define intervals $\mathcal{I}_r$ between change-points in such a way that their lengths are at least of order $O(T)$, and they are separated from the change-points also by distances at least of order $O(T)$. To fix ideas, define $\mathcal{I}_r = [\eta_{r-1} + \frac{1}{3}(\eta_r - \eta_{r-1}), \eta_{r-1} + \frac{2}{3}(\eta_r - \eta_{r-1})]$, $r = 1, \ldots, N + 1$. For each interval $I_{i,j}^{k,l}$, we are interested in the following event

$$A_{i,j}^{k,l} = \{\exists_{m_0 \in \{1,\ldots,\tilde{M}\}} \ \exists_{r \in \{i+1,\ldots,j-1\}} \ (s_{m_0}, e_{m_0}) \in \mathcal{I}_r \times \mathcal{I}_{r+1}\}.$$

Note that

$$\begin{aligned} P\{(A_{i,j}^{k,l})^c\} &\leq \prod_{m=1}^{\tilde{M}} P\left\{(s_m, e_m) \notin \bigcup_{r=i+1}^{j-1} \mathcal{I}_r \times \mathcal{I}_{r+1}\right\} \\ &\leq \prod_{m=1}^{\tilde{M}} \max_{r \in \{i+1,\ldots,j-1\}} (1 - P\{(s_m, e_m) \in \mathcal{I}_r \times \mathcal{I}_{r+1}\}) \leq (1 - \delta^2/9)^{\tilde{M}}, \end{aligned}$$

and hence

$$P\left(\bigcap_{i,j,k,l} A_{i,j}^{k,l}\right) \geq 1 - \sum_{i,j,k,l} P\{(A_{i,j}^{k,l})^c\} \geq 1 - \frac{1}{2} N(N+1)(2C_1(\Delta)(\underline{f}_T)^{-2} \log T + 1)^2 (1 - \delta^2/9)^{\tilde{M}}.$$

Define further the following event

$$\mathcal{D}_{\Delta,T} = \left\{\forall\ 1 \leq s \leq b < e \leq T \qquad |\tilde{\varepsilon}_{s,e}^b| \leq 2\sigma\{(1 + \Delta) \log T\}^{1/2}\right\},$$

where $\tilde{\varepsilon}_{s,e}^b$ is defined as in formula (2) with ε_t in place of X_t. It is the statement of Lemma A.1 in Fryzlewicz (2018) that $\mathcal{B}_{\Delta,T} \subseteq \mathcal{D}_{\Delta,T}$.

The following arguments apply on the set $\bigcap_{i,j,k,l} A_{i,j}^{k,l} \cap \mathcal{D}_{\Delta,T}$ for any fixed $\Delta > 0$. Proceeding exactly as in the proof of Theorem 3.2 in Fryzlewicz (2014), at the start of the procedure we have $s = 1$, $e = T$, and in view of the fact that we are on $A_{0,N+1}^{0,0} \cap \mathcal{D}_{\Delta,T}$, the procedure finds a $b_{m_0} \in [s_{m_0}, e_{m_0}] \subseteq [s, e]$ such that

(a) there exists an $r \in \{1, 2, \ldots, N\}$ such that $|\eta_r - b_{m_0}| \leq C_1(\Delta)(\underline{f}_T)^{-2} \log T$, and

(b) $|\tilde{X}^{b_{m_0}}_{s_{m_0}, e_{m_0}}| \gtrsim T^{1/2} \underline{f}_T$,

where the $\gtrsim$ symbol means "of the order of or larger". Again by arguments identical to those in the proof of Theorem 3.2 in Fryzlewicz (2014), on each subsequent segment containing previously undetected change-points, we are on one of the sets $A^{k,l}_{i,j} \cap \mathcal{D}_{\Delta,T}$, and therefore the procedure again finds a b_{m_0} satisfying properties (a), (b) above for a certain previously undetected change-point η_r, until all change-points have been identified in this way. Once all change-points have been identified, by Lemma A.5 of Fryzlewicz (2014), all maximisers b_{m_0} of the absolute CUSUM statistics $|\tilde{X}^b_{s_m, e_m}|$ (over b) are such that $|\tilde{X}^{b_{m_0}}_{s_{m_0}, e_{m_0}}| \leq C_2(\Delta) \log^{1/2} T$. As $\log^{1/2} T = o(T^{1/2} \underline{f}_T)$, for T large enough, the sorting of the elements of $\tilde{\mathcal{P}}$ does not move any elements from the first N to the last $T - 1 - N$ or vice versa, which completes the proof of part (iii). □

Proof of Theorem 3.2.

