Detecting Abrupt Changes in the Presence of Local Fluctuations and Autocorrelated Noise

Gaetano Romano, Guillem Rigaill, Vincent Runge, Paul Fearnhead

JOURNAL OF THE AMERICAN STATISTICAL ASSOCIATION
2021, VOL. 00, NO. 0, 1–16: Theory and Methods
https://doi.org/10.1080/01621459.2021.1909598

Taylor & Francis
Taylor & Francis Group

Detecting Abrupt Changes in the Presence of Local Fluctuations and Autocorrelated Noise

Gaetano Romano[a], Guillem Rigaill[b,c], Vincent Runge[c], and Paul Fearnhead[a]

[a]Department of Mathematics and Statistics, Lancaster University, Lancaster, UK; [b]Université Paris-Saclay, CNRS, INRAE, Univ Evry, Institute of Plant Sciences Paris-Saclay (IPS2), Orsay, France; [c]Université Paris-Saclay, CNRS, Univ Evry, Laboratoire de Mathématiques et Modélisation d'Evry, Evry-Courcouronnes, France

ABSTRACT
While there are a plethora of algorithms for detecting changes in mean in univariate time-series, almost all struggle in real applications where there is autocorrelated noise or where the mean fluctuates locally between the abrupt changes that one wishes to detect. In these cases, default implementations, which are often based on assumptions of a constant mean between changes and independent noise, can lead to substantial over-estimation of the number of changes. We propose a principled approach to detect such abrupt changes that models local fluctuations as a random walk process and autocorrelated noise via an AR(1) process. We then estimate the number and location of changepoints by minimizing a penalized cost based on this model. We develop a novel and efficient dynamic programming algorithm, DeCAFS, that can solve this minimization problem; despite the additional challenge of dependence across segments, due to the autocorrelated noise, which makes existing algorithms inapplicable. Theory and empirical results show that our approach has greater power at detecting abrupt changes than existing approaches. We apply our method to measuring gene expression levels in bacteria. Supplementary materials for this article are available online.

ARTICLE HISTORY
Received May 2020
Accepted March 2021

KEYWORDS
Breakpoints; Changepoints; Dynamic programming; FPOP; Optimal partitioning; Structural breaks

1. Introduction

Detecting changes in data streams is a ubiquitous challenge across many modern applications of statistics. It is important in such diverse areas as bioinformatics (Olshen et al. 2004; Futschik et al. 2014), ion channels (Hotz et al. 2013), climate records (Reeves et al. 2007), oceanographic data (Killick et al. 2010), and finance (Kim, Morley, and Nelson 2005). The most common and important change detection problem is that of detecting changes in mean, and there have been a large number of different approaches to this problem that have been proposed (e.g., Olshen et al. 2004; Killick, Fearnhead, and Eckley 2012; Fryzlewicz 2014; Frick, Munk, and Sieling 2014; Maidstone et al. 2017; Eichinger and Kirch 2018; Fearnhead and Rigaill 2019; Fryzlewicz 2018*b*, amongst many others). Almost all of these methods are based on modeling the data as having a constant mean between changes and the noise in the data being independent. Furthermore, all change-point methods require specifying some threshold or penalty that affects the amount of evidence that there needs to be for a change before an additional changepoint is detected. In general, the methods have default choices of these thresholds or penalties that have good theoretical properties under strong modeling assumptions.

Although these methods perform well when analyzing simulated data where the assumptions of the method hold, they can be less reliable in real applications, particularly if the default threshold or penalties are used. Reasons for this include the noise in the data being autocorrelated, or the underlying mean fluctuating slightly between the abrupt changes that one wishes to detect. To see this, consider change detection for the well-log data (taken from Fearnhead and Liu 2011; Ruanaidh and Fitzgerald 2012) shown in Figure 1. These data come from lowering a probe into a bore-hole, and taking measurements of the rock structure as the probe is lowered. The data we plot has had outliers removed. As the probe moves from one rock strata to another, we expect to see an abrupt change in the signal from the measurements, and it is these changes that an analyst would wish to detect. Previous analyses of this data have shown that, marginally, the noise in the data is very well approximated by a Gaussian distribution; but by eye we can see local fluctuations in the data that suggest either autocorrelation in the measurement error, or structure in the mean between the abrupt changes.

The top plot shows an analysis of the well-log data that uses wild binary segmentation (Fryzlewicz 2014) with the standard cusum test for a change in mean, and then estimates the number of changepoints based on a strengthened Schwarz information criteria. Both the cusum test and the strengthened Schwarz information criteria are based on modeling assumptions of a constant mean between changepoints and independent, identically distributed (IID) Gaussian noise, and are known to consistently estimate the number and location of the changepoints if these assumptions are correct. However, in this case, we can see that it massively overfits the number of changepoints. Similar results are obtained for standard implementation of other

CONTACT Paul Fearnhead p.fearnhead@lancaster.ac.uk Department of Mathematics and Statistics, Lancaster University, Lancaster, UK.
Supplementary materials for this article are available online. Please go to *www.tandfonline.com/r/JASA*.

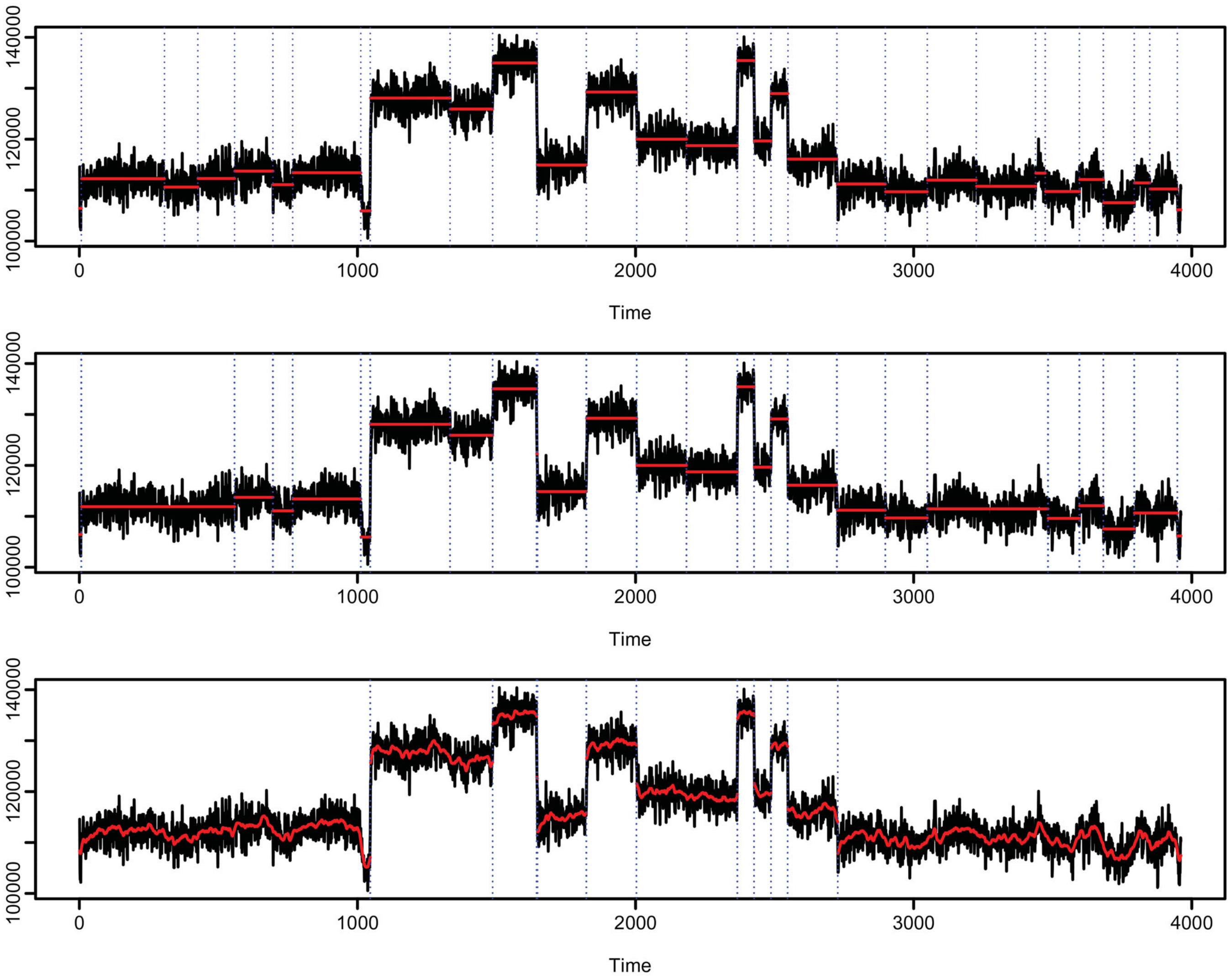

Figure 1. Segmentations of well-log data: wild binary segmentation using the strengthened Schwarz information criteria (top); segmentation under square error loss with penalty inflated to account for autocorrelation in measurement error (middle); optimal segmentation from DeCAFS with default penalty (bottom). Each plot shows the data (black line) the estimated mean (red line) and changepoint location (vertical blue dashed lines).

algorithms for detecting changes in mean, see Figure 13 in the supplementary material.

Lavielle and Moulines (2000) and Bardwell et al. (2019) suggested that if we estimate changepoints by minimizing the squared error loss of our fit with a penalty for each change, then we can correct for potential autocorrelation in the noise by inflating the penalty used for adding a changepoint. The middle plot of Figure 1 shows results for such an approach (Bardwell et al. 2019); this gives an improved result but it still noticeably overfits.

By comparison, the method we propose models both autocorrelation in the noise and local fluctuations in the mean between changepoints—and analysis of the data using default settings produces a much more reasonable segmentation of the data (see bottom plot of Figure 1). This method is model-based, and assumes that the local fluctuations in the mean are realizations of a random walk and that the noise process is an AR(1) process. We then segment the data by minimizing a penalized cost that is based on the log-likelihood of our model together with a BIC penalty for adding a changepoint.

The key algorithmic challenge with our approach is minimizing the penalized cost. In particular, many existing dynamic programming approaches (e.g., Jackson et al. 2005; Killick, Fearnhead, and Eckley 2012) do not work for our problem due to the dependence across segments caused by the autocorrelated noise. We introduce a novel extension of the functional pruned optimal partitioning algorithm of Maidstone et al. (2017), and we call the resulting algorithm DeCAFS, for Detecting Changes in Autocorrelated and Fluctuating Signals. It is both computationally efficient (analysis of the approx 4000 data points in the well-log data taking a fraction of a second on a standard laptop) and guaranteed to find the best segmentation under our criteria.

While we are unaware of any previous method that tries to model both autocorrelation and local fluctuations, Chakar et al. (2017) introduced AR1Seg which aims to detect changes in mean in the presence of autocorrelation. Their approach is similar to ours if we remove the random walk component, as they aim to minimize a penalized cost where the cost is the negative of the log-likelihood under a model with an AR(1) noise process. However they were unable to minimize this penalized cost, and instead minimized an approximation that removes the dependence across segments. One consequence of using this approximation is that it often estimates two consecutive changes at each changepoint, and AR1Seg uses a further post-processing step to try and correct this. Moreover, our simulation results show that using the approximation leads to a loss of power, particularly when the autocorrelation in the noise is high.

Particularly if interest lies in estimating how the underlying mean of the data varies over time, natural alternatives to DeCAFS are trend filtering methods (Kim et al. 2009; Tibshirani et al. 2014) which estimate the mean function under an L_1 penalty on a suitably chosen discrete derivative of the mean. Depending on the order of the derivative, these methods can fit a piecewise constant (in this case trend filtering is equivalent to the fused lasso of Tibshirani et al. 2005), linear, or quadratic etc. function to the mean. One advantage of trend filtering is its flexibility: depending on the application one can fit mean functions with different degrees of smoothness. However, it does not allow one to jointly estimate a smoothly changing mean and abrupt changes—the L_1 penalty must be chosen to capture one or other of these effects. This is particularly an issue if the main interest is in detecting abrupt changes rather than estimating the mean. These issues are investigated empirically in Appendix F.3.

