Anomaly Detection in the Internet of Things

Ziyang Yang, B.Sc.(Hons.), M.Sc.(Hons.),

M.Res

Submitted for the degree of Doctor of

Philosophy at Lancaster University.

September 2025

STOR-i

excellence with impact

Abstract

The first part of this thesis addresses the challenge of efficiently detecting changes within a network of sensors, where minimizing communication between sensors and the cloud. We proposed two online, communication-efficient methods to detect such changes. We provide an asymptotic theory for the first method, disMOSUM, concerning consistency and the asymptotic distribution if there are no changes. Simulation results suggest that our method can achieve similar performance to the idealised setting, where we have no constraints on communication between sensors, but substantially reduce the transmission costs. The second approach, mixFOCuS, addresses the scenario where post-change parameters are unknown and the data belong to the exponential family, while still maintaining computational efficiency. A simulation study is conducted to evaluate the performance of our method with state-of-the-art approaches.

In the second part, we consider Bayesian approaches to online changepoint detection in linear regression models. Such methods are less common than frequentist methods due to their perceived higher computational cost, but they have advantages in terms of more naturally quantifying uncertainty and the ability to incorporate prior information about the type of change. We proposed a fast online Bayesian changepoint algorithm that can be applied to a wide range of problems, including detecting changes in mean or slope, and detecting changes in the presence of seasonal effects. Simulation suggests that the algorithm has a similar speed but higher accuracy compared to a benchmark pruning approach, which only prunes the candidate with the lowest posterior probability.

Acknowledgements

In the summer of 2019, I was sitting alone at the airport, waiting for my flight to the UK. It was my first time travelling and studying abroad by myself. I cried, and I did not know that moment was the changepoint of my life. Now, in 2025, I am at the end of my PhD. This thesis summarises the work I have done over the past few years, but what I gained goes far beyond these words.

A big thanks to my supervisors, Idris Eckley and Paul Fearnhead. I guess I am the biggest challenge in your supervision careers, and I truly appreciate your support, guidance, patience, and your continuous efforts in correcting my grammar. Idris, thanks for your support and for sharing your life experiences with me when I was lost. Paul, it has been a wonderful experience working with you. Thanks for your effort in reading and commenting on draft after draft. I can imagine how much time and energy it took to work through my horrible writing. Working with both of you has been the best part of my PhD journey. You have shown me what a good researcher is: staying humble, down-to-earth, and working hard. I have learned a lot from you, and I will carry these with me throughout my life.

I also want to thank BT for sponsoring my PhD. Special thanks to Dave and Rika for your support throughout these years. Travelling to Ipswich was exhausting, but our conversations have always been a source of inspiration.

Jon, thanks a lot for interviewing me and offering me a position at STOR-i. Joining the STOR-i programme has been the best opportunity and decision I have ever made.

During the interview, you asked me what would happen if I could not solve the problem during the PhD, and my answer was that I did not care. But the truth was, I struggled and doubted myself almost every single day. Thank you for the conversations and the support along the way. Now, I think I finally understand what that question really meant, and I believe I have gained the resilience to face whatever comes next.

To all my lecturers, I would not consider doing a PhD without you. Thanks to my lecturers during my undergraduate, Xue and Qi, for sparking my interest in statistics and inspiring me to pursue a PhD. At the University of Southampton, I was lucky to be taught and supported by incredible mentors. Olga, you were the first person who told me to be proud of myself. Jessie, you are an amazing supervisor and thanks for all your support.

To all my STOR-i cohorts and colleagues, I could not have done this without you. Thank you all for making STOR-i feel like a second home to me when everything around me was new and unfamiliar. Owen, the conversation we had after my first terrible presentation still stays with me. Jacob, I truly enjoyed our time in the reading group. Dan, Robyn, Maddie, Lídia, Katie and Rebecca, thank you for bringing me joy and lighting up my everyday life. I feel so lucky to have shared this journey with all of you.

Thanks to my family for your love and support throughout this journey. Without you, I would not have had the courage to board that flight. I hope I have made you proud. I also want to thank my Jellycats for keeping me company each night and listening patiently to all my rants, worries, and late-night thoughts. And, to my cat, Cube, thank you for always being there for me.

Finally, to my husband and best friend, Francis, thank you for your unconditional love and support. You were with me every single day, from the very beginning until now. You saw all my failures, my mistakes, and all my sad moments, and you were always with me. This journey was ours as much as it was mine.

Again, thank you all for walking with me.

Declaration

I declare that the work in this thesis has been done by myself and has not been submitted elsewhere for the award of any other degree. The word count of the main text is approximately 22,409 words. Chapters 3 and 4 are based on work that has been submitted for publication, as listed below.

Ziyang Yang

Chapter 3 has been accepted for publication as Yang, Z., Eckley, I.A. and Fearnhead, P., 2024. A communication-efficient, online changepoint detection method for monitoring distributed sensor networks. Statistics and Computing, 34(3), pp.1-16. The authors are grateful to Lawrence Bardwell who played a key role in inspiring this work.

Chapter 4 has been accepted by the Journal of Time Series Analysis for publication as Yang, Z., Eckley, I.A. and Fearnhead, P., to appear. mixFOCuS: A Communication-Efficient Online Changepoint Detection Method in Distributed System for Mixed-Type Data.

We gratefully acknowledge the financial support of the EPSRC via the STOR-i Centre for Doctoral Training (EP/S022252/1), EPSRC grant EP/R004935/1 (Eckley) and BT Research (Eckley, Fearnhead, Yang). We are also grateful to Dave Yearling (BT) for several helpful conversations that helped shape these researches.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Dr. Sheldon Cooper, a fictional character in *The Big Bang Theory*, once said "the inevitability of change might be a universal constant". This is a changing world, where even small shifts can lead to significant consequences if left unnoticed. The challenge lies not only in preventing change but also in recognizing the moment it occurs. How can we detect these shifts as soon as they happen, allowing us to react before their impacts? This is the motivation behind anomaly detection, which focuses on identifying unusual or unexpected patterns in data. Extensive research has been conducted on this task, and existing methods can be broadly classified based on learning paradigms (e.g. supervised or unsupervised), methodological strategies (such as statistical methods, clustering-based techniques, classification models, and reconstruction-based approaches), and the types of data they are applied to, including time series, images, and graph-structured data.

Within the broader scope of anomaly detection, our work focuses on changepoint detection, a statistical approach aimed at identifying points in time where the underlying properties of a process change. While not all changepoints are anomalous, anomalous changepoints can be identified by comparing current behaviour against a model of expected change. When we have training data, a changepoint in the deviations (e.g., large

residuals between observed and expected values) signals an anomaly.

Since Page (1954) first introduced changepoint detection algorithms for quality control, research in this area has expanded significantly. Early work focused primarily on offline methods, which analyse entire datasets to efficiently and accurately identify changepoints. More recently, attention has shifted toward developing online (real-time) algorithms, which aim to detect shifts quickly as data streams in. A detailed overview of these methods will be presented in Chapter 2.

Nowadays, with the advent of the Internet of Things (IoT), it is increasingly common to have large networks of sensors, where each sensor may collect different types of data, has limited local computing resources and the ability to transmit data to a central cloud. Detecting events that trigger changes in sensor data properties is a key concern. However, minimizing sensor-to-cloud communication might be necessary due either to privacy constraints or limited battery resources. Whilst there has been extensive research on both the univariate changepoint and multivariate changepoint, few methods explicitly account for communication constraints. In Chapters 3 and 4, we propose two real-time communication-efficient methods, disMOSUM and mixFOCuS, to detect changepoints in such a system.

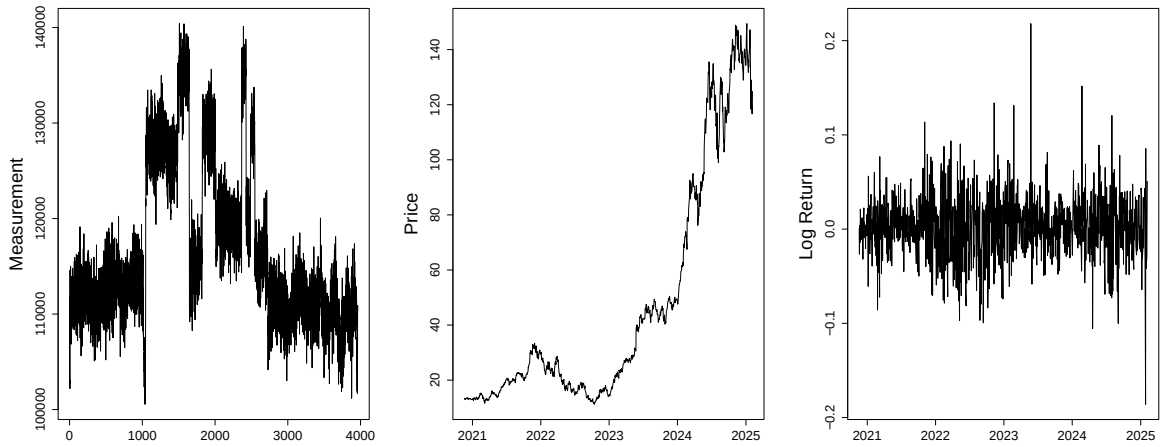
Although disMOSUM and mixFOCuS can efficiently and quickly detect changes, they lack a natural way to quantify uncertainty or incorporate prior information about the change. To address these limitations, in Chapter 5 we propose an online Bayesian changepoint detection method named Bagel. This approach can effectively detect the changes for linear models with constant time complexity per time step.

Chapter 2

Literature review

Changepoint detection, or breakpoint detection, is the process of identifying points in time where the statistical properties of a dataset undergo significant shifts. These shifts can take various forms, including changes in mean, variance, slope, or more complex structural transformations. Detecting such changes is important in numerous domains, as they often signal critical transitions with practical implications. For instance, as illustrated in Figure 2.1, abrupt changes in financial time series can indicate shifts in investor sentiment, economic events, or structural breaks in asset pricing, often leading to changes in market volatility and risk. In geophysics, changepoint detection helps identify subsurface layer boundaries in well-log data, aiding in resource exploration and geological analysis.

Over the past decade, extensive research has been conducted on changepoint detection, leading to a diverse set of methodologies tailored to different applications and data characteristics. These techniques can be categorized along several dimensions, including online versus offline detection, univariate versus multivariate analysis, frequentist versus Bayesian frameworks, and exact versus approximate solutions. The choice of method depends on multiple considerations, such as computational efficiency, the need for real-time processing, the robustness to noise and underlying statistical properties.



(a) (b) (c)

Figure 2.1: Examples of Different Changes. Figure (a) demonstrates a change-in-mean example in cleaned well-log data, where a shift in average values indicates transitions between geological layers. Figure (b) illustrates a change-in-slope example observed in Nvidia's closing stock price, reflecting an upward trend after 2023, followed by a decline in 2025. Figure (c) highlights a change in variance in the log returns of Nvidia's stock price. Increased variance suggests higher market volatility and risk.

For instance, online methods are ideal for real-time applications where immediate detection is crucial, while offline methods conduct analysis of historical data. Likewise, Bayesian approaches provide a probabilistic framework for handling uncertainty and incorporating prior knowledge, whereas frequentist methods offer deterministic solutions based purely on observed data.

In this chapter, we focus on univariate changepoint detection methods aimed at detecting changes in the mean. We consider both frequentist and Bayesian methodologies that are particularly foundational to the techniques introduced in the subsequent chapters. The chapter begins with a review of offline changepoint detection, followed by an introduction to online detection approaches.

2.1 Univariate offline changepoint detection

In the offline problem, we can access the entire data of size n, $X_1, \ldots, X_n$. The objective is to determine the number of changepoints and identify their locations. For simplicity, we assume that the data points are independent and identically distributed following a standard Gaussian distribution. While we omit discussions of changepoint detection in other contexts—such as autoregressive models, nonparametric approaches, or robust methodologies—we refer the reader to comprehensive reviews, such as Truong et al. (2020); Cho and Kirch (2024).

In the at most one change (AMOC) scenario, there is a single changepoint at an unknown time τ. The observations follow

$$X_1, \ldots, X_\tau \overset{\text{iid}}{\sim} f(\cdot|\theta) \quad \text{and} \quad X_{\tau+1}, \ldots, X_n \overset{\text{iid}}{\sim} f(\cdot|\theta + \delta),$$

where θ represents the mean before the changepoint, δ is the magnitude of the change in the mean after the changepoint, and $f(\cdot|\mu)$ is the density for an observation with mean μ. The goal of changepoint detection in this setting is to determine whether a significant change in mean has occurred and, if so, to estimate τ. In Section 2.1.1, we introduce common solutions to this problem.

If there are m changepoints, $\tau_{1:m}$, such that $0 = \tau_0 < \tau_1 < \tau_2 < \ldots < \tau_m < \tau_{m+1} = n$, the observations can be partitioned into $m+1$ segments, each following a distinct mean level. Specifically, we can denote that

$$X_{\tau_i+1}, \ldots, X_{\tau_{i+1}} \overset{\text{iid}}{\sim} f(\theta_i), \quad \text{for} \quad i = 0, 1, ..., m.$$

Here, $\theta_i (\forall i, \theta_i \neq \theta_{i+1})$ denotes the mean specific to its corresponding segment. In this more general multiple changepoint detection setting, the challenge extends to estimating both the number of changepoints and their locations. This often requires heavy

computing, so it is crucial to ensure that the model effectively captures significant shifts. In Sections 2.1.2 and 2.1.3, we introduce algorithms that have been specifically developed to tackle this problem efficiently.

2.1.1 Loglikelihood ratio test and offline CUSUM

We first consider the case where there is at most one changepoint in the sequence. When there is at most one changepoint, a straightforward approach to detection is to perform a hypothesis test. The null hypothesis assumes no change in the data, while the alternative hypothesis assumes the presence of a changepoint. At a candidate changepoint location $1 \leq j < n$, the log-likelihood ratio is given by

$$LR_j = -2 \log \frac{\max_\theta \prod_{i=1}^{n} f(x|\theta)}{\max_{\theta,\delta} \prod_{i=1}^{j} f(x|\theta) \prod_{i=j+1}^{n} f(x|\theta+\delta)}.$$

Substituting the maximum likelihood estimators of the segment parameters, this can be simplified as:

$$LR_j = -\left\{ \sum_{i=1}^{n} (x_i - \bar{x}_{1:n})^2 - \sum_{i=1}^{j} (x_i - \bar{x}_{1:j})^2 - \sum_{i=j+1}^{n} (x_i - \bar{x}_{j+1:n})^2 \right\}, \tag{2.1}$$

where $x_{1:t}$ is the set of observations from time 1 to t, and $\bar{x}_{1:t} = \frac{\sum_{i=1}^{t} x_i}{t}$ is the sample mean. Alternatively, equation 2.1 can be rewritten as

$$\sqrt{LR_j} = C_j = \left| \sqrt{\frac{n-j}{nj}} \sum_{i=1}^{j} x_i - \sqrt{\frac{j}{n(n-j)}} \sum_{i=j+1}^{n} x_i \right|.$$

This formula gives the offline version of CUSUM (online CUSUM is discussed in Section 2.2.1), which is equivalent to the square root of the log-likelihood ratio test statistic. Both test statistics measure the deviations in the sample mean before and after a potential changepoint at j. A larger value indicates stronger evidence of a change

occurring at j.

As the changepoint location, j, is unknown, we need to maximise the test statistic over the possible values of j:

$$U = \max_{1 \leq j < n} \sqrt{LR_j} = \max_{1 \leq j < n} C_j.$$

If $U < c$, there is no change. Otherwise, the estimated changepoint location is given by

$$\hat{\tau} = \arg\max_{1 \leq j < n} C_j.$$

The constant c is a predefined threshold that controls the significance level of the test. While theoretical bounds for c exist, they are often loose in practice (Fearnhead and Fryzlewicz, 2022). A practical approach to determining the threshold is through Monte Carlo simulation, which simulates the behaviour of U under the null hypothesis (Fearnhead and Fryzlewicz, 2022). The procedure involves the following steps:

1. Generate R independent data of fixed size of n under the null;
2. Calculate the test statistic U^r for rth replicate;
3. The threshold c is the empirical quantile of $\{U^r\}_{r=1}^R$ corresponding to the desired significance level.

2.1.2 Binary Segmentation, constrained and penalized approaches

When multiple changepoints are present, we must estimate both their number and locations simultaneously. How can we extend the AMOC approaches mentioned in Section 2.1.1 to handle the multiple changepoints case? An intuitive solution follows the "divide and conquer" principle, which leads to one of the most widely used methods: Binary Segmentation.

Binary segmentation

Binary Segmentation (BS) can extend single change detection algorithms, such as CUSUM or likelihood ratio test, to detect multiple changes. This approach was first introduced by Scott and Knott (1974) in the context of clustering. In BS, the test for a single changepoint is repeatedly applied. If evidence for a changepoint is found, the data is split at the estimated location, and the procedure is recursively applied to each interval until no further changepoints are identified. The algorithm is shown in Algorithm 1.

Algorithm 1: Binary Segmentation

Initialize: threshold c**, changepoint** $cp \leftarrow \{\}$**, interval sets** $I \leftarrow \{[0, n]\}$**.**
while $I \neq \emptyset$ **do**
  for i *in* $|I|$ **do**
    Set the start and the end of the segment:
    $s = \min\{t : t \in I_i\}, e = \max\{t : t \in I_i\}$.
    Stop if segment is too small:
    if $e - s < 2$ **then**
      $I \leftarrow I \setminus I_i$
    end
    else
      Calculate $U = \max_{s \leq j \leq e} C_j$.
      if $U > c$ **then**
        $\hat{\tau} = \arg\max_{j \in [s,e]} C_j$,
        $cp \leftarrow cp \cup \{\hat{\tau}\}$,
        Split the interval into two parts:
        $I \leftarrow I \setminus I_i \cup [s, \hat{\tau}] \cup [\hat{\tau} + 1, t]$.
      end
      else
        $I \leftarrow I \setminus I_i$
      end
    end
  end
end
return cp.

BS is computationally efficient with a time complexity of $O(n \log(n))$ on univariate data. It has also been extended into multivariate, and high-dimensional changepoint

detection, see Cho and Fryzlewicz (2014). However, it struggles to detect changes within short intervals. As demonstrated by Fryzlewicz (2014), when the sample size increases, $n \to \infty$, BS asymptotically fails to detect changepoints if the segment length is shorter than $O(n^{3/4})$.

To overcome the drawbacks of the BS, one solution is Circular Binary Segmentation (CBS), which was introduced by Olshen et al. (2004). CBS assumes that the mean before the first changepoint equals the mean after the last changepoint, so the data is treated as circular. CBS has been widely used in detecting the variations in DNA copy numbers, such as in Venkatraman and Olshen (2007); Hsu et al. (2011); Olshen et al. (2011); Mouliere et al. (2017); Hoadley et al. (2018). However, it has expensive computational complexity.

Another solution is Wild Binary Segmentation (WBS), introduced by Fryzlewicz (2014). Unlike BS, which partitions the data based on the decision rule, WBS improves detection power by scanning a set of h randomly selected subintervals. On each subinterval, we perform a statistical test for the presence of a single change, recording both the test statistic value and the estimated location of the change. Subintervals with test statistics below a predefined threshold are discarded. Among the remaining subintervals, the one with the largest test statistic is selected, its estimated changepoint is added to the list of detected changes, and all subintervals overlapping this location are removed. This process is repeated iteratively until all subintervals have been either processed or eliminated. However, the computational efficiency and accuracy of WBS depends on the number of sub-intervals chosen. More intervals lead to improved detection at the cost of increased computational cost.

There are some major extensions after WBS. Narrowest-Over-Threshold (Baranowski et al., 2019, NOT) selects the shortest interval whose test statistic exceeds the threshold, instead of choosing the interval with the highest test statistic. To improve the computational efficiency, WBS2 (Fryzlewicz, 2020) extends WBS by recursively se-

lecting informative intervals. More recently, Seeded Binary Segmentation (SBS) was introduced by Kovács et al. (2023) to improve both the computational efficiency and reproducibility of WBS. Compared to WBS, SBS employs deterministic intervals, ensuring consistent results while maintaining computational efficiency. The selection process for deterministic intervals is outlined below:

- The first layer interval is $[0, n]$.
- For the rest of the layers k for $k = 2, \cdots, \lceil \log_{1/a} n \rceil$, we have $n_k = 2\lceil (1/a)^{k-1} \rceil - 1$ intervals. The kth layer and ith intervals are

$$I_k = \bigcup_{i=1}^{n_k} \{(\lfloor (i-1)s_k \rfloor, \lceil (i-1)s_k + l_k \rceil)\}, \quad i = 1, 2, \ldots, n_k.$$

  Here $l_k = na^{k-1}$ is the interval lengths, $s_k = \frac{n - l_k}{n_k - 1}$ is the shift and a is a decay parameter. $I = \bigcup_{k=1}^{\lceil \log_{1/a} n \rceil} I_k$ is the collection of the seeded intervals for kth layer.

By strategically selecting intervals, SBS achieves the time complexity of $O(n \log n)$ while maintaining the same accuracy as the WBS. The whole SBS algorithm is presented in Algorithm 2 and the graphical explanation is shown in Figure 2.2.

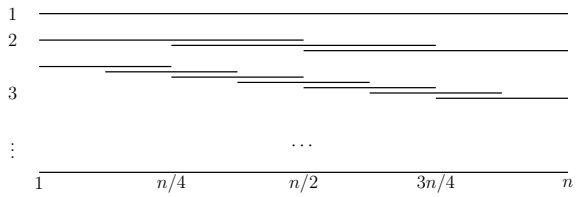


Figure 2.2: Example intervals in seeded binary segmentation at each recursion ($\alpha = 0.5$).

Constrained and penalised cost approaches

Another idea for detecting multiple changes is classification, where the data sequence is partitioned into different segments. That is, given data $X_1, X_2, \cdots, X_n$, the partition

Algorithm 2: Seeded Binary Segmentation

Initialize: threshold c**, changepoints** $cp \leftarrow \{\}$**, the intervals** I_k **for** $k = 1, \cdots, \lceil \log_{1/a} n \rceil$ **layers, and** $I = \bigcup_{k=1}^{\lceil \log_{1/a} n \rceil} I_k$**.**
$k = 1$
while $I \neq \emptyset$ *or* $k \leq \lceil \log_{1/a} n \rceil$ **do**
  for $i = 1, \cdots, n_k$ **do**
    Calculate U_i for each interval in I_k.
  end
  $U = \max_i U_i$,
  if $U > c$ **then**
    $\hat{\tau} = \{t : \arg\max U > c\}$,
    $cp \leftarrow cp \cup \{\hat{\tau}\}$,
    Remove all the intervals containing $\hat{\tau}$
    $I_{k'} \leftarrow I_{k'} \setminus \{(\lfloor (i-1)s_{k'} \rfloor, \lceil (i-1)s_{k'} + l_{k'} \rceil) \mid \hat{\tau} \in (\lfloor (i-1)s_{k'} \rfloor, \lceil (i-1)s_{k'} + l_{k'} \rceil)\}$,
    $\forall k' > k, i = 1, \ldots, n_{k'}$.
  end
  $k = k + 1$
end
return *cp*.

rule aims to maximize the gain or minimize the cost

$$\sum_{i=0}^{m} C(x_{(\tau_i+1):\tau_{i+1}}).$$

where m is the number of changepoints, $(\tau_i + 1) : \tau_{i+1}$ is the partition, and $C(\cdot)$ is the cost function. The cost function is often defined as twice the negative log-likelihood (with additive constants ignored). For example, in the case of Gaussian data with known standard deviation σ, the cost for m segment is given by:

$$\sum_{i=0}^{m} C(x_{(\tau_i+1):\tau_{i+1}}) = \sum_{i=0}^{m} \sum_{k=\tau_i+1}^{\tau_{i+1}} \left\{ \frac{1}{\sigma^2} (x_k - \bar{x}_{(\tau_i+1):\tau_{i+1}})^2 \right\}.$$

However, a key issue with this objective function is that it inherently favours $m = n-1$, as modelling each data point alone incurs no additional cost or achieves maximum gain, resulting in over-fitting. To address this, one can either restrict m explicitly or introduce

a regularization term. This leads to two key approaches that we will introduce below: the constrained cost approach and the penalized cost approach.

The Segmentation Neighbourhood Search approach (Auger and Lawrence, 1989) takes the idea of the constrained cost approach in that the number of the changepoints is fixed and the goal is to determine the optimal segmentation. Let $\mathcal{C}_{m,t}$ denote the minimum segmentation cost with mth changepoint is at time t, $\tau_0 = 0$ and $\tau_{m+1} = n$. At time t, the cost of $\mathcal{C}_{m,t}$ can be calculated recursively as:

$$\begin{aligned}
\mathcal{C}_{m,t} &= \min_{\tau_1:\tau_{m-1}} \left[\sum_{i=0}^{m-2} C(x_{\tau_i+1:\tau_{i+1}}) + C(x_{\tau_{m-1}+1:t}) \right] \\
&= \min_{\tau_{m-1}} \left[\min_{\tau_1:\tau_{m-2}} \sum_{i=0}^{m-2} C(x_{\tau_i+1:\tau_{i+1}}) + C(x_{\tau_{m-1}+1:t}) \right] \\
&= \min_{\tau_{m-1}} \left[\mathcal{C}_{m-1,\tau_{m-1}} + C(x_{\tau_{m-1}+1:t}) \right] \\
&= \min_{\tau \in \{m,\cdots,t-1\}} \left[\mathcal{C}_{m-1,\tau} + C(x_{\tau+1:t}) \right].
\end{aligned}$$

In practical scenarios where the number of changepoints m is unknown, we have to evaluate a range of values from 1 to M. This results in a total time complexity of $O(Mn^2)$, making it computationally expensive.

The optimal partition method (Jackson et al., 2005) follows the idea of the penalized cost approach, which introduces a regularization term β into the objective function to avoid over-fitting. The objective function is given by

$$\min_{\tau_1:\tau_m, m} \sum_{i=0}^{m} C(x_{\tau_i+1:\tau_{i+1}}) + m\beta.$$

Let $\mathcal{F}_t$ denotes the minimum penalized cost at time t. Then this function can be

rewritten as:

$$\begin{aligned}
\mathcal{F}_t &= \min_{\tau_1:\tau_m,m} \left[\sum_{i=0}^{m-1} \left[C(x_{\tau_i+1:\tau_{i+1}}) + \beta \right] + C(x_{\tau_m+1:t}) \right] \\
&= \min_{\tau_1:\tau_m,m} \left\{ \sum_{i=0}^{m-1} \left[C(x_{\tau_i+1:\tau_{i+1}}) + \beta \right] - \beta + C(x_{\tau_m+1:t}) + \beta \right\} \\
&= \min_{\tau_m} \left\{ \min_{\tau_1:\tau_{m-1},m-1} \sum_{i=0}^{m-1} \left[C(x_{\tau_i+1:\tau_{i+1}}) + \beta \right] - \beta + C(x_{\tau_m+1:t}) + \beta \right\} \\
&= \min_{\tau_m} \left\{ \mathcal{F}_{\tau_m} + C(x_{\tau_m+1:t}) + \beta \right\}.
\end{aligned}$$

Thus, this optimization problem can be solved recursively as:

$$\mathcal{F}_t = \min_{0 \le \tau < t} \left\{ \mathcal{F}_\tau + C(x_{\tau+1:t}) + \beta \right\},$$

with $O(n^2)$ computation. The penalty term β plays a crucial role in determining the balance between model complexity and fit. Two common choices are Akaike's Information Criterion (Akaike, 1974, AIC), where $\beta = 2$ and Bayesian Information Criterion (Schwarz, 1978, BIC), where $\beta = \log n$. The theoretical properties of BIC for independently and identically distributed Gaussian data are examined in Yao (1988), while Ninomiya (2015) explores the application of AIC across different data types. However, AIC tends to favor more complex models and may lead to overfitting, BIC is more conservative and can result in underfitting.

To address the issue introduced by the choice of penalty, Haynes et al. (2017) introduced Changepoints for a Range of Penalties (CROPS) that efficiently identifies changepoints across a continuum of penalty values. Instead of relying on a single, predefined penalty, CROPS explores a range of penalties to construct a penalty-segmentation profile, illustrating how the number of detected changepoints varies with different penalty values. By avoiding redundant computations and leveraging previously identified segmentations, CROPS provides a more comprehensive view of potential changepoint

structures, helping to mitigate the risks of both overfitting and underfitting.

However, both Segmentation Neighbourhood Search and Optimal Partitioning suffer from expensive computing costs, making them impractical for analysing large datasets. To improve the efficiency, four major works have been proposed: the pruned Dynamic Programming Algorithm (Rigaill, 2015, pDPA), Segment Neighbourhood with Inequality Pruning (Maidstone et al., 2017, SNIP), Pruned Exact Linear Time Algorithm (Killick et al., 2012, PELT), and Functional Pruning Optimal Partitioning (Maidstone et al., 2017, FPOP). As Figure 2.3 shows, these four methods are based on either functional pruning or inequality rule to reduce the computational burden. In the following, we will briefly introduce the key ideas behind these two pruning techniques and refer readers to the respective papers for further details.

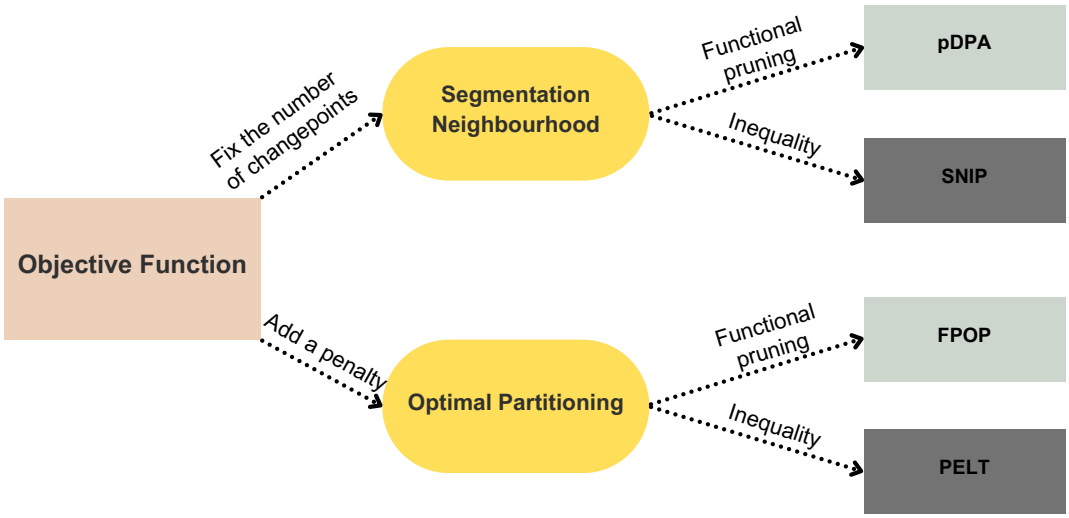


Figure 2.3: The relationships among offline algorithms.

An efficient pruning mechanism allows us to discard candidate changepoints when certain conditions are met, say, if the cost function satisfies

$$C(x_{r:s}) + C(x_{s+1:t}) > C(x_{r:t})$$

for all $r < s < t$. Inequality-based pruning checks the condition

$$\mathcal{F}_s + C(x_{s+1:t}) + \beta > \mathcal{F}_t$$

at every iteration. Here the $\mathcal{F}_s$ represents the minimum cost of segmentation up to time s, $C(x_{s+1:t})$ is the cost of modelling data from $s+1$ to t as one segment. If this inequality condition holds, s can never be part of an optimal segmentation for any future time $n \geq T > t$, as partitioning with the most recent change at s would always incur a higher cost than partitioning with the most recent change at t. Algorithm 3 presents the PELT algorithm, which applies the inequality rule after each iteration. By systematically discarding suboptimal changepoints, PELT improves the efficiency of the Optimal Partitioning approach, ensuring that its computational complexity remains at most $O(n^2)$. In practice, PELT often achieves an expected runtime that scales linearly, i.e., $O(n)$, making it highly efficient for large datasets.

Algorithm 3: PELT

Initialize: changepoint $cp \leftarrow \{\}$**, candidates set** $R = \{0\}$**,** $\mathcal{F}_0$**=0.**
for $t = 1, \cdots, n$ **do**
  Calculate $\mathcal{F}_t = \min_{\tau \in R} \{\mathcal{F}_\tau + C(x_{\tau+1} : t) + \beta\}$,
  $\hat{\tau} = \{\tau \in R : \arg\min \mathcal{F}_t\}$,
  $cp \leftarrow cp \cup \{\hat{\tau}\}$,
  Prune out the candidates that satisfy the inequality condition.
  $R \leftarrow t \cup \{\tau \in R : \mathcal{F}_\tau + C(x_{\tau+1:t}) + \beta \leq \mathcal{F}_t\}$.
end
return *cp*.