With the notation as in the statement of the theorem and in the proof of Theorem 3.1, the following arguments apply on the set $\bigcap_{i,j,k,l} A^{k,l}_{i,j} \cap \mathcal{D}_{\Delta,T} \cap \Theta_{\theta,T}$ for any fixed $\Delta > 0$. Let T be large enough for the assertion of Theorem 3.1 to hold. Further, let $\tilde{C}$ be large enough for the following inequality to hold

$$\tilde{\zeta}_T = \tilde{C} \hat{\sigma}_T \{2 \log T\}^{1/2} > \max(C_2(\Delta) \log^{1/2} T, 2\sigma\{(1 + \Delta) \log T\}^{1/2}) \tag{6}$$

(this is always possible on $\Theta_{\theta,T}$). If $N = 0$, then in view of inequality (6) and the fact that we are on $\mathcal{D}_{\Delta,T}$, from the definition of the SDLL algorithm we necessarily have $\hat{N} = 0$. If $N > 0$, then by part (iii) of Theorem 3.1, we must have $K + 1 \geq N$. If $|\tilde{X}^{b_{K+1}}_{s_{K+1}, e_{K+1}}| > \tilde{\zeta}_T$, then by inequality (6), we have $N = K + 1$ and the SDLL procedure correctly identifies $\hat{N} = K + 1 = N$. If $|\tilde{X}^{b_{K+1}}_{s_{K+1}, e_{K+1}}| < \tilde{\zeta}_T$, then we necessarily have $N < K + 1$, and by part (iii) of Theorem 3.1, we have, for $Z_k = \log |\tilde{X}^{b_k}_{s_k, e_k}| - \log |\tilde{X}^{b_{k+1}}_{s_{k+1}, e_{k+1}}|$,

$$\begin{aligned} Z_N &\sim \log T, \\ Z_k &\leq \log \left(\frac{(1 + \theta) C_2(\Delta)}{\beta \tilde{C} \sigma 2^{1/2}} \right) \quad \text{for} \quad k = N + 1, \ldots, K \quad \text{if this range is non-empty.} \end{aligned}$$

Therefore, from the definition of the SDLL and by part (iii) of Theorem 3.1, for T large enough, $\hat{N} = N$ will be chosen. This completes the proof of the Theorem. □

References

A. Amiri and S. Allahyari. Change point estimation methods for control chart postsignal diagnostics: A literature review. *Quality and Reliability Engineering International*, 28: 673–685, 2012.

A. Anastasiou and P. Fryzlewicz. Detecting multiple generalized change-points by isolating single ones. *Preprint*, 2018a.

A. Anastasiou and P. Fryzlewicz. *IDetect: Detecting multiple generalized change-points by isolating single ones*, 2018b. URL `https://CRAN.R-project.org/package=IDetect`. R package version 1.0.

E. Andreou and E. Ghysels. Detecting multiple breaks in financial market volatility dynamics. *Journal of Applied Econometrics*, 17:579–600, 2002.

S. Arlot. Minimal penalties and the slope heuristics: a survey. *Journal de la Societe Française de Statistique*, 160:1–106, 2019.

S. Arlot, V. Brault, J.-P. Baudry, C. Maugis, and B. Michel. *capushe: CAlibrating Penalities Using Slope HEuristics*, 2016. URL `https://CRAN.R-project.org/package=capushe`. R package version 1.1.1.

J. Bai. Estimating multiple breaks one at a time. *Econometric Theory*, 13:315–352, 1997.

J. Bai and P. Perron. Computation and analysis of multiple structural change models. *Journal of Applied Econometrics*, 18:1–22, 2003.

R. Baranowski and P. Fryzlewicz. *wbs: Wild Binary Segmentation for Multiple Change-Point Detection*, 2015. URL `https://CRAN.R-project.org/package=wbs`. R package version 1.3.

R. Baranowski, Y. Chen, and P. Fryzlewicz. Narrowest-Over-Threshold detection of multiple change-points and change-point-like features. *J. Roy. Stat. Soc. Ser. B*, 81:649–672, 2019.

J.-P. Baudry, C. Maugis, and B. Michel. Slope heuristics: overview and implementation. *Statistics and Computing*, 22:455–470, 2012.