Distinguishing between local fluctuations and abrupt changes is possible by methods that model the mean as a sum of functions (Jalali, Ravikumar, and Sanghavi 2013)—for example one smoothly varying and one piecewise constant. And trend-filtering, or other regularized estimators can then be used to estimate each component. The LAVA approach of Chernozhukov, Hansen, and Liao (2017) is one such approach, fitting the piecewise constant function and the smoothly varying function using, respectively, an L_1 and L_2 penalty on the function's first discrete derivative. The main difference between LAVA and DeCAFS is thus that LAVA has an L_1 penalty for abrupt changes, and DeCAFS has an L_0 penalty. If interest is primarily in detecting changes, previous work has shown the use of an L_0 penalty to be preferable to an L_1 penalty – as the latter can often over-fit the number of changes (see, e.g., the empirical evidence in Fearnhead, Maidstone, and Letchford 2018; Jewell et al. 2020), and a post-processing step is often needed to correct for this (Lin et al. 2017; Safikhani and Shojaie 2020).

One important feature of modeling the random fluctuations in the mean via a random walk, is that the information that a data point y_s has about a change at time t decays to 0 as $|t-s|$ gets large. This is most easily seen if we consider applying DeCAFS to detect a single abrupt change. We show in Section 5 that whether DeCAFS detects a change at a time t depends on some contrast of the data before and after t. This contrast compares a weighted mean of the data before t to a weighted mean of the data after t, with the weights decaying (essentially) geometrically with the length of time before/after t. This is appropriate for the random walk model which enables, for example, the mean of y_{t+h} to be very different to the mean at y_{t+1} if $h > 0$ is large, and thus y_{t+h} contains little to no information about whether there has been an abrupt change in the mean of y_{t+1} compared to the mean of y_t. By contrast, standard CUSUM methods for detecting a change in mean quantify the evidence for a change at t by a contrast of the *unweighted mean* of the data before and after t – thus each data point, y_{t+h}, has the same amount of information about the change regardless of the value of h. In situations where there are local fluctuations in the mean but through some stationary process, such as a mean-reverting random walk, data far from a change would still have a non-negligible amount of information about it. In such situations, particularly if the segments are long, DeCAFS could have lower power than CUSUM or other methods that ignore any local fluctuations in the data. We investigate this empirically in Appendix F.4.

The outline of the article is as follows. In the next section, we introduce our model-based approach and the associated penalized cost. In Section 3 we present DeCAFS, a novel dynamic programming algorithm that can exactly minimize the penalized cost. To implement our method we need estimates of the model parameters, and we present a simple way of preprocessing the data to obtain these in Section 4. We then look at the theoretical properties of the method. These justify the use of the BIC penalty, show that our method has more power at detecting changes when our model assumptions are correct than standard approaches, and also that we have some robustness to model error—in that we can still consistently estimate the number and location of the changepoints in such cases by adapting the penalty for adding a changepoint. Sections 6 and 7 evaluate the new method on simulated and real data; and the article ends with a discussion.

Code implementing the new algorithm is available in the R package `DeCAFS` on CRAN. The package and full code from our simulation study is also available at *github.com/gtromano/DeCAFS*.

2. Modeling and Detecting Abrupt Changes

2.1. Model

Let $y_{1:n} = (y_1, \dots, y_n) \in \mathbb{R}^n$ be a sequence of n observations, and assume we wish to detect abrupt changes in the mean of this data in the presence of local fluctuations and autocorrelated noise. We take a model-based approach where the signal vector is a realization of a random walk process with abrupt changes, and we super-impose an AR(1) noise process.

So for $t = 1, \dots, n$,

$$y_t = \mu_t + \epsilon_t, \tag{1}$$

where for $t = 2, \dots, n$

$$\mu_t = \mu_{t-1} + \eta_t + \delta_t, \quad \text{with } \eta_t \underset{\text{iid}}{\sim} \mathcal{N}(0, \sigma_\eta^2),\ \delta_t \in \mathbb{R}, \tag{2}$$

and $\delta_t = 0$ except at time points immediately after a set of m changepoints, $0 < \tau_1 < \dots < \tau_m < n$. That is $\delta_t = 0$ unless $t = \tau_j + 1$ for some j. This model is unidentifiable at changepoints. If τ is a changepoint, then whilst the data is informative about $\mu_{\tau+1}$ and μ_τ, we have no further information about the specific value of $\delta_{\tau+1}$ relative to $\eta_{\tau+1}$. We thus take the convention that $\delta_{\tau+1} = \mu_{\tau+1} - \mu_\tau$ and $\eta_{\tau+1} = 0$, which is the most likely value of $\eta_{\tau+1}$ under our model, and consistent with maximizing the likelihood criteria we introduce below. The noise process, ϵ_t is a stationary AR(1) process with, for $t = 2, \dots, n$,

$$\epsilon_t = \phi\epsilon_{t-1} + \nu_t \quad \text{with } \nu_t \underset{\text{iid}}{\sim} \mathcal{N}(0, \sigma_\nu^2), \tag{3}$$

for some autocorrelation parameter, ϕ with $|\phi| < 1$ and $\epsilon_1 \sim \mathcal{N}(0, \sigma_\nu^2/(1-\phi^2))$. Special cases of our model occur when $\phi = 0$ or when $\sigma_\eta^2 = 0$. When $\phi = 0$ our noise process ϵ_t is then IID, and the model is equivalent to a random walk plus noise with abrupt changes. When $\sigma_\eta^2 = 0$ we are detecting changes in mean with an AR(1) noise process, resulting in a formulation equivalent to the one of Chakar et al. (2017).

2.2. Penalized Maximum Likelihood Approach

In the following, we will assume that ϕ, σ_η^2 and σ_ν^2 are known; we consider robust approaches to estimate these parameters from the data in Section 4. We can then write down a type of likelihood for our model, defined as the joint density of the observations, $y_{1:n}$, and the local fluctuations in the mean, $\eta_{2:n}$. We will express this as a function of $\mu_{1:n}$ and $\delta_{2:n}$. Writing $f(\cdot|\cdot)$ for a generic conditional density, we have that this is

$$\begin{aligned}\mathcal{L}(y_{1:n}; \mu_{1:n}, \delta_{2:n}) &= \left(\prod_{t=2}^{n} f(\mu_t|\mu_{t-1}, \delta_t)\right) f(y_1|\mu_1) \\ &\times \left(\prod_{t=2}^{n} f(y_t|y_{t-1}, \mu_{t-1}, \mu_t)\right) \\ \propto &\left(\prod_{t=2}^{n} \exp\left\{-\frac{(\mu_t - \mu_{t-1} - \delta_t)^2}{2\sigma_\eta^2}\right\}\right) \exp\left\{-\frac{(y_1 - \mu_1)^2}{2\sigma_\nu^2/(1-\phi^2)}\right\} \\ &\times \left(\prod_{t=2}^{n} \exp\left\{-\frac{((y_t - \mu_t) - \phi(y_{t-1} - \mu_{t-1}))^2}{2\sigma_\nu^2}\right\}\right).\end{aligned}$$

We have used the specific Gaussian densities of our model, and dropped multiplicative constants, to get the second expression.

If we knew the number of changepoints, then we could estimate their position by maximizing this likelihood subject to the constraints on the number of nonzero entries of $\delta_{2:n}$. However, as we need to also estimate the number of changepoints we proceed by maximizing a penalized version of the log of the likelihood where we introduce a penalty for each changepoint—this is a common approach to changepoint detection, see, for example, Maidstone et al. (2017). It is customary to restate this as minimizing a penalized cost, rather than maximizing a penalized likelihood, where the cost is minus twice the log-likelihood. That is, we estimate the number and location of the changepoints by solving the following minimization problem:

$$\begin{aligned}\mathcal{F}_n = \min_{\substack{\mu_{1:n}\\ \delta_{2:n}}} \Bigg\{&(1-\phi^2)\gamma(y_1 - \mu_1)^2 + \\ &\sum_{t=2}^{n} \Big[\lambda(\mu_t - \mu_{t-1} - \delta_t)^2 + \gamma\Big((y_t - \mu_t) - \phi(y_{t-1} - \mu_{t-1})\Big)^2 \\ &+ \beta\, \mathbb{1}_{\delta_t \neq 0}\Big]\Bigg\}, \qquad (4)\end{aligned}$$

where $\beta > 0$ is the penalty for adding a changepoint, $\lambda = 1/\sigma_\eta^2$, $\gamma = 1/\sigma_\nu^2$, and $\mathbb{1} \in \{0, 1\}$ is an indicator function. For the special case of a constant mean between changepoints, corresponding to $\sigma_\eta^2 = 0$, we require $\mu_t = \mu_{t-1} + \delta_t \ \forall\ t = 2, \dots, n$ and simply drop the first term in the sum.

2.3. Dynamic Programming Recursion

We will use dynamic programming to minimize the penalized cost Equation (4). The challenge here is to deal with the dependence across changepoints due to the AR(1) noise process which means that some standard dynamic approaches for changepoint detection, such as optimal partitioning (Jackson et al. 2005) and PELT (Killick, Fearnhead, and Eckley 2012), cannot be used. To overcome this, as in Rigaill (2015) or Maidstone et al. (2017), we define the function $\mu \mapsto Q_t(\mu)$ to be the minimum penalized cost for data $y_{1:t}$ conditional on $\mu_t = \mu$,

$$\begin{aligned}Q_t(\mu) = \min_{\substack{\mu_{1:t}\\ \delta_{2:t}, \mu_t = \mu}} \Bigg\{&(1-\phi^2)\gamma(y_1 - \mu_1)^2 + \\ &\sum_{i=2}^{t} \Big[\lambda(\mu_i - \mu_{i-1} - \delta_i)^2 \\ &+ \gamma\Big((y_i - \mu_i) - \phi(y_{i-1} - \mu_{i-1})\Big)^2 + \beta\, \mathbb{1}_{\delta_i \neq 0}\Big]\Bigg\}.\end{aligned}$$

So $\mathcal{F}_n = \min_{\mu \in \mathbb{R}} Q_n(\mu)$; and the following proposition gives a recursion for $Q_t(\mu)$.

Proposition 1. The set of functions $\{\mu \mapsto Q_t(\mu)\,,\ t = 1, \dots, n\}$ satisfies

$Q_1(\mu) = (1-\phi^2)\gamma(y_1 - \mu)^2$ and, for $t = 2, \dots, n$,

$$\begin{aligned}Q_t(\mu) = \min_{u \in \mathbb{R}} \Big\{&Q_{t-1}(u) + \min\{\lambda(\mu - u)^2, \beta\} \\ &+ \gamma\Big((y_t - \mu) - \phi(y_{t-1} - u)\Big)^2\Big\}. \qquad (5)\end{aligned}$$

The intuition behind the recursion is that we first condition on $\mu_{t-1} = u$, with the term in braces being the minimum penalized cost for $y_{1:t}$ given u and $\mu_t = \mu$, and then minimize over u. The cost in braces is the sum of three terms: (i) the minimum penalized cost for $y_{1:t-1}$ given u; (ii) the cost for the change in mean from u to μ; and (iii) the cost of fitting data point y_t with μ_t. The cost for the change in mean, (ii), is just the minimum of the constant cost for adding a change and the quadratic cost for a change due to the random walk. The recursion applies to the special case of a constant mean between changepoints, where $\lambda = \infty$, if we replace $\min\{\lambda(\mu - u)^2, \beta\}$ with its limit as $\lambda \to \infty$, which is $\beta \mathbb{1}_{\mu \neq u}$.

Although recursions of this form have been considered in earlier changepoint algorithms (e.g., Maidstone et al. 2017; Hocking et al. 2020), the dependence between the current mean u and the previous mean μ that appears in terms (ii) and (iii) makes our recursion more challenging to solve. We next show one efficient way of solving by combining existing functional pruning dynamic programming ideas with properties of infimal convolutions.

3. Computationally Efficient Algorithm

3.1. The DeCAFS Algorithm

Algorithm 1 gives pseudo code for solving the dynamic programming recursion introduced in Proposition 1. The key to implementing this algorithm is performing the calculations in line 5, and how this can be done efficiently will be described below. Throughout we give the algorithm for the case where there is a random walk component, that is, $\lambda < \infty$, though it is trivial to adapt the algorithm to the $\lambda = \infty$ case.

As well as solving the recursion for $Q_t(\mu)$, Algorithm 1 shows how we can also obtain the estimate of the mean, through a standard back-tracking step. The idea is that our estimate of μ_n, $\hat{\mu}_n$, is just the value of μ that maximizes $Q_n(\mu)$. We then loop backwards through the data, and our estimate of μ_t is the value that minimizes the penalized cost for the data $y_{1:t}$ conditional on $\mu_{t+1} = \hat{\mu}_{t+1}$, which can be calculated as $B_t(\mu)$ in line 11.