Unlike inequality-based pruning, functional pruning considers the cost $C(\cdot)$ as a function of its parameter of the last segment. For example, in the Gaussian mean change case, the cost function can be rewritten in terms of the mean μ:

$$C(x_{s+1:t}) = \min_{\mu} \sum_{i=s+1}^{t} q(x_i|\mu)$$

where $q(\cdot)$ is the negative log-likelihood function, which is a quadratic in μ. Consequently, minimizing the segmentation cost over potential changepoints at each time step is equivalent to minimizing this quadratic function with respect to μ. For example, recall the recursion in Segmentation Neighbourhood Search, we have

$$\begin{aligned}\mathcal{C}_{m,t} &= \min_{\tau} \left[\mathcal{C}_{m-1,\tau} + C(x_{\tau+1:t})\right] \\ &= \min_{\tau} \left[\mathcal{C}_{m-1,\tau} + \min_{\mu} \sum_{i=\tau+1}^{t} q(x_i|\mu)\right] \\ &= \min_{\tau} \min_{\mu} \left[\mathcal{C}_{m-1,\tau} + \sum_{i=\tau+1}^{t} q(x_i|\mu)\right].\end{aligned}$$

Denote $\mathcal{Q}^{\tau}_{m,t}(\mu)$ as the segmentation cost when partitioning data from τ to t, and $\mathcal{Q}^{*}_{m,t}(\mu)$ as the optimal segmentation cost up to time t with parameter μ. These are defined as

$$\mathcal{Q}^{\tau}_{m,t}(\mu) = \mathcal{C}_{m-1,\tau} + \sum_{i=\tau+1}^{t} q(x_i|\mu), \quad \mathcal{Q}^{*}_{m,t}(\mu) = \min_{\tau} \mathcal{Q}^{\tau}_{m,t}(\mu).$$

So that

$$\mathcal{C}_{m,t} = \min_{\mu} \min_{\tau} \mathcal{Q}^{\tau}_{m,t}(\mu) = \min_{\mu} \mathcal{Q}^{*}_{m,t}(\mu).$$

$\mathcal{Q}^{*}_{m,t}(\mu)$ can be calculated recursively depending on whether a changepoint occurs before t:

$$\mathcal{Q}^{*}_{m,t}(\mu) = \min \left\{ \underbrace{\min_{\tau} \mathcal{Q}^{\tau}_{m,t-1}(\mu) + q(x_t|\mu)}_{\text{change happens before time } t}, \quad \underbrace{\mathcal{C}_{m-1,t}}_{\text{change happens at time } t} \right\}$$

Thus, at time t, we can track m quadratics corresponding to different segmentations. At this step, the time complexity remains quadratic. The idea behind pDPA is that if for all μ,

$$\mathcal{Q}^{\tau}_{m,t}(\mu) > \mathcal{Q}^{*}_{m,t}(\mu),$$

the candidate τ will not contribute to the optimal partition and thus we can prune it out. Figure 2.4 illustrates the denotation and the process of functional pruning, and Algorithm 4 is the pseudocode for the pDPA.

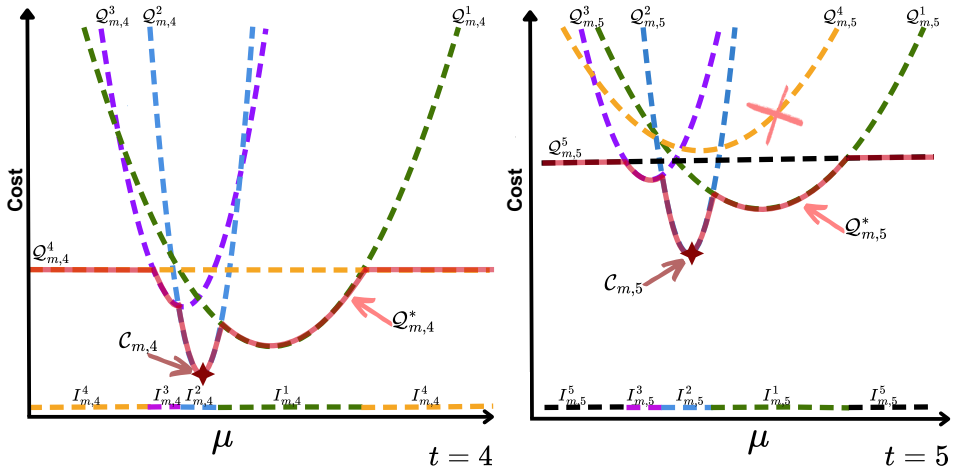


Figure 2.4: The diagram illustrates the process of functional pruning from time $t = 4$ to 5. The dashed lines are the cost $\mathcal{Q}^\tau_{m,t}$, the red curve is the $\mathcal{Q}^*_{m,t}$, the red star is the $\mathcal{C}_{m,t}$, the coloured line at the bottom is the mean interval $I^\tau_{m,t}$ that contributes to the optimal partition as described in Algorithm 4, and the red cross represents the candidate we want to prune out. We can see from time 4 to 5, the cost of $\mathcal{Q}^4_{m,5}$ is larger than any other costs for any μ, therefore we can prune out candidate 4 from the searching space.

Although functional pruning reduces computational cost, the worst-case complexity remains bounded by $O(Mn^2)$. Empirical results (Rigaill, 2015) suggest that it can achieve an improved complexity of $O(Mn \log n)$. FPOP applies the same pruning idea to the Optimal Partitioning problem. In the best-case scenario, it can reach a complexity of $O(n \log n)$. Compared to PELT, FPOP is more efficient, pruning more or at least as many candidate changepoints. However, functional pruning requires minimizing a function, which is efficient when dealing with one-dimensional convex curves. For problems involving two or more dimensions, determining what to prune becomes infeasible.

Several extensions have been developed based on PELT and FPOP to address more complex scenarios. NP-PELT (Haynes et al., 2017) adapts the algorithm for non-parametric settings where the cost function is based on the empirical cumulative distri-

Algorithm 4: pDPA

Initialize: changepoint $cp_m \leftarrow \{\}$**, candidates set** $R_m = \{\}$**,** $\mathcal{Q}_0$**=0, the maximum number of changepoints** M**, and the mean invertal** $I^{\tau}_{m,t}$

for $m = 1, \cdots, M$ **do**
  Calculate $\mathcal{Q}^{m}_{m,1:(m+1)}(\mu) = \mathcal{C}_{m-1,1:m} + \sum_{i=\tau+1}^{t} q(x_i|\mu)$
  $R_m \leftarrow \{m\}$
  for $t = m+1, \cdots, n$ **do**
    Calculate $I^{\tau}_{m,t+1} = \{\mu : \mathcal{Q}^{\tau}_{m,t} \leq \mathcal{C}_{m-1,\tau}\}$ for $\tau \in R_m$
    Calculate $I^{t}_{m,t+1} = \cap_{\tau<t}\{\mu : \mathcal{Q}^{\tau}_{m,t} > \mathcal{C}_{m-1,\tau}\}$
    Prune out the candidate which is not contributing to the optimal partition
    The changepoint candidate sets $R_m \leftarrow \{\tau \in (R_m \cup t) : I^{\tau}_{m,t} \neq \emptyset\}$
    Update $\mathcal{Q}^{\tau}_{m,t+1}(\mu) = \mathcal{Q}^{\tau}_{m,t}(\mu) + q(x_i|\mu)$, for $\tau \in R_m$
    The recent changepoint location
     $cp_m \leftarrow cp_m \cup \{\tau \in R_m : \arg\min \mathcal{Q}^{*}_{m,t}(\mu)\}$
  end
end
return $\{cp_m\}$.

bution. Extensions have also been made to detect changes in the slope of linear models (Fearnhead et al., 2019; Runge et al., 2020), to incorporate autoregressive noise (Chakar et al., 2017; Romano et al., 2022), to enhance robustness to outliers (Fearnhead and Rigaill, 2019), and to improve the computational speed via parallelisation (Tickle et al., 2020).

A more recent and particularly interesting development might be the application in anomaly detection, with a notable approach called CAPA (Fisch et al., 2022b). The key difference between anomaly detection and changepoint detection lies in the definition of the baseline. In changepoint detection, the baseline is dynamic—it is always the previous segment, with the focus on identifying shifts between consecutive data segments. In contrast, anomaly detection assumes a constant baseline, typically representing a normal or expected period of behaviour. The goal is to identify deviations from this fixed baseline, whether as isolated point anomalies or as collective anomalies spanning multiple data points. CAPA extends the concept of optimal partitioning

while introducing a fixed baseline. Assuming there are m anomalous segments, each characterized by its own mean μ_m, and there is a known baseline segment with mean μ_0. The objective function is to minimize the following costs:

$$\underbrace{\sum_{t \notin \cup [s_i:e_i]} C(x_t|\mu_0)}_{\text{Cost of modelling a normal segment}} + \underbrace{\sum_{i=1}^{m} \left[\min_{\mu_i} C(x_{s_i:e_i}|\mu_i) + \beta \right]}_{\text{Cost of modelling an anomalous segment.}},$$

where s_i and e_i are the start and the end of the segment with $s_{i+1} = e_i + 1$. This objective function can be efficiently solved using dynamic programming, with computational speed significantly improved by applying the PELT rule.

2.1.3 Bayesian offline changepoint detection

In addition to the frequentist approach introduced earlier, Bayesian methods are attractive because they incorporate expert knowledge via prior distributions and quantify the uncertainty in both the number and locations of changepoints. The first work of Bayesian changepoint analysis can be traced to Chernoff's work (Chernoff and Zacks, 1964), where the posterior mean is estimated for a single change. Later studies expanded this framework, applying Bayesian methods to different contexts, i.e., linear models, time series models, and the Poisson process (Chin Choy and Broemeling, 1980; Salazar, 1982; Akman and Raftery, 1986; Carlin et al., 1992; Broemeling, 2017).

In this section, we consider the general framework of the Bayesian changepoint algorithm, where both the number and locations of changepoints are random rather than fixed. To model this, we specify priors for the parameters including the number of changepoints m, denoted as $f(m)$. m changepoints will separate the sequence into $m + 1$ segments, where we set $\tau_0 = 1$ and $\tau_{m+1} = n$. Each segment is associated with segment-specific parameters θ_i for $i = 1, \cdots, m + 1$, where θ_i are the segment-specific parameters. Assume the observations x are conditionally independent given

the parameters θ, the joint posterior distribution of interest is given by

$$f(m, \tau_{1:m}, \theta_{1:m+1}|x_{1:n}) \propto f(x_{1:n}|\theta_{1:m+1}, m, \tau_{1:m}) f(\theta_{1:m+1}|\tau_{1:m}, m) f(\tau_{1:m}|m) f(m).$$

In the following sections, we review two approaches: Markov Chain Monte Carlo and the direct simulation.

Markov Chain Monte Carlo

When m is known, the distribution of the number and the location of changepoints can be estimated via Gibbs sampling, as demonstrated in Carlin et al. (1992), Stephens (1994), and Chib (1996). However, when m is unknown, the problem becomes more complex. Because the distribution contains multiple subspaces with different numbers of changepoints. One solution is the Reversible Jump Markov Chain Monte Carlo (RJMCMC) algorithm, introduced by Green (1995). In this algorithm, each model incorporates different numbers of changepoints, and the RJMCMC can jump between models with different numbers of changepoints. Specifically, at each iteration we have a state $(m_i, \theta^i_{1:m_i+1})$, the general procedure of jumping to a random new state $(m_j, \theta^j_{1:m_j+1})$ is given by:

- Propose a new move to m_j with probability $q_m(m_i \to m_j)$,
- Generate a random vector u from the proposal distribution $q(u)$,
- A bijective function $g : (\theta^i_{1:m_i+1}, u) \to (\theta^j_{1:m_j+1}, u')$ is introduced to map the dimension.
- We will accept this move with an acceptance probability

$$\min\left\{1, \frac{f(m_j, \theta^j_{1:m_j+1}) q_m(m_j \to m_i) q(u')}{f(m_i, \theta^i_{1:m_i+1}) q_m(m_i \to m_j) q(u)} \left|\frac{\partial g(\theta^i_{1:m_i+1}, u)}{\partial(\theta^i_{1:m_i+1}, u)}\right|\right\},$$

  otherwise stay at the current state.

In this procedure, if $m_j > m_i$, changepoints are introduced; otherwise, if $m_j < m_i$, changepoints are eliminated. This process is referred to as the birth-death procedure in the Green (1995). The empirical marginal probability of having i changepoints then can be estimated by

$$P(m = i) = \frac{1}{r} \sum_{j=1}^{r} I(m_j = i)$$

where r is the total number of samples from the posterior distribution,m_j is the number of changepoints in the j-th sample. The marginal probability of the location of a changepoint can be estimated conditional on the number of changepoints. However, RJMCMC faces challenges, particularly in designing proposal distributions that mix well across models. Chib (1998) proposed an alternative approach using approximate Bayes factors derived from MCMC output to compare models. However, these methods can be computationally expensive. More recently, Benson and Friel (2018) developed an adaptive RJMCMC algorithm that learns from previous states, improving efficiency for large datasets while maintaining ergodicity. Nevertheless, all MCMC-based approaches suffer from issues related to computational efficiency and convergence.

Direct Simulation

Different from the MCMC-based approaches, direct simulation approaches directly simulate the true posterior without approximating it. The ideas for direct simulation are based on exact methods for calculating posterior means which originated in Barry and Hartigan (1993) and extended in Fearnhead (2006). The assumption of these approaches is to model the data as contiguous blocks, where being in the same block, conditional on the most recent changepoint, indicates no change has occurred. Therefore, the probability that the data points $x_{t:s}$ belong to the same segment is given by:

$$p(t, s) = \int \prod_{i=t}^{s} f(x_i \mid \theta)\, \pi(\theta)\, d\theta,$$

where $\pi(\theta)$ is the prior distribution for the segment parameter of interest. If we use conjugate priors, then this integral can be calculated analytically. We denote by $g(\cdot)$ and $g_0(\cdot)$ the probability mass functions of the segment length after a changepoint and the length until the first changepoint, respectively. Common choices for these priors include the geometric and negative binomial distributions. Similarly, $G(\cdot)$ and $G_0(\cdot)$ represent the corresponding cumulative distribution functions of the segment lengths. Fearnhead (2006) proposes a recursive method for computing the probability of observed data following a changepoint, given its location. Denote $Q(t)$ as the probability of observed data from t to n, given the changepoint occurs at $t-1$, where $t = 2, \cdots, n$. The $Q(t)$ can be expressed as:

$$\begin{aligned} Q(t) =& p(x_{t:n}) \\ =& \sum_{s=t}^{n-1} p(x_{t:n}|\text{next change at } s)p(\text{next change at } s) \\ &+ p(x_{t:n}|\text{next change at } n)p(\text{next change at } n) \end{aligned} \tag{2.2}$$

The first term represents the probability of introducing a changepoint within the range $[t, n-1]$ after a change at $t-1$, it can be calculated as:

$$\begin{aligned} &\sum_{s=t}^{n-1} p(x_{t:n}|\text{next change at } s)p(\text{next change at } s) \\ =& \sum_{s=t}^{n-1} p(\text{next change at } s)p(x_{t:s} \text{ in the same segment}) \\ &p(x_{s+1:n}|\text{ changepoint at } s) \\ =& \sum_{s=t}^{n-1} g(s+1-t)p(t,s)Q(s+1). \end{aligned}$$

The second term in Equation 2.2 accounts for the probability of having no further changepoints after $t-1$, which is given by:

$$p(x_{t:n}|\text{next change at } n)p(\text{next change at } n) = p(t,n)(1-G(n-t)).$$

Thus, $Q(t)$ can be calculated recursively, with the initial condition:

$$Q(1) = \sum_{s=1}^{n-1} g_0(s)p(1,s)Q(s+1) + p(1,n)(1-G(n-1)).$$

With a computed list of $Q(t), \forall t = 1, \cdots, n$, one can infer the *maximum-a-posteriori* estimates of the changepoint locations. This recursive method directly provides the probability distribution of changepoint locations while allowing the number of changepoints to vary. Additionally, Fearnhead (2006) presents a version of this method that conditions a fixed number of changepoints.

Further work has extended this approach to various settings, including its adaptation to linear regression models with dependence across segments (Fearnhead and Liu, 2011), its extension to online detection frameworks (Fearnhead and Liu, 2007; Adams and MacKay, 2007), and its extension to the multivariate case with dependent structures (Xuan and Murphy, 2007). This approach has also been applied in diverse fields such as financial markets, epidemiology, climate studies and finance (Thies and Molnár, 2018; Verma et al., 2020; Thomson et al., 2010; Thies and Molnár, 2018; Held et al., 2006). A recent extension (Bardwell and Fearnhead, 2017) introduced a framework for anomaly detection by additionally introducing two states: normal segment and abnormal segment.

However, Bayesian changepoint detection approaches can be computationally demanding. Although the Bayesian approach has the same order of computational cost as penalized approaches, its constant factor is typically larger. Furthermore, these methods often impose stronger model assumptions, which can limit their flexibility.

However, as we will see later, research in offline changepoint detection has laid a crucial foundation for the development of online changepoint detection.

2.2 Online changepoint detection

As sampling the data becomes faster and cheaper, the need for quick decisions grows. Nowadays, most tasks not only require identifying the changepoints, but also require real-time change-point detection as each new observation arrives. Specifically, the observations $X_1, X_2, ...$ arrive sequentially over time. We will monitor the data stream and stop the algorithm immediately when the change is detected. Therefore, we only consider at most one change that happened at an unknown time τ.

Effective algorithms should have the ability to quickly identify the change and also be cheap to train and run. Online changepoint detection addresses these challenges by applying sequential statistical tests, or Bayesian inference to determine whether a new point is a changepoint. Furthermore, unlike most deep learning approaches, changepoint algorithms are lightweight. In this section, we will explore the key methodologies and advancements in this popular and growing area.

2.2.1 Frequentist approaches

We begin by specifying the model. Assume the observations follow

$$X_i \overset{\text{iid}}{\sim} f(X|\theta), \text{for } i = 1, \cdots, \tau,$$

before the changepoint, and follow

$$X_i \overset{\text{iid}}{\sim} f(X|\theta + \delta), \text{for } i = \tau + 1, \cdots, n,$$

after the change occurs at an unknown time τ. Here θ represents the pre-change parameter, δ denotes the magnitude of the change in the parameter after the changepoint, and n is the pre-specified monitoring period. In the following subsections, without being explicitly stated, we assume the pre-change data follows a standard Gaussian distribution, with the change occurring in the mean.

Detection criteria

The rationale behind frequentist approaches is fundamentally the same: at each time step, a hypothesis test is performed, where the null hypothesis assumes no change and the alternative hypothesis assumes a change has occurred. The algorithm stops and alarms a changepoint when the test statistic reaches the threshold c. A crucial question, therefore, is how to determine this threshold? We answer this question before introducing online changepoint detection algorithms.

Average run length (ARL) is a commonly used criterion. It measures the expected number of observations between false alarms when no changepoints are present. It is defined as:

$$\text{ARL} = \mathbb{E}_\infty[\hat{\tau}],$$

where $\hat{\tau}$ denotes the first time a change is detected, and the expectation is under the null hypothesis of no change ($\tau = \infty$). A higher ARL indicates less sensitivity to false positives. In practice, ARL is user-specified and defines the threshold when all other model parameters are held constant. This relationship guides us in choosing the threshold based on a desired ARL value, which is often larger than the user-specified monitoring time.

Another criterion is to control the false alarm rate (FAR)

$$\text{FAR}_n = p_n(\hat{\tau} < \infty | \tau = \infty),$$

defined as the probability of triggering an alarm when no change has occurred within the monitoring period n. A lower FAR improves precision but it also means the algorithm may take longer to signal a true change. Similarly, FAR is typically user-specified, reflecting the level of risk the user is willing to tolerate. In many cases, n is taken to be ∞, so the monitoring period is unbounded. Under this setting, a fixed detection threshold would result in $\hat{\tau} < \infty$ with probability 1, leading to a false alarm. To control FAR properly, the detection threshold must increase with time t in the long run. Various methods exist for selecting time-varying thresholds, or equivalently, weighting the test statistics with a function $w(\cdot) < 1$ to control the false alarm rate (Leisch et al., 2000; Zeileis et al., 2005; Horváth et al., 2008; Aue et al., 2012; Kirch and Kamgaing, 2015; Weber, 2017; Kengne and Ngongo, 2022).

Figure 2.5 provides a graphical illustration of the two criteria. Neither criterion is inherently superior; the choice depends on the specific requirements of the task.

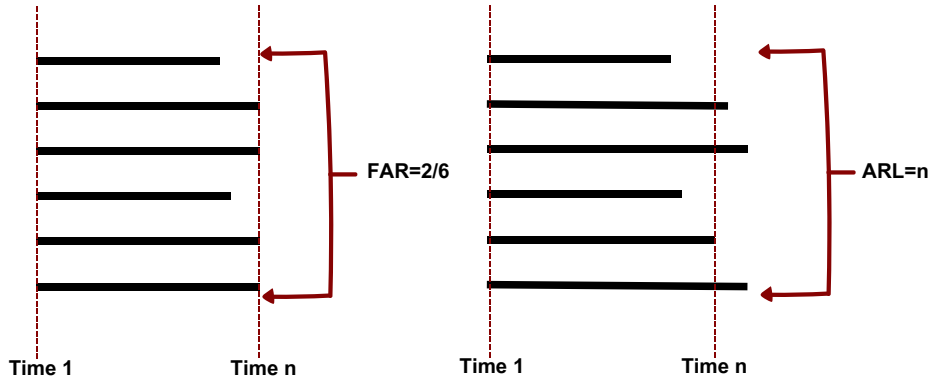


Figure 2.5: Graphical explanation of the false alarm rate and average run length under the null. The black lines represent the monitoring time for a fixed threshold until the first alarm in each replicate, where the first alarm occurs when the local test statistic exceeds the threshold. In the left Figure, for a monitoring period of n, the threshold results in a false alarm rate of 2/6. On the right hand side, the average of these five stopping times gives the average run length of n.

After answering the question of determining the threshold, we now introduce different test statistics.

Sequential probability ratio test

One of the most significant contributions to sequential testing, the Sequential Probability Ratio Test (SPRT), was introduced by Wald (1945). In the Wald test, the likelihood ratio

$$W_t = \frac{f(x_{1:t}|\theta+\delta)}{f(x_{1:t}|\theta)} = W_{t-1}\frac{f(x_t|\theta+\delta)}{f(x_t|\theta)}$$

is the test statistic and can be updated sequentially. The decision is to:

- stop and reject H_0 when $W_t \geq B$,
- stop and not reject H_0 when $W_t \leq A$,
- continue to observe the new data when $A < W_t < B$,

where

$$B \approx \frac{1-\beta}{\alpha}, \quad A \approx \frac{\beta}{1-\alpha},$$

and α and β are the desired Type I and Type II errors. The SPRT is easy to apply and is optimal in terms of minimizing the expected sample size under both hypotheses Wald (1945). However it needs both the pre-change parameter and the size of change to be known.

Online CUSUM

Another important development in sequential analysis is the Cumulative Sum (CUSUM) procedure. The original online CUSUM procedure, introduced by Page (1954), is designed to detect the positive changes in parameters of interest within quality control contexts, such as the mean, range, or number of defectives. Define the cumulative score as $S_t = \sum_{i=1}^{t} X_i$, the change is declared as

$$S_t - \min_{0\leq i<t} S_i \geq c_1.$$

The above formula indicates that if S_t increases significantly compared to its past minimum, it suggests a positive shift in the mean of the process. To avoid storing a list of $S_i, \forall i \leq t$, this formula can be rewritten as a recursive defined process:

$$Z_t = S_t - \min_{0 \leq i < t} S_i = \max\{Z_{t-1} + X_t, 0\},$$

with $Z_0 = 0$. The algorithm alarms when $Z_t \geq c_1$.

It is worth noting that this CUSUM only detects the increase in the mean. To detect changes in both directions (positive and negative), the doubled CUSUM procedure is introduced (Waldmann, 1996). In the doubled CUSUM procedure, two test statistics

$$Z_t^+ = \max\{Z_{t-1}^+ + X_t, 0\}, \quad Z_n^- = \max\{Z_{t-1}^- - t_n, 0\}, \quad Z_0^+ = Z_0^- = 0$$

are used to detect positive changes and negative changes respectively, and the stopping rule is defined as raising an alarm if

$$Z_t^+ \vee Z_t^- \geq c_2.$$

In the original paper, Page uses the cumulative sum as the score function in Page (1954), but he also noted that the score function can also be other statistics, i.e., the likelihood ratio test statistics if the parameter of pre-change and post-change is known. If we use the log-likelihood ratio test statistic as the score function, the one side test statistic will be

$$Z_t = \max\left\{Z_{t-1} + \log\frac{f(x|\theta+\delta)}{f(x|\theta)}, 0\right\}, \tag{2.3}$$

with $Z_0 = 0$, so the previous observations can be discarded. The algorithm alarms the change when $Z_t > c_3$. We can see that the online CUSUM procedure is similar to SPRT tests as Z_t is actually the logarithm of the Wald likelihood ratio bounded by 0 and c_3.

Built upon the ideas of Page and Wald, the cumulative sum of log-likelihood ratios has become a cornerstone in sequential testing. This approach is computationally efficient, with a cost of $\mathcal{O}(1)$ per observation, and it has strong optimality properties as discussed in Lorden (1971); Moustakides (1986); Ritov (1990); Lai (1998). Recent work has extended the CUSUM procedure into various areas, including regression models (Chu et al., 1996; Horváth et al., 2004; Hušková and Kirch, 2012; Horváth et al., 2022), time series (Gombay and Serban, 2009; Gösmann et al., 2021; Kurozumi, 2020, 2021, 2023), multivariate or high dimensional case (Mei, 2010; Weber, 2017; Chen et al., 2022, 2024) and so on. We refer readers to the review by Aue and Kirch (2024) and the book by Basseville and Nikiforov (1993) for further details on CUSUM statistics and its recent advances. In this thesis, we will introduce some of the above approaches, such as the works of Mei (2010) in Chapters 3 and 4.

However, CUSUM, perhaps the most popular online changepoint algorithm, faces the issue of requiring the knowledge of pre-change and post-change distributions. A poor specification can lead to reduced power or failure in detection. Several methods have been proposed to relax the assumption of known pre-change and post-change parameters while aiming to speed up the algorithm, such as Bell et al. (1994); Pollak and Siegmund (1991); Gordon and Pollak (1997); Krieger et al. (2003); Gordon and Pollak (1995); however, many of these methods are computationally intensive.

A possible efficient solution, the recursive register approach, was introduced by Lorden and Pollak (2008). This approach approximates the post-change parameter under the assumption that the pre-change parameter is known. The idea is to estimate the post-change mean after the candidate changepoint. Denote v as the candidate changepoint location, $T_v = t - v$ as the total number between candidates v and $t-1$, and $S_v = \sum_{i=v}^{t} X_i$ as the sum of observations in this interval, the algorithm is described as in Algorithm 5. This algorithm only considers positive changes. A later work Liu et al. (2019) accounts for changes in both directions (positive and negative) which shares

the same idea of doubled CUSUM. Although the recursive register algorithm can detect

Algorithm 5: Recursive Register Algorithm (Lorden and Pollak, 2008; Liu et al., 2019)

Initialize: Set registers $Z_0 = S_0 = T_0 = X_0 = 0$, prior knowledge of s and t, and pre-specified smallest post-change mean ρ.
for $t = 1, 2, \ldots, n$ **do**
  Update the registers:
  if $Z_t = 0$ **then**
    $S_t = T_t = 0$.
  end
  else
    $S_t = S_{t-1} + X_t$ and $T_t = T_{t-1} + 1$.
  end
  Estimate post-change mean: $\hat{\delta} = \max\left(\rho, \frac{s+S_t}{t+T_t}\right)$,
  Calculate Z_t
  Substitute the estimated post-change mean $\hat{\delta}$ into Formula 2.3.
  Check the stopping rule:
  if Z_t *exceeds the threshold or* $t = n$ **then**
    Terminate the procedure.
  end
end

the changes with unknown post-change parameters efficiently, it requires accurate prior information about the change. This makes it less efficient when the prior guesses are poorly specified.

Generalized likelihood ratio procedure and FOCuS

Another efficient solution, FOCuS, is based on the Generalized Likelihood Ratio test procedure (GLR). At time t, assuming the pre-change parameter θ is known, the GLR test statistic is given by:

$$GLR_t = \max_{0\le\tau<t} \sup_{\delta\in\Delta} \log \frac{f(x_{1:\tau}|\theta) f(x_{\tau+1:t}|\theta+\delta)}{f(x_{1:t}|\theta)} = \max_{0\le\tau<t} \sup_{\delta\in\Delta} \sum_{i=\tau+1}^{t} \log \frac{f(x_i|\theta+\delta)}{f(x_i|\theta)}.$$

For example, in standard Gaussian cases, the summation of the log-likelihood ratio test statistic for having the changepoint at τ is $Q_t^\tau = \delta(\sum_{i=\tau+1}^{t} x_i - (t-\tau)\frac{\delta}{2})$ at time t, and the GLR test statistic is $\max_{0\le\tau<t} \sup_{\delta\in\Delta} Q_t^\tau$.

GLR procedure searches the post-change parameter through the region Δ, and can be extended to pre-change unknown variance. Its asymptotic optimality is proved in Lorden (1971). While the GLR statistic can be calculated in $O(1)$ cost per iteration if θ and δ are both known via Page's recursion, this no longer works when one or both are unknown. Since this requires solving a separate optimization problem at every iteration, the computational complexity of nth iteration is $O(n)$.

To improve the efficiency, functional pruning is used in the FOCuS algorithm (Romano et al., 2023b). We introduce the scenario where the pre-change distribution is known, but the post-change parameter is unknown, with a particular focus on cases where the mean undergoes a positive shift. The same approach can be applied to detect negative shifts. In the standard Gaussian case, at time t, the cost of introducing a change after the time τ (where $\tau = 0$ indicating no change), can be written as a function of unknown parameter δ:

$$\mathcal{Q}_t^\tau(\delta) = \max\left\{Q_{t-1}^\tau(\delta) + \delta(x_t - \frac{\delta}{2}), 0\right\}.$$

As a result, at any given time t, there exist t quadratic functions. Define $\mathcal{Q}_t^*$ as the supremum of these t quadratic and the zero line, the test statistic at each time t is given by $\max_\delta \mathcal{Q}_t^*$.

So far this procedure costs $O(t)$ for tth iteration. To improve efficiency, similar to the FPOP algorithm, the core idea in Romano et al. (2023b) is to prune quadratics that do not contribute to Q_t^* for any δ. Figure 2.6 illustrates the pruning idea. Define the optimal region I_t^i as the set of intervals over the possible values of δ at time t where the ith quadratic function Q_t^i is greater than all other quadratic functions. We can then find the optimal intervals I_t^0, where the zero line dominates. I_t^0 is given by:

$$[2\max_\tau \frac{S_t - S_\tau}{t - \tau}, \infty)$$

where the infimum is the the value that one quadratic can maximize its intersection with 0, and $S_t = \sum_{i=1}^{t} x_t$ is the cumulative sum. This follows from the fact that the cost function at time t for a change at τ is given by:

$$\delta(\sum_{i=\tau}^{t} x_i - (t-\tau)\frac{\delta}{2}) = \delta(S_t - S_\tau - (t-\tau)\frac{\delta}{2}),$$

which has roots at 0 and $2\max_\tau \frac{S_t - S_\tau}{t-\tau}$. Then we can prune out the quadratics that do not contribute to the Q_t^*. Specifically any candidates that its optimal interval I_t^i is either entirely enclosed by the other quadratics ($I_t^i = \infty$) or partially enclosed with the remaining portion below 0, is pruned out ($\min I_t^i > \min I_t^0$). By doing so, FOCuS is exact and the average cost is $\mathcal{O}(\log t)$. The whole procedure is outlined in Algorithm 6.

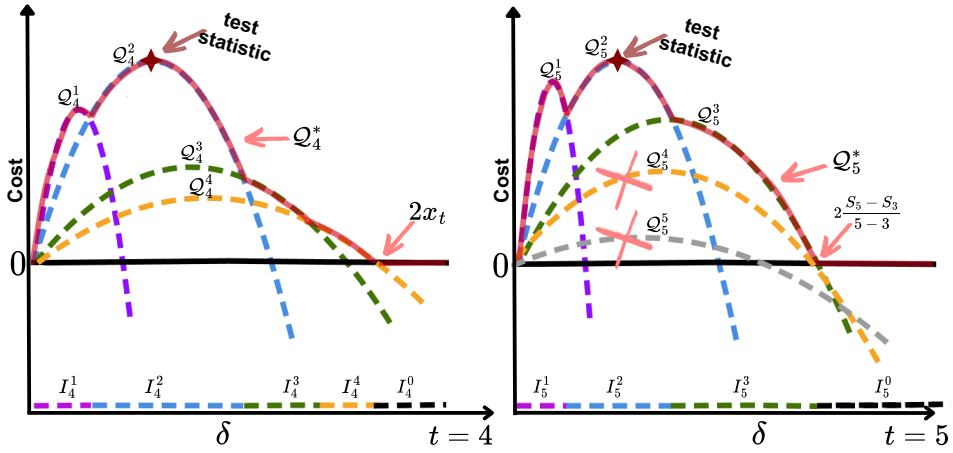


Figure 2.6: The diagram illustrates the process of FOCuS from time $t = 4$ to $t = 5$. The dashed lines are the zero line, the red curve is the $\mathcal{Q}_t^*$, the red star is test statistic $\max_\delta \mathcal{Q}_t^*$, the coloured line at the bottom is the set I_t^τ which contributes to the optimal partition, $2x_t$ and $2\frac{S_t - S_3}{5-3}$ are the starting points of the partition where zero line is the optimal, and the red cross represents the candidate we want to prune out. We can see from time 4 to 5, the cost of $\mathcal{Q}_t^4$ and $\mathcal{Q}_t^5$ is smaller than other costs or zero line for any δ, therefore we can prune out candidate 4 and 5 from the searching space.