L. Birgé and P. Massart. Gaussian model selection. *J. European Math. Soc.*, 3:203–268, 2001.

L. Birgé and P. Massart. Minimal penalties for Gaussian model selection. *Prob. Th. Rel. Fields*, 138:33–73, 2007.

D. Bosq. *Nonparametric Statistics for Stochastic Processes*. Springer-Verlag, New York, 2 edition, 1998.

L. Boysen, A. Kempe, V. Liebscher, A. Munk, and O. Wittich. Consistencies and rates of convergence of jump-penalized least squares estimators. *Annals of Statistics*, 37:157–183, 2009.

J. Braun and H.-G. Mueller. Statistical methods for DNA sequence segmentation. *Statistical Science*, 13:142–162, 1998.

J. Braun, R. Braun, and H.-G. Mueller. Multiple changepoint fitting via quasilikelihood, with application to dna sequence segmentation. *Biometrika*, 87:301–314, 2000.

B. Brodsky and B. Darkhovsky. *Nonparametric Methods in Change-Point Problems*. Kluwer Academic Publishers, 1993.

K.-M. Chen, A. Cohen, and H. Sackrowitz. Consistent multiple testing for change points. *Journal of Multivariate Analysis*, 102:1339–1343, 2011.

H. Cho and P. Fryzlewicz. Multiscale interpretation of taut string estimation and its connection to Unbalanced Haar wavelets. *Statistics and Computing*, 21:671–681, 2011.

H. Cho and P. Fryzlewicz. Multiscale and multilevel technique for consistent segmentation of nonstationary time series. *Statistica Sinica*, 22:207–229, 2012.

H. Cho and P. Fryzlewicz. Multiple change-point detection for high-dimensional time series via Sparsified Binary Segmentation. *Journal of the Royal Statistical Society Series B*, 77: 475–507, 2015.

G. Ciuperca. A general criterion to determine the number of change-points. *Statistics & Probability Letters*, 81:1267–1275, 2011.

G. Ciuperca. Model selection by LASSO methods in a change-point model. *Statistical Papers*, 55:349–374, 2014.

A. Cleynen, G. Rigaill, and M. Koskas. *Segmentor3IsBack: A Fast Segmentation Algorithm*, 2016. URL `https://CRAN.R-project.org/package=Segmentor3IsBack`. R package version 2.0.

M. D'Angelo, R. Palhares, R. Takahashi, R. Loschi, L.Baccarini, and W. Caminhas. Incipient fault detection in induction machine stator-winding using a fuzzy-Bayesian change point detection approach. *Applied Soft Computing*, 11:179–192, 2011.

P. L. Davies and A. Kovac. Local extremes, runs, strings and multiresolution. *Ann. Statist.*, 29:1–48, 2001.

R. Davis, T. Lee, and G. Rodriguez-Yam. Structural break estimation for nonstationary time series models. *Journal of the American Statistical Association*, 101:223–239, 2006.

C. Du, C.-L. Kao, and S. Kou. Stepwise signal extraction via marginal likelihood. *Journal of the American Statistical Association*, 111:314–330, 2016.

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32:407–499, 2004.

B. Eichinger and C. Kirch. A MOSUM procedure for the estimation of multiple random change points. *Bernoulli*, 24:526–564, 2018.

K. Frick, A. Munk, and H. Sieling. Multiscale change-point inference (with discussion). *Journal of the Royal Statistical Society Series B*, 76:495–580, 2014.

P. Fryzlewicz. Wild Binary Segmentation for multiple change-point detection. *Ann. Stat.*, 42:2243–2281, 2014.

P. Fryzlewicz. *breakfast: Multiple Change-Point Detection and Segmentation*, 2017. URL `https://CRAN.R-project.org/package=breakfast`. R package version 1.0.0.

P. Fryzlewicz. Tail-greedy bottom-up data decompositions and fast multiple change-point detection. *Ann. Stat.*, 46:3390–3421, 2018.

P. Fryzlewicz and S. Subba Rao. Multiple-change-point detection for auto-regressive conditional heteroscedastic processes. *Journal of the Royal Statistical Society Series B*, 76: 903–924, 2014.

E. Galceran, A. Cunningham, R. Eustice, and E. Olson. Multipolicy decision-making for autonomous driving via changepoint-based behavior prediction. In *2015 Robotics: Science and Systems Conference, RSS 2015*, volume 11, 2015.

A. Guntuboyina, D. Lieu, S. Chatterjee, and B. Sen. Adaptive risk bounds in univariate total variation denoising and trend filtering. *Ann. Stat.*, 48:205–229, 2020.