Finally, as we obtain the estimates of the mean, we can also directly obtain the estimated changepoint locations. It is straightforward to see, by examining the form of the penalized cost, that the optimal solution for $\delta_{2:n}$ has $\delta_{t+1} \neq 0$ (and hence, t is a changepoint) if and only if $\lambda(\hat{\mu}_{t+1} - \hat{\mu}_t)^2 > \beta$.

```
Algorithm 1: DeCAFS
    Data: $\mathbf{y} = y_{1:n}$ a time series of length $n$
    Input: $\beta > 0, \lambda > 0, \gamma > 0$ and $0 \leq \phi < 1$.
1  begin Initialization
2      $Q_1(\mu) \leftarrow (1 - \phi^2)\gamma(y_1 - \mu)^2$
3  end
4  for $t = 2$ to $n$ do
5      $Q_t(\mu) \leftarrow \min_u \Big\{ Q_{t-1}(u) + \min\{\lambda(\mu - u)^2, \beta\} + \gamma\Big((y_t - \mu) - \phi(y_{t-1} - u)\Big)^2 \Big\}$
6  end
7  begin Backtracking
8      $\hat{\mu}_n \leftarrow \mathrm{argmin} Q_n(\mu)$
9      $\hat{\tau} \leftarrow n$
10     for $t = n - 1$ to $1$ do
11         $B_t(\mu) \leftarrow Q_t(\mu) + \min\{\lambda(\mu - \hat{\mu}_{t+1})^2, \beta\} + \gamma\Big((y_{t+1} - \hat{\mu}_{t+1}) - \phi(y_t - \mu)\Big)^2$
12         $\hat{\mu}_t \leftarrow \mathrm{argmin} B_t(\mu)$
13         if $(\hat{\mu}_t - \hat{\mu}_{t+1})^2 > \beta/\lambda$ then
14             $\hat{\tau} \leftarrow (t, \hat{\tau})$
15         end
16     end
17 end
18 Return $\hat{\mu}_{1:n}$, $\hat{\tau}$
```

3.2. The Infimal Convolution

The main challenge with Algorithm 1 is implementing line 5. First, this needs a compact way of characterizing $Q_t(\mu)$. This is possible as $Q_1(\mu)$ is a quadratic function; and the recursion maps piecewise quadratic functions to piecewise quadratic functions. Hence $Q_t(\mu)$ will be piecewise quadratic and can be defined by storing a partition of the real-line together with the coefficients of the quadratics for each interval in this partition.

Next we can simplify line 5 of Algorithm 1. As written line 5 involves minimizing a two-dimensional function, in $(u, \mu) \in \mathbb{R}^2$, over the variable u. We can recast this operation into a one-dimensional problem by introducing the concept of an infimal convolution (see Chapter 12 of Bauschke and Combettes 2011).

Definition 1. Let f be a real-valued function defined on $\mathbb{R}$ and ω a nonnegative scalar. We define $\mathsf{INF}_{f,\infty}(\theta) = f(\theta)$ and for $\omega > 0$,

$$\mathsf{INF}_{f,\omega}(\theta) = \min_{u \in \mathbb{R}} \left(f(u) + \omega(u - \theta)^2 \right), \tag{6}$$

as the infimal convolution of f with a quadratic term.

The following proposition presents a reformulation of the update-rule into a minimization involving infimal convolutions, for the case $\phi \geq 0$. The proof is in Appendix B, together with details of equivalent results when $\phi < 0$.

Proposition 2. Assume $\phi \geq 0$. The functions $\{Q_t(\mu), t = 2, \ldots, n\}$ can be written as

$$Q_t(\mu) = \min\left\{ Q_t^{=}(\mu),\ Q_t^{\neq}(\mu) \right\},$$

where

$$Q_t^{=}(\mu) = \mathsf{INF}_{\mathbb{Q}_{t-1}, \gamma\phi + \lambda}(\mu) + \frac{\gamma}{1 - \phi}\Big(y_t - \phi y_{t-1} - (1 - \phi)\mu\Big)^2,$$

$$Q_t^{\neq}(\mu) = \mathsf{INF}_{\mathbb{Q}_{t-1}, \gamma\phi}(\mu) + \frac{\gamma}{1 - \phi}\Big(y_t - \phi y_{t-1} - (1 - \phi)\mu\Big)^2 + \beta,$$

and

$$\mathbb{Q}_{t-1}(u) = Q_{t-1}(u) - \gamma\phi(1 - \phi)\left(u - \frac{y_t - \phi y_{t-1}}{1 - \phi}\right)^2.$$

3.3. Fast Infimal Convolution Computation

As noted above we can represent Q_t by $\mathsf{Q}_t = (q_t^1, \ldots, q_t^s)$ where each q_t^i is a quadratic defined on some interval $[d_i, d_{i+1}[$ with $d_1 = -\infty$ and $d_{s+1} = +\infty$. It is this representation of Q_t that we update at each time step. Some operations involved in solving the recursion, such as adding a quadratic to a piecewise quadratic, or calculating the pointwise minimum of two piecewise quadratics are easy to perform with a computational cost that is linear in the number of intervals (see, e.g., Rigaill 2015). The following theorem shows that a fast update for the infimal convolution of a piecewise quadratic is also possible, and is important for developing a fast algorithm for solving the dynamic programming recursions.

Theorem 1. Let $\mathsf{Q}_t = (q_t^1, \ldots, q_t^s)$ be the representation of the functional cost Q_t. For all $\omega \geq 0$, the representation returned by the infimal convolution $\mathrm{INF}_{\mathsf{Q}_t,\omega}$ has the following order-preserving form:

$$\mathrm{INF}_{\mathsf{Q}_t,\omega} = (\mathrm{INF}q_t^{u_1}, \mathrm{INF}q_t^{u_2}, \ldots, \mathrm{INF}q_t^{u_{s^*-1}}, \mathrm{INF}q_t^{u_{s^*}}),$$

with $1 = u_1 < u_2 < \ldots < u_{s^*-1} < u_{s^*} = s$ and $s^* \leq s$.

The key part of this result is that the order of the quadratics is not changed when we apply the infimal convolution, and thus we can calculate $\mathrm{INF}_{\mathsf{Q}_t,\omega}$ using a linear scan over the real-line. The proof of this theorem is given in Appendix C, and an example algorithm for calculating $\mathrm{INF}_{\mathsf{Q}_t,\omega}$ with complexity that is linear in s is shown in Appendix D.

4. Robust Parameter Estimation

Our optimization problem Equation (4) depends on three unknown parameters: σ_η^2, σ_ν^2 and ϕ. We estimate these parameters by fitting to robust estimates of the variance of the k-lag differenced data, $z_t^k = y_{t+k} - y_t$, for $k \geq 1$.

Proposition 3. With the model defined by Equations (1)–(3),

$$z_t^k \sim \mathcal{N}\Big(\sum_{i=t+1}^{t+k} \delta_i, k\sigma_\eta^2 + 2\frac{1-\phi^k}{1-\phi^2}\sigma_\nu^2\Big), \quad t = 1, \dots, n-k.$$

Providing k is small relative to the average length of a segment, the mean of z_t^k will be zero for most t: if there are m changes then at most km of the z_t^k's will overlap a change and have a nonzero mean. This suggests that we can estimate the variance of z_t^k using a robust estimator, such as the median absolute difference from the median, or MAD, estimator.

Fix K, and let v_k be the MAD estimator of the variance of z_t^k for $k = 1, \dots, K$. We estimate the parameters by minimizing the least-square fit to these estimates,

$$\mathcal{S}_\phi(\sigma_\eta^2, \sigma_\nu^2) = \sum_{k=1}^{K} \left(k\sigma_\eta^2 + 2\frac{1-\phi^k}{1-\phi^2}\sigma_\nu^2 - v_k\right)^2.$$

In practice, we can minimize these criteria by using a grid of values for ϕ and then for each ϕ value analytically minimize with respect to $\sigma_\eta^2 \geq 0$ and $\sigma_\nu^2 \geq 0$. Obviously, if we are fitting a model without the random walk component, then we can set $\sigma_\eta^2 = 0$, or if we wish to have uncorrelated noise, then we set $\phi = 0$.

An empirical evaluation of this method for estimating the parameters is shown in Section F of the supplementary material. These include an investigation of the accuracy in situations where we have changepoints. In our simulation study, we use $K = 10$, though similar results were obtained as we varied K.

5. Theoretical Properties

We can reformulate our model as a linear-regression. To do this it is helpful to introduce new variables, $\tilde{\eta}_{1:n}$, that give the cumulative effect of the random-walk fluctuations. To simplify exposition, it is further helpful to define this process so it has an invertible covariance matrix. So we will let $\tilde{\eta}_1 \sim \mathcal{N}(0, \sigma_\eta^2)$ and $\tilde{\eta}_t = \tilde{\eta}_{t-1} + \eta_t$ for $t = 2, \dots, n$. For a set of m changepoints $\tau_{1:m}$, and defining $\tau_0 = 0$, we can introduce a $n \times (m+1)$ matrix $X_{\tau_{0:m}}$ where the ith column is a column of τ_{i-1} zeros followed by $n - \tau_{i-1}$ ones. Our model is then

$$y_{1:n} = X_{\tau_{0:m}}\Delta + \zeta_{1:n}, \tag{7}$$

where $\zeta_{1:n}$ is a vector of Gaussian random variables with

$$\text{var}(\zeta_{1:n}) = \text{var}(\epsilon_{1:n}) + \text{var}(\tilde{\eta}_{1:n}) := \Sigma_{\text{AR}} + \Sigma_{\text{RW}}$$

the sum of the variance matrices for the AR component of the model, $\epsilon_{1:n}$, and the random walk component of the model, $\tilde{\eta}_{1:n}$; and Δ is a $(m+1) \times 1$ vector whose first entry is $\mu_1 - \tilde{\eta}_1$ and whose ith entry is $\delta_{\tau_{i-1}+1}$ the change at the $(i-1)$th changepoint. This formulation allows us to consider the impact of model error on DeCAFS. Later, when we consider its asymptotic properties, we will allow for the data-generating process to be Equation (7) but with $\text{var}(\zeta_{1:n})$ different from that assumed by DeCAFS.

As shown in Section E of the supplementary material, the unpenalized version of the cost that we minimize, conditional on a specific set of changepoints, can be written as

$$\mathcal{C}(\tau_{1:m}) = \min_{\Delta, \tilde{\eta}_{1:n}, \tilde{\eta}_1 = 0} \Big[(y_{1:n} - X_{\tau_{0:m}}\Delta - \tilde{\eta}_{1:n})^T \times \Sigma_{\text{AR}}^{-1}(y_{1:n} - X_{\tau_{0:m}}\Delta - \tilde{\eta}_{1:n}) + \tilde{\eta}_{1:n}^T \Sigma_{\text{RW}}^{-1} \tilde{\eta}_{1:n}\Big],$$

where $\tilde{\eta}_{1:n}$ is assumed to be a column vector. Thus, the penalized cost Equation (4) is $\mathcal{F}_n = \min_{m, \tau_{1:m}} [\mathcal{C}(\tau_{1:m}) + m\beta]$. In the remainder of this section, we will call $\mathcal{C}(\tau_{1:m})$ the cost, and $\mathcal{C}(\tau_{1:m}) + m\beta$ the penalized cost.

While our cost is obtained by minimizing over $\eta_{2:n}$, the following result shows that it is equal to the weighted residual sum of squares from fitting the linear model defined in Equation (7).

Proposition 4. The cost for fitting a model with changepoints, $\tau_{1:m}$ is

$$\mathcal{C}(\tau_{1:m}) = \min_{\Delta}(y_{1:n} - X_{\tau_{0:m}}\Delta)^T (\Sigma_{\text{AR}} + \Sigma_{\text{RW}})^{-1} (y_{1:n} - X_{\tau_{0:m}}\Delta) \tag{8}$$

Let $\mathcal{C}_0$ denote the cost if we fit a model with no changepoints. The following corollary, which follows from standard arguments, gives the behavior of the cost under a null model of no changepoints. This includes a bound on the impact of mis-specifying the covariance matrix, for example due to mis-estimating the parameters of the AR(1) or random walk components of the model, or if our model for the residuals is incorrect.