In Romano et al. (2023b), FOCuS has also been extended to the unknown pre-change mean case; for details, we refer readers to Romano et al. (2023b). Ward et al. (2024) further reduced the computational cost to constant by recycling the calculations

Algorithm 6: FOCuS

Initialize: candidates set $R \leftarrow \{\}$, $\mathcal{Q}_0^0 = 0$, **threshold** c, **monitoring time** n, **borderline** B, **the cumulative sum** S **and the optimal interval** I

for $t = 1, \cdots, n$ **do**
  Update the candidate set $R \leftarrow \{t-1\}$
  Calculate $\mathcal{Q}_t^\tau(\delta) = \mathcal{Q}_{t-1}^\tau(\delta) + \delta\left(x_t - \frac{\delta}{2}\right)$ for $\tau \in R$
  Calculate $B_t(\delta) = \sup_{\tau \in R} \mathcal{Q}_t^\tau$
  for $\tau \in R$ **do**
    Calculate the optimal region for each candidate $I_t^\tau = \{\delta : B_t(\delta) = \mathcal{Q}_t^\tau\}$ for $\tau \in R$
    Calculate the optimal region for zero line $I_t^0 = \{\delta : \delta \geq 2\max_\tau \frac{S_t - S_\tau}{t-\tau}\}$ for $\tau \in R$
    Prune out the candidate which does not contribute to the optimal partition
    The changepoint candidate sets $R \leftarrow R \setminus \{\tau \in R : I_t^\tau = \emptyset\} \setminus \{\tau \in R : \min I_t^\tau > \min I_t^0\}$
  end
  if $\max_\delta B_t(\delta) > c$ **then**
    Alarm
  end
end

from previous steps. The subsequent developments of FOCuS focus on its applications to different distributions, such as Possion distribution (Ward et al., 2023), exponential family model (Ward et al., 2024), and nonparametric density (Romano et al., 2023a).

MOSUM

An alternative approach to CUSUM is the Moving Sum (MOSUM) procedure, which employs a sliding window to scan the data and test for changes. Since specific details and formulas are shown in Chapter 3, we provide a brief overview now. The basic idea is to estimate the pre-change mean $\hat{\theta}$ from training data where no change is assumed to occur. Then, the MOSUM procedure calculates the mean of the absolute differences between the average within the sliding window h and the estimated mean,

$$M_t = \left| \frac{1}{h} \sum_{i=t}^{t+h-1} X_i - \hat{\theta} \right|,$$

as the evidence of a change. A large M_t indicates a potential changepoint. Compared with CUSUM, MOSUM does not need to know the post-change distribution. However, the performance of MOSUM highly depends on the choice of window size h. A large window may delay the detection of big changes, while a small window may struggle to identify subtle shifts. To reduce the bias caused by window size, one can simultaneously evaluate multiple window sizes.

Theoretical results on the asymptotic behaviour of MOSUM statistics under the null hypothesis can be found in Eichinger and Kirch (2018) and Aue et al. (2012). Recent developments on MOSUM include bootstrap methods for constructing confidence intervals for locations (Cho and Kirch, 2022), detecting changes under piecewise linearity (Kim et al., 2024), and under high dimensional factor model (Barigozzi et al., 2024).

2.2.2 Bayesian online changepoint detection

In Section 2.1.3, we introduced Bayesian offline changepoint detection approaches. These methods are computationally expensive because they have to infer the joint distribution of the number of changepoints and their locations. In contrast, the online setting focuses solely on estimating the location of the change, so the task becomes easier and computationally efficient. The online Bayesian approach is particularly appealing as it not only incorporates prior knowledge but also provides the posterior distribution to quantify the uncertainty of the detected change.

Fearnhead and Liu (2007) proposed an online Bayesian changepoint detection method. Under the assumption that the parameters before the change and after the change are conditionally independent, the exact posterior distribution can be calculated efficiently. Specifically, denote C_t as the index of the last changepoint, the probability of having

change at $j, j > C_t$ given observations $x_{1:t+1}$ can be calculated recursively as

$$p(C_{t+1} = j|x_{1:t+1}) \propto \underbrace{p(x_{t+1}|C_{t+1} = j, x_{1:t})}_{\text{predictive distribution}} p(C_{t+1} = j|x_{1:t}),$$

$$\text{where} \quad p(C_{t+1} = j|x_{1:t}) = \sum_{i=0}^{t-1} \underbrace{p(C_{t+1} = j|C_t = i)}_{\text{transition probability}} p(C_t = i|x_{1:t}).$$

$p(C_{t+1} = j|C_t = i)$ is the transition probability of having a changepoint at j since the last changepoint i, while $p(x_{t+1}|C_{t+1} = j, x_{1:t})$ is the predictive distribution:

$$p(x_{t+1}|C_{t+1} = j, x_{1:t}) = \int p(x_{t+1}|\theta)p(\theta|C_{t+1} = j, x_{1:t})d\theta,$$

$$\text{where} \quad p(\theta|C_{t+1} = j, y_{1:t}) \propto p(y_t|\theta)p(\theta|C_{t+1} = j, y_{1:t-1}).$$

If we use conjugate priors then this predictive distribution can be calculated analytically.

The $p(C_{t+1} = j|C_t = i)$ term reflects the prior knowledge about the potential locations of changepoints. A common choice for this prior is:

$$p(C_{t+1} = j|C_t = i) = \begin{cases} \frac{1-G(t-i)}{1-G(t-i-1)} & \text{if no change since } i, \text{ so } j = i, \\ \frac{G(t-i)-G(t-i-1)}{1-G(t-i-1)} & \text{if a new change at } t, \text{ so } j = t, \\ 0 & \text{otherwise.} \end{cases}$$

where $G(\cdot) = \sum g(\cdot)$ is the cumulative distribution function between two changepoint candidates, and $g(\cdot)$ represents the probability mass function. If $g(\cdot)$ is modelled as a geometric distribution, this assumes that the probability of a changepoint occurring at each time step is constant and independent of previous observations.

By running the algorithm, at time n, we will have n probabilities $P(C_n = i|x_{1:n})$ for $i = 0, 1, 2, .., n-1$. So we can simulate the joint posterior distribution of locations backwards, see Fearnhead (2006); Fearnhead and Liu (2007).

Another work from the same year (Adams and MacKay, 2007), named OBCPD,

builds on the same idea but models the run length r, the time elapsed since the last changepoint, instead of using a Markov process C_t. The connection between the two approaches is the run length r_t is equivalent to $t - C_t$.

Subsequent works have extended these approaches to various settings, including different fidelity (Gundersen et al., 2021), more robust to outliers (Altamirano et al., 2023), detecting the changes in distributions of rewards in bandit problem (Alami, 2023), Applications to Hawkes processes and neural networks (Detommaso et al., 2019). However, these methods often involve a higher-order quadratic cost, making them computationally expensive.

Not only these extensions, but both algorithms introduced in 2007, have a linearly increasing computational cost per iteration, which limits their scalability for long-term or real-time monitoring applications. To address this limitation, existing approaches have explored pruning out changepoint candidates with the least probability of change (Adams and MacKay, 2007) or using a stratified rejection control algorithm during the resampling step (Fearnhead and Liu, 2007). While these techniques can reduce computational complexity to a constant cost, they do so at the sacrifice of accuracy.

So far, we have introduced key works in the field of online changepoint detection, which has gained increasing attention in recent years. Table 2.1 summarises the algorithm we introduced. While frequentist approaches have seen significant advancements, Bayesian methods have received comparatively less attention. In the following chapters, we will apply frequentist approaches in a new setting - a distributed system, and propose strategies to reduce the time complexity of the online Bayesian algorithm.

Algorithm	Unknown pre-change	Unknown post-change	Computational cost
SPRT	N	N	$O(1)$
Page-CUSUM	N	N	$O(1)$
GLR	N/Y	Y	$\geq O(t)$
FOCuS	N/Y	Y	$\leq O(\log t)$
MOSUM	N	Y	$O(1)$
BOCPD	need a prior	need a prior	$\leq O(t)$

Table 2.1: Summary of online changepoint detection algorithms. The computational cost is measured at tth time step.

Chapter 3

A communication-efficient, online changepoint detection method for monitoring distributed sensor networks

3.1 Introduction

During the last decade, there has been a significant focus on the important challenge of efficient and accurate detection of changes in both univariate and multivariate data sequences (Cho and Fryzlewicz, 2014; Fisch et al., 2022b; Kovács et al., 2023; Truong et al., 2020; Tveten et al., 2022; Wang and Samworth, 2018). More recently, focus has turned to translating the efficiency of such approaches to the online setting, typically motivated by an applied challenge such as how to deal with limited computational power (e.g. Ward et al., 2023). Recent major contributions to the online setting include Adams and MacKay (2007), Tartakovsky et al. (2014), Yu et al. (2020), Chen et al. (2022), and Romano et al. (2023b). In this paper we consider a less studied scenario, monitoring

edge-behaviour within distributed sensor networks, which are common architectures within the Internet of Things framework (IoT). The importance of efficiently detecting changes at the edge efficiently, whilst minimising communication between sensors and the cloud is perhaps best appreciated by considering two key applications: detecting cyber-attacks on smart cities (Alrashdi et al., 2019) and optimising the performance of base stations (Wu et al., 2019).

Consider, by way of example, Figure 3.1 which shows a schematic representation of real-time monitoring within a distributed network. Here we assume that d data streams are monitored, each by its own sensor. Communication between the sensors and the centre is possible as shown by the dashed lines. An unusual event happens at time τ, and we want to detect this event as quickly as possible. However, in modern sensor networks that deploy IoT devices the computational resources of the sensors can be substantial. Moreover, communication between the sensors and the cloud can be problematic due to the heavy energy usage involved with transmitting data (Varghese et al., 2016; Pinto and Castor, 2017). As such, we need algorithms that can identify the time when it is important for information to be shared with the cloud. More specifically, in this article, we seek to develop a new method to detect changes within such a network in real time with high statistical power and as little communication and computation as possible.

Changepoint methods which can be applied in the fully centralised problem, when the data from the sensors is processed and transmitted to the centre (cloud) at every time step, are well studied. Approaches typically seek to calculate the maximum or the sum of all the test statistics (see, e.g., Mei, 2010; Xie and Siegmund, 2013a; Chan, 2017; Chen et al., 2022; Gösmann et al., 2022). The rationale behind these methods is to set thresholds and raise the alarm if the aggregated test statistics from multiple streams exceed pre-defined thresholds. Numerical experiments (Mei, 2010) indicate that taking the maximum is the optimal method when there are only a few affected data streams -

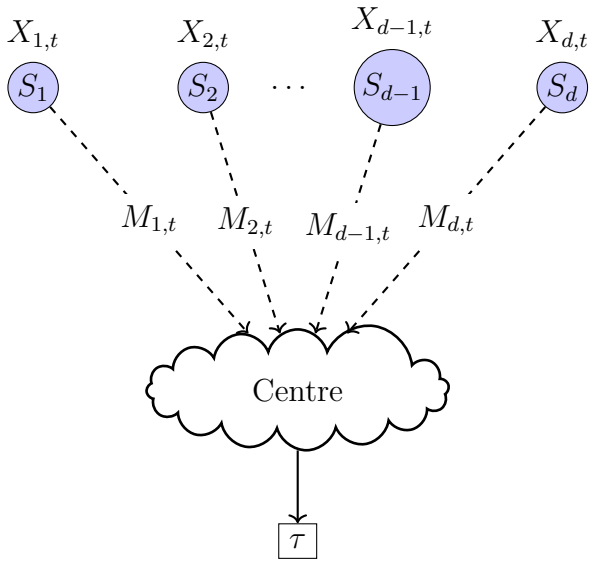


Figure 3.1: Schematic representation of a sensor network made up of d sensors, where S_i is the index for sensor i, $X_{i,t}$ is the data observed at sensor i, and $M_{i,t}$ is the message transmitted from sensor i to centre at time t.

what we will term a sparse change. Conversely, taking the sum is optimal when most data streams are affected, also known as a dense change.

Recent contributions to this distributed problem include (Rago et al., 1996; Veeravalli, 2001; Appadwedula et al., 2005; Mei, 2005, 2011; Tartakovsky and Kim, 2006; Banerjee and Veeravalli, 2015). Among them, two recent papers of particular interest develop *communication efficient* schemes for monitoring a large number of data streams (Zhang and Mei, 2018; Liu et al., 2019). The key idea is that each sensor computes a local monitoring statistic and then employs a thresholding step, only sending the statistic to the centre if there is some evidence of a change. The information from multiple sensors is then combined at the centre. This approach reduces unnecessary transmission by ignoring streams with little evidence for a change, while only focusing on data streams that show signs of change.

Although computationally feasible, existing works assume that the pre- and post-change mean are known. In practice, the pre-change mean can be estimated based on historical data. However assuming a known post-change mean is typically unrealistic

in practice, with an incorrect value potentially leading to a failure to detect, or poor detection power. Liu et al. (2019) approximate the post-change mean recursively but, as a consequence, somewhat sacrifice statistical power of the algorithm.

Our approach builds on recent work developing the moving sum (MOSUM) as a window-based changepoint methods (see, e.g., Aue et al., 2012; Kirch and Kamgaing, 2015; Kirch and Weber, 2018). Specifically, we propose an online communication-efficient changepoint detection algorithm (distributed MOSUM) to detect changes in real-time within the distributed network setting. A local threshold is chosen to filter out unimportant information and only transmit the statistically important test statistic to the centre. The change will be alarmed when the aggregated test statistic exceeds the pre-defined global threshold in the central cloud. The low time complexity and communication efficient scheme of our proposed method makes it suitable for online monitoring. We also establish that the proposed method can achieve similar statistical power as the idealistic setting, where there is no communication constraint, at detecting large changes whilst substantially reducing the transmission cost. Moreover, we also show how to make the detection performance of distributed MOSUM close to that of the idealised setting by increasing the window size, which will only sacrifice the storage cost and a little transmission cost.

The key differences between our work and previous distributed changepoint detection contributions (e.g., Liu et al., 2019) are: Firstly, a moving window-based test statistic MOSUM is chosen to avoid the requirement of knowledge of the post-change mean. Secondly, earlier works have been based on the framework that controls the average run length (ARL) - the average amount of time until incorrectly detect a change. However, such a metric gives a somewhat limited amount of information since the distribution of run length is usually unknown. For instance, if multiple procedures end quickly while a few replications stop significantly longer, the ARL would be the same if all the replications terminated around the same time. Conversely, in this work, we

present methods in terms of controlling the error rate under the null at a specific level, and with asymptotic power 1 under alternatives. Furthermore, our ideas generalise trivially to methods controlling the average run length.

The structure of this paper is as follows. In Section 3.2, the problem setting is outlined, before introducing the distributed MOSUM methodology in Section 3.3. Several theoretical results for this new approach are given in Section 3.4. Simulation studies are carried out in Section 3.5, before ending with some concluding remarks (Section 3.6).

3.2 Problem setting

We begin by assuming that we have d sensors, each of which is observed as follows: $\mathbf{X_t} = (X_{1,t}, X_{2,t}, X_{3,t}..., X_{d,t})$ at every time point $t \in \mathbb{N}$. Here $\mathbf{X_t}$ could be raw data or the residuals after pre-processing the data. These observations are assumed to be identically distributed and independent across series. Such assumptions are common in the problem of detecting changes within a distributed system setting (Mei, 2010; Xie and Siegmund, 2013a; Liu et al., 2019). We do not strictly assume time independence here, but our method is optimal when this assumption holds. Moreover, the impact of time dependence will be numerically studied in Section 3.5.3.

We begin by assuming that at some unknown time, τ, the distribution of some unknown subsets of d sensors will change. For simplicity, we only consider change in mean, but the ideas below are easily extended to other changepoint settings. Therefore, in this illustrative change in mean setting, the model for the data is expressed as follows:

$$X_{i,t} = \mu_i + \delta_i \mathbb{1}_{\{t>\tau\}} + \epsilon_{i,t}, \quad t \in \mathbb{N}, 1 \leq i \leq d, \tag{3.1}$$

where μ_i is the known pre-change mean, δ_i is the mean shift, and $\{\epsilon_{i,t} : t \in \mathbb{N}\}$ are strictly stationary error sequences. After time τ, the mean of the i-th data stream

changes immediately from μ_i to $\mu_i + \delta_i$. Here it is useful to note that our setting also permits some $\delta_i = 0$, which means that only a subset of data streams are affected by the change. Without loss of generality, we assume $\mu_i = 0$. Under the null hypothesis, the model for the data can be rewritten as

$$X_{i,t} = \epsilon_{i,t}, \quad t \in \mathbb{N}, 1 \leq i \leq d. \tag{3.2}$$

Moreover under the alternative hypothesis, the model is $X_{i,t} = \delta_i + \epsilon_{i,t}, t \in \mathbb{N}, 1 \leq i \leq d$. Our aim is to monitor such a system and raise the alarm as soon as possible following the event at time τ. One way of achieving this is to perform hypothesis testing sequentially, i.e., evaluate the null hypothesis of no change in mean at each time point $t \in \mathbb{N}$. The algorithm will stop and declare a change when we can reject the null hypothesis.

In the classical sequential changepoint detection problem, we evaluate the performance of an algorithm subject to a constraint on its false alarm rate. First, consider an open-ended stopping rule where we have an infinite time-window of measurements and the algorithm never halts until it detects a change. The false alarm rate can be evaluated in two ways. Assume there is no change, and let $\widehat{\tau}$ be the time at which we detect a change, with the convention that $\widehat{\tau} = \infty$ if we detect no change. One approach is to control the average run length, $E^{\infty}(\widehat{\tau})$, the expected time of to a false alarm. This makes sense for procedures with a constant threshold for detection, for which we are certain to detect a change under the Null if we monitor for an infinite time period. Alternatively, one can control the false alarm rate, $P^{\infty}(\widehat{\tau} < \infty)$, the probability of a false alarm. To control this over an infinite time horizon requires increasing the threshold for detecting a change over time. Equivalently, this can be achieved by multiplying the test statistic with a weight function $w(\cdot) < 1$. See Leisch et al. (2000); Zeileis et al. (2005); Horváth et al. (2008); Aue et al. (2012); Kirch and Kamgaing (2015); Weber (2017); Yau et al. (2017); Kirch and Weber (2018); Kengne and Ngongo (2022) for examples of how to choose an appropriate weight function.

In our paper, we focus on controlling the false alarm rate. However Aue et al. (2012) states that "applying open-ended procedures built from the asymptotic critical values have a tendency to be too conservative in finite samples". Therefore, our paper considers a close-ended stopping rule. In this approach, the algorithm will stop either upon detecting a change or upon reaching the predefined monitoring time T. We thus control the false alarm rate over a time window of length T. However, the ideas we present can easily be adapted to the open-ended setting, and also to methods which control the average run length.

Under the context of distributed changepoint detection problem, we additionally evaluate the index - the average transmission cost $\bar{\Delta}$. This is the average number of transmissions at each time step for d sensors, and should be smaller than the pre-specified transmission cost Δ.

Before introducing our proposed method, we first review relevant work. At time t, the local monitoring statistic, $\mathcal{T}_i$ is calculated for the ith stream. Then all the local statistics $\mathcal{T}_i$ can be combined into a global monitoring statistic $\mathcal{T}$ at the fusion centre. There are two common choices of message combinations for monitoring changes within the distributed system. One of these two types, the SUM scheme (Mei, 2010), declares a change when the sum of all the local monitoring statistics exceeds a pre-defined threshold, that is:

$$\hat{\tau}_{\text{sum}}(c_{\text{Global}}) = \inf\{t \geq 1 : \mathcal{T} \geq c_{\text{Global}}\} = \inf\left\{t \geq 1 : \sum_{i=1}^{d} \mathcal{T}_i \geq c_{\text{Global}}\right\},$$

where c_{Global} is global threshold. This way of combining statistics across streams is known to be good if the series are independent and the changes are dense. However, implementing this method on the distributed system requires sending every $\mathcal{T}_i$ to the fusion centre, which is expensive. A sum-shrinkage method (Liu et al., 2019) is proposed to reduce the communication cost by thresholding the test statistics before summing

them:

$$\hat{\tau}_{\text{sum}}(c_{\text{Local}}, c_{\text{Global}}) = \inf\{t \geq 1 : \mathcal{T} \geq c_{\text{Global}}\} = \inf\left\{t \geq 1 : \sum_{i=1}^{d} \mathcal{T}_i \mathbb{I}(\mathcal{T}_i \geq c_{\text{Local}}) \geq c_{\text{Global}}\right\}.$$

Empirically the sum-shrinkage method could achieve similar performance as the SUM scheme in the dense case and surprisingly performs better in the sparse case. When the change is sparse, it has been shown both theoretically and empirically (Mei, 2010; Liu et al., 2019; Chen et al., 2022) that monitoring the maximum of the test-statistics across series is best. In such a setting, the MAX procedure (Tartakovsky and Veeravalli, 2002) monitors the maximum of test statistics and raises the alarm when the maximum of the local test statistics exceeds the thresholds, that is:

$$\hat{\tau}_{\max}(c_{\text{Global}}) = \inf\{t \geq 1 : \mathcal{T} \geq c_{\text{Global}}\} = \inf\left\{t \geq 1 : \max_{1 \leq i \leq d} \mathcal{T}_i \geq c_{\text{Global}}\right\}.$$

The best choice of different schemes depends on the sparsity of changes which is based on the number of affected data streams p. This can be made precise if we consider an asymptotic setting where $p \to \infty$ (Enikeeva and Harchaoui, 2019), and define a change to be sparse if the number of affected streams is $p = o(\sqrt{d})$, and it to be a dense change otherwise. A recent paper (Chen et al., 2022) combines both SUM procedure and MAX procedure to achieve good performance regardless of the sparsity. In the context of distributed monitoring, the MAX procedure is trivially implemented without any communication. Specifically, each sensor has the threshold for the max-statistic and flags a change if their local statistic is above this threshold. Therefore, within this paper, we only focus on developing a communication-efficient version of the SUM scheme. Our aim is a method that performs well for dense changes, but limits the communication cost. We will use the SUM scheme as the ideal method to compare against since it has no restrictions on communication.

3.3 Distributed change point detection method

Our proposed methodology is summarized in Algorithm 7, and described in detail below; the associated notations are defined in a later section. The method essentially comprises of three steps. The first step involves the parallel local monitoring of each data stream by the sensors. As the monitoring unfolds, messages are occasionally sent from the sensors to the centre to indicate the presence of a potential change. Finally, at the centre, these messages are aggregated to find changes that occur across a number of data streams.

Algorithm 7: Centralized and distributed MOSUM

input : historic data $x_{i,t}$ for $i = 1, 2, ..., d$, and $1 \leq t \leq m$

Estimating the baseline parameters // can be done offline

for $i = 1$ **to** d **do**

  estimate $\hat{\mu}_i$ and $\hat{\sigma}_i$

end

Data: $x_{i,t}$ for $i = 1, 2, ..., d$ at time t

while *change is detected or reached the maximum monitoring time* T **do**

  Local monitoring // parallel computing

  for $i = 1$ **to** d **do**

    $\mathcal{T}_i(m, k, h) = \frac{1}{\hat{\sigma}_i} \left| \sum_{t=m+k-h+1}^{m+k} (X_{i,t} - \hat{\mu}_i) \right|$.

  end

  Message passing

  if $w(k, h)\mathcal{T}_i(m, k, h) > c_{Local}$ **then**

    $M_{i,t} = \mathcal{T}_i(m, k, h)$ // centralized scheme:set $c_{\text{Local}} = 0$

    else $M_{i,t} = 0$;

  end

  Global monitoring

  if $w(k, h)\sqrt{\sum_i^d M_{i,t}} > c_{Global}$ **then**

    stop the algorithm

    output: $\hat{\tau} = t$

  end

  $t \longleftarrow t + 1$

end

3.3.1 Local monitoring

Estimating the baseline parameters

Our sequential testing approach requires a historic data set of length m to estimate the baseline parameters. Theoretical results are obtained later in the paper when $m \to \infty$. The parameters of interest are the mean of each data stream μ_i and the variance of the errors σ_i^2. For the ith data stream these estimates are,

$$\begin{aligned} \hat{\mu}_i &= \frac{1}{m}\sum_{t=1}^{m} X_{i,t}, \\ \hat{\sigma}_i^2 &= \frac{1}{m}\sum_{t=1}^{m} \left(X_{i,t} - \hat{\mu}_i\right)^2. \end{aligned} \tag{3.3}$$

If the errors cannot be assumed to be independent we can estimate the long run variance. This requires specifying a kernel function $K(\cdot)$:

$$\hat{\sigma}_i^2 = \frac{1}{m}\sum_{t=1}^{m} \left(X_{i,t} - \hat{\mu}_i\right)^2 + 2\sum_{j=1}^{m-1} K\left(\frac{j}{l}\right)\hat{\gamma}_j^{(i)}, \tag{3.4}$$

$$\text{where } \hat{\gamma}_j^{(i)} = \frac{1}{m-j}\sum_{t=1}^{m-j} \left(X_{i,t} - \hat{\mu}_i\right)\left(X_{i,t+j} - \hat{\mu}_i\right).$$

In this setting, the Kernel function can be seen as a weighting function for sample covariance $\hat{\gamma}_j^{(i)}$. The kernel function must be symmetric and such that $K(0) = 1$. Various kernel functions are proposed. Standard kernel functions include Truncated (White and Domowitz, 1984), Bartlett (Newey and West, 1986) and Parzen (Gallant, 2009) amongst others. Among them, the Bartlett kernel is frequently used in Econometrics. This kernel takes the form:

$$K_{\text{Bartlett}}\left(\frac{j}{l}\right) = \begin{cases} 1 - \frac{j}{l}, & \text{for } 0 \le j \le l-1, \\ 0, & \text{otherwise.} \end{cases}$$

For more details, see Horváth and Hušková (2012); Kiefer and Vogelsang (2002a); Kiefer and Vogelsang (2002b).

Starting local monitoring

Once the baseline parameters have been estimated, beginning at time $m+1$ data $X_{i,m+1}, X_{i,m+2}, \ldots$ are observed sequentially and monitored for a change. This is achieved using a MOSUM statistic which at monitoring time, k, takes a window containing the most recent h observations. Therefore, at time $t = m + k$:

$$\mathcal{T}_i(m,k,h) = \frac{1}{\hat{\sigma}_i}\left|\sum_{t=m+k-h+1}^{m+k}(X_{i,t} - \hat{\mu}_i)\right|. \tag{3.5}$$

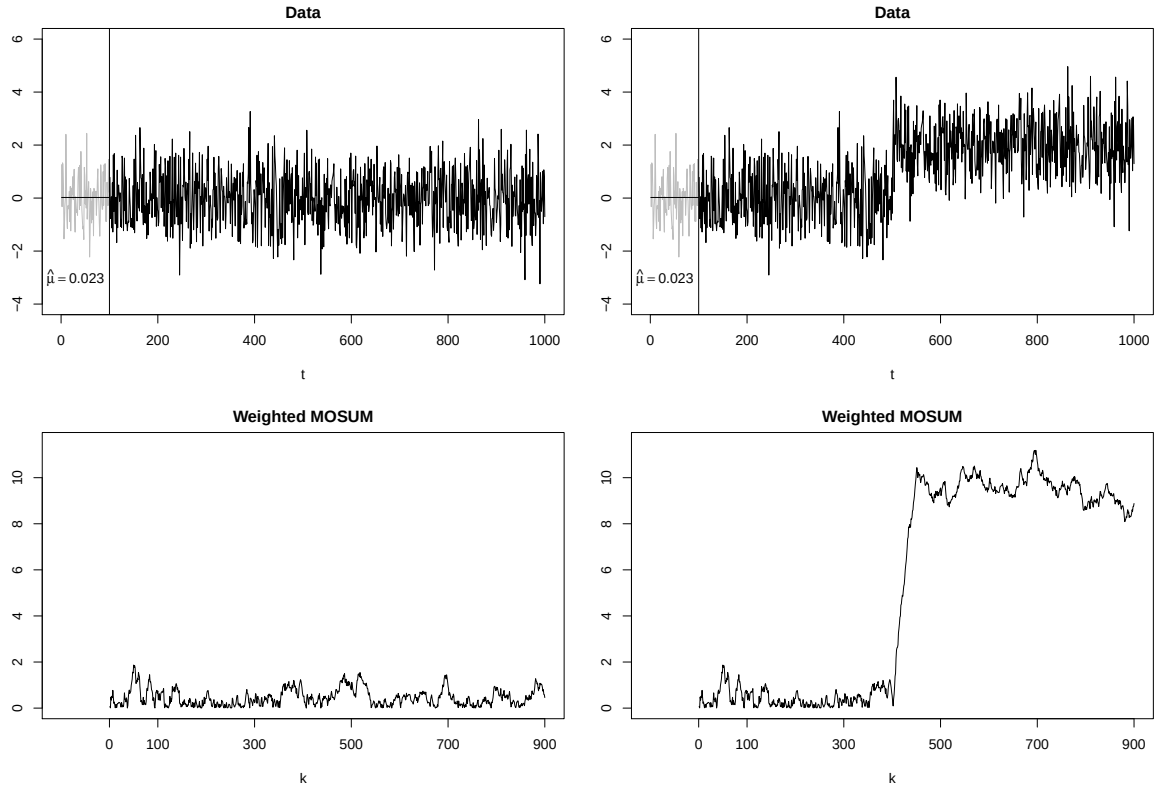
Following Aue et al. (2012), the MOSUM statistic will declare a change at time k when the weighted local MOSUM statistic $w(k,h)\mathcal{T}_i(m,k,h)$ exceeds a pre-defined threshold. A weight function $w(\cdot,\cdot)$ is introduced to control the asymptotic size of the detection procedure. Typically $w(\cdot,\cdot)$ depends on the monitoring time k, and the window size h,

$$w(k,h) = \frac{1}{\sqrt{h}}\rho\left(\frac{k}{h}\right), \tag{3.6}$$

for some appropriate $\rho(\cdot)$. The choice of the weight function controls the sensitivity of the test. A wide range of weight functions can be used as long as they are continuous functions that satisfy $\inf_{0\leq t\leq T}\rho(t) > 0$. In this paper, we use the weight function proposed in Leisch et al. (2000) and Zeileis et al. (2005):

$$\rho(t) = \max(1, \log(1+t))^{-1/2}.$$

Intuitively, if there is no change the weighted MOSUM will remain small, but it will be large if there is a change. Figure 3.2 gives the behavior of weighted MOSUM statistic under the null and the alternative assumptions for one data stream.



(a) (b)

Figure 3.2: Example time series with no change (a) and a single change (b) in the top row. The bottom row shows the weighted MOSUM statistic with a historic period of length $m = 100$ and a window size of $h = 50$.

3.3.2 Message passing

The local monitoring described in the previous section is applied to each sensor independently. In order to make global decisions about the state of the system, messages from the sensors must be passed to the central hub (see Figure 3.1). However, since there are constraints on communication in the system, the message passing process must be carefully designed.

At time $t = m + k$, where m is the historic period of length m, and k is the monitoring time, each sensor makes a decision as to whether or not to transmit a message to the centre. This message vector is denoted as $\mathbf{M}_t = (M_{1,t}, M_{2,t}, \ldots, M_{d,t})$. We consider two different messaging regimes:

- Centralized messaging regime: $\mathbf{M}_t = \mathcal{T}_i(m, k, h), \quad \text{for} \quad 1 \leq i \leq d.$

- Distributed messaging regime:

$$M_{i,t} = \begin{cases} \mathcal{T}_i(m, k, h) & \text{if } w(k, h)\mathcal{T}_i(m, k, h) > c_{\text{Local}}, \\ NULL & \text{otherwise.} \end{cases} \tag{3.7}$$

The centralized massaging regime is one where there is no constraint on the communication between the sensors and the centre, so all sensors send a message to the centre at each time instant. This is similar to the "SUM" scheme changepoint detection method proposed by Mei (2010). However, when communication is expensive, a "distributed" messaging regime can be used where each of the sensors only send local monitoring statistics that exceed a chosen threshold. The $NULL$ means no message is sent. The threshold c_{Local} can be chosen to control the fraction of transmitting sensors when there is no change. It is worth noting that when $c_{\text{Local}} = 0$, the "distributed" messaging regime is equivalent to "centralized" messaging regime.

3.3.3 Global monitoring

In our paper, we assume that there is no communication delay between sensors and the central hub, so the message could be immediately received by the centre at time t. Based on the messages received, the centre will make the decision as to whether or not to flag a change.

Combining messages

Depending on different messaging regimes, the global MOSUM statistics are constructed as follows:

- Centralized global MOSUM statistic:

$$\mathcal{T}(m,k,h) = \sqrt{\sum_{i=1}^{d} M_{i,t}^2}, \tag{3.8}$$

  This is similar to the SUM scheme mentioned in Section 3.2. By using such a scheme, Formula 3.8 is the idealistic scheme under dense change.

- Distributed global MOSUM statistic:

$$\mathcal{T}(m,k,h) = \sqrt{\sum_{i=1}^{d} M_{i,t}^2 \mathbb{1}_{\mathcal{T}_i(m,k,h) > c_{\text{Local}}}}, \tag{3.9}$$

  where $NULL$ values in Formula 3.7 are taken to be zeros in the sum. The form of Equation 3.9 is taken from the multivariate MOSUM (Kirch and Kamgaing, 2015; Weber, 2017; Kirch and Weber, 2018).