B. Hansen. The new econometrics of structural change: Dating breaks in U.S. labour productivity. *Journal of Economic Perspectives*, 15:117–128, 2001.

Z. Harchaoui and C. Lévy-Leduc. Multiple change-point estimation with a total variation penalty. *Journal of the American Statistical Association*, 105:1480–1493, 2010.

C.-Y. Huang and M. Lyu. Estimation and analysis of some generalized multiple change-point software reliability models. *IEEE Transactions on Reliability*, 60:498–514, 2011.

M. Huskova and A. Slaby. Permutation tests for multiple changes. *Kybernetika*, 37:605–622, 2001.

N. James and D. Matteson. ecp: An R package for nonparametric multiple change point analysis of multivariate data. *Journal of Statistical Software*, 62:1–25, 2014.

R. Killick, P. Fearnhead, and I. Eckley. Optimal detection of changepoints with a linear computational cost. *Journal of the American Statistical Association*, 107:1590–1598, 2012.

R. Killick, K. Haynes, and I. Eckley. *changepoint: An R package for changepoint analysis*, 2016. URL `https://CRAN.R-project.org/package=changepoint`. R package version 2.2.2.

K. Korkas and P. Fryzlewicz. Multiple change-point detection for non-stationary time series using wild binary segmentation. *Statistica Sinica*, 27:287–311, 2017.

M. Lavielle. Detection of multiple changes in a sequence of dependent variables. *Stochastic Processes and their Applications*, 83:79–102, 1999.

M. Lavielle. Using penalized contrasts for the change-point problem. *Signal Processing*, 85: 1501–1510, 2005.

M. Lavielle and E. Moulines. Least-squares estimation of an unknown number of shifts in a time series. *J. Time Ser. Anal.*, 21:33–59, 2000.

E. Lebarbier. Detecting multiple change-points in the mean of Gaussian process by model selection. *Signal Processing*, 85:717–736, 2005.

C.-B. Lee. Estimating the number of change points in a sequence of independent normal random variables. *Statistics and Probability Letters*, 25:241–248, 1995.

H. Li and A. Munk. FDR-control in multiscale change-point segmentation. *Electronic Journal of Statistics*, 10:918–959, 2016.

H. Li and H. Sieling. *FDRSeg: FDR-Control in Multiscale Change-Point Segmentation*, 2017. URL `https://CRAN.R-project.org/package=FDRSeg`. R package version 1.0-3.

K. Lin, J. Sharpnack, A. Rinaldo, and R. Tibshirani. A sharp error analysis for the fused lasso, with application to approximate changepoint screening. In *Advances in Neural Information Processing Systems*, pages 6884–6893, 2017.

D. Liu, X. Chen, Y. Lian, and Z. Lou. Impacts of climate change and human activities on surface runoff in the Dongjiang River basin of China. *Hydrological Processes*, 24: 1487–1495, 2010.

R. Maidstone, T. Hocking, G. Rigaill, and P. Fearnhead. On optimal multiple changepoint algorithms for large data. *Statistics and Computing*, 27:519–533, 2017.

C. Mallows. Another comment on O'Cinneide. *The American Statistician*, 45, 1991.

D. Matteson and N. James. A nonparametric approach for multiple change point analysis of multivariate data. *Journal of the American Statistical Association*, 109:334–345, 2014.

A. Meier, H. Cho, and C. Kirch. *mosum: Moving Sum Based Procedures for Changes in the Mean*, 2018. URL `https://CRAN.R-project.org/package=mosum`. R package version 1.2.0.

V. Muggeo. Estimating regression models with unknown break-points. *Statist. Med.*, 22: 3055–3071, 2003.

V. Muggeo. *cumSeg: Change point detection in genomic sequences*, 2012. URL `https://CRAN.R-project.org/package=cumSeg`. R package version 1.1.

V. Muggeo and G. Adelfio. Efficient change point detection for genomic sequences of continuous measurements. *Bioinformatics*, 27:161–166, 2011.

National Research Council. *Frontiers in Massive Data Analysis*. The National Academies Press, Washington, DC, 2013. doi: 10.17226/18374.

A. Olshen, E.S. Venkatraman, R. Lucito, and M. Wigler. Circular binary segmentation for the analysis of array-based DNA copy number data. *Biostatistics*, 5:557–572, 2004.

J. Pan and J. Chen. Application of modified information criterion to multiple change point problems. *Journal of Multivariate Analysis*, 97:2221–2241, 2006.