Corollary 1. Assume that data is generated from the model defined in Equation (7), with $m = 0$ but with $\zeta_{1:n}$ a mean-zero Gaussian vector with $\text{var}(\zeta_{1:n}) = \Sigma$. Let α_n^+ be the largest eigenvalue of $(\Sigma_{\text{AR}} + \Sigma_{\text{RW}})^{-1}\Sigma$. If $\Sigma = \Sigma_{\text{AR}} + \Sigma_{\text{RW}}$ then $\mathcal{C}_0 - \mathcal{C}(\tau_{1:d}) \sim \chi_d^2$. Otherwise, for any x

$$\Pr(\mathcal{C}_0 - \mathcal{C}(\tau_{1:d}) > x) \leq \Pr(\chi_d^2 > x/\alpha_n^+).$$

Furthermore, if we estimate the number of changepoints using the penalized cost Equation (4) with penalty $\beta = C\alpha_n^+ \log n$ for any $C > 2$, then the estimated number of changepoints, $\hat{m}$, satisfies $\Pr(\hat{m} = 0) \to 1$ as $n \to \infty$.

To gain insight into the behavior of the procedure in the presence of changepoints, and how it differs from standard standard change-in-mean procedures, it is helpful to consider the reduction in cost if we add a single changepoint.

Proposition 5. Given a fixed changepoint location τ_1:

(i) The reduction in cost for adding a single changepoint at τ_1 can be written as $\mathcal{C}_0 - \mathcal{C}(\tau_1) = (v^T y_{1:n})^2$, for some vector v defined as

$$v = \frac{1}{\sqrt{c_{\tau_1} - c_{0,\tau_1}^2/c_0}}\Big\{(\Sigma_{\text{AR}} + \Sigma_{\text{RW}})^{-1}u_{\tau_1} - \frac{c_{0,\tau_1}}{c_0}(\Sigma_{\text{AR}} + \Sigma_{\text{RW}})^{-1}u_0\Big\},$$

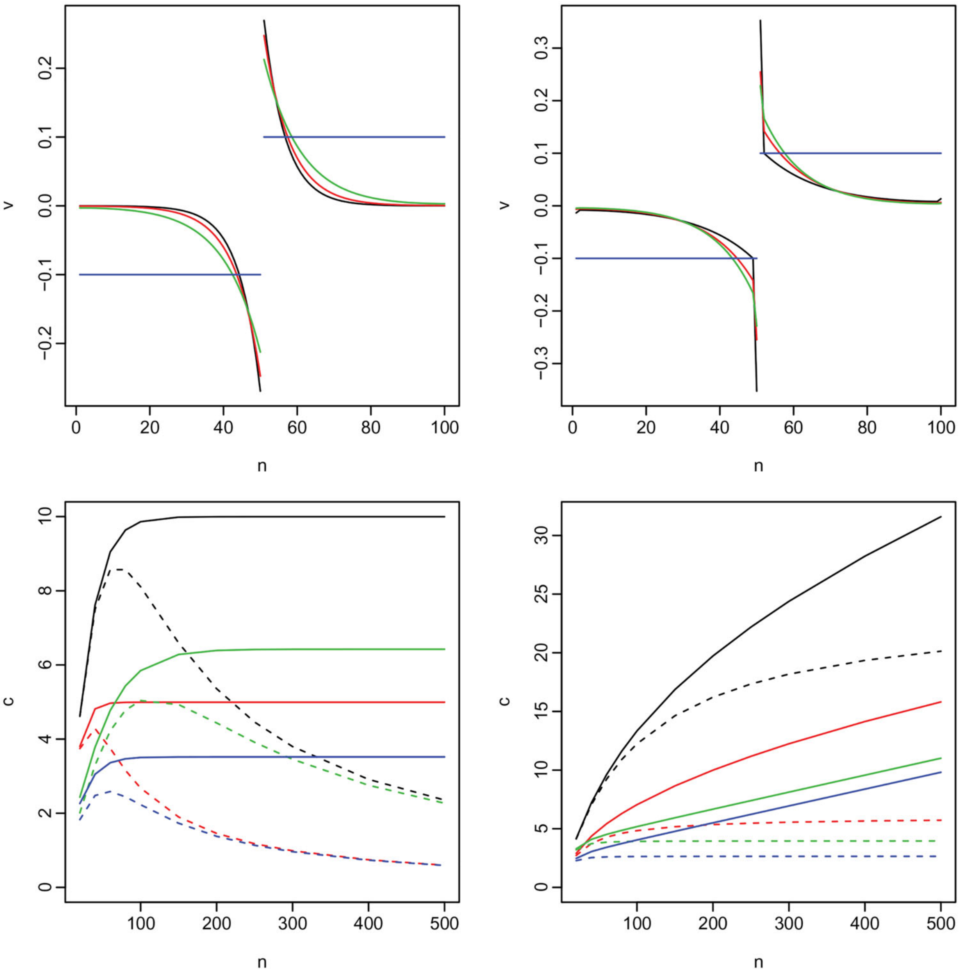

Figure 2. Top row: projections of data v for detecting a change in the middle of $n = 100$ data-points. Random walk model (top-left) for varying σ_η^2 of 0.03 (black), 0.02 (red) and 0.01 (green); AR(1) plus random walk model (top-right) for $\sigma_\eta^2 = 0.01$ and varying ϕ of 0.4 (black), 0.2 (red) and 0.1 (green). In both plots the blue line shows the standard cusum projection. Bottom row: noncentrality parameter for a χ_1^2 test of a change using the optimal projection (solid line) and the cusum projection (dashed line) for a change of size 1 in the middle of the data as we vary n. Out-fill asymptotics (bottom-left) where (σ_η^2, ϕ) is (0.0025,0) (black), (0.01,0) (red), (0.0025,0.5) (green) and (0.01,0.5) (blue); In-fill asymptotics (bottom-right) where for $n = 50$ (σ_η^2, ϕ) is (0.0025,0) (black), (0.01,0) (red), (0.0025,0.5) (green) and (0.01,0.5) (blue).

where u_0 is a column vector of n ones, u_{τ_1} is a column vector of τ_1 zeroes followed by $n - \tau_0$ ones, and

$$c_0 = u_0^T(\Sigma_{\mathrm{AR}} + \Sigma_{\mathrm{RW}})^{-1}u_0, \quad c_{0,\tau_1} = u_0^T(\Sigma_{\mathrm{AR}} + \Sigma_{\mathrm{RW}})^{-1}u_{\tau_1},$$
$$c_{\tau_1} = u_{\tau_1}^T(\Sigma_{\mathrm{AR}} + \Sigma_{\mathrm{RW}})^{-1}u_{\tau_1}.$$

(ii) The vector v in (i) satisfies $\sum_{i=1}^n v_i = 0$ and $v^T(\Sigma_{\mathrm{AR}} + \Sigma_{\mathrm{RW}})v = 1$.

(iii) For any vector w that satisfies $\sum_{i=1}^n w_i = 0$ and $w^T(\Sigma_{\mathrm{AR}} + \Sigma_{\mathrm{RW}})w = 1$,

$$\left(\sum_{i=\tau_1+1}^{n} w_i\right)^2 \leq \left(\sum_{i=\tau_1+1}^{n} v_i\right)^2.$$

The vector v in part (i) of this proposition defines a projection of the data that is used to determine whether to add a changepoint at τ_1. The properties in part (ii) mean that this projection is invariant to shifts of the data, and that the distribution of the reduction in cost if our model is correct and there are no changes will be χ_1^2. The statistic $v^T y_{1:n}$ can be viewed as analogous to the cusum statistic (Hinkley 1971) that is often used for a standard change-in-mean problem, and in fact if we set $\phi = 0$ and $\sigma_\eta = 0$ so as to remove the auto regressive and random-walk aspects of the model, $|v^T y_{1:n}|$ is just the standard cusum statistic. The power of our method to detect a change at τ_1 will be governed by the distribution of this projection applied to the data in the segments immediately before and after τ_1. For a single changepoint where the mean changes by δ this distribution is a noncentral chi-squared with 1 degree of freedom and noncentrality parameter $\delta^2(\sum_{i=\tau_1+1}^n v_i)^2$. Thus, part (iii) shows that v is the best linear projection, in terms of maximizing the non-centrality parameter, over all projections that are invariant to shifts in the data and that are scaled so that the null distribution is χ_1^2.

To gain insight into how the auto regressive and random-walk parts of the model affect the information in the data about a change we have plotted different projections v for different model scenarios in the top row of Figure 2. The top-left plot shows the projections if we have $\phi = 0$ for different ,values of the random walk variance. The projection, naturally, places more weight to data near the putative changepoint, and the weight decays essentially geometrically as we move away from the putative changepoint. In the top-right plot we show the impact of increasing the autocorrelation of the AR(1) process, with the absolute value of the weight given to data points immediately before and after the putative change increasing with ϕ.

A key feature of the random walk model is that for any fixed $\sigma_\eta^2 > 0$ the amount of information about a change will be bounded as we increase the segment lengths either side of the change. This is shown in the bottom-left plot of Figure 2 where we show the noncentrality parameter for detecting a change in the middle of the data as we vary n. For comparison we also show the non-centrality parameter of a test based on the cusum statistic (scaled so that it also has a χ_1^2 distribution under the null of no change). We can see that ignoring local fluctuations in the mean, if they exist and come from a random walk model, by using the cusum statistic leads to a reduction of power as segment lengths increase. For comparison in the bottom right we show an equivalent comparison where we consider an infill asymptotic regime, so that as n increases we let the random walk variance decay at a rate proportional to $1/n$ and we increase the lag-1 autocorrelation appropriately. In this case using the optimal projection gives a noncentrality parameter that increases with n, whereas the cusum statistic has power that can be shown to be bounded as we increase n.

We now turn to the property of our method at detecting multiple changes. Based on the above discussion, we will consider in-fill asymptotics as $n \to \infty$.

(C1) Let $y_1, \dots, y_n$ be generated as a finite sample from a Gaussian process on $[0, 1]$; that is $y_i = z(i/n)$ where, for $t \in [0, 1]$ $z(t) = \mu(t) + \zeta(t)$, $\mu(t)$ is a piecewise constant with m^0 changepoints at locations $r_1, \dots, r_{m^0}$, and $\zeta(t)$ is a mean zero Gaussian process. For a given n define the true changepoint locations as $\tau_i^0 = \lfloor nr_i^0 \rfloor$. The change in mean at each changepoint is fixed and non-zero.

(C2) Assume there exists strictly positive constants c_η, c_ν and c_ϕ, such that we implement DeCAFS with $\sigma_\eta^2 = c_\eta/n$ and either (i) $\phi = 0$ and $\sigma_\nu^2 = c_\nu$; or (ii) $\phi = \exp\{-c_\phi/n\}$ and $\sigma_\nu^2 = c_\nu(1 - \exp\{-2c_\phi/n\})$.

(C3) There exists an α such that for any large enough n if Σ_n^0 is the covariance of the noise in the data generating model (C1), and $\Sigma_{\text{AR}}^{(n)} + \Sigma_{\text{RW}}^{(n)}$ is the covariance assumed by DeCAFS in (C2) then the largest eigenvalue of $(\Sigma_{\text{AR}}^{(n)} + \Sigma_{\text{RW}}^{(n)})^{-1}\Sigma_n^0$ is less than α.

The two regimes covered by condition C2 are due to the different limiting behavior of an AR(1) model under in-fill asymptotics, depending on whether the AR(1) noise is independent, case (i), or there is autocorrelation, case (ii). The form of σ_ν^2 in each case ensures that the AR(1) process has fixed marginal variance, c_ν, for all values of n.

The key condition here is (C3) which governs how accurate the model assumed by DeCAFS is to the true data-generating procedure. Clearly, if the model is correct then (C3) holds with $\alpha = 1$. The following proposition gives upper bound on α in the case where the covariance of the data-generating model is that of a random walk plus AR(1) process, but with different parameter values to those assumed by DeCAFS in (C2), for example, due to misestimation of these parameters.

Proposition 6. Assume the noise process $\zeta(t)$ of the data-generating process (C1) is equal to a random walk plus an AR(1) process.

(i) If $\text{cov}(\zeta(t), \zeta(s)) = c_\eta^0 \min(t, s)$ for $t \neq s$ and $\text{var}(\zeta(t)) = c_\eta^0 t + c_\nu$, and DeCAFS is implemented as in (C2)(i), then (C3) holds with $\alpha = \max\{c_\nu^0/c_\nu, c_\eta^0/c_\eta\}$.

(ii) If $\text{cov}(\zeta(t), \zeta(s)) = c_\eta^0 \min(t, s) + c_\nu^0 \exp\{-c_\phi^0|t - s|\}$ and DeCAFS is implemented as in (C2)(ii), then for any $\epsilon > 0$ (C3) holds with

$$\alpha = \max\left\{\frac{c_\nu^0}{c_\nu}\frac{c_\phi^0}{c_\phi}(1 + \epsilon), \frac{c_\nu^0}{c_\nu}\left(1 + \frac{c_\phi}{c_\phi^0}\right)(1 + \epsilon), \frac{c_\eta^0}{c_\eta}\right\}.$$

The following result shows that we can consistently estimate the number of changepoints and gives a bound on the error in the estimate of changepoint locations, if we use DeCAFS under an assumption of a maximum number of changepoints (the assumption of a maximum number changes is for technical convenience, though is common in similar results, e.g., Yao 1988).