Declaring the change

Similar to the local monitoring procedure, a change is declared as soon as the weighted global MOSUM exceeds a threshold. A closed-end stopping rule can be used when the aim is to monitor changes within a fixed time. This can be formalised as

$$\tau_{m,\tilde{T}} = \min\left\{1 \le k \le \lfloor m\tilde{T} \rfloor : w(k,h)\mathcal{T}(m,k,h) > c_{\text{Global}}\right\}, \tag{3.10}$$

where $\min\{\emptyset\} = \infty$ and the total length of the data $T = m\tilde{T}$. If no change is detected by this stopping rule prior to $\lfloor m\tilde{T} \rfloor$, the monitoring procedure is terminated. The parameter $\tilde{T} > 0$ governing the length of the monitoring period is chosen in advance (Horváth et al., 2008; Aue et al., 2012). Figure 3.3 shows the weighted global MOSUM statistic for the distributed and centralized messaging regimes on the same dataset. Whenever the weighted global MOSUM of distributed regime hits zero, there is no

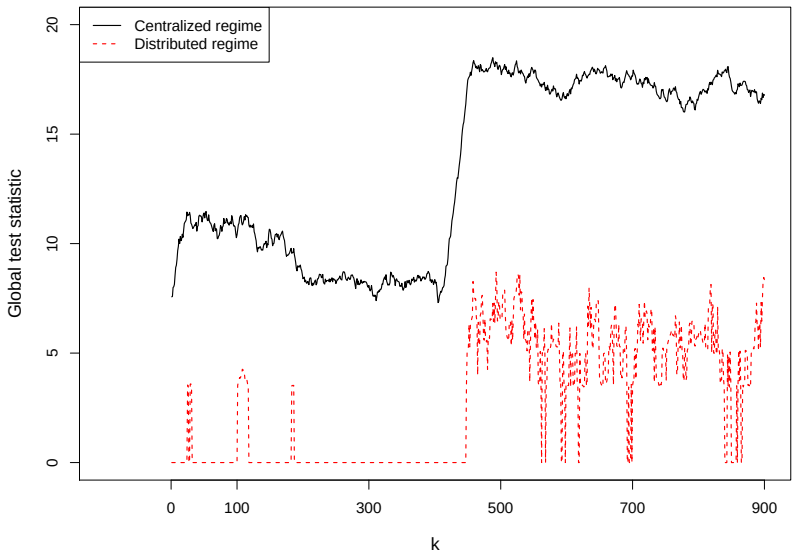


Figure 3.3: Example of the weighted global MOSUM statistic for the distributed (red dashed line) and centralized (black line) regime. The result is obtained with $T = 1000, d = 100, m = 100,\ h = 50, \delta = 0.5$ and the number of affected sensors $p = 50$. A value of $c_{\text{Local}} = 3.44$ was used in the distributed regime.

communication between the edges and the centre at that time. In the next section, we will show the theoretical properties of our proposed method under H_0 and H_A.

3.4 Theoretical properties for distributed MOSUM

This section considers the theoretical properties of the closed-end stopping rule, $\tau_{m,\tilde{T}}$ defined in Equation 3.10 as $m \to \infty$. Firstly, in Section 3.4.1 we find the limiting distribution under the null hypothesis for the different procedures. Then, appropriate choices for the thresholds, c_{Local} and c_{Global} are given in Section 3.4.2 using these results. Finally, in Section 3.4.3 we prove that the detection procedures we have studied are consistent under alternatives.

Three key assumptions are made in order to derive asymptotic results, which are the same in Horváth et al. (2008), Aue et al. (2012), and Weber (2017):

Assumption 1 (Clean historic data)**.** *$h \to \infty$ as $m \to \infty$ and the location of the changepoint $\tau > m$ for $1 \leq i \leq d$.*

This assumption is to guarantee we can get good estimators based on the training dataset, and it can be easily achieved in real applications.

Assumption 2 (Asymptotic regime)**.** *$h \to \infty$ as $m \to \infty$ and*

$$\lim_{m\to\infty} \frac{h}{m} \to \beta \in (0, 1].$$

This assumption quantifies the long run connection between the length of the historical period m and the window size $h := h(m)$.

Assumption 3 (FCLT on errors)**.**

$$\lim_{m\to\infty} \frac{1}{\sqrt{m}} S_i(mt) \xrightarrow{\mathcal{D}} \sigma_i W_i(t)$$

where $\sigma_i > 0$, $\{W_i(t), 0 \leq t < \infty\}$ is a standard Brownian motion when $h \to \infty$, and $S_i(x) = \sum_{t=1}^{\lfloor x \rfloor} \epsilon_{i,t}$. σ_i can be estimated by $\hat{\sigma}_i$. Furthermore, $\hat{\sigma}_i$ satisfying $\hat{\sigma}_i \xrightarrow{\mathcal{P}} \sigma_i$ as $m \to \infty$.

This assumption is a functional central limit theorem on the errors, ϵ, in the model for the data 3.1.

3.4.1 Asymptotics under the null

In this part, the asymptotic theories of our proposed method will be given, which can help guide the choice of thresholds.

The local monitoring process of our proposed method within each sensor is the same as univariate MOSUM detection process. Thus, Theorem 3.4.1 and Corollary 3.4.2 of the local MOSUM can be directly cited from Horváth et al. (2008), Aue et al. (2012)

and Weber (2017). For simplicity, we denote

$$Z_i(t) = \left| W_i\left(\frac{1}{\beta}+t\right) - W_i\left(\frac{1}{\beta}+t-1\right) - \beta W_i\left(\frac{1}{\beta}\right)\right|, \quad 1 \le i \le d \tag{3.11}$$

where $\{W_i(t), 0 \le t < \infty\}$ are independent standard Brownian motions.

Theorem 3.4.1 (*Local MOSUM*). *If assumption 1-3, and model 3.2 holds, then under* H_0, *let* $k = ht$ *for any* $t > 0$

$$\lim_{m\to\infty} w(k,h)\mathcal{T}_i(m,k,h) \xrightarrow{\mathcal{D}} \rho(t)Z_i(t).$$

Corollary 3.4.2 (*Local MOSUM - asymptotic type-I error*). *Under* H_0, *for any* $\tilde{T} > 0$ *and ith data stream,*

$$\lim_{m\to\infty} P\left(\tau^{(i)}_{m,\tilde{T}} < \infty\right) = P\left(\sup_{0\le t\le \tilde{T}/\beta} \rho(t)Z_i(t) > c_{Local}\right).$$

Thus, the false alarm rate for one data stream is asymptotically equal to a pre-specified type-I-error $\in (0,1)$.

Following the results of local MOSUM, similar results for global MOSUM follow readily. These can be used to choose thresholds given the pre-defined Type-I-error. Below we obtain two limiting distributions, for the centralized and distributed regime settings of Section 3.3 respectively.

Theorem 3.4.3 (*Global MOSUM*). *Let* $k = ht$ *for any* $t > 0$, *then under* H_0,

$$\lim_{m\to\infty} w(k,h)\mathcal{T}(m,k,h) \xrightarrow{\mathcal{D}} \rho(t) \begin{cases} \sqrt{\sum_{i=1}^d Z_i(t)^2} & \textit{centralized case,} \\ \sqrt{\sum_{i=1}^d Z_i(t)^2 \mathbb{1}_{\rho(t)Z_i(t) > c_{Local}}} & \textit{distributed case.} \end{cases}$$

Proof. See Appendix A.1. □

Thus, their limiting distribution will be a function of Gaussian process. Using the Theorem 3.4.3, the following may be obtained:

Corollary 3.4.4 (*Global MOSUM - asymptotic type-I error*)**.** *Under H_0, for any $\tilde{T} > 0$,*

$$\lim_{m\to\infty} P\left(\tau_{m,\tilde{T}} < \infty\right) = \begin{cases} P\left(\sup_{0\le t\le \tilde{T}/\beta} \rho(t)\sqrt{\sum_{i=1}^{d} Z_i(t)^2} > c_{Global}\right) & \textit{centralized case,} \\ P\left(\sup_{0\le t\le \tilde{T}/\beta} \rho(t)\sqrt{\sum_{i=1}^{d} Z_i(t)^2 \mathbb{1}_{\rho(t)Z_i(t)>c_{Local}}} > c_{Global}\right) & \textit{distributed case.} \end{cases}$$

This result can lead us to find the local and global thresholds which can obtain the pre-defined type-I-error.

3.4.2 Obtaining critical values

Using the results of the previous section, appropriate critical values can be found such that the asymptotic type-I error is controlled for the different procedures. To achieve this the stochastic processes $\{Z_i(t), 0 \le t \le \tilde{T}/\beta, 1 \le i \le d\}$ need to be approximated on a fine grid. This is done in the same way as Aue et al. (2012), simulating the component standard Brownian motions using ten thousand i.i.d. standard normal random variables. The parameters used were $\beta = 1/2$ and $\tilde{T} = 10$. Tables 3.1 and 3.2 give critical values for $\alpha \in \{0.10, 0.05, 0.01\}$.

α	c_{Local}	$c_{\text{Global}}^{\text{Centralized}}(\alpha)$
0.10	0	14.1
0.05	0	14.4
0.01	0	15.0

Table 3.1: Critical values for the centralized procedures, results averaged over five thousand replications.

Since the critical values obtained above are valid asymptotically (in m), an important question to consider is how they perform in finite samples. Numerical results of

α	c_{Local}	$c_{\text{Global}}^{\text{Distributed}}(\alpha)$
	3.15	7.48
0.10	3.44	6.70
	4.05	5.20
	3.15	7.89
0.05	3.44	7.16
	4.05	6.02
	3.15	8.74
0.01	3.44	8.01
	4.05	6.59

Table 3.2: Critical values for the distributed procedure with different values for c_{Local}, results averaged over five thousand replications.

empirical size in the finite sample are shown in Table 3.3. These indicate that the implementation in the finite sample setting can be conservative, as per Aue et al. (2012). However, approximately, the type-I error is controlled at the correct level for both of the global procedures in finite samples.

Method	c_{Local}	c_{Global}	Proportion of false alarms		
			$m = 200$	$m = 400$	$m = 500$
	3.15	7.89	5.14%	5.18%	5.5%
Distributed	3.44	7.16	5.3%	5.38%	5.12%
	4.05	6.02	5.5%	5.64%	5.16%
Centralized	-	14.4	5.92%	5.28%	5.3%

Table 3.3: Empirical size, results averaged over one thousand replications with $\alpha = 0.05, \tilde{T} = 10$, and $\beta = 1/2$.

The choice of local threshold c_{Local}

The values for c_{Local} used in Table 3.2 are somewhat arbitrary. The main influence of the value of local threshold is that it controls the proportion of messages that the system can pass (on average) per iteration. For d streams, the number of sensors passing message at each time step is:

Corollary 3.4.5 (*Transmission cost*)**.** *For any $t > 0$ and $k = ht$, the expected fraction*

of transmitting sensors at each time step is

$$\bar{\Delta}_t = dP\left(\rho(t)|Z| > c_{Local}\right).$$

where Z is the standard normal distribution.

Therefore, the local threshold can be chosen based on the restriction of the transmission cost. Combined with pre-defined type-I-error, the global threshold will be given based on Theorem 3.4.3.

3.4.3 Asymptotics under the alternative

Under the alternative it is assumed that there is a changepoint at monitoring time k^* and a subset $\mathcal{S}$ of the data streams have an altered mean

$$H_A : \tau = m + k^* \quad \& \quad \exists \mathcal{S} \subset \{1, 2, \ldots, d\} : \delta_i \neq 0 \quad \text{for } i \in \mathcal{S}.$$

Deriving sharp asymptotic results on the detection delay of the proposed method is challenging, and thus we focus only on giving consistency results. A procedure is consistent if it stops in finite time with probability approaching one as $m \to \infty$. In other words, the test statistic should tend to infinity as $m \to \infty$. In the asymptotic regime of interest, we additionally assume that the changepoint k^* grows at the same order as h, that is $\frac{k^*}{h} \to \gamma \geq 0$, and the size of change $\delta_{i,t}$ satisfies $\sqrt{h}|\delta_{i,t}| \to \infty$ as $m \to \infty$ and $h \to \infty$. These assumptions are the same in Aue et al. (2012).

Theorem 3.4.6 (*Global MOSUM: Consistency*)**.** *If the assumption above holds, under H_A,*

(i) the changepoint $k^* \leq \lfloor h\nu \rfloor$ for some $0 < \nu < \tilde{T}\frac{m}{h}$,

(ii) there exists a constant $c > 0$ such that $\rho(x + 1) \geq c$ for all $x \in (\nu, \tilde{T}\frac{m}{h} - 1)$.

Then, as $m \to \infty$ *and* $h \to \infty$

$$\max_{1 \le k \le \lfloor m\tilde{T} \rfloor} w(k,h)\mathcal{T}(m,k,h) \xrightarrow{\mathcal{P}} \infty.$$

Proof. See Appendix A.2. □

Thus, our proposed method is consistent.

3.5 Simulations

In this section, we will present the numeric performance of our algorithm. Since the SUM procedure that is optimal when the change is dense, we will evaluate the performance in the dense case, specifically when the affected data streams $p = d$. Firstly, the different practical choices of thresholds at fixed type-I-error will be investigated. Here the performance of our proposed method was also compared against the idealistic setting. Finally, the effect of parameters and the violation of the independence assumption are investigated.

The set-up of the simulations is as follows. For simplicity, the data generating process under the null is that $X_{i,t} \sim N(0,1)$ for $1 \le i \le d$ and $1 \le t \le T$. To compare fairly, the type-I-error of all procedures is controlled to be 0.05 under the null.

The family of alternatives considered is that

$$X_{i,t} \sim N(0,1) \text{ for } 1 \le i \le d, 1 \le t < \tau \text{ and}$$
$$X_{i,t} \sim N(\delta_i, 1) \text{ for } 1 \le i \le d, \tau \le t \le T.$$

We assume the change will affect all the sensors instantaneously. But the size of the change is unknown. We consider two scenarios of mean shift: 1) Same size: $\delta_i = \delta =$ some constant values for $1 \le i \le d$; 2) Random size: $\delta_i = \eta N(0,1)$, where η is the scale factor controlling the magnitude of size. The average detection delay (ADD) $\bar{D}$ and

average communication cost $\bar{\Delta}$ are then measured:

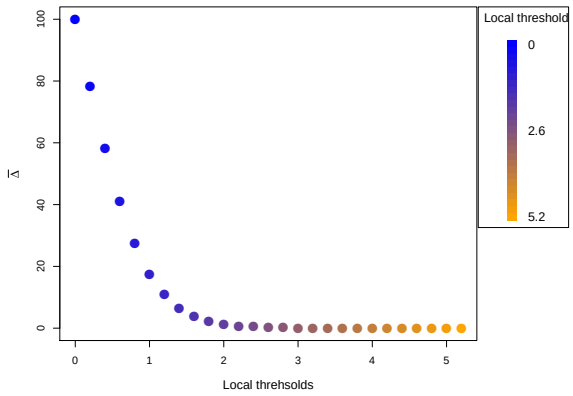
$$\bar{D} = E(\hat{\tau} - \tau | \hat{\tau} > \tau)$$
$$\bar{\Delta} = \sum_{t=1}^{\hat{\tau}} \frac{\sum_{i=1}^{d} \mathbb{I}(w(k,h)\mathcal{T}_i(m,k,h) > c_{\text{Local}})}{\hat{\tau} - m + 1},$$

3.5.1 The numerical dependency on local thresholds

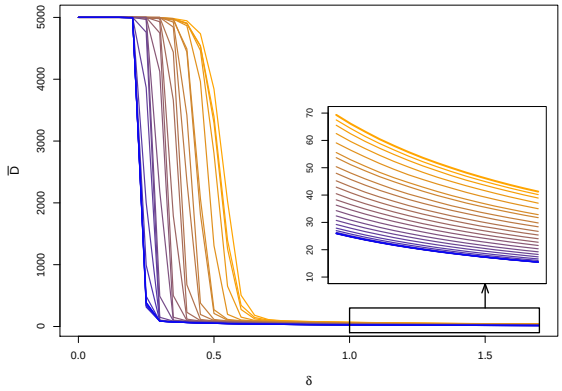
Our proposed method requires specifying two thresholds. Usually, c_{Global} can be given based on the Theorem 3.4.4 once α and c_{Local} are confirmed. Therefore, it is crucial to pick an appropriate local threshold. This section gives numeric results with different values of local thresholds, which may provide some guidance in choosing the local threshold.

Figure 3.4 gives the average detection delay and transmission cost for different values of local thresholds. There is a trade-off between communication savings and detection performance when choosing the local threshold. Larger local thresholds can reduce the transmission cost but will also lead to longer delays, especially when the change is small. However, with the increase in the mean shift, the detecting power of larger thresholds will close to that of small thresholds.

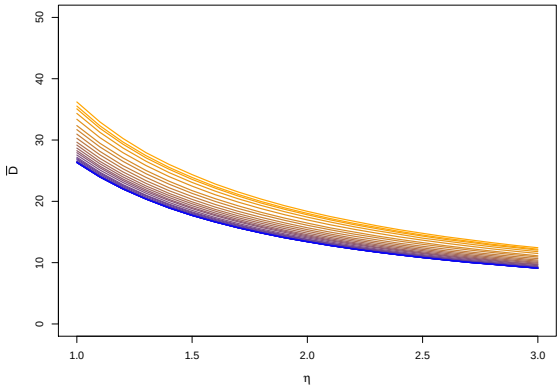
A centralized framework can be seen as an idealistic setting, which is equivalent to distributed setting when $c_{\text{local}} = 0$. Compared with the idealistic setting, the distributed MOSUM can achieve similar performance when the size of the change is not small but also reduces massive transmission costs. But we will lose power in detecting small changes. We show the result below that distributed MOSUM can approximate the performance of idealistic setting overall by increasing the window size.



(a) Transmission cost for $d = 100$ data streams.



(b) average detection delay versus fixed δ.



(c) average detection delay versus η.

Figure 3.4: The average number of messages transmitted to the centre (top) and average detection delay across varying mean shifts (bottom). Results are obtained when $m = 200$, $h = 100$, $T = 10000, \tau = 5000, \alpha = 0.05$. Each line corresponds to a different local threshold, which is labelled on the top right. The colour changes from blue to orange as the local thresholds increase from 0 to 5.2. When the local threshold is 5.2, the global threshold will be 0. So all possible combinations of thresholds are covered.

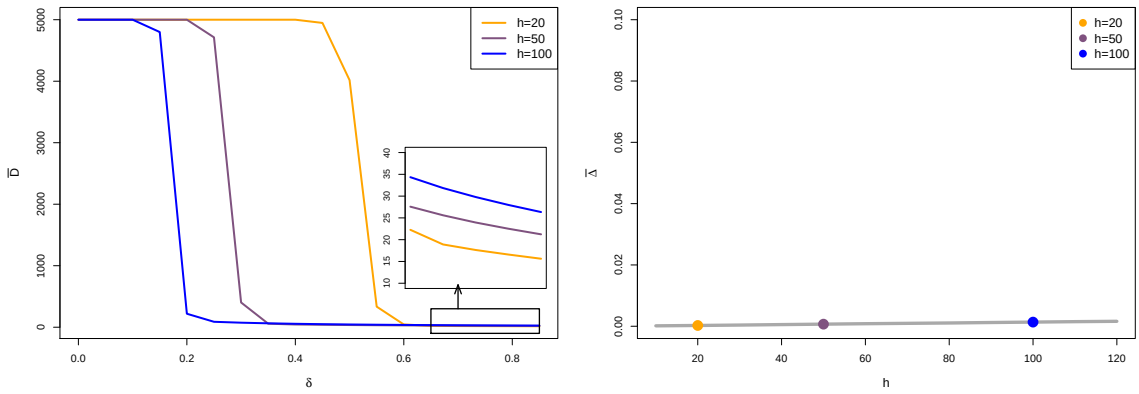
3.5.2 The numerical dependency on parameters

One advantage of using MOSUM statistics is that we do not need to specify the post-change mean. Instead, our proposed method requires specifying the window size h and the training size m. In this section, we will investigate the impact of bandwidth and training size.

The impact of bandwidth

As shown in Figure 3.5a, increasing the window size can increase the power of detecting small changes while leading to a slight delay in detecting large changes. Although increasing window size will increase the storage cost, it will not significantly increase the transmission cost as shown in Figure 3.5b. This drive us to think about whether we can improve the ability of distributed MOSUM with a large threshold to detect small changes by increasing the window size. Ideally, we would expect distributed MOSUM with increased window size can achieve similar performance as the idealistic setting.

Recovering detectability



(a) $\bar{D}$ versus δ for $h = 20, 50, 100$.

(b) $\bar{\Delta}$ versus h.

Figure 3.5: The influence of window size. Results are obtained over 1000 replications and take $m = 200, d = 100, T = 10000, \tau = 5000, \alpha = 0.05, c_{\text{Local}} = 3.44$.

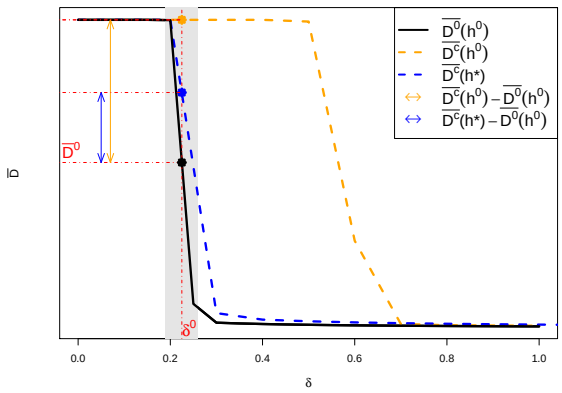
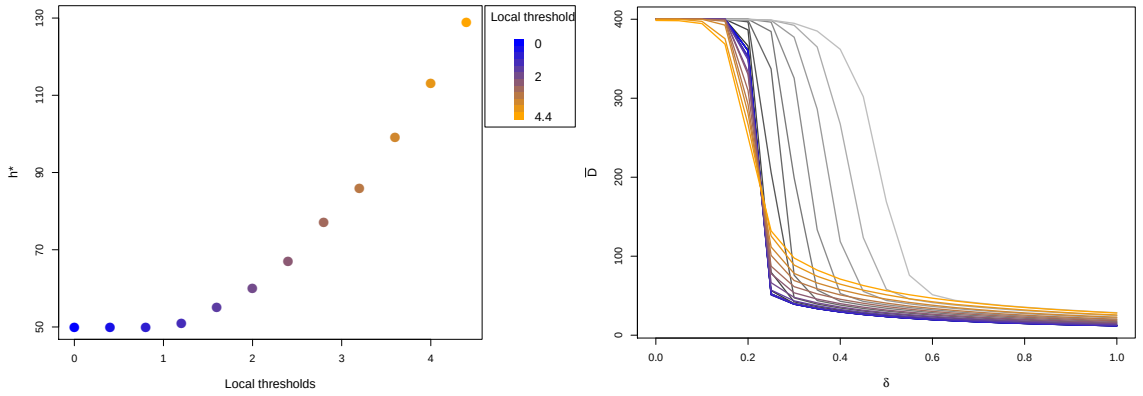


Figure 3.6: An graphic explanation of our proposed idea. Black line is the ADD for centralized MOSUM with window size h. Yellow dashed line is the ADD for distributed MOSUM with window size h; while blue dashed is the ADD for distributed MOSUM with window size $h*$.

For simplicity, we denote that the default window size for centralized MOSUM is h^0, and h^* is the smallest window size that would allow distributed MOSUM to have similar performance as the idealistic setting. It is difficult to develop a neat theoretical formula between h^* and h^0. But we can approximately find h^* under alternatives by simulation. Our idea can be summarized as follows, and Figure 3.6 is the graphic explanation:

- The behaviour of $\bar{D}$ will decrease dramatically when the mean shift is within a certain range (gray area). Therefore, we can find the median or mean δ of this certain range, denoted by δ^0. Also, the corresponding ADD $\bar{D}^0$ can be calculated.
- Fix δ^0, calculate $\bar{D}^{c_{\text{Local}}}(h)$ iteratively for distributed MOSUM, where $h \in [h^0, m]$.
- The optimal window size $h^* = \arg\min \left\{ \bar{D}^{c_{\text{Local}}}(h) - \bar{D}^0(h^0) \right\}$. See blue arrow ($h^*$) is shorter than yellow arrow (h).

Figure 3.7 displays the simulation results that, for distributed regime, we can recover the same detectability of the centralized statistic by inflating h.



(a) h^* found by our idea

(b) average detection delay for distributed MOSUM (fixed $h = 50$, gray line) and distributed MOSUM ($h = h^*$, colored line). Different lines corresponds to different c_{Local}

Figure 3.7: An simple example showing that distributed MOSUM could approximate the detection power of centralized MOSUM by inflating window size. Results are obtained over 500 replications and take $m = 200, d = 100, T = 1000, \tau = 600, \alpha = 0.05$, and $c_{\text{Local}} \in [0, 4.4]$. When $c_{\text{Local}} = 4.4$, $c_{\text{Global}} = 0$. So all possible local thresholds are covered. For centralized setting, window size $h^0 = 50$.

	$m = 80$	$m = 100$	$m = 500$	$m = 1000$
c_{Global}	9.039	8.159	6.014	5.708
Empirical size	0.007	0.007	0.009	0.006
MSE for mean	0.0125	0.01	0.002	0.001
MSE for sd	0.006	0.005	0.0001	0.0001

Table 3.4: Empirical size, and MSE for estimated mean and standard deviation, results averaged over one thousand replications with $c_{\text{Local}} = 3.44, h = 50, T = 6000$ and $\alpha = 0.05$.

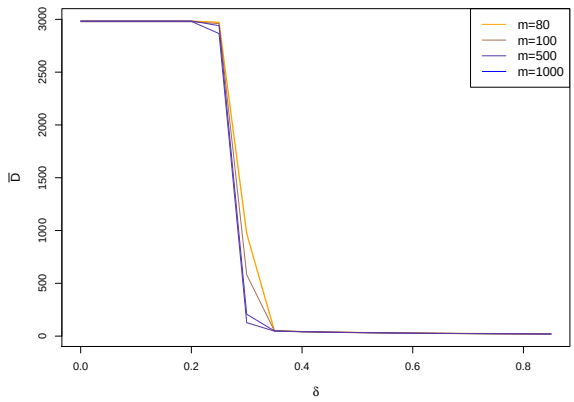


Figure 3.8: $\bar{D}$ versus δ when varing the size of training dataset. Result averaged over 500 replications with $\alpha = 0.05, c_{\text{Local}} = 3.44, T = 6000$, $\tau = 3000$ and $h = 50$. The corresponding global thresholds are shown in Table 3.4.

The impact of the training dataset

Fix the bandwidth h, the impact of the size of the training dataset can be investigated. Table 3.4 gives the thresholds, empirical size, and mean square errors (MSE) of estimated baseline parameters in our simulation. As we expected, the larger the training size is, the more accurate estimators are. Figure 3.8 indicates that overall the detection powers of four different sizes of training datasets are similar. A larger training size could slightly increase the detection power when detecting small changes, which is attributed to more accurate estimators. Thus, in the real application, it is beneficial to choose a large-size training dataset because it is not expensive that can be done offline.

3.5.3 The violation of the independence assumption

Before, we assume that there is temporal independence among observations. However, it may not always hold in the real application. This section will investigate the performance when this assumption is violated. Here we measure our algorithm under AR(1)

noise process, that is

$$X_{i,t} = \delta_{i,t}\mathbb{1}_{\{t>\tau\}} + \epsilon_{i,t},$$

where $\epsilon_{i,t} = \phi\epsilon_{i,t-1} + v_t$ with $v_t \sim N(0,1)$. $|\phi| < 1$ is used to measure the strength of the auto-correlation.

Auto-correlation will inflate the variance of data. There are two possible ways to handle this problem. The first one is to estimate the long-run variance as shown in Section 3.4. And one can also inflate the thresholds. We measure the false positives, average detection delay and the number of transmitted messages with fixed type-I-error of these two solutions under different scenarios. For better comparison, we also show the result of MOSUM without any adjustment. This will give us hints that to what extent our method fails to detect the change if we ignore the auto-correlation.

As Table 3.5 shows, our proposed method without adjustment can lose the ability to detect changes when introducing auto-correlation, that it fails to detect the change and always alarms. The performances of MOSUM with inflating thresholds are generally better than MOSUM with LRV since the former can detect the change in most scenarios. However, for those scenarios that the MOSUM with LRV can detect (usually δ is not small and ϕ is not large), it always has the lowest transmission cost and reasonable detection power. For example, when $p = 100, \delta = 1$, and $\phi = 0.25$, both solutions have similar false positive rates and average detection delay, while MOSUM with LRV has lower transmission cost. It is surprising that estimating LRV has the lowest false positive rates and average detection delay when $\phi = 0$ and $p = 100/50$. This may be because it underestimates the variance.

However, when the auto-correlation is serious, it is not appropriate to apply our method to the raw data. Instead, it is more reasonable to apply our method after pre-processing the data, such as the residuals of AR models.

ϕ	Methods	$p = 100$					$p = 50$					$p = 10$				
			$\delta = 0.5$		$\delta = 1$			$\delta = 0.5$		$\delta = 1$			$\delta = 0.5$		$\delta = 1$	
		FP	D	Trans	D	Trans	FP	D	Trans	D	Trans	FP	D	Trans	D	Trans
0	constant	0.30	90.67	0.03	45.75	0.02	0.29	116.68	0.03	50.07	0.03	0.23	3767.69	0.18	63.09	0.03
	inflating	0.30	90.67	0.03	45.75	0.02	0.29	116.68	0.03	50.07	0.03	0.23	3767.69	0.18	63.09	0.03
	LRV	0.19	85.85	0.05	43.41	0.05	0.28	105.74	0.06	48.42	0.05	0.32	4198.59	0.34	64.32	0.05
0.25	constant	63.49	79.82	0.19	40.36	0.18	62.04	92.62	0.19	45.90	0.18	61.95	2059.06	0.20	62.85	0.18
	inflating	0.33	104.78	0.22	49.04	0.20	0.21	1413.09	0.76	55.98	0.20	0.40	5000.00	0.48	81.90	0.21
	LRV	0.30	1294.64	0.21	57.21	0.05	0.21	4228.80	0.30	63.95	0.05	0.22	4995.26	0.09	85.23	0.05
0.5	constant	499.42	64.17	1.20	33.07	1.20	507.02	78.94	1.21	40.47	1.21	503.93	764.67	1.14	64.74	1.20
	inflating	0.57	4663.87	5.15	60.51	1.26	0.34	5000.00	3.06	73.66	1.28	0.59	5000.00	1.10	5000.00	4.28
	LRV	0.43	4996.68	0.09	83.57	0.05	0.38	5000.00	0.06	97.18	0.06	0.59	5000.00	0.03	3878.04	0.35
0.75	constant	4219.42	5.74	8.35	3.99	8.35	4225.67	7.56	8.35	5.35	8.35	4228.56	8.25	8.37	6.94	8.37
	inflating	0.76	5000.00	10.19	498.83	10.43	0.85	5000.00	7.84	5000.00	15.63	0.69	5000.00	6.02	5000.00	7.59
	LRV	0.64	5000.00	0.05	4980.28	0.25	0.59	5000.00	0.04	5000.00	0.14	0.78	5000.00	0.04	5000.00	0.06

Table 3.5: Results are obtained over 1000 replications with $T = 10000$, $m = 200$, $h = 100$, $\tau = 5000$, $d = 100$, $c_{\text{Local}} = 3.44$, and $\alpha = 0.05$ for all three methods. The blue colours are labelled when both the false positive rates and average detection delay are small.

3.6 Conclusion

Within this paper, we proposed an online communication-efficient distributed change-point detection method, and it can achieve similar performance as an idealistic setting but save many transmission costs. Numerically, we show that the local threshold and window size have an impact on the performance of our algorithm, and there is a trade-off in choosing a local threshold and window size. In application, we recommend choosing a large local threshold in general cases. But when the change is extremely small, the choice of the local threshold depends on the communication and storage budgets. If the communication budget is much more limited, choosing a large threshold with a large window size is sensible. If the storage cost is much more expensive, choosing a small threshold with small window size will approximately achieve the idealistic performance.

The violation of independent assumptions will negatively affect the power of our proposed method. We tried to solve this problem by inflating thresholds or estimating the long-run variance. Both ways can, to some extent, improve our algorithm when the auto-correlation problem is not severe. However, both approaches fail to detect changes in highly auto-correlated data. Therefore, one of the future research directions is how to detect change within highly auto-correlated data in real-time.

Chapter 4

mixFOCuS: A Communication-Efficient Online Changepoint Detection Method in Distributed System for Mixed-Type Data

4.1 Introduction

This article considers a novel changepoint problem arising from a contemporary, Internet of Things (IoT) setting. Within the IoT, modern technology allows industry to deploy numerous sensors to collect data at a relatively low cost. The detection of changes in such a setting is increasingly of interest. For example, detecting cyber attacks in smart cities (Alrashdi et al., 2019), attacks on network traffic (Lung-Yut-Fong et al., 2012), or abnormal cryptocurrency transactions for data trading (Zhao et al., 2023), amongst many others.