F. Pein, T. Hotz, H. Sieling, and T. Aspelmeier. *stepR: Multiscale change-point inference*, 2018. URL `https://CRAN.R-project.org/package=stepR`. R package version 2.0-2.

G. Pezzatti, T. Zumbrunnen, M. Bürgi, P. Ambrosetti, and M. Conedera. Fire regime shifts as a consequence of fire policy and socio-economic development: An analysis based on the change point approach. *Forest Policy and Economics*, 29:7–18, 2013.

M. Pierre-Jean, G. Rigaill, and P. Neuvial. *jointseg: Joint segmentation of multivariate (copy number) signals*, 2017. URL `https://CRAN.R-project.org/package=jointseg`. R package version 1.0.1.

A. Ranganathan. PLISS: labeling places using online changepoint detection. *Autonomous Robots*, 32:351–368, 2012.

J. Reeves, J. Chen, X. Wang, R. Lund, and Q. Lu. A review and comparison of changepoint detection techniques for climate data. *Journal of Applied Meteorology and Climatology*, 46:900–915, 2007.

G. Rigaill. A pruned dynamic programming algorithm to recover the best segmentations with 1 to k_{max} change-points. *Journal de la Societe Francaise de Statistique*, 156:180–205, 2015.

G. Rigaill and T.D. Hocking. *fpop: Segmentation using Optimal Partitioning and Function Pruning*, 2016. URL `https://R-Forge.R-project.org/projects/opfp/`. R package version 2016.10.25/r55.

A. Rinaldo. Properties and refinements of the fused lasso. *Annals of Statistics*, 37:2922–2952, 2009.

C. Rojas and B. Wahlberg. On change point detection using the fused lasso method. *Preprint*, 2014.

G. Ross. Parametric and nonparametric sequential change detection in R: The cpm package. *Journal of Statistical Software*, 66:1–20, 2015.

M. Salarijazi, A. Akhond-Ali, A. Adib, and A. Daneshkhah. Trend and change-point detection for the annual stream-flow series of the Karun River at the Ahvaz hydrometric station. *African Journal of Agricultural Research*, 7:4540–4552, 2012.

R. Tibshirani. Adaptive piecewise polynomial estimation via trend filtering. *Annals of Statistics*, 42:285–323, 2014.

R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight. Sparsity and smoothness via the fused lasso. *J. R. Statist. Soc. B*, 67:91–108, 2005.

C. Truong, L. Oudre, and N. Vayatis. Selective review of offline change point detection methods. *Signal Processing*, 167, 2020. Article no. 107299.

E. S. Venkatraman. Consistency results in multiple change-point problems. *Technical Report No. 24, Department of Statistics, Stanford University, available from* `https://statistics.stanford.edu/resources/technical-reports`, 1992.

E. S. Venkatraman and A. Olshen. A faster circular binary segmentation algorithm for the analysis of array CGH data. *Bioinformatics*, 23:657–663, 2007.

L. Vostrikova. Detecting 'disorder' in multidimensional random processes. *Soviet Math. Dokl.*, 24:55–59, 1981.

D. Wang, Y. Yu, and A. Rinaldo. Univariate mean change point detection: Penalization, CUSUM and optimality. *Preprint*, 2018.

T. Wang and R. Samworth. High dimensional change point estimation via sparse projection. *J. Roy. Statist. Soc. Ser. B*, 80:57–83, 2018.

Y. Wang. Jump and sharp cusp detection by wavelets. *Biometrika*, 82:385–397, 1995.

Y. Wu. Simultaneous change point analysis and variable selection in a regression problem. *Journal of Multivariate Analysis*, 99:2154–2171, 2008.

Y.-C. Yao. Estimating the number of change-points via Schwarz' criterion. *Stat. Prob. Lett.*, 6:181–189, 1988.

Y.-C. Yao and S. T. Au. Least-squares estimation of a step function. *Sankhya Series A*, 51:370–381, 1989.

L. Younes, M. Albert, and M. Miller. Inferring changepoint times of medial temporal lobe morphometric change in preclinical Alzheimer's disease. *NeuroImage: Clinical*, 5:178–187, 2014.

A. Zeileis, F. Leisch, K. Hornik, and C. Kleiber. strucchange: An R package for testing for structural change in linear regression models. *Journal of Statistical Software*, 7:1–38, 2002.

N. Zhang and D. Siegmund. A modified Bayes information criterion with applications to the analysis of comparative genomic hybridization data. *Biometrics*, 63:22–32, 2007.