Theorem 2. Assume data, $y_{1:n}$, is generated as described in (C1), and let $\hat{m}$ and $\hat{\tau}_{1:\hat{m}}$ be the estimated number and location of the changepoints from DeCAFS implemented with parameters given by (C2), penalty $\beta = C\alpha \log n$ for some $C > 2$, and a maximum number of changes $m_{\max} \geq m^0$. Then as $n \to \infty$: if $\phi > 0$

$$\Pr\left(\hat{m} = m^0, \max_{i=1,\dots,m^0} \left|\hat{\tau}_i - \tau_i^0\right| = 0\right) \to 1;$$

and if $\phi = 0$

$$\Pr\left(\hat{m} = m^0, \max_{i=1,\dots,m^0} \left|\hat{\tau}_i - \tau_i^0\right| \leq (\log n)^2\right) \to 1.$$

The most striking part of this result is the very different behavior between $\phi = 0$ and $\phi > 0$. In the latter case, asymptotically we detect the position of the changepoints without error. This is because the positive autocorrelation in the noise across the changepoint helps us detect it. In fact, as $n \to \infty$ the signal for a change at t comes just from the lag-1 difference, $y_{t+1} - y_t$. The variance of $(y_{t+1} - y_t)$ is $O(1/n)$, and its mean is 0 except at changepoints, where it takes a fixed non-zero value. A simple rule based on detecting a change at t if and only if $(y_{t+1} - y_t)^2$ is above some threshold, $c_1(\log n)/n$ for some suitably large constant c_1, would consistently detect the changes. For the infill asymptotics, we consider, empirically DeCAFS converges to such an approach as $n \to \infty$.

The theorem also gives insight into the choice of penalty β. It is natural to choose this to be the smallest value that ensures consistency, as larger values will mean loss of power for detecting changes. Assuming the DeCAFS model is correct and we have the true hyper parameters this suggests using $\beta = 2 \log n$, the infimum of the penalties that are valid according to the theorem. This is the value that we use within our simulation study—though slightly inflating the penalty may be beneficial to account for error in the estimated hyper parameters or if we want to account for substantial model error.

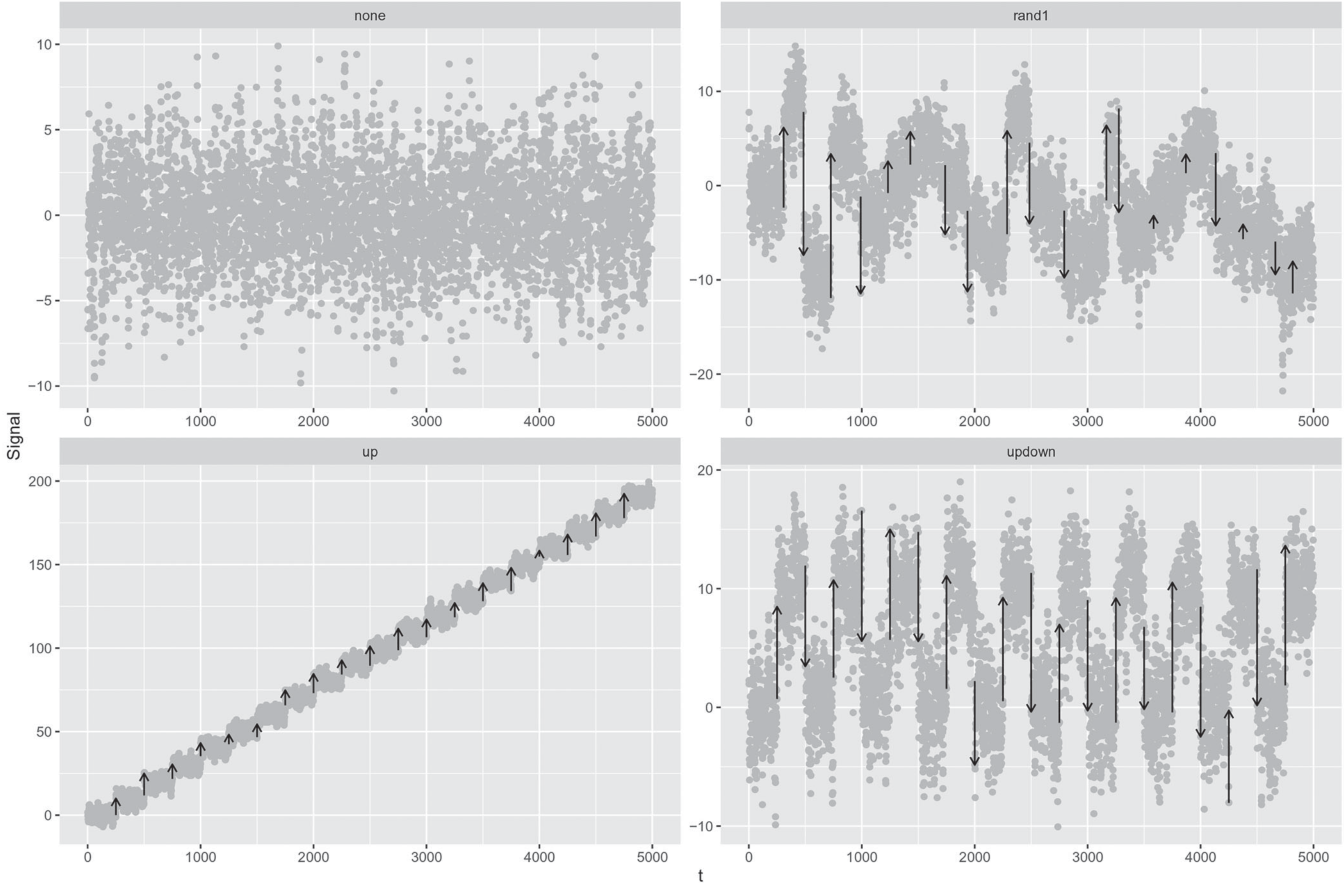

Figure 3. Four different change scenarios. Top-left, no change present, top-right, change pattern with 19 different changes, bottom-left up changes only, bottom-right, up-down changes of the same magnitude. In this particular example data were generated from an AR model with $\phi = 0.7$, $\sigma_\nu = 2$.

6. Simulation Study

6.1. Comparison with Changepoint Methods

We now asses the performance of our algorithm in a simulation study on four different change scenarios, illustrated in Figure 3. In all cases we run DeCAFS with $\beta = 2 \log n$, and estimate the parameters ϕ, σ_η, σ_ν as described in Section 4.

Simulations were performed over a range of evenly spaced values of ϕ, σ_η, σ_ν. There are no current algorithms that directly model local fluctuations in the mean, so we compare with two approaches the assume a constant mean between changes: FPOP (Maidstone et al. 2017) which also assumes IID noise, and AR1Seg (Chakar et al. 2017) that models the noise as an AR(1) process. We compare default implementation of each method, which involves robust estimates of the assumed model parameters. We also compare an implementation of FPOP with an inflated penalty (Bardwell et al. 2019) to account for the autocorrelated noise. To see the impact of possible misestimation of the model parameters, we also implement DeCAFS and AR1Seg using the true parameters when this is possible.

We focus on the accuracy of these methods at detecting the changepoints. We deem a predict change as correct if it is within ±2 observations of a true changepoint. As a measure of accuracy we use the F1 score, which is defined as the harmonic mean of the precision (the proportion of detected changes that are correct) and the recall (the proportion of true changes that are detected). The F1 score ranges from 0 to 1, where 1 corresponds to a perfect segmentation. Separate figures for precision and recall can be found in Section G of the supplementary material. Results reported are based over 100 replications of each simulation experiment, with each simulated data having $n = 5000$.

In Figure 4A, we report performances of the various algorithms as we vary ϕ for fixed values of $\sigma_\nu = 2$ and $\sigma_\eta = 0$. In Figure 4B, we additionally fix $\phi = 0.85$, but we vary the size of changes. In these cases, there is no random walk component and the model assumed by AR1Seg is correct.

There are a number of conclusions to draw from these results. First, we see that the impact of estimating the parameters on the performance of DeCAFS and AR1Seg is small. Second, we see that using a method which ignores autocorrelation but just inflates the penalty for a change does surprisingly well unless the autocorrelation is large, $\phi > 0.5$, this is inline with results on the robustness of using a square error cost for detecting changes in mean (Lavielle and Moulines 2000). For high values of ϕ, DeCAFS is the most accurate algorithm. The one exception are the simulations where there are no changes: the default penalty choice for AR1Seg is such that it rarely introduces a false positive.

In Figure 4C, we explore the effect of local fluctuations in the mean by varying σ_η. We see a quick drop off in performance for all methods as σ_η increases, consistent with the fact that it is harder to detect abrupt changes when the local fluctuations of the mean are greater. Across all experiments, DeCAFS was the most accurate algorithm.

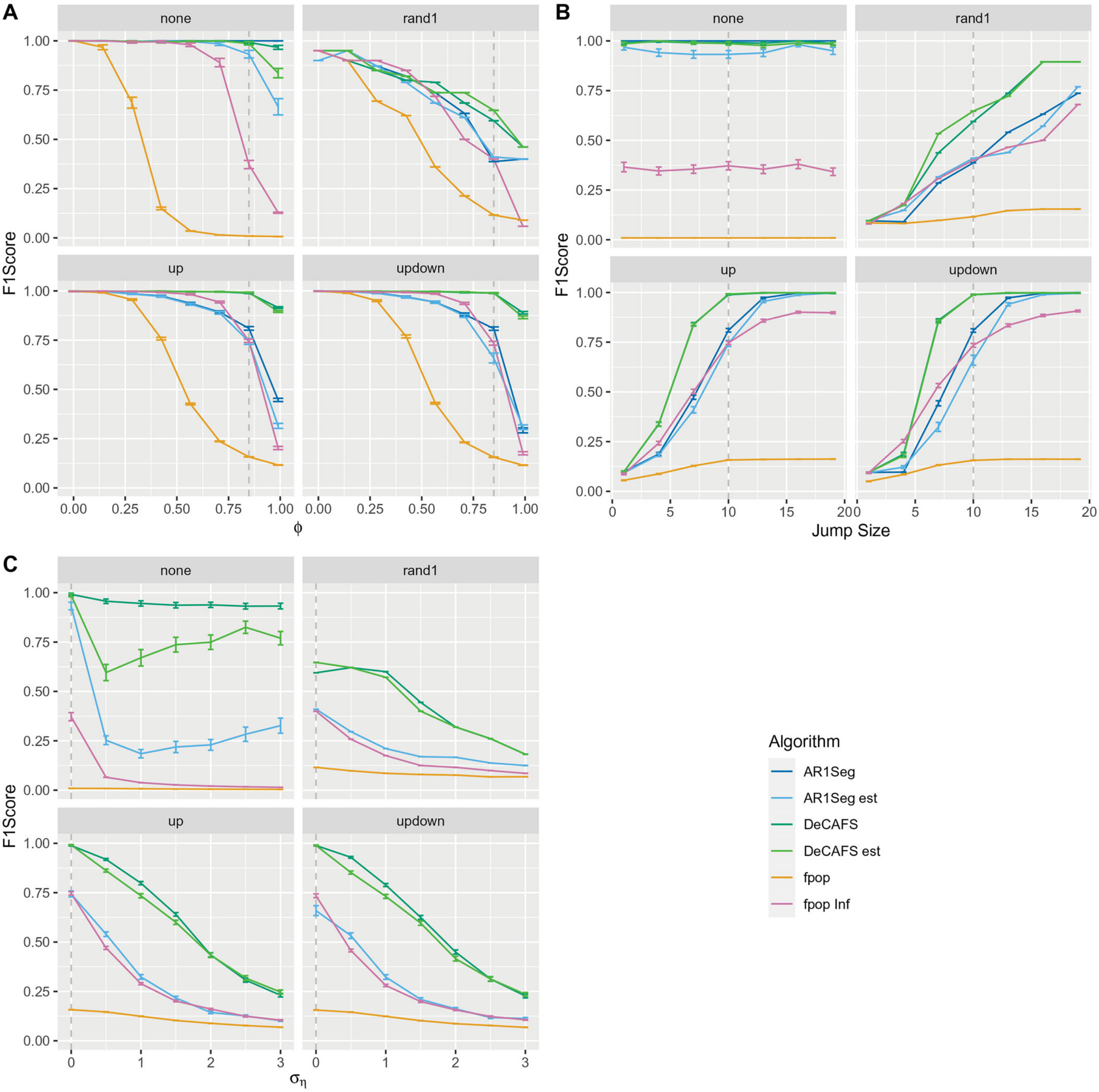

Figure 4. F1 Scores on the 4 different scenarios. In **A** a pure AR(1) over a range of values of ϕ, for fixed values of $\sigma_\nu = 2$, $\sigma_\eta = 0$ and a change of magnitude 10. In **B** a pure AR(1) process with fixed $\phi = 0.85$ and changes in the signal of various magnitudes. In **C** the full model with $\phi = 0.85$ for a range of values of σ_η. The gray line represents the cross-section between parameters values in **A**, **B**, and **C**. AR1Seg est. and DeCAFS est. refer to the segmentation of the relative algorithms with estimated parameters. Note, in **B** the results from DeCAFS and DeCAFS est overlap so only one line is visible. Other algorithms use the true parameter values.