To have the power to detect changes that affect multiple sensors, we need to share information. For some sensor networks this can be done by allowing sensors (edge devices) to communicate with a central cloud. Such a sensor-to-cloud communication is usually described as a "distributed system". However limitations on, e.g., battery life, transmission budget, or privacy concerns require limiting the frequency with which a sensor can transmit information to the cloud (Alrashdi et al., 2019; Varghese et al., 2016). For example, satellite data may be physically stored at different geographical locations, and thus transferring all the data over the network could incur substantial bandwidth and time costs (Bhaduri et al., 2010). Another concern is that edge devices can observe a large amount of streaming data, much of which may be private. Frequently sending raw data or even an "anonymized" dataset can still put user privacy at risk (McMahan et al., 2017).

When sensors collect data in real time, the task of identifying changepoints frequently moves to an online setting. Here, one ideally seeks to raise an alarm when there is a change in the distribution of the data. One, straightforward approach, to achieve this is to run a simple, efficient online changepoint algorithm at each edge device and then combine the results in the cloud. See Figure 4.1 for an example. The problem of detecting changes that affect multiple data streams, but without any constraints on sensor-to-cloud transmission frequency, has been studied extensively, and has led to a number of proposals of how to combine the statistics from different streams (e.g. Tartakovsky et al., 2006; Tartakovsky and Polunchenko, 2008; Mei, 2010; Xie and Siegmund, 2013b). Different methods of combining statistics have different properties. For example, detecting a change based on the maximum statistic across streams performs well if one, or only a small proportion, of data streams undergo a change. By comparison, detecting a change based on the sum of the statistics performs well if the change affects most or all of the streams (Mei, 2010; Enikeeva and Harchaoui, 2019).

To limit the transmission of data Liu et al. (2019) and Yang et al. (2024) suggest

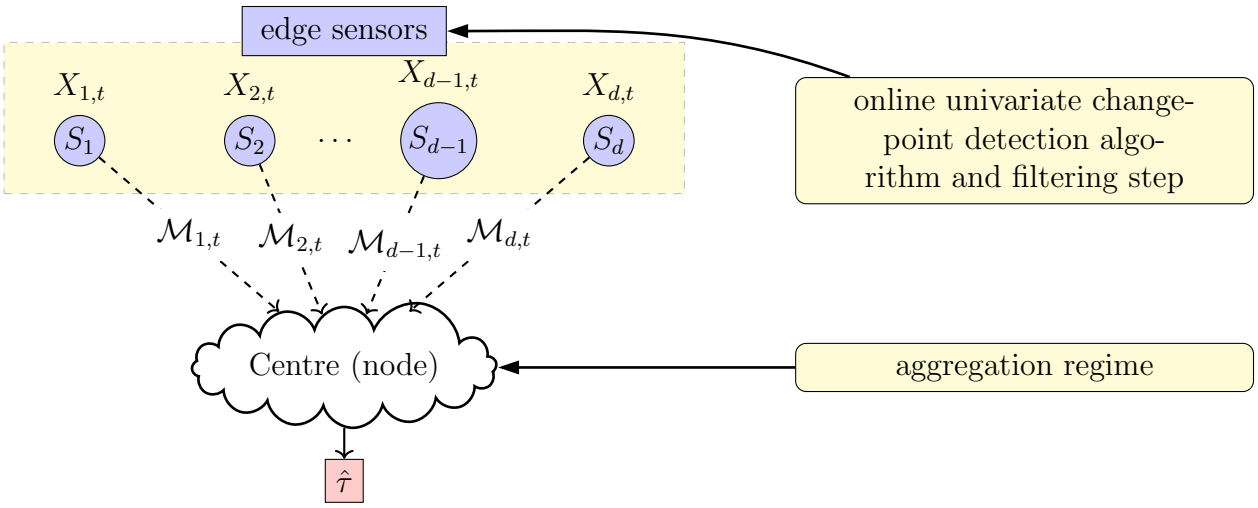


Figure 4.1: Schematic representation of a sensor network made up of d sensors, where S_i is the index for sensor i, $X_{i,t}$ is the data observed at sensor i, $\mathcal{M}_{i,t}$ is the message transmitted from sensor i to centre at time t, and $\hat{\tau}$ is the time the algorithm stops or alarms.

thresholding the local monitoring statistics, that is a sensor only sends information to the cloud when there is sufficient evidence (i.e., the value is above a threshold). The threshold can be chosen, for example, based on the frequency with which we are able to transmit information. Depending on the threshold value selected, the approaches proposed by Liu et al. (2019) and Yang et al. (2024) can work well for changes impacting either a small number of streams (high threshold) or many streams (low threshold). In the case of Liu et al. (2019) the CUSUM statistic for Gaussian data is used for each data stream and then the sum is taken at the fusion centre. Conversely,Yang et al. (2024) adopts the MOSUM statistic (Aue et al., 2012; Kirch and Weber, 2018). Both the CUSUM and MOSUM procedures have limitations. For example: 1) at each edge, both approaches suffer from the need to specify the pre-change distribution, and either the post-change distribution (for CUSUM) or a window size (for MOSUM). If any of these are poorly specified we may lose power to detect a change, or, in the case of a misspecified pre-change distribution, have an inflated false-positive rate; 2) both methods assume collected data are independent and identically Gaussian distributed, which may not be realistic in practice; 3) at the centre, taking the sum of the statistics

can lead to a longer detection delay when only one or a few sensors have changed.

Motivated by the above challenges, we propose a novel approach for the efficient detection and communication of changes within a distributed system, mixFOCuS. This work builds upon the recent development of the FOCuS algorithm (Romano et al., 2023b) and its extensions (Ward et al., 2023, 2024), recent contributions to the literature built on Page's recursion (Page, 1954, 1955). Whereas Page's recursion assumes we know precisely both the pre-change and post-change distribution of the data; the FOCuS algorithms only assume we know the family of these distributions, and that they are from the one-parameter exponential family. When calculating the test statistic for a change, FOCuS then maximises over both the pre-change and post-change parameter. Consequently, the approach that we introduce in this article, mixFOCuS, can be seen as a further development of the seminal work of Page (1954) to the setting of monitoring data streams from distributed networks.

As we will later demonstrate, our contribution differs from previous works in two key respects:

1. **Flexible local monitoring.** mixFOCuS allows data streams to follow the exponential family. It also does not require prior knowledge of the pre-change or post-change distribution. Instead, mixFOCuS uses the expFOCuS algorithm (Ward et al., 2024) to analyse the streams. This algorithm can be applied if the data is modelled as coming from any one-parameter exponential family with an unknown post-change parameter, or even an unknown pre-change parameter. Helpfully, expFOCuS has a small per-iteration computational cost that increases logarithmically with data size. Thus, for Gaussian data, at time T, its cost is roughly that of maximising a $\log T$ quadratic. Furthermore, there are approaches to reduce this cost to a constant, either using a simple approximation (Romano et al., 2023b), by using bounds on the statistics (Ward et al., 2024) or fixing a minimum size of change we wish to detect (Ward et al., 2023).

2. **Robust global monitoring.** mixFOCuS monitors both the maximum and the sum of the test statistics. In so doing, it is able to detect the location of both sparse changes, i.e. where few streams are affected, and dense changes, i.e. where many streams are affected. Further, for the distributed network setting, there is no additional cost for monitoring the maximum of the test statistic.

The outline of the paper is as follows. In Section 4.2 we briefly summarise the link between the FOCuS algorithms we build on and the algorithm of Page (1954). Then, in Section 4.3 we introduce the problem setting and our proposed method - mixFOCuS. In Section 5.3, we evaluate the trade-off between transmission constraints and detection power under various scenarios, considering both homogenous and mixed data types. We also show the improved detection power of mixFOCuS for handling mixed-type data, rather than assuming and approximating data with Gaussian distributions and compare the proposed algorithm compared to those of Liu et al. (2019) and Yang et al. (2024). In Section 4.5 we apply the method to an anomaly benchmark dataset and evaluate the impact of reducing data transmission on the speed at which we detect changes. The paper ends with a discussion.

4.2 Background: From Page (1954) to expFOCuS

Consider a data stream at a single sensor. Denote this by $X_1, X_2, \ldots$. We assume the data are independent and drawn from a single parameter exponential family distribution. We wish to detect if and when the parameter of this distribution changes, and will perform a test for a change sequentially at each time $t = 2, 3, \ldots$.

Let $f(x;\theta)$ denote the density function of our one-parameter exponential family distribution with parameter value θ. Then $f(x;\theta) = h(x)\exp[\eta(\theta)T(x) - A(\theta)]$, for some appropriate scalar functions $h(\cdot)$, $\eta(\cdot)$, $T(\cdot)$ and $A(\cdot)$. Table 4.1 provides examples of these one-parameter exponential family distributions and the corresponding functions

Distribution	$\eta(\theta)$	$T(x)$	$A(\theta)$
Gaussian (fixed variance σ^2)	$\frac{\theta}{\sigma^2}$	x	$\frac{\theta^2}{2\sigma^2}$
Gaussian (fixed mean 0)	$-\frac{1}{2\theta^2}$	x^2	$\log\theta$
Bernoulli	$\log\frac{\theta}{1-\theta}$	x	$-\log(1-\theta)$
Poisson	$\log\theta$	x	θ
Exponential	$-\theta$	x	$-\log\theta$
Gamma (fixed shape α)	$-\theta$	x	$-\alpha\log(\theta)$

Table 4.1: Examples of one-parameter exponential families as natural parameter form. We are interested in detecting the change in the mean with fixed variance σ^2 and the change in variance with fixed mean 0, for Gaussian distributions, the change in probability for Bernoulli distributions, the change in rate for Poisson distributions, the change in rate for Exponential distribution, and the change in rate for Gamma distributions with fixed shape parameter α.

(with $h(x)$ being disregarded as it cancels out during the subsequent calculations).

Formally we can model our problem at time t as testing between two models for the data. If there is no change then

$$X_1, \ldots, X_t \sim f(x;\theta),$$

whereas if there is a change, then for some $\tau \in \{1, \ldots, t-1\}$,

$$X_1, \ldots, X_\tau \sim f(\cdot;\theta), \quad X_{\tau+1}, \ldots, X_t \sim f(\cdot;\theta+\delta),$$

for some $\delta \neq 0$, that corresponds to the change in parameter. Ideally we would base a test for a change on the likelihood-ratio test statistic

$$\mathcal{T}_t(\delta) = 2\left\{\max_{\theta,\delta,\tau}\left\{\sum_{j=1}^{\tau}\log f(x_j|\theta) + \sum_{j=\tau+1}^{t}\log f(x_j|\theta+\delta)\right\} - \max_{\theta}\left\{\sum_{j=1}^{t}\log f(x_j|\theta)\right\}\right\}.$$

The computational challenge is that calculating $\mathcal{T}_t(\delta)$ from scratch is an $O(t)$ calculation, due to the maximisation over τ. This is computationally prohibitive in a streaming setting. Page (1954) proposed a computationally efficient algorithm for calculating $\mathcal{T}_t$ recursively, if θ and δ are known.

In some settings it is natural to assume θ is known or can be well estimated due to sufficient pre-change data. If we consider this setting, and denote $\mathcal{T}_t(\delta)$ to be the likelihood-ratio test-statistic at time t for a specified change δ, then Page (1954) showed that for $t = 1, \ldots,$ with $\mathcal{T}_0(\delta) = 0$,

$$\mathcal{T}_t(\delta) = \max\{0, \mathcal{T}_{t-1}(\delta)\} + 2\left[\{\eta(\theta+\delta) - \eta(\theta)\}T(x_t) - \{A(\theta+\delta) - A(\theta)\}\right]. \quad (4.1)$$

This can be interpreted as replacing the previous statistic by 0 if it was negative, and then adding the contribution to the likelihood ratio-test statistic of the tth observation. Importantly this recursion can be calculated with fixed computational cost after each observation is received.

The disadvantage of Page's recursion is that the test will have lower power if we mis-specify the size of change δ. The idea of the FOCuS algorithm (Romano et al., 2023b) was to solve Formula 4.1 simultaneously for all δ. This was initially done for a Gaussian model, and then extended to Poisson data (Ward et al., 2023) and to the exponential-family model (Ward et al., 2024). In all cases, the solution to Formula 4.1 can be written as a function of δ in terms of the maximum of curves of the form $a\eta(\theta+\delta)+bA(\theta+\delta)$ for scalar coefficients a and b that depend on the data. Each curve corresponds to a different possible value of τ. Importantly there are only a small number of possible values of τ that contribute curves to the definition of $\mathcal{T}_t(\delta)$. The values of τ that contribute curves correspond to vertices on the convex hull of the convex hull of the random walk with points $(s, \sum_{i=1}^{s} T(x_i))$ – and the expected number of vertices is known to increase like $O(\log T)$. Moreover the computational cost of updating the solution of $\mathcal{T}_{t-1}(\delta)$ to that of $\mathcal{T}_t(\delta)$ is $O(1)$. To perform our test we then need to find $\max_\delta \mathcal{T}_t(\delta)$, which needs a sum over all curves and has an $O(\log T)$ cost: though if we simply want to detect whether the test statistic is above a threshold we can re-cycle calculations to have a cost that is empirically $O(1)$ (see Ward et al., 2024).

Finally, because the link between the values of τ that contribute a curve to $\mathcal{T}_t(\delta)$

and the vertices of the convex hull of the random walk of $T(X_i)$ holds for all θ – it is possible to extend this idea to give a similarly efficient algorithm for calculating $\mathcal{T}_t(\delta)$ simultaneously for θ. By maximising $\mathcal{T}_t(\delta)$ over both θ and δ we can calculate our test statistics $\mathcal{T}_t$ recursively in an efficient manner. The resulting algorithm is given in Ward et al. (2024) and is called expFOCuS. Finally we remark that the link between the FOCuS algorithm and the convex hull is a specific example of a link between online change-point algorithms and computational geometry that is discussed in more detail in Pishchagina et al. (2023).

4.3 Problem setting and our proposed method

4.3.1 Problem setting

By way of introduction to the problem, let us assume that there are d sensors deployed in a system, each producing a single stream of data through time. At a given time $t \in \mathbb{N}$, observations or pre-processed data, $X_{i,t}$, are observed by the sensor $i \in \{1, \ldots, d\}$. The data streams are assumed to be independent and follow the exponential family of distributions.

At an unknown time, τ, some unknown subset of data streams undergo a change. Therefore, the i-th stream with change can be modelled as follows:

$$X_{i,1}, X_{i,2}, ..., X_{i,\tau} \sim f_i(\cdot; \theta_i), \quad X_{i,\tau+1}, X_{i,\tau+2}, ... \sim f_i(\cdot; \theta_i + \delta_i).$$

Here $f_i(x; \theta_i) = h_i(x) \exp[\eta_i(\theta_i) T_i(x) - A_i(\theta_i)]$ is the density function or mass function of a one-parameter exponential family for the i-th sensor; θ_i is the pre-change parameter, δ_i is the magnitude of change, and we assume that the distribution is known while its parameters are assumed unknown. This assumption is reasonable in many practical settings, since users may typically know the type of data collected by each sensor. For

example, counting sensors collect data following a Poisson distribution, etc. Alternatively, the distributions can also be estimated through training data. Finally we will focus on the more general situation, where both θ_i and δ_i are unknown. This is in contrast to, e.g. Liu et al. (2019) and Yang et al. (2024), where the θ_i are assumed known (or readily estimable).

Within the context of our distributed changepoint detection problem, the task is to monitor sensors with the pre-specified transmission constraint in real-time and report changes $\hat{\tau}$ as soon as possible. The transmission constraint concerns the average frequency that each sensor is allowed to transmit a message to the cloud. The detection ability will be evaluated by the false alarm rate (that is the algorithm raises alarms and stops before a true change), and the detection delay (this refers to the difference between a true change and an estimated stopping time). Ideally, we seek an approach with low false alarm rates and small detection delays within the transmission constraint.

We first focus on the computation at each edge node, where we employ the online univariate changepoint detection method, expFOCuS, introduced in Section 4.2. For one data stream, i, the log-likelihood ratio test statistic at time t can be written as:

$$\mathcal{T}_{i,t} = 2\left\{\max_{\theta_i,\delta_i,\tau}\left\{\sum_{j=1}^{t}\log f_i(x_{i,j}|\delta_i,\theta_i,\tau)\right\} - \max_{\theta_i}\left\{\sum_{j=1}^{t}\log f_i(x_{i,j}|\theta_i)\right\}\right\}, \tag{4.2}$$

where $\sum_{j=1}^{t}\log f_i(x_{i,j}|\delta_i,\theta_i,\tau) = \sum_{j=1}^{\tau}\log f_i(x_{i,j}|\theta_i) + \sum_{j=\tau+1}^{t}\log f_i(x_{i,j}|\theta_i+\delta_i)$. If this test statistic is larger than a pre-specified threshold, we reject the null hypothesis. Calculating $\mathcal{T}_{i,t}$ can be done efficiently at each edge device using the expFOCuS algorithm.

4.3.2 The mixFOCuS approach

We now introduce mixFOCuS, the proposed monitoring approach that will be used within the thresholding step to reduce the transmission and monitor the maximum and sum of the test statistics. The approach consists of three key parts: local monitoring,

local screening, and global monitoring.

Local monitoring. The local test statistics $\mathcal{T}_{i,t}$ for i-th data stream at time t can be calculated via equation 4.2.

Local screening. Subsequently, a local screening procedure is deployed at each sensor to filter out unimportant messages through thresholding. We introduce a variable, $\mathcal{M}_{i,t}$, representing the message passed from the local sensor to the cloud:

$$\mathcal{M}_{i,t} = \begin{cases} \mathcal{T}_{i,t} & \text{if} \quad \mathcal{T}_{i,t} > c_{\text{Local}}, \\ 0 & \text{otherwise} \end{cases}$$

Here, "0" indicates that no transmission occurs. Therefore, c_{Local} controls the frequency of transmission. We illustrate the transmission regime with hard thresholding of the contribution of each test statistic to the SUM statistic we define below. However, we should highlight that other regimes, such as soft thresholding, could also be employed. Our experience suggests that the form of thresholding has little impact on the properties of the method.

Global monitoring. Once the message is sent to the global centre, we monitor both the sum and maximum of the received test statistics, $\mathcal{T}^{\text{SUM}}$ and $\mathcal{T}^{\text{MAX}}$. At time t, these are defined as:

$$\mathcal{T}_t^{\text{SUM}} = \sum_{i=1}^{d} \mathcal{M}_{i,t}, \qquad \text{and} \qquad \mathcal{T}_t^{\text{MAX}} = \max_{1 \le i \le d} \mathcal{M}_{i,t}.$$

The decision rule at the global level is defined to be $\mathbb{1}\left\{\mathcal{T}_t^{\text{SUM}} > c_{\text{Global}}^{\text{SUM}}\right\} \vee \mathbb{1}\left\{\mathcal{T}_t^{\text{MAX}} > c_{\text{Global}}^{\text{MAX}}\right\}$, where $c_{\text{Global}}^{\text{SUM}}$ and $c_{\text{Global}}^{\text{MAX}}$ are chosen to control the average run length (ARL), i.e., the average stopping time when there is no change. The MAX procedure can also be applied at the edge instead of the centrally. The whole algorithm is described in Algorithm 9, whilst guidance on the choice of thresholds is provided in Sections 4.3.3 and 4.3.4.

Algorithm 8: mixFOCuS

input : historic data $x_{i,t}$ for $i = 1, 2, ..., d$, and $1 \leq t \leq m$

while *the algorithm is not stopped* **do**
  Local monitoring
  for $i = 1$ **to** d **do**
    Calculate the local likelihood ratio test statistic $\mathcal{T}_{i,t}$ for i-th data stream at time t as Formula 4.2
  end
  Message passing
  if $\mathcal{T}_{i,t} > c_{Local}$ **then**
    hard-thresholding: $\mathcal{M}_{i,t} = \mathcal{T}_{i,t}$ // centralized scheme:set $c_{\text{Local}} = 0$
    else $\mathcal{M}_{i,t} = 0$ // No message sent;
  end
  if $\mathcal{M}_{i,t} > 0$ **then**
    send message to the central cloud
  end
  Global monitoring
  if $\sum_{i=1}^{d} \mathcal{M}_{i,t} > c_{\text{Global}}^{\text{SUM}}$ *or* $\max_{1 \leq i \leq d} \mathcal{M}_{i,t} > c_{\text{Global}}^{\text{MAX}}$ **then**
    alarm and stop
    else
      $t \longleftarrow t + 1$
    end
  end
end

4.3.3 The choice of the local threshold

Let us assume that d data streams are observed, and denote the average fraction of sensors sending messages as $\Delta \in [0, 1]$ at each time step. Ideally, $d\Delta$ should be smaller than ω, where ω is the pre-specified average number of transmissions allowed at any time instance. Δ represents the probability that a test statistic exceeds the pre-specified threshold. Local test statistics are often modeled as asymptotically following a Gumbel distribution (Yao and Davis, 1986). However, the rate convergence to the Gumbel distribution can be slow in practice (Fearnhead and Fryzlewicz, 2022).

In practice, one straightforward approach to choosing the local threshold is to approximate the average transmitted fraction $\hat{\Delta}$ using Monte Carlo simulations under the null hypothesis, assuming a fixed monitoring time, γ. Specifically, for a given distribution $f(x|\theta)$, we generate r sequences and record the test statistics. The threshold that gives $\hat{\Delta}$ is approximately the $(1 - \Delta)$-th quantile of ascending-ordered sequence

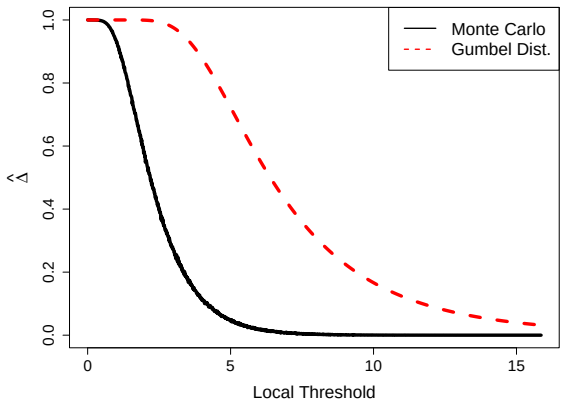


Figure 4.2: The numerical transmission frequency of local test statistic for simulated standard Gaussian distribution based on 1000 replications with $\gamma = 10000$.

$(\mathcal{T}^1_{i,1:\gamma}, \mathcal{T}^2_{i,1:\gamma}, ..., \mathcal{T}^r_{i,1:\gamma})$, where $\mathcal{T}^r_{i,1:\gamma}$ is the local test statistics for the r-th replication with data length γ. Figure 4.2 displays the approximated empirical $\hat{\Delta}$ values and their corresponding local thresholds obtained through Monte Carlo simulation and theory. We note that the thresholds suggested by the theory are too conservative.

4.3.4 The choice of the global threshold

We next turn to consider the question of how to choose the global threshold. Our approach is based on the average run length criteria, i.e., the expected time until the first false alarm approximately equal to the predetermined average run length (ARL) γ, under the null hypothesis of no change. To this end let $\kappa^{\text{SUM}} = \inf\{t : \mathcal{T}^{\text{SUM}}_t > c^{\text{SUM}}_{\text{Global}}\}$, $\kappa^{\text{MAX}} = \inf\{t : \mathcal{T}^{\text{MAX}}_t > c^{\text{MAX}}_{\text{Global}}\}$, and $\kappa^{\text{comb}} = \min\left\{\kappa^{\text{SUM}}, \kappa^{\text{MAX}}\right\}$ be the time to detection under the null hypothesis. In an adaptive procedure, two global thresholds need to be chosen such that $E^{\infty}(\kappa^{\text{comb}}) = \gamma$, where $E^{\infty}(\cdot)$ is the expectation when there is no change.

To avoid the computational cost tuning both the thresholds, $c^{\text{SUM}}_{\text{Global}}$ and $c^{\text{MAX}}_{\text{Global}}$, simultaneously we suggest tuning each in turn. Previous empirical results (Chen et al., 2022) suggest that the time to detection for online changepoint is approximately ge-

ometrically distributed. Empirically we see this is approximately true for our setting (see Appendix B.1) for both the MAX and SUM tests. If we have two independent geometrically distributed random variables with mean $(1/p)\gamma$ and $(1/(1-p))\gamma$ respectively, for some $\gamma > 0$ and probability $0 < p < 1$, then distribution of the minimum has mean γ and the probability that the minimum is attained by the first is $\approx p$.

Whilst the time to detection for our two test statistics is not independent, we can use this result to guide our choice of threshold. That is we can choose our require ARL, γ, and the preference we have for the SUM test, specified by a probability p. We then tune $c^{\text{SUM}}_{\text{Global}}$ so that the SUM test has ARL $(1/p)\gamma$. We then tune $c^{\text{MAX}}_{\text{Global}}$ so that the overall procedure has ARL γ. In most applications we would give equal weight to the two tests, so $p = 1/2$. Our final procedure would have the correct the ARL of γ due to how we tune the second threshold - and the approximation of the time to detection of the tests by independent geometric distributions will only affect the relative priority given to the two tests. This procedure is described pictorially in Figure 4.3.

To tune each of the thresholds we use Monte Carlo. Our approach is to simulate R replicates of data sets of length n, each simulated with no change. Let T_r, for $r = 1, \ldots, R$, denote the time at which a change is detected, with $T_r = n$ if no change is detected. Further let I_r indicate whether a change is detected in the rth data set, so $I_r = 1$ if a change is detected and $I_r = 0$ otherwise. Then the we can estimate the ARL as the MLE under a geometric distribution of the the time to detection, this gives an estimate that is $\sum_{r=1}^{R} T_r / \sum_{r=1}^{R} I_r$. We can then tune the threshold until this estimate the smallest value greater than the required value. If we choose n to be much larger than the required ARL then in all or almost all data-sets for the chosen threshold we will have detected a change – making our estimate of the ARL robust to the assumption of a geometric distribution for the time to detection. Furthermore, we can make the search for the threshold efficient by running the algorithm on each data-set once and storing the test statistic values at each time. We can the easily post-process these values

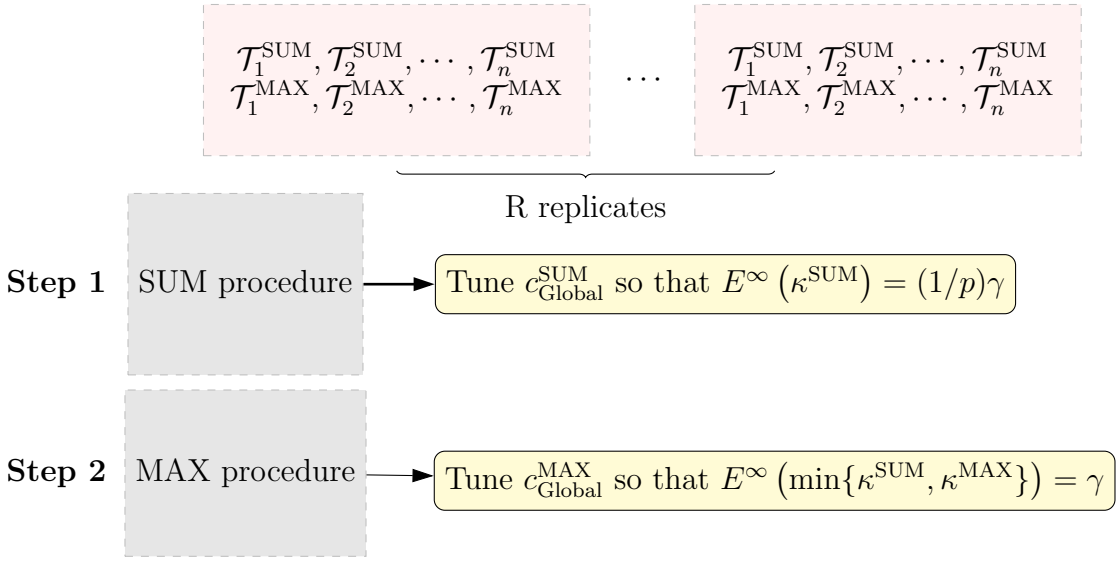


Figure 4.3: An example diagram illustrating the procedure for determining global thresholds to achieve a target ARL of γ. We begin by simulating R $n \times d$ datasets under the null, where $n \gg \gamma$. For each dataset, we compute and store global test statistics at every time point. We first isolate the SUM procedure and tune its global threshold $c_{\text{Global}}^{\text{SUM}}$ such that the ARL satisfies $E^{\infty}(\kappa^{\text{SUM}}) = \gamma/p$. We then tune the threshold of MAX procedure $c_{\text{Global}}^{\text{MAX}}$ based on the combined test statistics, so that $E^{\infty}\left(\min\{\kappa^{\text{SUM}}, \kappa^{\text{MAX}}\}\right) = \gamma$. The order of the two steps can be swapped.

to evaluate T_r and I_r, for $r = 1, \ldots, R$, as we vary the threshold.

The MAX procedure can also be applied locally where each data stream has different thresholds. This can be useful when data streams are of different qualities. For instance, one might need a larger threshold for a sensor with higher auto-correlation. In such cases, different MAX thresholds can be selected for individual sequences. The tuning procedure remains the same as described earlier. For simplicity, we consider the same MAX threshold case in the following simulation study.

4.4 Simulation results

In this section, we begin by evaluating the performance of our proposed approach under restricted transmission frequencies, on both homogeneous and mixed-type data. We also show the loss of detection power and inflation in false alarms that can arise from

Setting	Index of data streams	Pre-change distribution	Post-change distribution
Homogeneous data	1-100	$N(0,1)$	$N(\delta,1)$
Mixed-type data	1-20	$N(0,1)$	$N(\delta,1)$
	21-40	Ber(0.4)	Ber($\delta\sqrt{0.4(1-0.4)}+0.4$)
	41-60	Pois(5)	Pois($\delta\sqrt{5}+5$)
	61-80	Exp(1/3)	Exp($\frac{1}{3\delta+3}$)
	81-100	Gamma(2,3)	Gamma(2,$\frac{2}{\delta\sqrt{2/3^2+2/3}}$)

Table 4.2: Set-up of simulated data.

approximating mixed-type data using a Gaussian distribution, a common assumption in previous studies. This highlights the benefit of using a mixed distribution approach. Finally, we present a comparison between our proposed mixed method and existing approaches on Gaussian data.

4.4.1 Detection power of mixFOCuS

To evaluate the performance of mixFOCuS with different transmission constraints, we consider simulating data with scenarios $S1/S2$: pre-change known/unknown, the sparsity level $p = 1, 0.5, 0.1, 0.01$, the fixed magnitude of change $\delta_i = \delta = 0.1, 0.25, 0.5, 1$, and the change happened at the same time $\tau = 1000$ or 3000 for affected $p \times d$ sensors. The parameters of homogeneous and mixed-type data are listed in Table 4.2. Under the alternative, to make sure each distribution in the mixed-type data can have the same size of change, we keep the ratio of the change in mean (post-change mean minus pre-change mean) to the standard deviation constant for all distributions. For normal/gamma distribution, the standard deviation/shape remains constant, and the change solely occurs in its mean/scale.

Both local and global thresholds are chosen based on 1000 Monte Carlo experiments. Local thresholds $\{0, 4.18, 4.94, 6.77\}$ and $\{0, 4.85, 5.62, 7.37\}$ are selected to limit the frequency at levels $\hat{\Delta} = \{100\%, 10\%, 5\%, 1\%\}$ in the pre-change known and unknown case respectively, and global thresholds are chosen to achieve $E^{\infty}\left(\kappa^{\text{MAX}}\right) = 20000$ and

$E^{\infty}\left(\kappa^{\text{comb}}\right) = 10000$. To evaluate the impact of limiting transmission, we will measure the false alarm rate (FAR), average detection delay (ADD) and proportion of missed alarms (the percentage of times the algorithm failed to raise an alarm when there was a change) with the benchmark being the case without any transmission constraints (100% transmission) under the alternative.

Table 4.3 and 4.4 present a summary of the detection power of mixFOCuS under different transmission constraints when the change is located at 3000. Notably, our proposed method exhibits slightly better detection power when the change is not located at the beginning of the data set when the change is small. Specifically, at location of 3000, there is a significantly reduced detection delay compared to the results obtained when the change is located at 1000, especially in the pre-change unknown case (The result of $\tau = 1000$ is shown in the Appendix B.2). This is because FOCuS algorithms estimate unknown parameters as more data comes in, more data can lead to more accurate estimators.

Generally speaking, compared with the benchmark, imposing restrictions on transmission frequency leads to a slightly longer detection delay in the dense case but a smaller delay in the sparse case. Limiting transmission can lead to increasing false alarms, but most time will still give a similar detection delay as the benchmark. When encountering a large mean shift ($\delta \geq \sigma$), all the methods may demonstrate similar detection power as the benchmark.

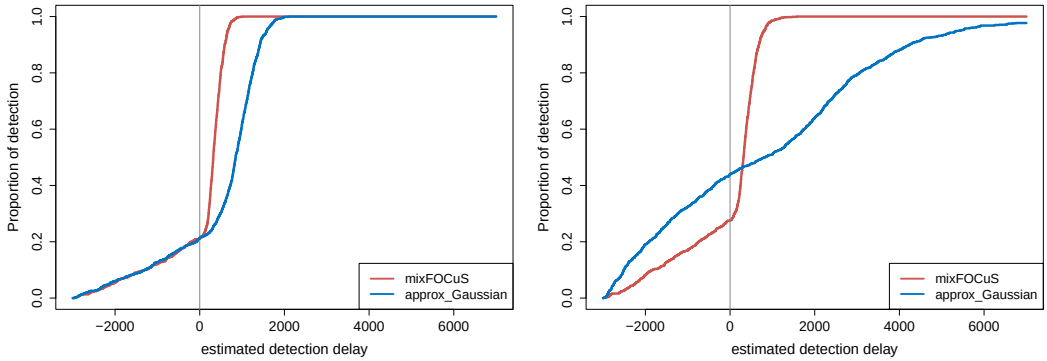
Compared with pre-change known cases, there will be a decrease in detection power in the unknown case. This is because the process of estimating the pre-change parameter could introduce bias.

Scenarios		pre-change known(S1)				pre-change unknown(S2)			
Transmission Constraint		100%	10%	5%	1%	100%	10%	5%	1%
FAR(%)		10.8	12.4	12.7	16.5	13.1	16.8	20.4	23.3
p	δ	ADD				ADD			
1	0.1	**118**	119	127	157	144	**143**	154	197
	0.25	**24**	25	27	33	**28**	29	31	38
	0.5	**7**	8	9	11	**8**	9	10	12
	1	**3**	**3**	**3**	4	**3**	**3**	**3**	4
0.5	0.1	213	**204**	210	242	259	**244**	251	301
	0.25	**43**	**43**	**43**	50	49	**47**	48	55
	0.5	**13**	**13**	**13**	15	**14**	**14**	15	16
	1	**4**	**4**	**4**	5	**4**	5	5	5
0.1	0.1	741	655	618	**608**	994	841	**805**	808
	0.25	152	138	129	**122**	160	145	137	**129**
	0.5	43	41	38	35	44	40	38	36
	1	12	12	11	**10**	12	12	11	**10**
0.01	0.1	2295	2199	2151	**2006**	3194 (20.2%)	2925 (17.1%)	**2818** (17.2%)	2867 (12.3%)
	0.25	420	416	413	**405**	501	487	485	**469**
	0.5	**110**	**110**	**110**	**110**	114	114	113	**112**
	1	**29**	**29**	**29**	**29**	**29**	**29**	**29**	**29**

Table 4.3: Results on the homogeneous Gaussian data are averaged over 1000 replications. The smallest ADD in each scenario is given in bold. The percentage within the bracket represents the proportion of missed alarms.

Scenarios		pre-change known(S1)				pre-change unknown(S2)			
Transmission Constraint		100%	10%	5%	1%	100%	10%	5%	1%
FAR(%)		12	13.1	14.7	19.6	12.6	17.8	16.9	27
p	δ	ADD				ADD			
1	0.1	**125**	126	132	157	**154**	**154**	167	204
	0.25	**26**	27	28	34	**30**	31	33	40
	0.5	**8**	9	9	11	**9**	10	11	12
	1	**3**	**3**	4	4	3	4	4	5
0.5	0.1	218	**210**	215	246	278	**261**	275	310
	0.25	46	**45**	46	51	53	**51**	53	59
	0.5	**14**	**14**	**14**	16	**16**	**16**	**16**	18
	1	**5**	**5**	**5**	6	**5**	**5**	6	6
0.1	0.05	763	677	637	**622**	1070	900	889	**856**
	0.25	164	150	140	**130**	179	161	154	**145**
	0.5	47	44	41	**38**	48	45	43	**40**
	1	14	14	13	**12**	14	14	13	**12**
0.01	0.1	2330	2231	2135	**1985**	3255 (16.9%)	2900 (14.2%)	2905 (16.9%)	**2769** (8.9%)
	0.25	421	417	415	**405**	497	489	486	**466**
	0.5	109	108	108	**107**	112	112	112	**110**
	1	29	29	29	**28**	**29**	**29**	**29**	**29**

Table 4.4: Average detection delay on mixed-type data against the levels of sparsity and the strengths of the signal. Results are averaged over 1000 replications. The smallest ADD in each scenario is given in bold. The percentage within the bracket represents the proportion of missed alarms.



(a) $\delta = 0.25, p = 0.01$ pre-change known (b) $\delta = 0.25, p = 0.01$ pre-change unknown

Figure 4.4: The proportions of experiments where a change was detected by time step $t - \tau$. Data are generated with $p_{ber} = 0.4, \lambda_{pois} = 5, \lambda_{exp} = \frac{1}{3}, \beta = 2, n = 10000$ and $\tau = 3000$. Results are obtained over 1000 repetitions. In cases where the pre-change distribution is known, data streams are normalized based on their theoretical mean and variance of the true distribution. When the pre-change distribution is unknown, the mean and variance are estimated from the training dataset.

4.4.2 Assessing detection power with true distribution versus Gaussian approximation

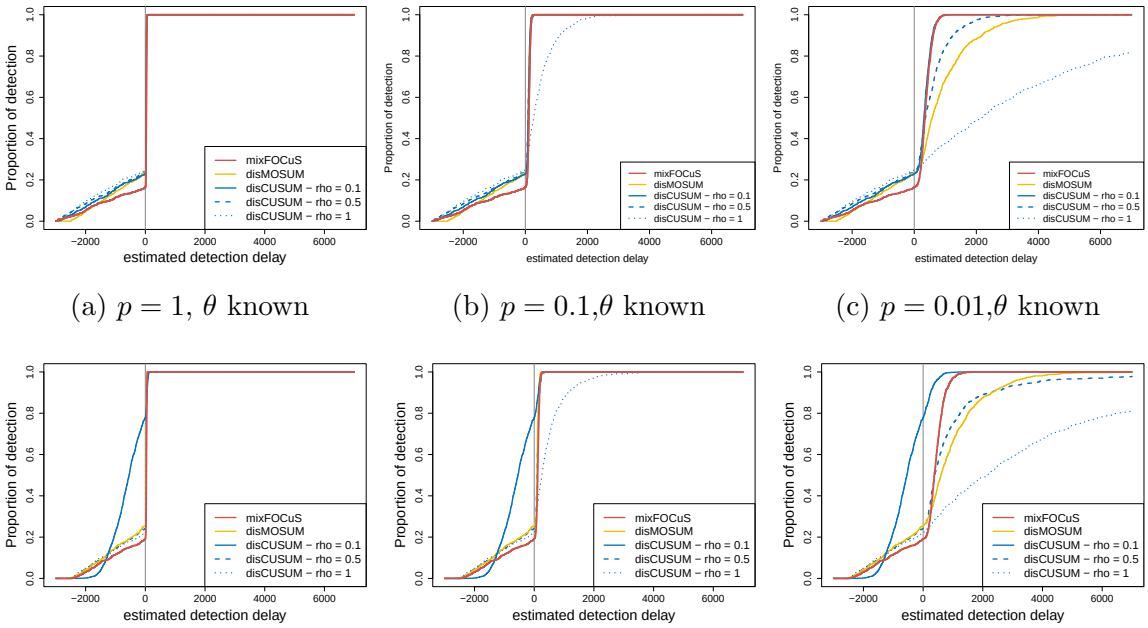
As previously mentioned in Section 1, existing methods typically assume that the data follows Gaussian distribution, which may not be realistic in the real world. One possible (naïve) solution of addressing this might be to approximate data using a Gaussian distribution. We next explore if our proposed method, which uses the true distributions, significantly improves detection power compared with this approximation approach. As we shall see, in dense cases both methods show similar detection delays, but mixFOCuS has fewer false alarms (see Appendix B.3). The improvement is more noticeable in sparse cases (Figure 4.4). When the pre-change distribution is known, both approaches may have similar false alarm rates, but the Gaussian-approximation approach leads to longer detection delays. In cases with unknown pre-change parameters, our proposed method can result in lower false alarms and smaller detection delays, while the Gaussian approximation approach can fail to detect the change.

4.4.3 Comparing with the current state of the art

We next turn to compare our proposed method, mixFOCuS, with two recently proposed communication efficient changepoint algorithms: disCUSUM (Liu et al., 2019) and disMOSUM (Yang et al., 2024). Within the simulations we consider Gaussian data with parameters $n = 10000, d = 100, \tau = 3000, \delta = 0.25$. Both competitor approaches, disCUSUM and disMOSUM, are similar in spirit to mixFOCuS, but they run CUSUM (with the recursive register approach of Lorden and Pollak (2008) to approximate the post-change mean) or MOSUM on each data stream. For each data stream, both methods are assumed to follow a Gaussian assumption. In the centre, both methods only monitor the sum of test statistics.

To compare fairly, thresholds in each method are chosen to achieve similar average run length $\gamma = 10070$ based on Monte Carlo simulation. The detailed tuning procedure is explained in the Appendix B.4. In the pre-change unknown case, disCUSUM and disMOSUM will use the first m observations to estimate the mean and variance. To keep the result simple, we only consider the case when the transmission constraint is 1%. Under the alternative, the stopping time for each replication is recorded and we will measure the distribution of the detection delay $(\hat{\tau} - \tau)$. A negative detection delay indicates a false alarm, while a positive detection delay shows a delay in raising the alarm. In general, a method with smaller false alarms and lower detection delay will be preferred. For simplicity, we only show the case when the transmission constraint is 1%.

Figure 4.5 illustrates that mixFOCuS exhibits robustness across various scenarios and consistently demonstrates good performance. In dense cases, disMOSUM and disCUSUM may have similar detection delay as mixFOCuS when an appropriate window size or possible changes ρ is chosen, but they have high false alarms. Notably, mixFOCuS outperforms the other methods, particularly in sparse change scenarios or when the pre-change mean is unknown. It is worth noting that disCUSUM can lose power to



(a) $p = 1$, θ known (b) $p = 0.1$,θ known (c) $p = 0.01$,θ known

(d) $m = 500, p = 1$, θ unknown (e) $m = 500, p = 0.1$, θ unknown (f) $m = 500, p = 0.01$, θ unknown

(g) $m = 100, p = 1$, θ unknown (h) $m = 100, p = 0.1$, θ unknown (i) $m = 100, p = 0.01$, θ unknown

Figure 4.5: The $x-$axis represents the detection delay $(\hat{\tau} - \tau)$, and the $y-$axis represents the cumulative percentage across 1000 repetitions. The grey line represents a detection delay of 0, with lines to the left indicating false alarm rates. A faster convergence of the lines to the right of the grey line towards 1, indicates a quicker detection.

detect changes in the pre-change unknown case when ρ is misspecified. This is because ρ estimates the possible change. A larger estimator or smaller estimator can lead to an underestimated global threshold under the null, resulting in more false alarms under the alternative.

4.4.4 Detection power of mix-FOCuS on time series data

We now investigate the detection capability of mix-FOCuS when the data streams are auto-correlated. There are two general approaches for applying mix-FOCuS in such a setting. The first is to inflate the thresholds, or equivalently normalise the test statistics, to take account of the the auto-correlation. The alternative is to pre-process the data to remove the auto-correlation.

We will take the second approach, and consider an AR(1) noise process, defined as follows:

$$X_{i,t} = \delta_i \mathbf{1}_{[t>\tau]} + \epsilon_{i,t},$$

where δ_i is the change size for ith data, $\epsilon_{i,t}$ is the noise term modelled as an AR(1) process:

$$\epsilon_{i,t} = \phi\epsilon_{i,t-1} + v_t,$$

with ϕ ($|\phi| < 1$) being the autoregressive coefficient that determines the strength of the temporal correlation, and $v_t \sim N(0,1)$ representing a Gaussian white noise process. If we consider data $\tilde{X}_{i,t} = X_{i,t} - \phi X_{i,t_1}$, then

$$\tilde{X}_{i,t} = \begin{cases} v_t & \text{if } t \leq \tau, \\ \delta_i + v_t & \text{if } t = \tau + 1, \\ (1-\phi)\delta_i v_t & \text{if } t > \tau + 1 \end{cases},$$

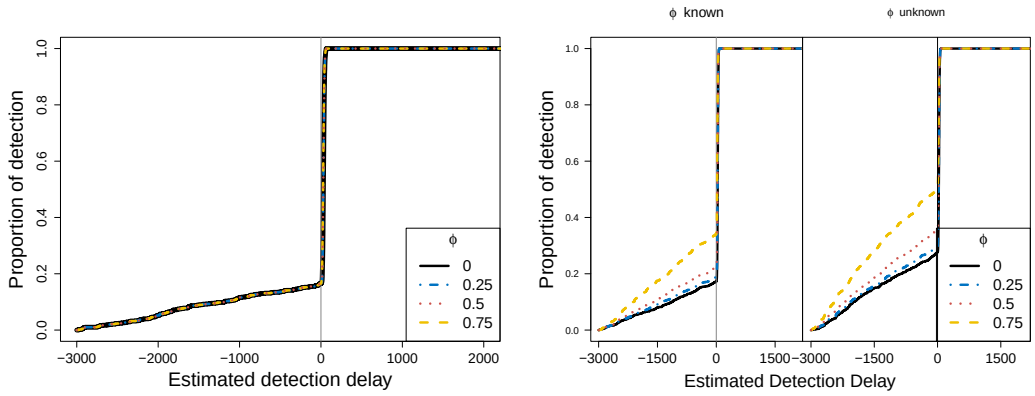
where the noise v_t is uncorrelated. We can see that using mixFOCuS is this transformed data is an approximation as the post-change mean is different at $t = \tau + 1$ than at

$t > \tau + 1$, where as mixFOCuS assumes a constant post-change parameter. Also, we see the affect that the level of autocorrelation, as determined by ϕ, will have on the evidence for a change – to maintain roughtly similar power we would require δ_i to increase as ϕ gets closer to 1 so that $\delta_i(1-\phi)$ remains constant.

Figure 4.6 measures the performance of mixFOCuS while maintaining an average run length $\gamma = 10000$, with the change occurring at 3000, transmission constraints set at 1%. We compare across different values of ϕ, and we set a change magnitude of $\delta = \frac{0.25}{1-\phi}$, so there is similar evidence for a change across different scenarios. We consider two cases: (i) the pre-change parameter and the autocorrelation coefficient are assumed to be known; and (ii) these are unknown with the autocorrelation coefficient estimated from m training data. The results are shown in Figure 4.6. When the autocorrelation coefficient is known, detection power remains comparable across all levels of autocorrelation. However, in the unknown case, bias in the estimated autocorrelation coefficient leads to an increased false alarm rate, and this is more pronounced as ϕ increases. Therefore, in practice, having a large training dataset for accurate autocorrelation estimation is crucial for maintaining detection power.

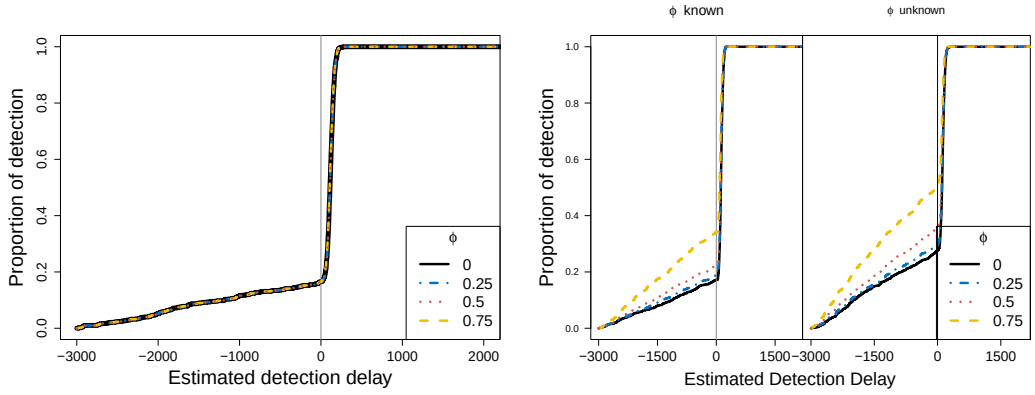
4.5 Skoltech anomaly benchmark

The Skoltech anomaly benchmark (Katser and Kozitsin, 2020) is designed to evaluate outlier detection and changepoint detection algorithms. This dataset includes measurements of two vibration accelerations, electric motor current, water pump pressure, engine temperature, fluid thermocouple readings, electric motor voltage, and fluid circulation rate. We present results for the first dataset from the `valve2` file, which captures an experiment where there is an anomalous region that relates to the closure of a pump valve. From this we extract two segments of data that contain a change: one corresponding to the start and one to the end of the anomalous region. Data from



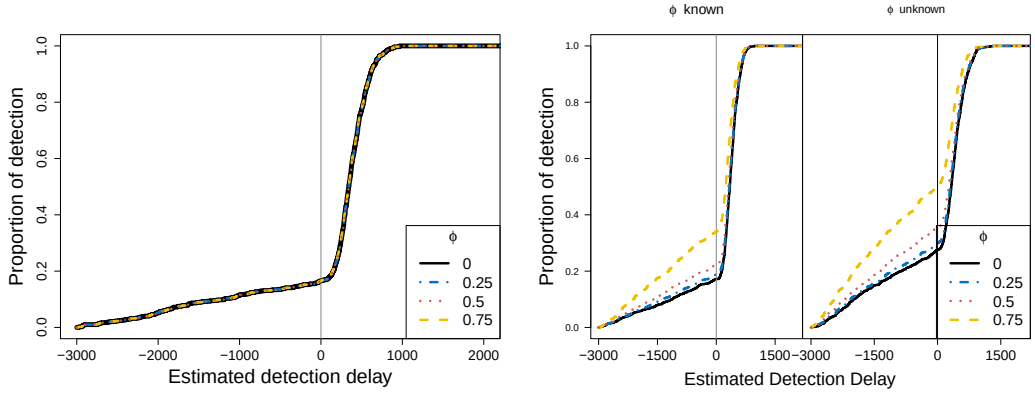
(a) $p = 1$, (ϕ, θ) known

(b) $m = 500, p = 1$, θ unknown



(c) $p = 0.1$,(ϕ, θ) known

(d) $m = 500, p = 0.1$, θ unknown



(e) $p = 0.01$,(ϕ, θ) known

(f) $m = 500, p = 0.01$, θ unknown

Figure 4.6: The $x-$axis represents the detection delay $(\hat{\tau} - \tau)$, and the $y-$axis represents the cumulative percentage across 1000 repetitions. The grey line represents a detection delay of 0, with lines to the left indicating false alarm rates. A faster convergence of the lines to the right of the grey line towards 1, indicates a quicker detection.

these two segments for the eight sensors are shown in Figure 4.7.

Before applying mixFOCuS we preprocess the data. Pump pressure and electric

motor voltage have values concentrated around three distinct states, we categorize them into three groups: low, middle, and high. For both series we then create three series of binary indicators, indicating whether, or not, the series is in each of the three groups respectively. We standardize the remaining continuous variables, take differences to reduce the impact of the autocorrelation and model them as Gaussian. To account for any remaining auto-correlation we inflate the local penalties for each series based on an estimate of the autocorrelation, as suggested in Fisch et al. (2022b). Finally, the global thresholds are selected to ensure that the average run length exceeds the monitoring time of the test dataset.

We train the model on the initial 300 data points from the normal segment to establish a baseline before the onset of anomalies. Upon detecting a change, we retrain the model on the 100 data points from the anomalous segment to learn the new behaviour and monitor the end of this anomalous segment. Figure 4.7 shows the detection procedure of the algorithm. At the start of the anomalous segment, the signal is strongly reflected on Sensor 1 (vibration acceleration sensor). Algorithms with 100% and 5% transmission alarm simultaneously at the true change time. At the end of the change, the flow rate returns to a high level, while the pressure remains stable. Models with 100% and 5% transmission experience a delay of 4 and 6 time steps (4 and 6 seconds). Overall, both approaches achieve the similar detection performance; however, the latter significantly reduces transmission costs.

4.6 Conclusions

In this article we have proposed a flexible, computational-feasible and communication-efficient online changepoint detection method that can be used to detect changes in a distributed system for data from the exponential family. In practice, data may show cross-correlation, which violates the independent assumptions. Future research could

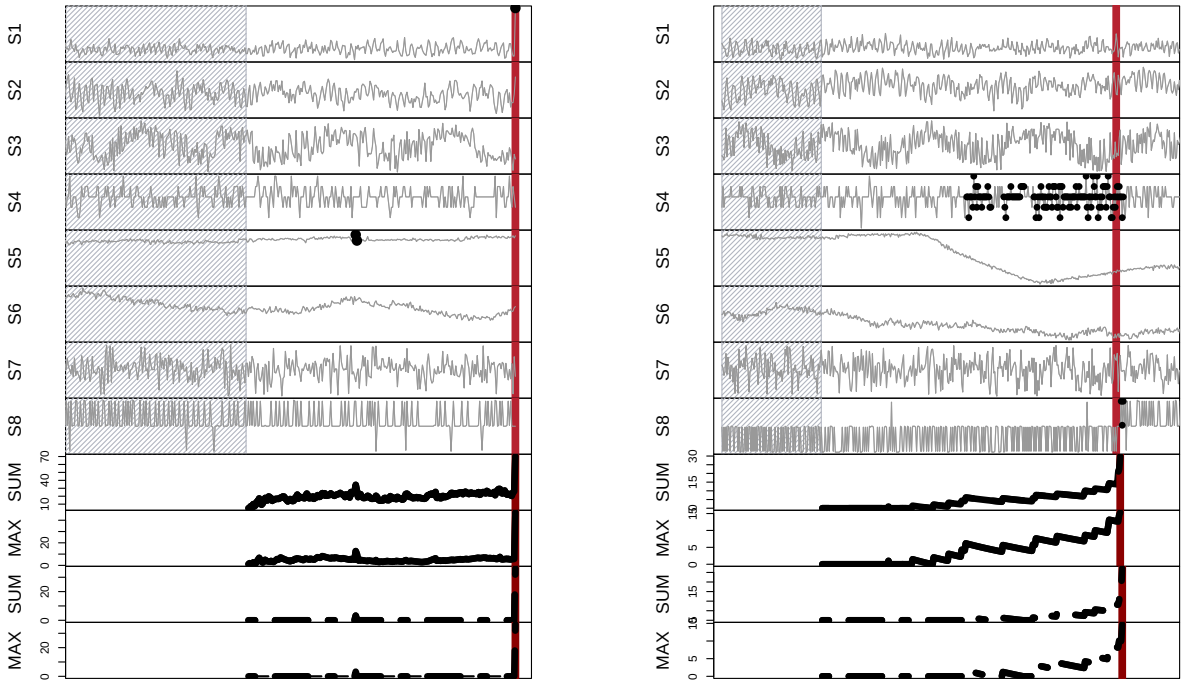


Figure 4.7: Detection results for identifying the start (left) and the end (right) of the change under 100% and 5% transmission constraints are presented. Black dots indicate the time points when the sensor transmits test statistics for 5% transmission model. The monitored global test statistics are displayed in the 9th and 10th rows for the 100% transmission model and in the 11th and 12th rows for the 5% transmission model. The true change time is represented by the light red line, while the dark red line in the bottom four rows indicates the detected change time identified by mixFOCuS with the two levels of transmission.

explore the consequences of such violations and potentially consider relaxing the assumptions.

Chapter 5

Bagel: A Fast Bayesian Online Changepoint Detection Algorithm for Linear Models

5.1 Introduction

Detecting changes in real time is a common problem. This has been well studied in the frequentist world, see for example (Page, 1954, 1955; Aue et al., 2012; Kirch and Weber, 2018; Romano et al., 2023b; Ward et al., 2023, 2024; Yu et al., 2020). Whilst these approaches often work well, they do not quantify uncertainty or allow the inclusion of prior information about the type of change. To provide extra information about the change, it is intuitive to consider a Bayesian framework as it can incorporate prior knowledge and produces the posterior distribution which quantifies the uncertainty of the change.

Most Bayesian approaches to the changepoint detection problem consider the offline setting, where we have the whole data and have to infer the location of changepoints or the joint distribution of the number of changepoints and their locations. Existing

approaches are either based on the Markov Chain Monte Carlo (Stephens, 1994; Green, 1995; Chib, 1998; Benson and Friel, 2018), or based on the direction simulation (Barry and Hartigan, 1993; Fearnhead, 2006; Fearnhead and Liu, 2011). These approaches have been used for modelling disease outbreak (Verma et al., 2020), financial volatility (Ross, 2013; Thies and Molnár, 2018), environmental data (Kim and Cheon, 2010) amongst many other applications.

There has also been work on online formulations in models that allow multiple changes. In this case, under the assumption that the parameters specifying the model for data in one segment do not depend on the parameters for other segments, there are efficient algorithms for exactly calculating the posterior. These have a computational cost that increases linearly with time – so processing the observation at time T has an $O(T)$ cost. This cost can be reduced to $O(1)$ by introducing an appropriate approximation (Fearnhead and Liu, 2007).

In this paper, we consider the case of online detection of a single changepoint. We derive efficient algorithms for calculating the posterior distribution in this case under a wider range of models than previous works – for example, including the problem of detecting a change in slope, when the post-change parameters of the model depend on the pre-change parameters. Our general framework encompasses any model that can be written as a change in regression: which allows for detecting changes in mean, slope, slope with seasonality, or splines amongst many others.

The exact algorithm we have has a linear-increasing cost, which makes it unsuitable for long-term or real-time monitoring. To reduce the time complexity, existing approaches include pruning the least probable changepoints (Adams and MacKay, 2007) or using resampling to prune the set of potential changepoints that are being considered (Fearnhead and Liu, 2007). Instead, we proposed approximations based on the merging idea of Shamp et al. (2021) to reduce the cost per-iteration to constant. Our merging procedure takes account of the difference in the conditional distribution of the

parameters given the change location into account, and is amenable to the wider range of changepoint models we consider.

The outline of the paper is as follows. In Section 2 we define the Bayesian real-time changepoint detection problem and introduce our proposed algorithm with two examples. Section 3 and Section 4 show the performance of our algorithm on simulated data and a real dataset. More examples, and all proofs, can be found in the Appendix. Throughout this paper, all vectors are assumed to be column vectors unless explicitly stated otherwise. We denote by $1:t$ the column vector $(1, \cdots, t)^\top$, $1_{1:t}$ and $0_{1:t}$ are column vectors of length t of ones and zeros respectively, and $y_{1:t}$ represents the column vector $(y_1, \cdots, y_t)^\top$. The probability density function (pdf) of the normal distribution with mean μ and variance σ^2 is denoted by $N(x; \mu, \sigma^2)$, the pdf of the Student-t distribution with ν degrees of freedom, location l, and scale e is denoted by $t_\nu(x; l, e)$, and the pdf of the Inverse-Gamma distribution with shape ν and scale ι is denoted by $IG(x; \nu, \iota)$. For a matrix A, we use $A_{i:j \times k:l}$ to denote the sub-matrix formed by rows i to j and columns k to l.

5.2 Univariate real-time Bayesian changepoint detection

5.2.1 The changepoint problem

The model

Assume an independent data stream $y_1, y_2, \ldots, y_t$ is observed in real-time, we wish to detect the presence, or not, of a changepoint at each time step. More precisely, at a given time t, we wish to detect whether there has been a change prior to t under an appropriate model for the data. We will consider a class of linear models that encompasses the standard change-in-mean and change-in-slope problem, as well as more complex models

that allow for incorporating seasonality or autocorrelation as in Robbins et al. (2016).

Let τ be the unknown time of the changepoint, where $\tau = \infty$ denotes no change. The observed data y_t at any time t is modeled as:

$$y_t = a_t^\top \beta + b_{t,\tau}^\top \gamma + \sigma \epsilon_t.$$

Here a_t and $b_{t,\tau}$ are known d_1 and d_2 dimensional vectors, while β and γ are unknown d_1 and d_2 dimensional parameters. The term $a_t^\top \beta$ gives the mean of the model if there has been no change prior to time t, while $b_{t,\tau}^\top \gamma$ specifies the change in this mean due to a changepoint. To be consistent with this we have $b_{t,\tau}^\top = 0_{1:d_2}$ if $\tau \geq t$. Finally, ϵ_t, is the realisation of a standard Gaussian random variable, and $\sigma > 0$ is the standard deviation of the noise in the model.

Consequently, at any time t, the model for the data up to time t, $y_{1:t}$, can be written as:

$$y_{1:t} = A_{1:t}\beta + B_{1:t,\tau}\gamma + \sigma \epsilon_{1:t},$$

where $\epsilon_{1:t}$ is a column vector of i.i.d Gaussian $N(0, 1)$ realisations, and the tth rows of matrices $A_{1:t}$ and $B_{1:t,\tau}$ are $a_t^\top$ and $b_{t,\tau}^\top$ respectively. If $\tau \geq t$ then $B_{1:t,\tau}$ is the zero matrix. Since τ is unknown, at each time t we consider t potential models: one corresponding to no change and others where change occurs at $\tau = 1, 2, ..., t-1$. Our task is to determine whether there is significant evidence supporting the presence of a change and to infer the posterior distribution of the location of any change.

The prior for the model parameters

The likelihood of the observation depends on the parameters (β, γ), if the variance of the observations is known, or (β, γ, σ) otherwise. We take a Bayesian approach and assume a prior for these parameters. Furthermore, to enable efficient calculation of the posterior for the presence and location of a changepoint we will use conjugate priors.

For our model, the conjugate priors are Gaussian for (β, γ) conditional on σ, with, if it is unknown, an independent inverse gamma prior for σ. We will have fixed priors for β (and if unknown, σ), but we will allow the conditional prior for γ to potentially depend on both τ and β. Specifically, the joint prior for (β, γ) given σ and τ is

$$\beta, \gamma | \sigma^2, \tau \sim N(\mu^\tau, \sigma^2 \Sigma^\tau)$$

where mean $\mu^\tau \in \mathbb{R}^{d_1+d_2}$ and scaled covariance matrix $\Sigma^\tau \in \mathbb{R}^{(d_1+d_2)\times(d_1+d_2)}$. If σ is unknown, the prior is given by:

$$\sigma^2 \sim IG(\nu, \iota).$$

The constraint on the having a fixed prior for β given σ means that $\mu^\tau_{1:d_1}$ and $\Sigma^\tau_{1:d_1 \times 1:d_1}$, that is the marginal prior mean and variance of β, are constant as we vary τ.

While it is often possible to use improper priors for β and σ - the conditional prior for γ must be proper in order to have a well-defined posterior probability for whether there is a changepoint prior to any given time t. The choice of the prior for γ will affect the power of detecting different sizes and types of change.

Here we give two examples to illustrate our model framework:

Example 1. Change-in-mean model.

We first consider the change-in-mean model where $d_1 = d_2 = 1$. The model can be written as :

$$A_{1:t} = 1_{1:t}, B_{1:t,\tau} = \begin{bmatrix} 0_{1:\tau} \\ 1_{\tau+1:t} \end{bmatrix}.$$

Here β is the mean before the change, γ represents the shift in the mean after the changepoint. There are two natural choices for the prior distribution in this setting: one where the pre-change mean and the change in mean are independent, and another where the pre-change and post-change means are independent.

For the former case we have $\mu = \begin{bmatrix} \mu_1 \\ 0 \end{bmatrix}$ and $\Sigma = \begin{bmatrix} \delta_1 & 0 \\ 0 & \delta_2 \end{bmatrix}$. For the latter case we have the same prior mean, but the covariance structure changes to

$$\Sigma = \begin{bmatrix} \delta_1 & -\delta_1 \\ -\delta_1 & 2\delta_1 \end{bmatrix}.$$

See Appendix C.1 for the derivation. This framework can also be extended to the unknown variance case by incorporating an inverse Gamma prior to the variance.

Example 2. Change-in-slope models.

Next, we consider the change-in-slope with a known variance case where $d_1 = d_2 = 2$. The model can be written as:

$$A_{1:t} = [1_{1:t} \quad 1:t], B_{1:t,\tau} = \begin{bmatrix} 0_{1:\tau} & 0_{1:\tau} \\ 1_{(\tau+1):t} & (\tau+1):t \end{bmatrix}$$

Here β defines the linear trend before the change, and γ specifies the shift in the intercept or slope at the changepoint.

There are two natural types of change we can allow in this model, depending on whether or not we require continuity of the trend line at the changepoint, as illustrated in Figure 5.1. If we wish to detect a continuous chance, also known as a change-in-slope (Fearnhead et al., 2019), then a natural model is that the change-in-slope, γ_2, has mean 0 and is independent of the pre-change trend. The distribution of γ_1 conditional on γ_2 is then deterministic due to the continuity constraint. This constraint implies for a change at τ that the trend is continuous at τ: $\gamma_1 + \tau\gamma_2 = 0$. This gives a prior for (β, γ)

that has mean and covariance defined by the parameters:

$$\mu^\tau = (\mu_1, \mu_2, 0, 0)^\top, \text{ and } \quad \Sigma^\tau = \begin{bmatrix} \delta_1 & 0 & 0 & 0 \\ 0 & \delta_2 & 0 & 0 \\ 0 & 0 & \tau^2\delta_3 & -\tau\delta_3 \\ 0 & 0 & -\tau\delta_3 & \delta_3 \end{bmatrix}.$$

For a discontinuous change there are two natural models for the type of change. One is that the change in the intercept and the slope are both mean 0 and independent of the pre-change model, in which case $\mu^\tau = \begin{bmatrix} \mu_1 \\ \mu_2 \\ 0 \\ 0 \end{bmatrix}$ and

$$\Sigma = \begin{bmatrix} \delta_1 & 0 & 0 & 0 \\ 0 & \delta_2 & 0 & 0 \\ 0 & 0 & \delta_3 & \frac{1}{2\tau}(\delta_1 - \delta_3 - \tau^2\delta_4) \\ 0 & 0 & \frac{1}{2\tau}(\delta_1 - \delta_3 - \tau^2\delta_4) & \delta_4 \end{bmatrix}.$$

The other would be to assume that the distribution of the trend line starting at time 0 is independent of and has the same distribution of the trend line starting at time τ. This requires the value of the mean function at τ, $\beta_1 + \gamma_1 + \tau(\beta_2 + \gamma_2)$ has the same distribution as β_1, and the new slope $(\beta_2 + \gamma_2)$ has the same distribution as β_2. This implies a prior with

$$\mu^\tau = (\mu_1, \mu_2, -\mu_2\tau, 0)^\top, \text{ and } \quad \Sigma^\tau = \begin{bmatrix} \delta_1 & 0 & -\delta_1 & 0 \\ 0 & \delta_2 & 0 & -\delta_2 \\ -\delta_1 & 0 & 2\delta_1 + \tau^2\delta_2 & -\tau\delta_2 \\ 0 & -\delta_2 & -\tau\delta_2 & 2\delta_2 \end{bmatrix}.$$

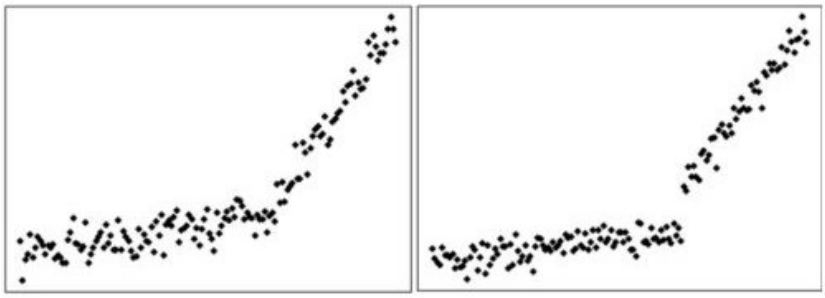


Figure 5.1: An example of continuous change (left) and discontinuous change (right) in linear trend model.