One word of caution when fitting the full DeCAFS model, is that when σ_η is large it can be difficult to estimate the parameters, as a model with a very high random walk variance produces data similar to that of a model with constant mean but high autocorrelation. Whilst the impact on detecting changes of any errors when estimating the parameters is small, it can lead to larger errors in the estimate of the signal, μ_t: as different parameter estimates mean that the fluctuations in the data are viewed as either fluctuations in the noise process or in the signal. An example of this is shown in Section F of the supplementary material.

6.2. Robustness to Model Mis-specification

We now investigate the performance of DeCAFS when its model is incorrect. First we follow Chakar et al. (2017) and simulate data with a constant mean between changes but with the noise process being AR(2), i.e. $\epsilon_t = \phi_1\epsilon_{t-1} + \phi_2\epsilon_{t-2} + \nu_t$. In Figure 5 we report F1 Scores for DeCAFS and AR1Seg as we vary range ϕ_2. Obviously as $|\phi_2|$ increases, all algorithms perform worse, but the segmentations returned from DeCAFS are the more reliable as we increase the level of model error.

Second, we consider local fluctuations in the mean that are generated by a sinusoidal process rather than the random walk

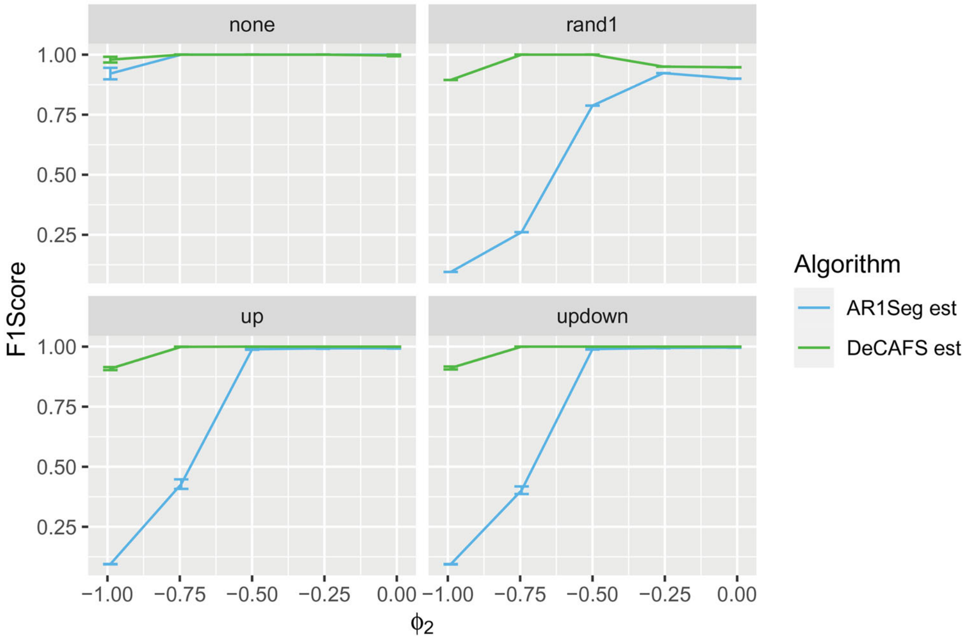

Figure 5. F1 score on different scenarios with AR(2) noise as we vary ϕ_2. Data simulated fixing $\sigma_\nu = 2$, $\sigma_\eta = 0$ and $\phi_1 = 0.3$ over a change of size 20.

model, see Figure 6B. In Figure 6A, we compare performance of DeCAFS and AR1Seg as we vary the frequency of the sinusoidal process. Again we see that DeCAFS gives more reliable segmentations in these cases. In the three change scenarios performance decrease as we increase the frequency of the process. In these cases, it becomes significantly harder to detect any changepoints, however, DeCAFS still has higher scores than AR1Seg since it is more robust and returns fewer false positives.

Additional simulation results, showing the robustness of DeCAFS to the mean fluctuations being from an Ornstein-Uhlenbeck process or to the noise being AR(1) within a segment but independent across segments are shown in Sections F and G in the Supplementary Material.

6.3. *Comparison to LAVA*

The LAVA method of Chernozhukov, Hansen, and Liao (2017) can be applied to model a signal as the sum of a piecewise constant function and a locally fluctuating function: and is thus a natural alternative to DeCAFS. If we let X denote the $n \times n$ matrix whose ith column has $i-1$ zeroes followed by $n-i+1$ ones then LAVA can estimate the mean $\mu_{1:n}$ as $X(f_{1:n}+g_{1:n})$ where the vectors f and g minimize

$$(y-X(f_{1:n}+g_{1:n}))^T(y-X(f_{1:n}+g_{1:n}))+\lambda_1||f_{1:n}||_1+\lambda_2||g_{1:n}||_2^2,$$

with $||\cdot||_1$ and $||\cdot||_2$ denoting, respectively, the L_1 and L_2 norms.

The interpretation of this is that $(f+g)_i$ is the change in the mean of the data from time $i-1$ to time i. The penalties are such that $f_{1:n}$ is sparse, and thus is modeling the abrupt changes in the mean, while $g_{1:n}$ is dense and is accounting for the local fluctuations. It is possible to show that DeCAFS is equivalent to LAVA if we have independent noise and replace the L_1 norm by an L_0 norm.

We implemented LAVA using the `lavash` package in R. This implementation of LAVA is substantially slower than DeCAFS, and empirically the computational cost appears to increase with the cube of the number of data points. As a result we compare DeCAFS with LAVA on simulated data of length $n = 1000$ (for which LAVA takes, on average, 132 seconds and DeCAFS 0.02 seconds per dataset).

LAVA tunes λ_1 by cross-validation. We used a plug-in value for λ_2 based on the oracle choice suggested in Chernozhukov, Hansen, and Liao (2017). This choice depends on the variance of the local fluctuations, i.e. the variance of the random walk component in our model, and we implemented LAVA using both the true variance (denoted LAVA), and using the same estimate of the variance as we use for DeCAFS (denoted LAVA_est). The plug-in approach seemed to give better results, and was substantially faster than using cross-validation to tune λ_2. It also makes a comparison with DeCAFS easier, as in situations where neither method estimates any abrupt changes, the two methods then give essentially identical estimates of the mean.

Results from analyzing data simulated under a pure random-walk model (so $\phi = 0$) are shown in Figure 7. While both methods perform similarly at estimating the underlying mean, we see that LAVA is less reliable at estimating the changepoint locations—and often substantially over-estimates the number of changepoints. As summarized in the introduction, this is not unexpected as it is common for methods that use ℓ_1 penalties on the size of an abrupt change to overestimate the number of changes (see, e.g., Fearnhead, Maidstone, and Letchford 2018; Jewell et al. 2020).

7. Gene Expression in *Bacilus subtilis*

We now evaluate DeCAFS on estimating the expression of cells in the bacteria *Bacilus subtilis*. Specifically, we analyze data from Nicolas et al. (2009), which is data from tiling arrays with a resolution of less than 25 base pairs. Each array contains several hundred thousand probes which are ordered according to their

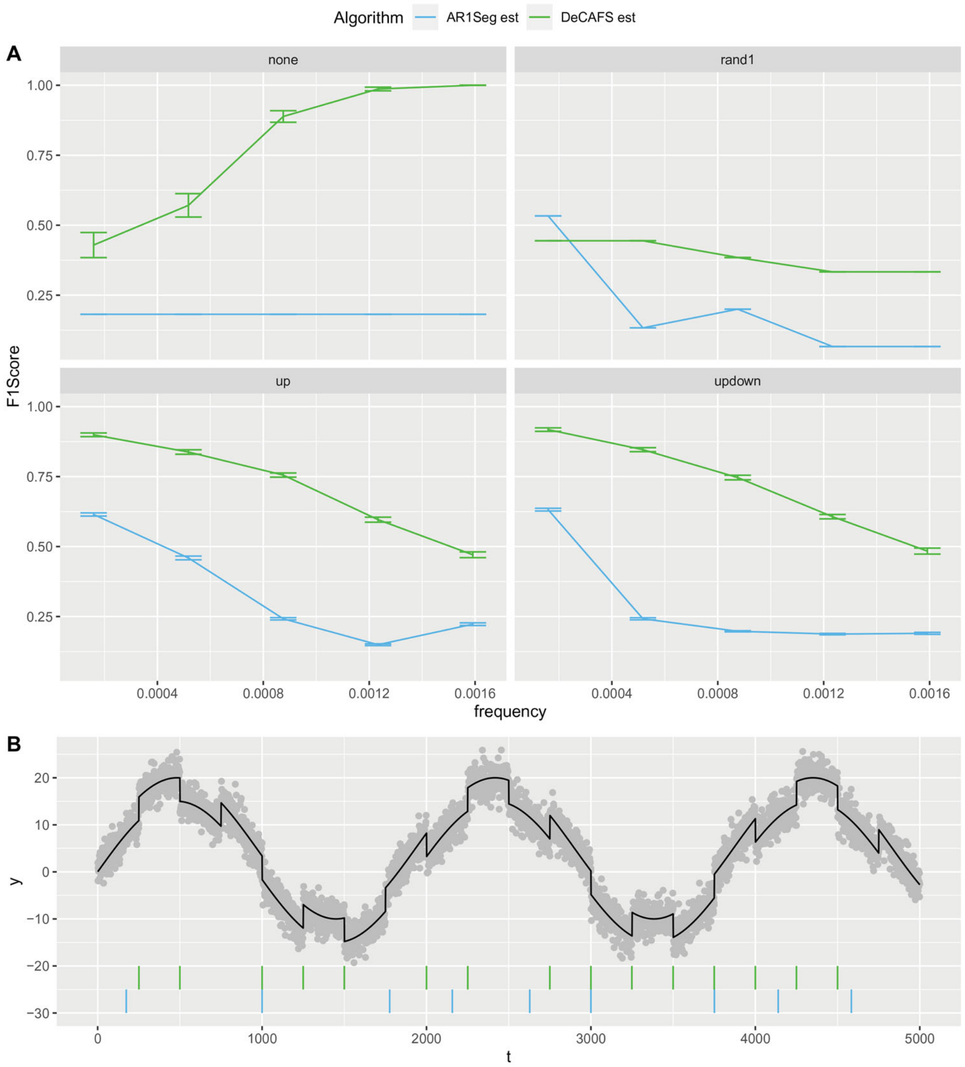

Figure 6. In **A** the F1Score on the 4 scenarios for the Sinusoidal Model for fixed amplitude of 15, changes of size 5 and IID Gaussian noise with a variance of 4, as we vary the frequency of the sinusoidal process. In **B** an example of a realization for the updown scenario, vertical segments refer to estimated changepoint locations of DeCAFS (in light green) and AR1Seg (in blue).

position on the bacterial chromosome. For a probe, labeled t say, we get an RNA expression measure, Y_t. Figure 8 shows data from 2000 probes. Code and data used in our analyses, presented below, are available on forgemia : *https://forgemia.inra.fr/guillem.rigaill/decafsrna*.

The underlying expression level is believed to undergo two types of transitions, large changes which Nicolas et al. (2009) call shifts and small changes which they call drifts. Thus, it naturally fits our modeling framework of abrupt changes, the shifts, between which there are local fluctuations caused by the drifts. To evaluate the performance of DeCAFS at estimating how the gene expression levels vary across the genome we will compare to the hmmTiling method of Nicolas et al. (2009). This method fits a discrete state hidden Markov model to the data, with the states being the gene expression level, and the dynamics of the hidden Markov model corresponding to either drifts or shifts. As a comparison of computational cost of the two methods, DeCAFS takes about 7 min to analyze data from one of the strands, each of which contains around 192,000 data points. Nicolas et al. (2009) reported a runtime of 5 h and 36 min to analyze both strands.