For fuller details see Appendix C.1. Again, this framework can also be extended to the unknown variance case by incorporating an inverse Gamma prior to the variance.

Prior for the changepoint

In order to calculate the posterior distribution of τ at any time t we will need a prior distribution. Denote the prior probabilities by $\Pr_t(\tau = i)$, for $i = 1, \ldots, t-1$ and $\Pr_t(\tau \geq t)$. The latter probability will encompass the event that a change has not occurred by time t.

One approach to defining the prior at time t is to construct a prior for τ taking values in $1, \ldots$ and $\tau = \infty$ (i.e. no change), and then define $\Pr_t(\tau)$ to be consistent with this. However, this is not necessarily appropriate in an online setting where we are monitoring a data stream and will intervene if we detect a change. In this case, there is information in the fact that we are still monitoring the data stream at time t that should affect our prior for τ. Intuitively, this information would mean that it is less likely for τ to take smaller values, for which there would be more power for detecting a change by t.

Thus we suggest the following approach. First let us define the Bayes factor for a change at $\tau = i$ against no change $\tau > t$ as

$$BF_t(i) = \frac{p(y_{1:t}|\tau = i)}{p(y_{1:t}|\tau \geq t)},$$

where

$$p(y_{1:t}|\tau) = \int\int p(y_{1:t}|\tau, \beta, \gamma, \sigma)p(\beta, \gamma|\tau, \sigma)\mathrm{d}\beta\mathrm{d}\gamma,$$

in the case where σ is known, and where we would further integrate out with respect to the prior for σ if it were unknown. The Bayes factor for a change prior to t is obtained by averaging these Bayes factors with respect to the conditional prior for τ assuming a change by time t,

$$BF_t = \sum_{i=1}^{t-1} BF_t(i) \Pr_t(\tau = i|\tau < t).$$

So the posterior probability of a change before time t is

$$p_t(\tau < t|y_{1:t}) = \frac{(1 - \Pr_t(\tau \geq t)BF_t}{\Pr_t(\tau \geq t) + (1 - \Pr_t(\tau \geq t))BF_t}.$$

If we use a monitoring scheme which detects a change if $p_t(\tau < t|y_{1:t}) \geq c_t$ for some constant c_t, this will be equivalent to detecting a change based on the Bayes Factor being greater than a constant, i.e.

$$BF_t \geq \frac{c_t \Pr_t(\tau \geq t)}{(1 - c_t)(1 - \Pr_t(\tau \geq t))}.$$

In the absence of specific prior information, it would be natural to detect a change based on the Bayes Factor being above a threshold that is constant over time. This is easiest to implement with c_t and $\Pr_t(\tau \geq t)$ both being constant. Furthermore, if we take these to be constant, and tune the threshold for detecting a change based on properties of the test under data where there is no change, then this would make our test invariant to the choice of prior probability $\Pr_t(\tau \geq t)$: as choosing a different prior probability would lead to tuning a different threshold such that we were implementing exactly the same test. We take this approach in our simulation study.

The test does depend on our prior for τ conditional on $\tau < t$. It is natural to define

this prior as

$$\Pr_t(\tau = t - k | \tau < t) \propto p^k, \text{ for } \quad k = 1, \ldots, t-1.$$

This is a (truncated geometric) prior for the time since the change. This prior is time dependent, and is updated sequentially as new observations arrive, ensuring that $\Pr_t(\tau < t)$ remains constant while the probabilities of having a change at each location are adjusted. A specific case, which we use in our simulations, is when $p = 1$ and we have a uniform distribution for this conditional distribution of the change location. In this case the Bayes Factor is similar to the Shiryaev-Roberts test statistics which is known to have good properties (Polunchenko and Tartakovsky, 2010).

5.2.2 Sequential Updating

Recall that we wish to calculate the posterior probability of a change, its location and the parameters of the model at each time t. To do this in a computationally efficient way, we need to update these posterior probabilities and distributions sequentially from time $t-1$ to time t given y_t.

The posterior probability for τ at time t satisfies

$$\Pr_t(\tau | y_{1:t}) \propto \Pr_t(\tau) p(y_{1:t} | \tau).$$

We will derive a recursion for the right-hand side, and define a set of weights $w_{i,t} = \Pr_t(\tau = i) p(y_{1:t} | \tau = i)$, for $i = 1, \ldots, t-1$. To simplify notation we have $w_{0,t} = \Pr_t(\tau \geq t) p(y_{1:t} | \tau \geq t)$. The following result gives the update for these weights in terms of the predictive density for y_t given $y_{1:t-1}$.

Theorem 5.2.1. *For $t > 1$,*

$$\begin{aligned} w_{i,t} &= w_{i,t-1}\frac{\Pr_t(\tau = i)}{\Pr_{t-1}(\tau = i)}p(y_t|y_{1:t-1}, \tau = i) \text{ for } i = 1, \ldots, t-2, \text{and } t > 2 \\ w_{0,t} &= w_{0,t-1}p(y_t|y_{1:t-1}, \tau \geq t), \text{ and} \\ w_{t-1,t} &= w_{0,t-1}\frac{\Pr_t(\tau = t-1)}{Pr_{t-1}(\tau \geq t-1)}p(y_t|y_{1:t-1}, \tau = t-1). \end{aligned}$$

With the weights initialised at $t = 1$ by

$$w_{0,t} = \Pr_t(\tau \geq t)p(y_{1:t}|\tau \geq t).$$

Proof. See Appendix C.2. □

We can implement these recursions by recursively calculating the posterior of the model parameters, and hence the predictive distribution, for each possible value of τ. The following result gives updates for the posterior for the parameters in the case σ is unknown, but these can be applied in the case σ is known by ignoring the update for the posterior of σ.

As the prior for the post-change parameter can depend on the time of change, in the case of no changepoint we will update only the marginal posterior for β. To introduce notation, at time $t-1$ let the posterior for β and σ given there is no change be

$$p(\beta|y_{1:t-1}, \sigma, \tau \geq t-1) \sim N\left(\mu^{t-1,0}, \sigma^2\Sigma^{t-1,0}\right), \text{ and } \quad p(\sigma^2|\tau \geq t-1) \sim IG(\nu^{t-1,0}, \iota^{t-1,0}),$$

where $\mu^{t-1,0}$ is a d_1 dimensional vector and $\Sigma^{t-1,0}$ is a $d_1 \times d_1$ matrix. Similarly, we denote the posterior given a change at $i = 1, \ldots, t-2$ as

$$p(\beta, \gamma|y_{1:t-1}, \sigma, \tau = i) \sim N\left(\mu^{t-1,i}, \sigma^2\Sigma^{t-1,i}\right), \text{ and } \quad p(\sigma^2|\tau = i) \sim IG(\nu^{t-1,i}, \iota^{t-1,i}),$$

where now $\mu^{t-1,i}$ is a $d_1 + d_2$ dimensional vector and $\Sigma^{t-1,i}$ is a $(d_1 + d_2) \times (d_1 + d_2)$

matrix. Finally, we will need the posterior distribution for (β, γ) and σ assuming a change at time $t-1$. Using the same notation as above, this can be obtained from the posterior for β and σ given no change prior to $t-1$.

Theorem 5.2.2. *Let the prior mean for a change at $t-1$ be $\mu^{t-1} = [\mu_\beta^\top, \mu_\gamma^\top]^\top$, and the prior covariance be*

$$\sigma^2 \Sigma^{t-1} = \sigma^2 \begin{bmatrix} \Sigma_{\beta,\beta} & \Sigma_{\beta,\gamma} \\ \Sigma_{\gamma,\beta} & \Sigma_{\gamma,\gamma} \end{bmatrix},$$

so that μ_β denotes the prior mean of β, $\Sigma_{\beta,\beta}$ the prior variance of β, $\Sigma_{\beta,\gamma}$ the prior covariance between β and γ, and so on. Then the distribution of (β, γ, σ) given $y_{1:t-1}$ and a change at $\tau = t-1$ has parameters

$$\mu^{t-1,t-1} = \begin{bmatrix} \mu^{t-1,0} \\ \mu_\gamma + \Sigma_{\gamma,\beta}\Sigma_{\beta,\beta}^{-1}(\mu^{t-1,0} - \mu_\beta) \end{bmatrix},$$

$$\Sigma^{t-1,t-1} = \begin{bmatrix} \Sigma^{t-1,0} & \Sigma_{\gamma,\beta}\Sigma_{\beta,\beta}^{-1}\Sigma^{t-1,0} \\ \Sigma^{t-1,0}\Sigma_{\beta,\beta}^{-1}\Sigma_{\beta,\gamma} & \Sigma_{\gamma,\gamma} + \Sigma_{\gamma,\beta}\Sigma_{\beta,\beta}^{-1}(\Sigma^{t-1,0} - \Sigma_{\beta,\beta})\Sigma_{\beta,\beta}^{-1}\Sigma_{\beta,\gamma} \end{bmatrix}$$

with $\nu^{t-1,t-1} = \nu^{t-1,0}$ and $\iota^{t-1,t-1} = \iota^{t-1,0}$.

Proof. See Appendix C.2. □

We can update these parameters as follows.

Theorem 5.2.3. *Fix iteration $t-1$, define h_i as the following feature vector:*

$$h_i = \begin{cases} [a_{t-1}^\top, b_{t-1,i}^\top]^\top & 0 < i \leq t-2, \\ a_{t-1} & i = 0, \end{cases}$$

For $i = 0, 1, \ldots, t-2$, define $e_i = y_{t-1} - h_i \mu^{t-1,i}$, $Q = h_i^\top \Sigma^{t-1,i} h_i + 1$, $A = \Sigma^{t-1,i} h_i / Q$,

The parameter updates for existing changepoint models after observing y_{t-1} follow:

$$\Sigma^{t,i} = \Sigma^{t-1,i} - A^\top AQ, \quad \mu^{t,i} = \mu^{t-1,i} + Ae,$$
$$\nu^{t,i} = \nu^{t-1,i} + \frac{1}{2}, \qquad \iota^{t,i} = \iota^{t-1,i} + \frac{1}{2}e_i^2/Q.$$

Finally, given the posterior for the parameters at time $t-1$, which is also the prior for the models at time t, we have the following calculations for the predictive density.

Theorem 5.2.4. *Using the same notation for the parameters of the posteriors as in Theorem 5.2.3, the predictive density can be calculated as*

$$P(y_t|y_{1:t-1}, \tau = i) = N\left(y_t; h_i\mu^{t,i}, \sigma^2(1 + h_i^\top \Sigma^{t,i} h_i)\right)$$

if σ is known, and

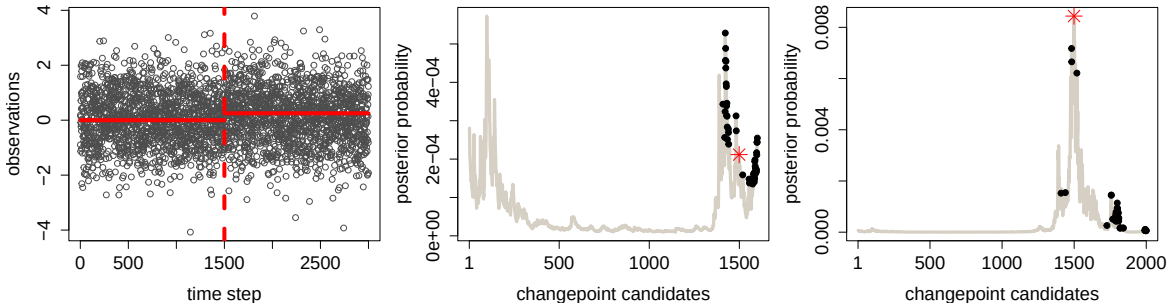
$$P(y_t|y_{1:t-1}, \tau = i) = t_{2\nu^{t,i}}\left(y_t; h_i\mu^{t,i}, \frac{\iota^{t,i}}{\nu^{t,i}}\left(1 + h_i^\top \Sigma^{t,i} h_i\right)\right)$$

if σ is unknown, where $t_\nu(y_t; l, e)$ is the location-scale t distribution with ν degrees of freedom, location l and scale parameter e.

In practice, we can sometimes simplify the updates by considering a suitable reparametrisation of the model to (β, θ) for some θ that is a linear function of β and γ. This can simplify the application of Theorem 5.2.2 if the prior for θ is independence of β – for example if we choose $\theta = \gamma + \beta$ in the chance-in-mean example with the independent mean prior. Moreover, if we choose a reparametrisation such that observations after the change depend on θ but not β, then we can simplify the updates for the changepoint cases by only updating the distribution of θ. Examples of such linear transformations are provided in Appendix C.1.

5.2.3 Reducing the computational complexity by merging

Implementing the univariate Bayesian changepoint detection methods gives a cost per iteration that increases with iteration t, so that the overall cost of analysing T data points is quadratic in T. To prevent this, we need to approximate the posterior distribution with fewer support points. A common and straightforward approach is to prune out the changepoint candidate with the posterior probability less than ϵ or the most M probable candidates (Adams and MacKay, 2007; Saatçi et al., 2010), then the computational complexity can be reduced to $O(\frac{T}{\epsilon})$ or $O(TM)$. However, such a method does not always work well. First, they reduce the support of the posterior for the change, which can be important if we wish to calculate credible intervals. If different changepoint values have substantially different posteriors for the post-change model then their relative probabilities can change substantially as we observe new data, as shown in Figure 5.2.



(a) Simulated data (b) At step 1600 (c) At step 2000

Figure 5.2: Figure (a) presents the simulated data with a change at 1500. Figures (b) and (c) show the posterior distribution obtained from the exact approach without pruning (gray line). The black dots represent the candidates not pruned by the benchmark approach with $M = 50$ at time steps 1600 and 2000, respectively. The red star indicates a candidate that was pruned at step 1600, but subsequently has the highest posterior probability after collecting more data.

Merging criteria

To overcome these issues, we first use the idea of merging the posterior distribution for the parameters for different change locations as a way to reduce the computational cost whilst keeping the full support of the posterior. Second, we take account of the similarity of the posterior distributions of the parameters when deciding which distributions to merge. This is similar to the approach in Shamp et al. (2021). We will look at the similarity of the posteriors in terms of the parameters, or the function of parameters, that affect the distribution of the data after the change. Denote such parameters as $\theta = g(\beta, \gamma)$, where $g(\cdot)$ is a linear transformation that maps the pre-change parameters (β, γ) to the post-change parameters θ as Examples in Appendix C.1.

To explain our approach, it is simplest to first consider what happens when we first merge posteriors. Fix the time, t, when this happens, and denote the posterior density function for the post-change parameter as $f_i(\theta)$ when change occurs at i. We have $t-1$ possible values for the location of a change, assuming a change has occurred, and these will have weight and posterior distribution denoted by the pair $\{w_{i,t}, f_i(\theta)\}, 1 \leq i \leq t-1$.

Our merge procedure will replace $f_i(\theta)$ with $f_{i+1}(\theta)$ for some appropriately chosen i. We will choose $i \in \{1, \ldots, t-2\}$ to be the value for which the total variation distance between the posterior for θ and the resulting approximation is minimised. Let $TV(h_1(\theta), h_2(\theta)) = \frac{1}{2}\int |h_1(\theta) - h_2(\theta)| \mathrm{d}\theta$ be the total variation distance between two densities, $h_1(\theta)$ and $h_2(\theta)$. Then if the true posterior is $h_1(\theta) = \sum_{i=1}^{t-1} w_{i,t} f_i(\theta)$, then the approximation after we replace $f_i(\theta)$ with $f_{i+1}(\theta)$ is $h_2(\theta) = h_1(\theta) - w_{i,t}(f_i(\theta) - f_{i+1}(\theta))$, and

$$\begin{aligned} TV\left(h_1(\theta), h_2(\theta)\right) &= \frac{1}{2}\int \left|h_1(\theta) - (h_1(\theta) - w_{i,t}(f_i(\theta) - f_{i+1}(\theta)))\right| \mathrm{d}\theta \\ &= \frac{1}{2}\int \left|w_{i,t}(f_i(\theta) - f_{i+1}(\theta))\right| \mathrm{d}\theta \\ &= w_{i,t} TV(f_i(\theta), f_{i+1}(\theta)). \end{aligned}$$

Thus we will choose to merge component i which minimises

$$w_{i,t}TV(f_i(\theta), f_{i+1}(\theta)). \tag{5.1}$$

This will maintain the same support for the changepoint locations, but will reduce the computational cost. At any time, if we have M distinct posterior densities for θ, then we only need to update these posteriors when we get a new observation. To update the weights associated with a given posterior, we store these as the sum of the weights for that posterior and the relative weight for each change location. The latter will not change, so we only need to update the former weight.

So at any iteration t, after applying merging, we will have consecutive runs of changepoint locations that will have the same posterior for θ. At the current iteration, denote the largest value of such runs by an ordered set $\mathcal{T} = \{j_1, j_2, \ldots, j_M\}$. Let $w_{i,t}$ for $i = 0, \ldots, M$ denote the set of weights such that for $i \in \{1, \ldots, M\}$, $w_{i,t}$ is the sum of the weights associated with change locations $\{j_{i-1} + 1, \ldots, j_i\}$, with $j_0 = 0$. Also we store a set of M ratio vectors $\mathcal{R} = \{r^{(1)}, \ldots, r^{(M)}\}$ which denote the proportion of the weight w_i associated with each changepoint location in the run from $j_{i-1} + 1$ to j_i. That the vector of weights associated with these candidate locations will be $w_i r^{(i)}$, or equivalently the vector of posterior probabilities for change locations would be

$$(w_0, w_1 r^{(1)}, \ldots, w_M r^{(M)}),$$

with, after normalisation, the first component being the probability of no change, and the remaining components being the probability of a change at $1, \ldots, t = 1$.

The idea is that we can update this representation by just updating the weights, and the parameters associated with each distinct posterior – which has a computational cost proportional to M. At each iteration we will also add a new candidate location for a change, and to keep the number of distinct posteriors as M we will need to merge a pair

of distinct posteriors for θ if $t > M$. As before let $f_i(\theta)$ denote the posterior associated with the ith run of changepoint locations. Using the same criteria on minimising the approximation based on the Total Variation distance, we will replace $f_i(\theta)$ with $f_{i+1}(\theta)$ for the value $i \geq 1$ which minimises $w_i TV(f_i(\theta), f_{i+1}(\theta)$. Algorithm 9 describes one iteration of the algorithm. Storing the relative ratios allows us to recover the pointwise mass distribution, but this leads to a linear increase in storage cost over time. If the exact posterior distribution is not required, this can be ignored, resulting in constant computational and storage costs.

Algorithm 9: one iteration of Bagel

Input: existing candidates set $\mathcal{T} = \{0, i_1, \ldots, i_N\}$, set of ratio vectors $\{r^{(1)}, \ldots, r^{(N)}\}^*$, weights $\{w_{i,t-1}\}$ and parameters for the posterior distributions Θ_i for $i \in \{0, \ldots, N\}$.
Data: Observe y_t at time t
Introducing new changepoint candidate
Calculate Θ_{N+1}, the parameters of the posterior distribution given $\tau = t - 1$ using Theorem 5.2.2.
$\mathcal{T} \longleftarrow \mathcal{T} \cup \{t - 1\}$
Updating weights and posterior distributions
for $i \in \{0, \ldots, N + 1\}$ **do**
    Calculate $w_{i,t}$ based on Theorems 5.2.1 and 5.2.4.
end
Update parameters of posterior distribution, $\Theta_0, \ldots, \Theta_{N+1}$ using Theorem 5.2.3.
Normalize $w_{i,t}$ for $i \in \{0, \ldots, N + 1\}$.
Decide if there is a changepoint
if $1 - w_{0,t} > c$ **then**
    output: Change detected; $\mathcal{T}, \{r^{(1)}, \ldots, r^{(N+1)}\}^*, \{\Theta_0, \ldots, \Theta_{N+1}\}, w_{0:N+1,t}$
end
Merging step
if $N = M$ **then**
    Search the index i that gives the minimum error (5.1) for $i \in \{1, \ldots, N\}$.
    *Update the ratio vector $\{r^{(i+1)}\} \longleftarrow (w_{i,t} + w_{i+1,t})^{-1}(w_{i,t} r^{(i)}, w_{i+1,t} r^{(i+1)})$.
    Combine $w_{i+1,t} \longleftarrow w_{i,t} + w_{i+1,t}$.
    Remove candidate i, Θ_i, $r^{(i)}$ and $w_{i,t}$.
    $N \longleftarrow N - 1$
end
output: $\mathcal{T}, \{r^{(1)}, \ldots, r^{(N+1)}\}^*, \{\Theta_0, \ldots, \Theta_{N+1}\}, w_{0:N+1,t}$

Note: The set marked with * can be omitted, resulting in constant memory complexity.

Calculating the total variation distance

Now we will show how to calculate the total variation, which is needed for our merging step. This can be calculated exactly if θ is 1-dimensional, but we need to resort to an approximation in higher dimensions.

One dimensional parameter

Calculating the total variation between two univariate distributions, such as in the change-in-mean case with known variance, is straightforward. In this case, θ will have a univariate Gaussian distribution, and we use the following result for the total variation distance between two univariate Gaussians.

Proposition 5.2.5. *The total variation between two univariate Gaussian distributions with means μ_i and μ_{i+1}, and variances $\sigma^2\Sigma_i$ and $\sigma^2\Sigma_{i+1}$ respectively is:*

$$2\Big[\Phi(a\sqrt{\sigma^2\Sigma_i^2}+b\sqrt{\sigma^2\Sigma_{i+1}^2})-\Phi(a\sqrt{\sigma^2\Sigma_i^2}-b\sqrt{\sigma^2\Sigma_{i+1}^2}) + \Phi(a\sqrt{\sigma^2\Sigma_{i+1}^2}-b\sqrt{\sigma^2\Sigma_i^2})-\Phi(a\sqrt{\sigma^2\Sigma_{i+1}^2}+b\sqrt{\sigma^2\Sigma_i^2})\Big].$$

where $a=\frac{\mu_i-\mu_{i+1}}{\sigma^2\Sigma_{i+1}^2-\sigma^2\Sigma_i^2}$ *and* $b=\frac{\sqrt{(\mu_i-\mu_{i+1})^2+(\sigma^2\Sigma_{i+1}^2-\sigma^2\Sigma_i^2)\log\frac{\sigma^2\Sigma_{i+1}^2}{\sigma^2\Sigma_i^2}}}{\sigma^2\Sigma_{i+1}^2-\sigma^2\Sigma_i^2}$.

Proof. See proof in Appendix C.3. □

Multi-dimensional parameters

In higher dimensions, it is often intractable to find the exact closed form for the total variation between two distributions. Instead, we use the following tractable bounds as approximations to the total variation distance.

For any pair of distributions for distributions θ over R^d with densities $f_i(\cdot)$ and $f_{i+1}(\cdot)$, Pinsker's inequality (Pinsker, 1964) states that

$$TV(f_i(\theta), f_{i+1}(\theta)) \leq \frac{1}{\sqrt{2}}\sqrt{\mathrm{KL}(f_i(\theta) \parallel f_{i+1}(\theta))} = \frac{1}{\sqrt{2}}\sqrt{\int f_i(\theta)\log\frac{f_i(\theta)}{f_{i+1}(\theta)}d\theta},$$

where KL denotes the Kullback–Leibler divergence. The KL distance for our posteriors, which will be either multivariate Gaussian or Normal Inverse Gamma, is given by the following results.

Theorem 5.2.6. *(Williams and Rasmussen, 2006) The KL between two multivariate Gaussian distributions with means μ_i and μ_{i+1}, and variances $\sigma^2\Sigma_i$ and $\sigma^2\Sigma_{i+1}$ respectively, is:*

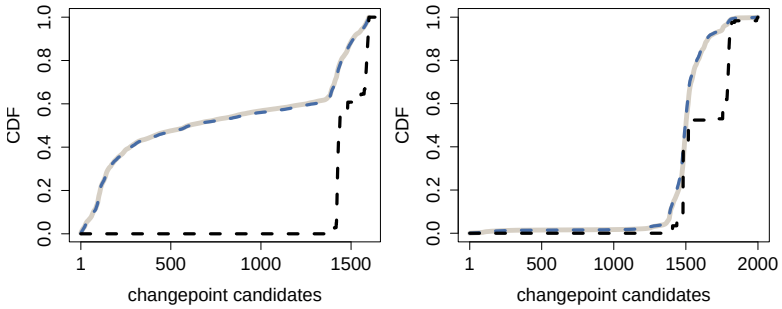
$$D_{KL}(N(\mu_i, \sigma^2\Sigma_i)||N(\mu_{i+1}, \sigma^2\Sigma_{i+1})) = \frac{1}{2}\left(tr(\Sigma_{i+1}^{-1}\Sigma_i - I) + \frac{1}{\sigma^2}(\mu_{i+1} - \mu_i)^\top \Sigma_{i+1}^{-1}(\mu_{i+1} - \mu_i) + \log\det(\Sigma_i^{-1}\Sigma_{i+1})\right)$$

Theorem 5.2.7. *(Soch and Allefeld, 2016) The KL distance between two normal inverse gamma distributions with means μ_i and μ_{i+1}, scaling variance Σ_i and Σ_{i+1}, shape ν_i and ν_{i+1}, scale ι_i and ι_{i+1} respectively is*

$$\begin{aligned} D_{KL}[NIG(\mu_i, \Sigma_i, \nu_i, \iota_i)||NIG(\mu_{i+1}, \Sigma_{i+1}, \nu_{i+1}, \iota_{i+1})] = \\ \frac{1}{2}\left[(\mu_{i+1} - \mu_i)^\top \Sigma_j^{-1}(\mu_{i+1} - \mu_i)\frac{\nu_i}{\iota_i} + tr(\Sigma_{i+1}^{-1}\Sigma_i - I) - \log\frac{\det(\Sigma_i)}{\det(\Sigma_{i+1})}\right] \\ + \nu_{i+1}\log\frac{\iota_i}{\iota_{i+1}} - \log\frac{\Gamma(\nu_i)}{\Gamma(\nu_{i+1})} + (\nu_i - \nu_{i+1})\psi(\nu_i) - (\iota_i - \iota_{i+1})\frac{\nu_i}{\iota_i}, \end{aligned}$$

where $\Gamma(\cdot)$ is the gamma function and $\psi(\cdot)$ is the digamma function.

Using the KL bounds, we revisit the example in Figure 5.2. Figure 5.3 presents the cumulative density function (CDF) of the exact approach, our recovered posterior, and the benchmark. It demonstrates that the recovered posterior from Bagelclosely matches the exact distribution.



(a) At step 1600 (b) At step 2000

Figure 5.3: The CDFs of the exact approach (grey), Bagelwith recovered posterior distribution (blue) and benchmark (black) when $M = 50$.

5.3 Simulation Results

We conduct a simulation study to evaluate the performance of different pruning methods, focusing on detection power and computational efficiency on the above examples. For simplicity, we assume the changepoint locations are uniformly distributed for changepoint models. For the change-in-mean case, we reparametrize the model in terms of the pre-change parameter (β) and post-change parameters ($\theta = \beta + \gamma$), and place independent priors on them. A Normal prior is used when the variance is known, and a Normal-Inverse-Gamma prior when it is unknown. For detecting changes in slope, we allow for both continuous and discontinuous changes, assigning equal prior probabilities to the two types of change. Therefore we consider three possible models: no-change, continuous change, and discontinuous change, with bivariate normal prior on (β, γ) as specified in Section 5.2.1. We can then implement the recursion and pruning rules of Section 2 for the two change models separately, and get an approximation to the posterior distribution. The estimated model is selected as the one with the highest marginal posterior probability. Further prior details are provided in Appendix C.4.1. Once the parameters are set, we simulate 2000 data points without change. The detection thresholds are chosen to control the false alarm rate, defined as the probability of falsely detecting a change, at 5%.

To evaluate the detection power of algorithms, we measure the following indices:

M	Algorithm	Known variance				Unknown variance			
		Coverage	Error	Delay	MAP	Coverage	Error	Delay	MAP
	Exact	0.95	0.05	281±174	1008±110	0.95	0.06	295 ± 189	1025 ± 106
50	Bagel	0.95	0.04	283±176	1010±108	0.95	0.06	299 ± 192	1022 ± 105
	Benchmark	0.38	0.08	308±198	1009±116	0.36	0.09	326 ± 213	1025 ± 106
100	Bagel	0.95	0.05	282±175	1009±107	0.95	0.06	296 ± 190	1024 ± 106
	Benchmark	0.56	0.06	300±195	1012±102	0.56	0.08	317 ± 207	1023 ± 104
200	Bagel	0.95	0.05	281±175	1009±109	0.94	0.06	295 ± 190	1024 ± 106
	Benchmark	0.74	0.05	294±186	1011±109	0.74	0.07	306 ± 202	1026 ± 111

Table 5.1: Simulation results for the Example 1 change-in-mean case with 500 replicates. The simulated data follows $N(0, 1)$ before $\tau = 1000$ and $N(0.25, 1)$ after the change.

- Coverage rate: the proportion in which the true changepoint falls within the estimated 95% highest posterior density intervals.
- Error rate of the algorithm: calculated as $\frac{\text{missed alarms + false alarms}}{\text{replications}}$.
- Average detection delay and its standard deviation: the detection delay measures the difference between the estimated true positive stopping time and the true changepoint location.
- MAP estimator and its standard deviation: the *maximum a posteriori* estimation gives the estimation of the changepoint location with the largest posterior probability.
- Speed: the average running time at each time step.

Table 5.1 and Table 5.2 present the simulation results for the two examples, using KL bounds when total variation needs to be approximated. In Example 1, the detection power of Bagel is close to that of the exact approach, while the benchmark consistently shows the lowest coverage rate, highest error rate, and longest detection delay. In Example 2, Bagel performs close to the exact approach. The benchmark algorithm again performs the worst and fails to identify the type of change. In terms of computational speed, Bagel and the benchmark are comparable and significantly faster than the exact approach, as shown in Appendix C.4.3. Results with different prior settings and different bounds approximation are provided in Appendix C.4.2.

M	Algorithm	Example 2 with continuous change				Example 2 with discontinuous change			
		Coverage	Error	Delay	MAP	Coverage	Error	Delay	MAP
	Exact	0.87	0.05	329 ± 73	1037 ± 190	0.93	0.05	198 ± 68	959 ± 145
50	Bagel	0.86	0.05	336 ± 76	1038 ± 152	0.93	0.05	205 ± 70	949 ± 137
	Benchmark	0.00	0.05	399 ± 96	1199 ± 101	0.05	0.05	253 ± 89	1062 ± 75
100	Bagel	0.88	0.05	331 ± 74	1042 ± 150	0.92	0.05	199 ± 69	954 ± 106
	Benchmark	0.00	0.05	373 ± 90	1140 ± 106	0.22	0.05	231 ± 80	1017 ± 83
200	Bagel	0.87	0.05	329 ± 74	1036 ± 159	0.92	0.05	198 ± 68	949 ± 116
	Benchmark	0.05	0.05	353 ± 83	1082 ± 136	0.55	0.05	214 ± 75	978 ± 105

Table 5.2: Simulation results for Example 2 change-in-slope scenario with known variance with 500 replicates. The simulated data follow a normal distribution $N(0, 1)$ before time $t = 1000$, and follow $-1.75+0.002t+N(0, 1)$ after the change for continuous change and follow $-2.2 + 0.002t + N(0, 1)$ after the change for discontinuous change.