A comparison of the estimated gene expression level from DeCAFS and from hmmTiling, for a 2000 base pair region of the genome, is shown in Figure 8. We see a close agreement in the estimated level for most of the region, except for a couple of regions where hmmTiling estimates abrupt changes in gene expression level that DeCAFS does not.

To evaluate which of DeCAFS and hmmTiling is more accurate, we follow Nicolas et al. (2009) and see how well the estimated gene expression levels align with bioinformatically predicted promoters and terminators. A promoter roughly corresponds to the start of a gene, and a terminator the end, and we expect gene expression to increase around a promoter and decrease around a terminator.

For promoters, consider all probe locations t from the tiling chip and consider a threshold parameter δ. We can count the number of probe locations with a predicted difference $\hat{d}_t = \hat{\mu}_{t+1} - \hat{\mu}_t$ strictly greater than δ. We call this $R(\delta)$. Among those probes, we can count how many have a promoter nearby (within 22 base pairs). We call this $M(\delta)$. By symmetry we can define an equivalent measure for terminators. A method is better than another if for the same $R(\delta)$ it achieves a larger $M(\delta)$.

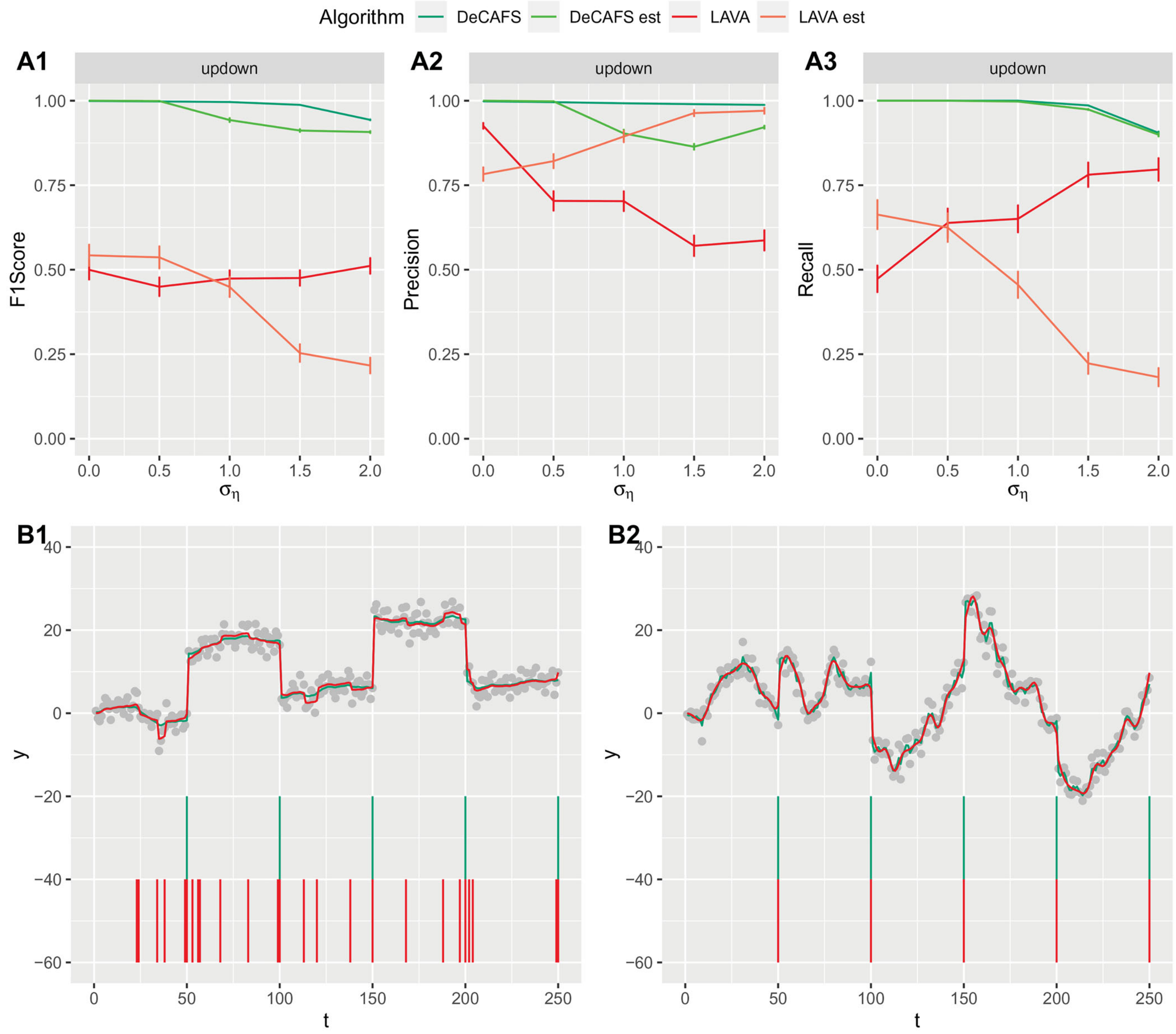

Figure 7. On top: comparison of the F1 Score in **A1**, Precision in **A2** and MSE in **A3**, for DeCAFS (in green) and LAVA (in red) with oracle initial parameters and the relative results with estimated initial parameters (in lighter colours), on the updown scenario for a random walk signal over a range of values of σ_η. On the bottom the first 250 observations of two realizations of the experiment with, in **B1**, σ_η equal to 0.5 and in **B2** σ_η equal to 2. Again, the continuous lines over the data points represent the signal estimates of DeCAFS and LAVA; and the vertical lines below show their estimated changepoint locations.

We used a data-driven approach to choose the penalty, β for DeCAFS, benefitting from having separate data from the plus and minus strand of the chromosome. For Figure 9 the penalty was learned on the minus strand data and tested on the plus strand data. More specifically we ran DeCAFS on the minus strand for $\beta = \{2\log(n), 2.5\log(n)...30\log(n)\}$. For each β we computed $M(\delta)$ for a fixed $R(\delta) = 750$ and took the β maximizing $M(\delta)$: $8\log(n)$ for promoters and $5\log(n)$ for terminators.

Figure 9 plots $M(\delta)$ against $R(\delta)$ for the plus strand as we vary δ for DeCAFS and two different estimates from hmmTiling. The first, hmmTiling.ori, are the prediction presented in Nicolas et al. (2009). The second, hmmtTiling.all, are those obtained when using all probes rather than only those called transitions by hmmTiling.

In the case of promoters, the prediction of hmmTiling is slightly better than DeCAFS for lower thresholds but noticeably worse for higher thresholds. In the case of terminators, the prediction of DeCAFS are clearly better than those of hmmTiling. Given that DeCAFS was not developed to analyze such data we believe that its relatively good performances for promoters and better performances for terminators is a sign of its versatility.

Unsurprisingly, we get different results and plots if we vary the value for $R(\delta)$ used to select the penalty or if we use the plus strand as a training dataset and the minus strand as a testing dataset. However, Figure 16 in Appendix F.5 shows that $M(\delta)$ is robust to these choices and that the range of penalty for which DeCAFS is close to or better than hmmTiling is quite large, particularly for terminators.

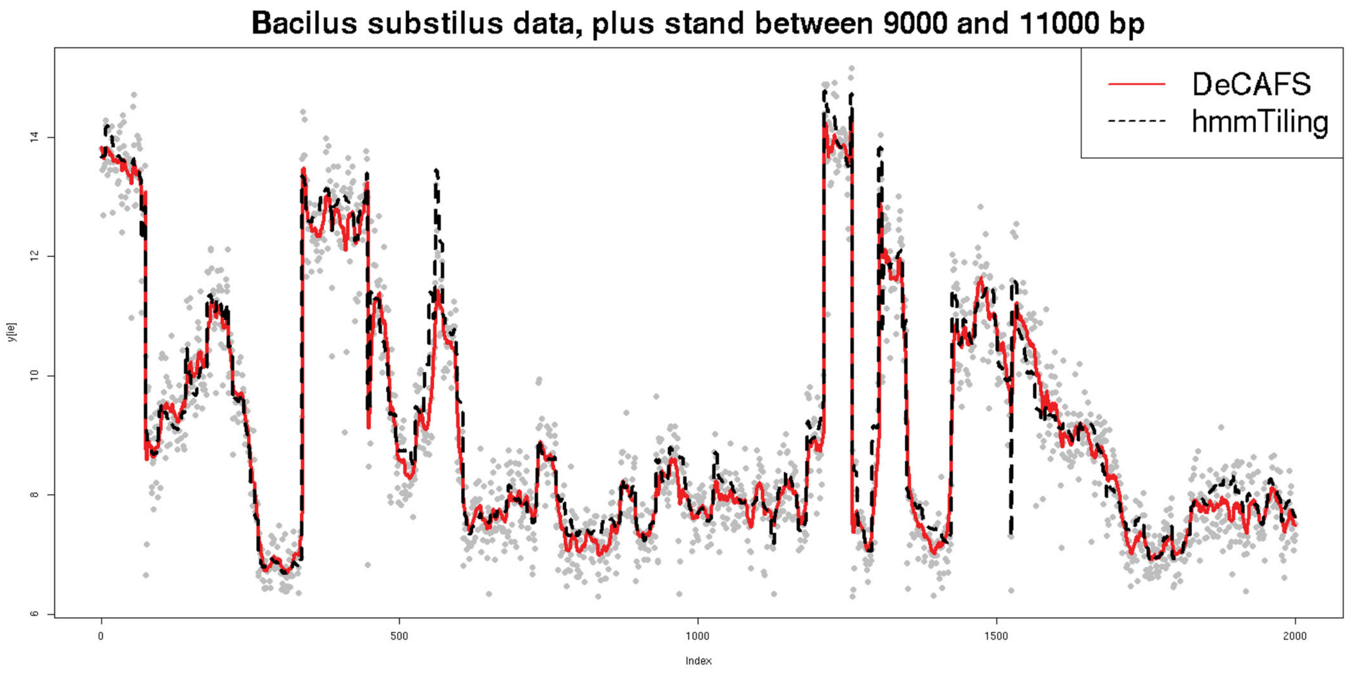

Figure 8. Data on 2000 bp of the plus-strand of the *Bacilus subtilis* chromosome. Gray dots show the original data. The plain red line represents the estimated signal of DeCAFS with a penalty of $10\log(n)$. The dashed black line represents the estimated signal of hmmTiling.

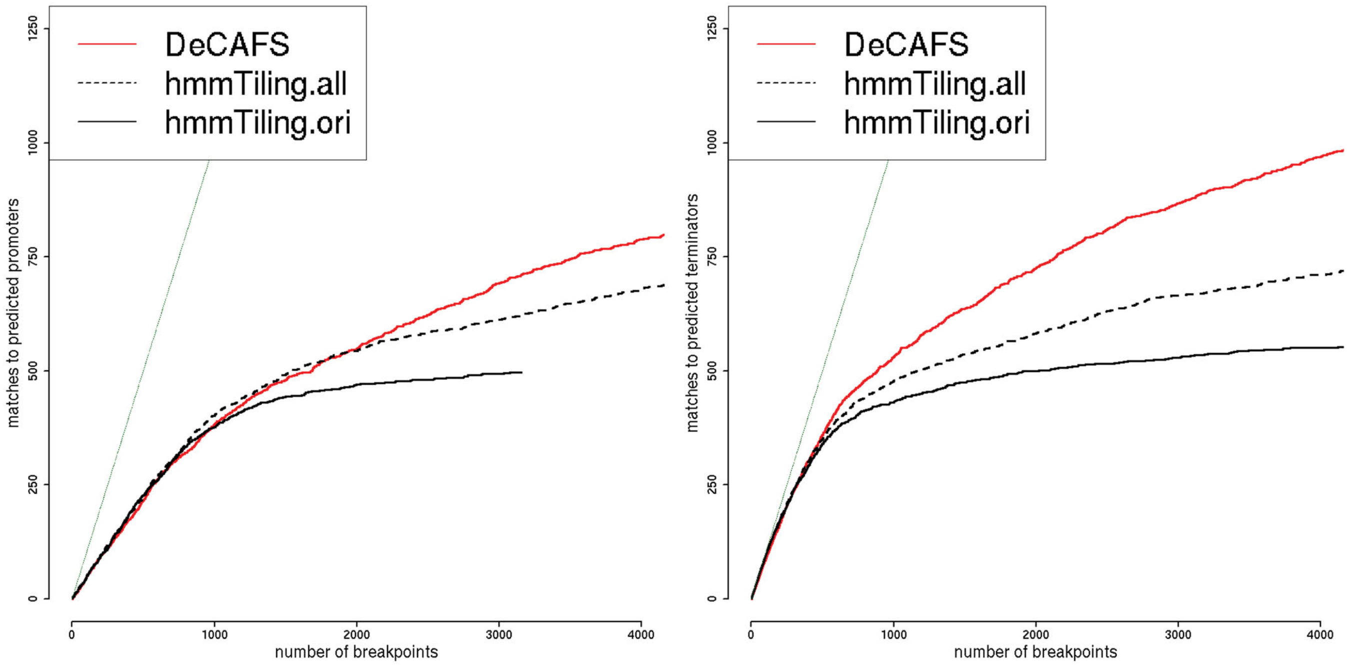

Figure 9. Benchmark comparisons. The number of promoters (left) and terminators (right) correctly predicted on the plus strand, $M(\delta)$ using a 22 bp distance cutoff, as a function of the number of predicted breakpoints, $R(\delta)$. Plain black lines are the results of hmmTiling (as reported in Figure 4 of Nicolas et al. 2009)). Dotted black lines are the results of hmmTiling when considering all probes rather than only those called transitions. Plain red lines are the results of DeCAFS using $\beta = 8\log(n)$ for promoters and $5\log(n)$ for terminators. These values were learned on the minus strand using a data-driven approach. The thin dark-green leaning line represent $y = x$.

8. Discussion

There are various ways of developing the DeCAFS algorithm, that build on other extensions of the functional pruning version of optimal partitioning. For example, to make the method robust to outliers, we can use robust losses, such as the biweight loss, instead of square error loss to measure our fit to the data (Fearnhead and Rigaill 2019). Alternatively, we can incorporate additional constraints on the underlying mean such as monotonicity (Hocking et al. 2020) or that the mean decays geometrically between changes (Jewell and Witten 2018; Jewell et al. 2020). Also it can be extended to allow for the noise to be independent across segments, but AR(1) within a segment. Finally, the algorithm is inherently sequential and thus should be straightforward to adapt to an online analysis of a data stream.

We do not claim that the method we present in Section 4 for estimating the parameters in our model is best. It is likely that more efficient or more robust methods are possible, for example using different robust estimates of the variances of the k-lag difference data (Rousseeuw and Croux 1993); or using iterative procedures where we estimate the changepoints, and then conditional on these changepoints re-estimate the parameters. Using better estimates should lead to further improvement on the statistical performance we observed in Section 6. Our theoretical results suggest that for estimating changes, misestimation of the parameters, or errors in our model for the noise or local fluctuations, can be corrected by inflating the penalty for adding a changepoint. As such, in applications we would suggest implementing the method for a range of penalty values, for example using the CROPS algorithm (Haynes, Eckley, and Fearnhead 2017), and then choosing the number of penalties using criteria that consider how the fit to data improves as we add more changes (e.g., Arlot et al. 2016; Fryzlewicz 2018*a*; Arlot 2019); or use training data to evaluate performance for different values of β, as we did in Section 7.

Acknowledgments

We thank Pierre Nicolas for providing the output of hmmTiling on the *Bacillus subtilis* data and his R code allowing us to generate Figure 9, which closely resembles Figure 4 of Nicolas et al. (2009).

Funding

This work was supported by EPSRC grant EP/N031938/1, and an ATIGE grant from Genopole. The IPS2 benefits from the support of the LabEx Saclay Plant Sciences-SPS.

ORCID

Gaetano Romano http://orcid.org/0000-0002-7751-9017
Guillem Rigaill http://orcid.org/0000-0002-7176-7511
Vincent Runge http://orcid.org/0000-0002-4857-1799
Paul Fearnhead http://orcid.org/0000-0002-9386-2341

References

Arlot, S. (2019), “Minimal Penalties and the Slope Heuristics: A Survey,” arXiv:1901.07277 . [15]

Arlot, S., Brault, V., Baudry, J.-P., Maugis, C., and Michel, B. (2016), *capushe: CAlibrating Penalities Using Slope HEuristics*. R package version 1.1.1. https://CRAN.R-project.org/package=capushe [15]

Bardwell, L., Fearnhead, P., Eckley, I. A., Smith, S. and Spott, M. (2019), “Most Recent Changepoint Detection in Panel Data,” *Technometrics*, 61, 88–98. [2,9]

Bauschke, H. H., and Combettes, P. L. (2011), *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*, Vol. 408. Berlin: Springer. [5]

Chakar, S., Lebarbier, E., Lévy-Leduc, C., and Robin, S. (2017), ‘A Robust Approach for Estimating Change-points in the Mean of an AR(1) Process,” *Bernoulli*, 23, 1408–1447. [2,4,9,10]

Chernozhukov, V., Hansen, C. and Liao, Y. (2017), “A Lava Attack on the Recovery of Sums of Dense and Sparse Signals,” *The Annals of Statistics*, 45, 39–46. [3,11]

Eichinger, B., and Kirch, C. (2018), “A Mosum Procedure for the Estimation of Multiple Random Change Points,” *Bernoulli*, 24, 526–564. [1]

Fearnhead, P. and Liu, Z. (2011), “Efficient Bayesian Analysis of Multiple Changepoint Models With Dependence Across Segments,” *Statistics and Computing*, 21, 217–229. [1]

Fearnhead, P., Maidstone, R. and Letchford, A. (2018), “Detecting Changes in Slope With an l 0 Penalty,” *Journal of Computational and Graphical Statistics*, 28, 265–275. [3,11]

Fearnhead, P., and Rigaill, G. (2019), “Changepoint Detection in the Presence of Outliers,” *Journal of the American Statistical Association*, 114, 169–183. [1,14]

Frick, K., Munk, A., and Sieling, H. (2014), “Multiscale Change-point Inference,” *Journal of the Royal Statistical Society*, Series B, 76, 495–580. [1]

Fryzlewicz, P. (2014), “Wild Binary Segmentation for Multiple Change-Point Detection’, *Annals of Statistics*, 42, 2243–2281. [1]

Fryzlewicz, P. (2018a), “Detecting Possibly Frequent Change-points: Wild Binary Segmentation 2 and Steepest-drop Model Selection,” arXiv:1812.06880. [15]

——— (2018b), “Tail-greedy Bottom-up Data Decompositions and Fast Multiple Change-point Detection,” *The Annals of Statistics*, 46, 3390–3421. [1]

Futschik, A., Hotz, T., Munk, A. and Sieling, H. (2014), “Multiscale DNA Partitioning: Statistical Evidence for Segments,” *Bioinformatics*, 30, 2255–2262. [1]

Haynes, K., Eckley, I. A., and Fearnhead, P. (2017), “Computationally Efficient Changepoint Detection for a Range of Penalties,” *Journal of Computational and Graphical Statistics*, 26, 134–143. [15]

Hinkley, D. V. (1971), “Inference About the Change-point From Cumulative Sum Tests,” *Biometrika*, 58, 509–523. [7]

Hocking, T. D., Rigaill, G., Fearnhead, P., and Bourque, G. (2020), “Constrained Dynamic Programming and Supervised Penalty Learning Algorithms for Peak Detection in Genomic Data,” *Journal of Machine Learning Research*, 21, 1–40. [4,14]

Hotz, T., Schütte, O. M., Sieling, H., Polupanow, T., Diederichsen, U., Steinem, C., and Munk, A. (2013), “Idealizing Ion Channel Recordings by a Jump Segmentation Multiresolution Filter,” *IEEE Transactions on Nanobioscience*, 12, 376–386. [1]

Jackson, B., Scargle, J. D., Barnes, D., Arabhi, S., Alt, A., Gioumousis, P., Gwin, E., Sangtrakulcharoen, P., Tan, L. and Tsai, T. T. (2005), “An algorithm for Optimal Partitioning of Data on an Interval,” *IEEE Signal Processing Letters*, 12, 105–108. [2,4]

Jalali, A., Ravikumar, P., and Sanghavi, S. (2013), “A Dirty Model for Multiple Sparse Regression,” *IEEE Transactions on Information Theory*, 59, 7947–7968. [3]

Jewell, S. W., Hocking, T. D., Fearnhead, P., and Witten, D. M. (2020), “Fast Nonconvex Deconvolution of Calcium Imaging Data,” *Biostatistics*, 21, 709–726. [3,11,14]

Jewell, S., and Witten, D. (2018), ‘Exact Spike Train Inference Via l_0 Optimization,” *The Annals of Applied Statistics*, 12, 2457–2482. [14]

Killick, R., Eckley, I. A., Ewans, K., and Jonathan, P. (2010), “Detection of Changes in Variance of Oceanographic Time-series Using Changepoint Analysis,” *Ocean Engineering*, 37, 1120–1126. [1]

Killick, R., Fearnhead, P., and Eckley, I. A. (2012), “Optimal Detection of Changepoints With a Linear Computational Cost,” *Journal of the American Statistical Association*, 107, 1590–1598. [1,2,4]

Kim, C.-J., Morley, J. C., and Nelson, C. R. (2005), “The Structural Break in the Equity Premium,” *Journal of Business & Economic Statistics*, 23, 181–191. [1]

Kim, S.-J., Koh, K., Boyd, S., and Gorinevsky, D. (2009), “ℓ_1 Trend Filtering,” *SIAM Review*, 51, 339–360. [3]

Lavielle, M., and Moulines, E. (2000), “Least-squares Estimation of an Unknown Number of Shifts in a Time Series,” *Journal of Time Series Analysis*, 21, 33–59. [2,9]

Lin, K., Sharpnack, J. L., Rinaldo, A. and Tibshirani, R. J. (2017), “A Sharp Error Analysis for the Fused Lasso, With Application to Approximate Changepoint Screening,” in *Advances in Neural Information Processing Systems*, eds. I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan and R. Garnett, Vol. 30, Red Hook, NY: Curran Associates Inc., 6884–6893. [3]

Maidstone, R., Hocking, T., Rigaill, G.. and Fearnhead, P. (2017), “On Optimal Multiple Changepoint Algorithms for Large Data,” *Statistics and Computing*, 27, 519–533. [1,2,4,9]

Nicolas, P., Leduc, A., Robin, S., Rasmussen, S., Jarmer, H., and Bessières, P. (2009), “Transcriptional Landscape Estimation From Tiling Array Data

Using a Model of Signal Shift and Drift," *Bioinformatics*, 25, 2341–2347. [11,12,13,14,15]

Olshen, A. B., Venkatraman, E. S., Lucito, R., and Wigler, M. (2004), "Circular Binary Segmentation for the Analysis of Array-Based DNA Copy Number Data," *Biostatistics*, 5, 557–572. [1]

Reeves, J., Chen, J., Wang, X. L., Lund, R., and Lu, Q. Q. (2007), "A Review and Comparison of Changepoint Detection Techniques for Climate Data," *Journal of Applied Meteorology and Climatology*, 46, 900–915. [1]

Rigaill, G. (2015), "A Pruned Dynamic Programming Algorithm to Recover the Best Segmentations With 1 to kmax Change-points," *Journal de la Societe Francaise de Statistique*, 156, 180–205. [4,5]

Rousseeuw, P. J., and Croux, C. (1993), 'Alternatives to the Median Absolute Deviation," *Journal of the American Statistical Association*, 88, 1273–1283. [15]

Ruanaidh, J. J. O., and Fitzgerald, W. J. (2012), *Numerical Bayesian Methods Applied to Signal Processing*, Springer Science & Business Media, New York: Springer-Verlag. [1]

Safikhani, A., and Shojaie, A. (2020), "Joint Structural Break Detection and Parameter Estimation in High-dimensional Non-stationary VAR Models," *Journal of the American Statistical Association*, DOI: 10.1080/01621459.2020.1770097, 1–26. [3]

Tibshirani, R. J. et al. (2014), 'Adaptive Piecewise Polynomial Estimation Via Trend Filtering," *The Annals of Statistics*, 42, 285–323. [3]

Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., and Knight, K. (2005), "Sparsity and Smoothness Via the Fused Lasso," *Journal of the Royal Statistical Society*, Series B, 67, 91–108. [3]

Yao, Y.-C. (1988), 'Estimating the Number of Change-points Via Schwarz's Criterion," *Statistics & Probability Letters*, 6, 181–189. [8]