5.4 Real data example - Machine Temperature Failure

The Numenta Anomaly Benchmark (NAB) contains a collection of datasets for evaluating real-time anomaly detection algorithms (Ahmad et al., 2017). We evaluate our proposed approaches on one of the datasets - machine temperature failure dataset. This dataset records temperature readings from a heat sensor, and has been used in several algorithm evaluations, such as those in Fisch et al. (2022a); Ahmad and Purdy (2016); Ahmad et al. (2017); Jesmeen et al. (2021), among others. It contains 22695 observations from 02/12/2013 to 19/02/2014, with data sampled every 5 minutes. Following the data preparation procedure in Lavin and Ahmad (2015); Fisch et al. (2022a), we use the first 15% to be training data. We preprocess the data by removing autocorrelation components and normalising with the median and the median absolute deviation estimated from the training set. To reduce the impact of extreme values, we remove outliers with modified Z-scores exceeding an absolute value of 3.5, as suggested in Iglewicz and Hoaglin (1993). The threshold for detection is empirically determined on simulated pseudo time series by resampling the block of observations (Politis and Romano, 1994). We perform 200 Monte Carlo replications to ensure that the false alarm rate remains 5% on 5000 observations.

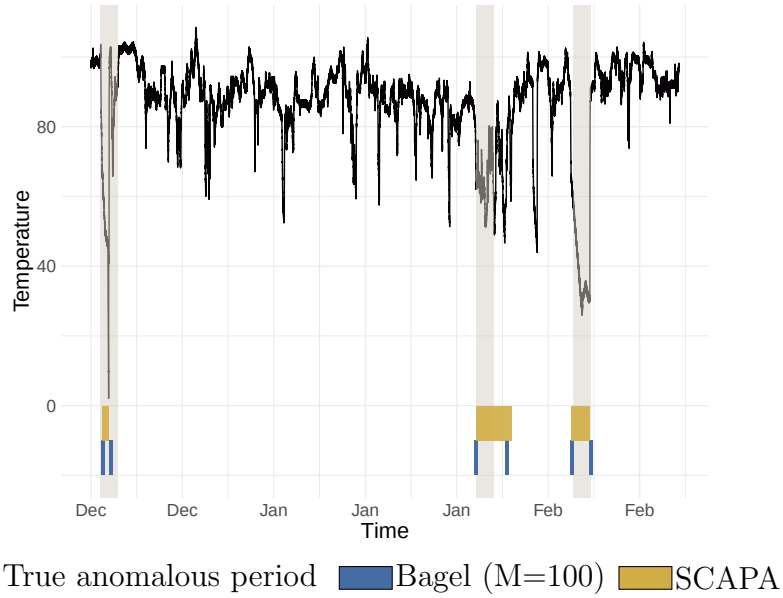


Figure 5.4: Machine temperature data with true anomalous period, MAP estimators of changepoints obtained from our proposed method, and the anomalous segment detected by SCAPA

Year	Start time	End time	SCAPA	Bagel(M=100)
2013	17:50 15/12	17:00 17/12	16:50 16/12	03:00 16/12
2014	14:20 27/01	13:30 29/01	21:25 28/01	07:15 28/01
2014	14:55 07/02	14:05 09/02	03:15 08/02	00:45 08/02

Table 5.3: Labelled anomalies along with the detection time of two approaches.

We use the change-in-slope models with known variance, which allow us to monitor both mean shifts and slope shifts. When Bagel detects a change, the algorithm restarts and is initialised using the MAP estimators from the posterior of the previous segment as its new prior. Figure 5.4 and Table 5.3 present the detection results of Bagel with $M = 100$, alongside the results of SCAPA as reported in their original paper Fisch et al. (2022a). Although SCAPA is proposed to detect the anomalous segment, we can treat the start and the end of the segment as two changepoints for comparison. Two algorithms give similar changepoint estimates, while Bagel is faster in detection. Moreover, the estimators from both methods outperform most of the results reported in Ahmad and Purdy (2016); Jesmeen et al. (2021), where changepoints were either missed or falsely triggered.

5.5 Discussion

We proposed a fast online Bayesian changepoint detection algorithm for identifying changes in linear models. The resulting method provides uncertainty estimates for both the change and its location through the recovered posterior distribution. Moreover, the algorithm maintains constant time complexity per time step, empirically at a millisecond level, making it suitable for real-time applications. Future work includes extending the method to multivariate settings, where the dependencies exist across the dimensions, and changes may propagate across dimensions.

Chapter 6

Conclusions And Future Work

Three novel online changepoint detection methods have been introduced in this thesis. The first two, disMOSUM and mixFOCuS, are designed to detect changes in distributed systems under constraints of limited computational and communication resources. Both approaches have their respective strengths and limitations. mixFOCuS outperforms disMOSUM in that it does not require selecting a window size and can accommodate data from the exponential family, rather than being restricted to Gaussian assumptions. On the other hand, disMOSUM benefits from a known limiting distribution, which provides theoretical guidance for threshold selection. Moreover, it is flexible enough to incorporate more complex modelling components, such as autocorrelation structures. One limitation for both approaches, as demonstrated through simulations, is the presence of autocorrelation in the data. While one solution is to inflate the threshold, we show the performance of the algorithms can be significantly affected when the autocorrelation is strong. Developing robust methods that explicitly handle autocorrelated inputs remains a direction for future research.

Another direction that could be considered in the future is to extend the current framework to decentralised systems, where sensors are allowed to communicate with neighbouring nodes according to a predefined network topology. In such settings, a key

question is whether changes can still be detected rapidly and reliably, with restricted communication bandwidth. Furthermore, if the change does not occur simultaneously across nodes, can the system still infer the path of change propagation through the network? A practical example of such a scenario is fault detection in power grids, where local sensors may detect anomalies at different times, and inferring the cascade of failures through limited communication can be critical. Another relevant application is a surveillance network using edge devices, where bandwidth is limited, but needs quick detection and localisation of unusual events.

In Section 5, we derived a fast online Bayesian changepoint detection algorithm. An extension is to generalise this framework to the multivariate case, where multiple data streams are monitored simultaneously. In so doing, several interesting research challenges arise. First, the model must be able to capture correlations across streams, which may reflect underlying structural or functional relationships in the system. Second, when changes occur at different times in different streams, a key question is whether we can infer the transmission or propagation of the change across the streams. Another possible application is to integrate the Bayesian changepoint detection framework with real-time decision-making systems, such as autonomous systems, adaptive control, or intelligent monitoring. The posterior distribution over changepoint locations and model parameters provides a quantifiable measure of uncertainty, which can be used in control policies or feedback loops.

Appendix A

Appendix for disMOSUM

A.1 Proof of Theorem 3.4.3

Squaring and expanding the weighted global MOSUM statistic in Equation (3.9) for the two different cases gives

$$\left(w(k,h)\mathcal{T}(m,k,h)\right)^2 = \begin{cases} \sum_{i=1}^{d} \left(w(k,h)\mathcal{T}_i(m,k,h)\right)^2 & \text{dense case,} \\ \sum_{i=1}^{d} \left(w(k,h)\mathcal{T}_i(m,k,h)\mathbb{1}_{\{w(k,h)\mathcal{T}_i(m,k,h)>c_{\text{Local}}\}}\right)^2 & \text{sparse case.} \end{cases} \tag{A.1}$$

From Theorem 3.4.1 with $k = ht$, the weighted local MOSUM and its hard-thresholded counterpart have limit

$$\begin{aligned} \lim_{m\to\infty} \left(w(k,h)\mathcal{T}_i(m,k,h)\right)^2 &\xrightarrow{\mathcal{D}} \rho(t)^2 Z_i(t)^2, \\ \lim_{m\to\infty} \left(w(k,h)\mathcal{T}_i(m,k,h)\mathbb{1}_{\{w(k,h)\mathcal{T}_i(m,k,h)>c_{\text{Local}}\}}\right)^2 &\xrightarrow{\mathcal{D}} \rho(t)^2 Z_i(t)^2 \mathbb{1}_{\{\rho(t)Z_i(t)>c_{\text{Local}}\}}, \end{aligned} \tag{A.2}$$

where $Z_i(t)$ is defined in Equation 3.11. Taking the limit of (A.1) as $m \to \infty$ gives the result.

A.2 Proof of Theorem 3.4.6

It is enough to show that

$$\left(w(\tilde{k},h)\mathcal{T}(m,\tilde{k},h)\right)^2 \xrightarrow{\mathcal{P}} \infty,$$

for a time $\tilde{k}$ later than the change point k^* but before the end of the monitoring time $\lfloor m\tilde{T}\rfloor$.

Since $k^* \leq \lfloor h\nu \rfloor$, we can choose $\tilde{k} = \lfloor x_0 h\rfloor + h$ where $\nu < x_0 < \tilde{T}\frac{m}{h} - 1$ so that $k^* \leq \tilde{k} - h$.

If data stream i is affected by the change, so that $i \in \mathcal{S}$ and $\delta_i \neq 0$ then

$$\begin{aligned}
\frac{1}{h}\mathcal{T}_i(m,\tilde{k},h) &= \frac{1}{h\hat{\sigma}_i}\left|\sum_{t=m+\lfloor x_0 h\rfloor+1}^{m+\lfloor x_0 h\rfloor+h}(X_{i,t}-\hat{\mu}_i)\right| \\
&= \frac{1}{h\hat{\sigma}_i}\left|\sum_{t=m+\lfloor x_0 h\rfloor+1}^{m+\lfloor x_0 h\rfloor+h}(\mu_i+\delta_i+\epsilon_{i,t}-\hat{\mu}_i)\right| \\
&= \frac{1}{h\hat{\sigma}_i}\left|h(\mu_i-\hat{\mu}_i)+h\delta_i+\sum_{t=m+\lfloor x_0 h\rfloor+1}^{m+\lfloor x_0 h\rfloor+h}\epsilon_{i,t}\right| \\
&= \frac{1}{\hat{\sigma}_i}\left|\mu_i-\hat{\mu}_i+\delta_i+\frac{1}{h}\sum_{t=m+\lfloor x_0 h\rfloor+1}^{m+\lfloor x_0 h\rfloor+h}\epsilon_{i,t}\right| \\
&= \frac{1}{\hat{\sigma}_i}\left|\delta_i+o_P(1)\right|.
\end{aligned}$$

On the other hand if $i \notin \mathcal{S}$,

$$\frac{1}{h}\mathcal{T}_i(m,\tilde{k},h) = \frac{1}{\hat{\sigma}_i}\left|o_P(1)\right|.$$

For the global dense procedure

$$\begin{aligned}\left(w(\tilde{k},h)\mathcal{T}(m,\tilde{k},h)\right)^2 &= w(\tilde{k},h)^2\sum_{i=1}^{d}\mathcal{T}_i(m,\tilde{k},h)^2\\ &= \left(\frac{1}{\sqrt{h}}\rho\left(\frac{\tilde{k}}{h}\right)\right)^2\times h^2\times\sum_{i=1}^{d}\left(\frac{1}{h}\mathcal{T}_i(m,\tilde{k},h)\right)^2\\ &= h\rho\left(\frac{\tilde{k}}{h}\right)^2\sum_{i=1}^{d}\left(\frac{1}{h}\mathcal{T}_i(m,\tilde{k},h)\right)^2\\ &= h\rho\left(x_0+1+o(1)\right)^2\sum_{i\in\mathcal{S}}\left(\frac{\delta_i}{\hat{\sigma}_i}\right)^2+o_P(1)\xrightarrow{\mathcal{P}}\infty,\end{aligned}$$

as $m, h \to \infty$.

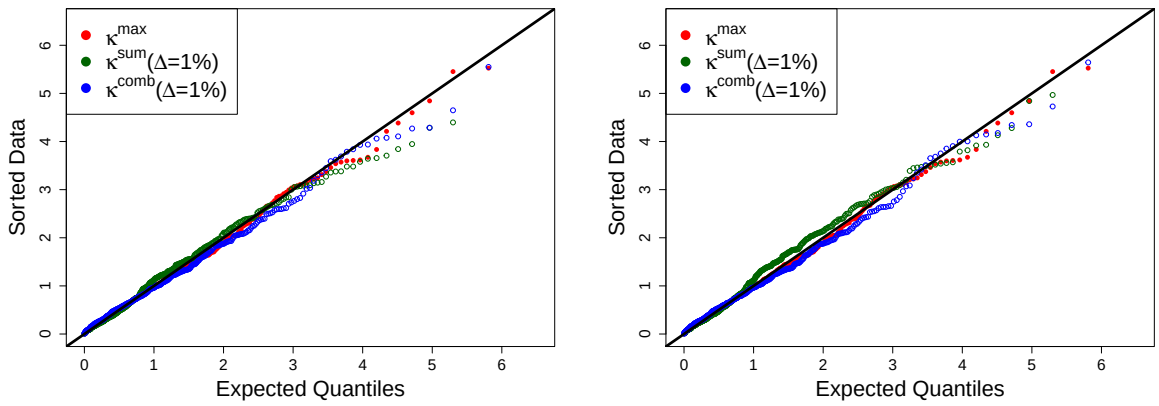
For the global sparse procedure the local MOSUM's $w(\tilde{k},h)\mathcal{T}(m,\tilde{k},h)$ are hard thresholded. Since these diverge to infinity individually then the same argument used for the dense procedure applies.

Appendix B

Appendix for mixFOCUS

B.1 Quantile-Quantile plots of the time to detection

We simulated 1000 Gaussian data sets with $n = 200000$ and $d = 100$ when there is no change and determined a local threshold that restricts the transmission frequency to 1%. Figure B.1 presents the QQ plot comparing the time to detection against an exponential (1) distribution. These sub-figures suggest the time to detection is approximately geometrically distributed under two large global thresholds.



(a) $c_{Global}^{MAX} = 16, c_{Global}^{SUM} = 28$ (b) $c_{Global}^{MAX} = 17, c_{Global}^{SUM} = 26$

Figure B.1: QQ plot of standardized $\kappa^{\max}$, κ^{sum} and κ^{comb} for different thresholds against exponential (1) distribution when $\Delta = 1\%$.

B.2 Detection power of mixFOCuS when $\tau = 1000$

Table B.1 and Table B.2 present the detection power of the mixFOCuS on the Gaussian data and mixed-type data when the change occurs at 1000. The detection power is slightly weaker compared to the cases where the change occurs later (e.g. at 3000, see Table 4.3 and Table 4.4). The trade-off between transmission constraints and detection efficiency, as well as the comparable performance of mixFOCuS on both mixed-type and homogeneous data, remains consistent.

Scenarios		pre-change known(S1)				pre-change unknown(S2)			
Transmission Constraint		100%	10%	5%	1%	100%	10%	5%	1%
FAR(%)		4.4	4.5	4.7	5.2	5.0	5.6	5.9	6.8
p	δ	ADD				ADD			
1	0.1	144	**133**	137	166	206	**185**	188	226
	0.25	**28**	**28**	29	34	35	**33**	34	40
	0.5	**9**	**9**	**9**	11	**10**	**10**	**10**	12
	1	**3**	**3**	**3**	4	**3**	**3**	4	4
0.5	0.1	262	233	**231**	256	414	341	**334**	391
	0.25	51	47	**46**	51	63	55	**54**	59
	0.5	15	**14**	**14**	15	17	**16**	**16**	17
	1	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**
0.1	0.1	887	753	689	**637**	2457 0.3%	1802 0.2%	1623 0.4%	**1579** 0.5%
	0.25	172	153	139	**126**	216	181	163	**147**
	0.5	49	45	40	**36**	52	46	42	**38**
	1	14	13	12	**11**	14	13	12	**11**
0.01	0.1	2420	2353	2268	**2094**	4690 (33%)	4064 (28.9%)	4789 (28.2%)	**4575** (28%)
	0.25	418	417	415	**407**	855	807 (0.1%)	780 (0.1%)	**722**
	0.5	108	108	108	**107**	124	124	123	**122**
	1	**29**	**29**	**29**	**29**	**30**	**30**	**30**	**30**

Table B.1: Results on Gaussian data are averaged over 1000 repetitions. Data are simulated with parameters $n = 10000$ and $d = 100$ and follow Gaussian distribution. The smallest ADD in each scenario is given in bold. The percentage represents the proportion of missed alarms.

B.3 The effect of a Gaussian approximation

Figure B.2 compares the detection power of mixFOCuS with a common approach that approximates the data as Gaussian, considering scenarios where all or 10% data streams

undergo the change. While the common approach shows improved performance compared to it under sparse change (Figure 4.4), it has higher false alarm rates compared to mixFOCuS, particularly when the pre-change distribution is unknown.

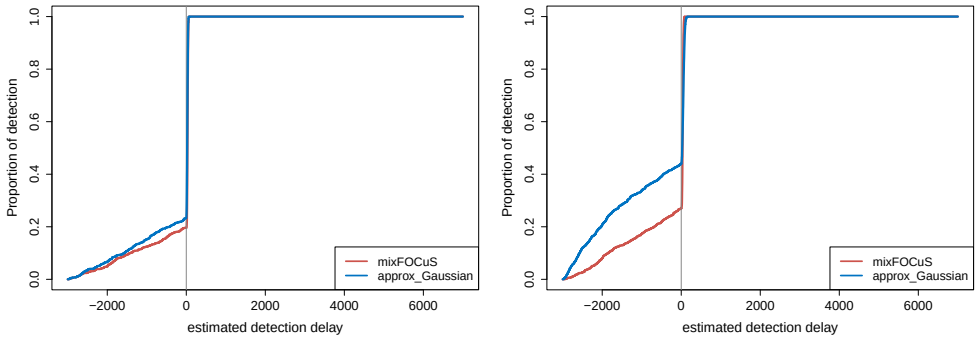
Scenarios		pre-change known(S1)				pre-change unknown(S2)			
Transmission Constraint		100%	10%	5%	1%	100%	10%	5%	1%
FAR(%)		4.2	4.2	4.5	5.9	4.6	5.1	5.1	7.4
p	δ	ADD				ADD			
1	0.1	149	**137**	140	163	216	**188**	202	231
	0.25	30	**29**	30	35	38	**35**	36	41
	0.5	10	**9**	10	11	**11**	**11**	**11**	13
	1	**3**	4	4	4	**4**	**4**	**4**	5
0.5	0.1	268	**238**	237	253	433	**351**	362	391
	0.25	54	49	**48**	52	67	**57**	**57**	61
	0.5	17	**15**	**15**	16	19	**17**	**17**	18
	1	**5**	**5**	**5**	**6**	**6**	**6**	**6**	6
0.1	0.05	899	768	709	**650**	2563 98.2%	1889 0.1%	1746 0.5%	**1529** 98.1%
	0.25	183	161	147	**130**	237	195	182	**154**
	0.5	52	48	44	**39**	56	50	47	**41**
	1	16	15	13	**12**	16	15	14	**12**
0.01	0.1	2378	2280	2226	**2031**	4502 (32.6%)	3809 (24.9%)	3745 (28.7%)	**3324** (22.9%)
	0.25	419	416	414	**408**	896 0.2%	843 (0.1%)	836 (0.1%)	**734**
	0.5	107	107	107	**106**	123	122	123	**120**
	1	**29**	**29**	**29**	**29**	30	30	30	**29**

Table B.2: Average detection delay on the mixed-type data against the levels of sparsity and the strengths of the signal. Results are averaged over 1000 repetitions. Data are simulated with parameters $n = 10000$ and $d = 100$. The smallest ADD in each scenario is given in bold. The percentage within the bracket is the proportion of missed alarms.

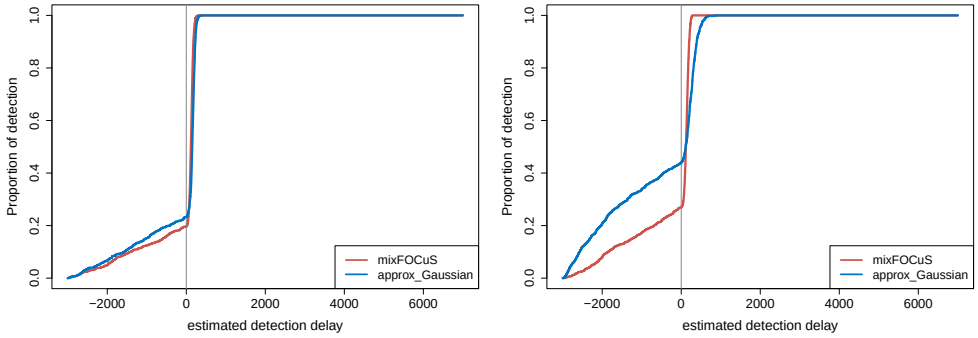
B.4 Model details

The performance of MOSUM depends on the choice of window size h. Here we run multiple window sizes at the same time, the one with the highest likelihood ratio statistics is selected. For training size 500, we evaluate window sizes $h = 20, 100, 250, 400$; as for training size 100, we pick window sizes $h = 20, 50, 80, 100$. Additionally, we set the weight function in disMOSUM to be 1, allowing us to control the average run length instead of the type-I error.

As for disCUSUM, we set parameters with $s = 1$ and $t = 4$ as in Lorden and Pollak



(a) $\delta = 0.25, p = 1$ pre-change known (b) $\delta = 0.25, p = 1$ pre-change unknown



(c) $\delta = 0.25, p = 0.1$ pre-change known (d) $\delta = 0.25, p = 0.1$ pre-change unknown

Figure B.2: The proportions of experiments where a change was detected by time step t. Data are generated with $p_{ber} = 0.4, \lambda_{pois} = 5, \lambda_{exp} = \frac{1}{3}, \beta = 2, n = 10000$ and $\tau = 3000$. Results are obtained under 1000 replications.

(2008) and Liu et al. (2019) and use soft-thresholding as in Liu et al. (2019). We consider different values for the possible change sizes of $\rho = 0.1, 0.5$ and 1.

Appendix C

Appendix for Bagel

C.1 Properties of the Prior for Examples 1 and 2.

Example 1

To see that the second prior for Example 1 gives the same distribution for the pre-change and post-change mean, and these are independent, we apply a change of variable. We have $(\beta, \gamma)^\top$ has a Gaussian distribution with mean $\begin{bmatrix} \mu_1 \\ 0 \end{bmatrix}$ and variance $\sigma^2 \begin{bmatrix} \delta_1 & -\delta_1 \\ -\delta_1 & 2\delta_1 \end{bmatrix}$.

Let the pre-change and post-change means be $(\beta, \theta)^\top$ then

$$\begin{bmatrix} \beta \\ \theta \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} \beta \\ \gamma \end{bmatrix}.$$

This is a linear transformation, so $(\beta, \theta)^\top$ will also have a Gaussian distributions with mean

$$\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} \mu_1 \\ 0 \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_1 \end{bmatrix},$$

and variance

$$\sigma^2 \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} \delta_1 & -\delta_1 \\ -\delta_1 & 2\delta_1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}^\top = \sigma^2 \begin{bmatrix} \delta_1 & 0 \\ 0 & \delta_1 \end{bmatrix},$$

as required.

Example 2

In the change-in-slope case, for a discontinuous change, denote the pre-change and post-change means of intercept and slope as $(\beta_1, \beta_2, \theta_1, \theta_2)^\top$, we have

$$\begin{bmatrix} \beta_1 \\ \beta_2 \\ \theta_1 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & \tau & 1 & \tau \\ 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \gamma_1 \\ \gamma_2 \end{bmatrix}.$$

With this linear transformation, we have

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & \tau & 1 & \tau \\ 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mu_1 \\ \mu_2 \\ -\mu_2\tau \\ 0 \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_1 \\ \mu_2 \end{bmatrix},$$

and variance

$$\sigma^2 \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & \tau & 1 & \tau \\ 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} \delta_1 & 0 & -\delta_1 & 0 \\ 0 & \delta_2 & 0 & -\delta_2 \\ -\delta_1 & 0 & 2\delta_1 + \tau^2\delta_2 & -\tau\delta_2 \\ 0 & -\delta_2 & -\tau\delta_2 & 2\delta_2 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & \tau & 1 & \tau \\ 0 & 1 & 0 & 1 \end{bmatrix}^\top = \sigma^2 \begin{bmatrix} \delta_1 & 0 & 0 & 0 \\ 0 & \delta_2 & 0 & 0 \\ 0 & 0 & \delta_1 & 0 \\ 0 & 0 & 0 & \delta_2 \end{bmatrix},$$

as required. So the pre-change and post-change parameters are independent and follow the same distribution.

As for the continuous change, we have

$$\begin{bmatrix} \beta_1 \\ \beta_2 \\ \theta_1 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \tau \\ 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \gamma_1 \\ \gamma_2 \end{bmatrix}.$$

So the $(\theta_1, \theta_2, \theta_3, \theta_4)^\top$ follows the multivariate Gaussian distribution with mean

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \tau \\ 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mu_1 \\ \mu_2 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ 0 \\ \mu_2 \end{bmatrix}$$

and variance

$$\sigma^2 \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \tau \\ 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} \delta_1 & 0 & 0 & 0 \\ 0 & \delta_2 & 0 & 0 \\ 0 & 0 & \tau^2\delta_3 & -\tau\delta_3 \\ 0 & 0 & -\tau\delta_3 & \delta_3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \tau \\ 0 & 1 & 0 & 1 \end{bmatrix}^\top = \sigma^2 \begin{bmatrix} \delta_1 & 0 & 0 & 0 \\ 0 & \delta_2 & 0 & \delta_2 \\ 0 & 0 & 0 & 0 \\ 0 & \delta_2 & 0 & \delta_2 + \delta_3 \end{bmatrix},$$

where the new intercept has mean 0 and variance 0, and the post-change slope is dependent on the pre-change slope.

C.2 Proof of the sequential updating

Proof of Theorem 5.2.1

For $i = 1, \ldots, t-2$ the recursion comes from the definition of the weights.

$$\begin{aligned} w_{i,t} &= \Pr_t(\tau = i) p(y_{1:t} | \tau = i) \\ &= \Pr_{t-1}(\tau = i) p(y_{1:t-1} | \tau = i) \frac{\Pr_t(\tau = i)}{\Pr_{t-1}(\tau = i)} p(y_t | y_{1:t-1}, \tau = i) \\ &= w_{i,t-1} \frac{\Pr_t(\tau = i)}{\Pr_{t-1}(\tau = i)}) p(y_t | y_{1:t-1}, \tau = i). \end{aligned}$$

A similar simple argument applies for $i = 0$. Finally for $i = t-1$ we have

$$\begin{aligned} w_{t-1,t} &= \Pr_t(\tau = t-1) p(y_{1:t} | \tau = t-1) \\ &= \Pr_{t-1}(\tau \geq t-1) p(y_{1:t-1} | \tau \geq t-1) \frac{\Pr_t(\tau = i)}{\Pr_{t-1}(\tau \geq t-1)} p(y_t | y_{1:t-1}, \tau = t-1) \\ &= w_{0,t-1} \frac{\Pr_t(\tau = i)}{\Pr_{t-1}(\tau \geq t-1)}) p(y_t | y_{1:t-1}, \tau = t-1). \end{aligned}$$

Proof of Theorem 5.2.2

If we have a change at $t-1$ then the prior for (β, γ) has mean μ^{t-1} and variance $\sigma^2 \Sigma^{t-1}$. Our assumption is that the marginal prior for β is the same as that assuming no change, but the conditional prior for γ given β can depend on the change location $t-1$.

The posterior distribution given $y_{1:t-1}$, conditional on a change at $t-1$ is

$$\begin{aligned} & p(\beta, \gamma, \sigma | y_{1:t-1}, \tau = t-1) \\ & \quad \propto p(\sigma) p(\beta | \sigma) p(\gamma | \beta, \sigma, \tau = t-1) p(y_{1:t-1} | \beta, \gamma, \sigma, \tau = t-1) \\ & \quad = p(\sigma) p(\beta | \sigma) p(\gamma | \beta, \sigma, \tau = t-1) p(y_{1:t-1} | \beta, \sigma, \tau = t-1) \\ & \quad \propto p(\beta, \sigma | y_{1:t-1}, \tau = t-1) p(\gamma | \beta, \sigma, \tau = t-1) \\ & \quad = p(\beta, \sigma | y_{1:t-1}, \tau \geq t-1) p(\gamma | \beta, \sigma, \tau = t-1). \end{aligned}$$

Here we have used the fact that the observations prior to the change do not depend on γ, and that as the prior for β and σ and the likelihood for $y_{1:t-1}$ are the same for

$\tau = t-1$ and $\tau \geq t-1$ so are the posteriors.

The posterior for (β, σ) given $y_{1:t-1}$ and $\tau \geq t-1$ is calculated at time $t-1$ for the case of no change by $t-1$ and conditional on σ the posterior for β is Gaussian with mean $\mu^{t-1,0}$ and variance $\Sigma^{t-1,0}$.

As $p(\beta, \gamma|\sigma, \tau = t-1)$ is multivariate Gaussian, the conditional prior $p(\gamma|\beta, \sigma, \tau = t-1)$ is also Gaussian, with a mean that is linear in β. Using standard results for the multivariate Gaussian gives that this conditional distribution is of the form

$$\gamma|\beta, \sigma, \tau = t-1 \sim \mathrm{N}(\mu_\gamma + \Sigma_{\gamma,\beta}\Sigma^{-1}_{\beta,\beta}(\beta - \mu_\beta), \sigma^2(\Sigma_{\gamma,\gamma} - \Sigma_{\gamma,\beta}\Sigma^{-1}_{\beta,\beta}\Sigma_{\beta,\gamma})),$$

Therefore, the marginal mean of γ after updating β is given by

$$E(\gamma) = E\left(E(\gamma \mid \beta)\right) = \mu_\gamma + \Sigma_{\gamma,\beta}\Sigma^{-1}_{\beta,\beta}(\mu^{t-1,0} - \mu_\beta).$$

The marginal scaled variance is

$$\begin{aligned}
\mathrm{Var}(\gamma) &= E\left(\mathrm{Var}(\gamma \mid \beta)\right) + \mathrm{Var}\left(E(\gamma \mid \beta)\right) \\
&= \sigma^2\left(\Sigma_{\gamma,\gamma} - \Sigma_{\gamma,\beta}\Sigma^{-1}_{\beta,\beta}\Sigma_{\beta,\gamma} + \Sigma_{\gamma,\beta}\Sigma^{-1}_{\beta,\beta}\Sigma^{t-1,0}\Sigma^{-1}_{\beta,\beta}\Sigma_{\beta,\gamma}\right) \\
&= \sigma^2\left(\Sigma_{\gamma,\gamma} + \Sigma_{\gamma,\beta}\Sigma^{-1}_{\beta,\beta}(\Sigma^{t-1,0} - \Sigma_{\beta,\beta})\Sigma^{-1}_{\beta,\beta}\Sigma_{\beta,\gamma}\right).
\end{aligned}$$

The covariance can be calculated as

$$\mathrm{Cov}(\beta, \gamma) = \mathrm{Cov}\left(\beta, \Sigma_{\gamma,\beta}\Sigma^{-1}_{\beta,\beta}\beta\right) = \Sigma_{\gamma,\beta}\Sigma^{-1}_{\beta,\beta}\Sigma^{t-1,0}.$$

Thus, the full updated joint distribution of (β, γ) is

$$N\left(\begin{bmatrix} \mu^{t-1,0} \\ \mu_\gamma + \Sigma_{\gamma,\beta}\Sigma^{-1}_{\beta,\beta}(\mu^{t-1,0} - \mu_\beta) \end{bmatrix}, \begin{bmatrix} \Sigma^{t-1,0} & \Sigma_{\gamma,\beta}\Sigma^{-1}_{\beta,\beta}\Sigma^{t-1,0} \\ \Sigma^{t-1,0}\Sigma^{-1}_{\beta,\beta}\Sigma_{\beta,\gamma} & \Sigma_{\gamma,\gamma} + \Sigma_{\gamma,\beta}\Sigma^{-1}_{\beta,\beta}(\Sigma^{t-1,0} - \Sigma_{\beta,\beta})\Sigma^{-1}_{\beta,\beta}\Sigma_{\beta,\gamma} \end{bmatrix}\right).$$

as required.

Proof of Theorems 5.2.3 and 5.2.4

The results in Theorems 5.2.3 and 5.2.4 follow from standard results. One way to see this is to view our model as a dynamic linear model (West and Harrison (2006)), where the hidden state is the parameter, but we define the dynamics such that the hidden state does not change over time.

For the case of no change, the dynamic linear model is

$$\begin{aligned} \beta_t &= \beta_{t-1} \\ y_t &= h_0^T \beta_t + \sigma \epsilon_t, \end{aligned}$$

where $\epsilon_t \sim \mathcal{N}(0,1)$, and h_0 is the known regression vector. At time $t-1$, given observations $y_{1:t-1}$, the posterior distribution is:

$$\beta_{t-1} \mid \sigma^2, y_{1:t-1}, \tau \geq t-1 \sim \mathcal{N}(\mu^{t-1,0}, \sigma^2 \Sigma^{t-1,0}),$$

$$\sigma^2 \mid y_{1:t-1}, \tau \geq t-1 \sim IG(\nu^{t-1,0}, \iota^{t-1,0}).$$

After observing y_t, the posterior distribution of (β, σ^2) is updated using the following formulas as show in West and Harrison (2006); Fearnhead and Liu (2011):

$$\begin{aligned} \Sigma^{t,0} &= \Sigma^{t-1,0} - A^\top A Q, \\ \mu^{t,0} &= \mu^{t-1,0} + A e_0, \\ \nu^{t,0} &= \nu^{t-1,0} + \frac{1}{2}, \\ \iota^{t,0} &= \iota^{t-1,0} + \frac{e_0^2}{2Q}, \end{aligned}$$

where $e_0 = y_t - h_0^\top \mu^{t-1,0}$ is the one-step-ahead forecast error, $Q = h_0^\top \Sigma^{t-1,0} h_0 + 1$ is the forecast error variance and $A = \Sigma^{t-1,0} h_0 / Q$ is the adaptive gain. Then the one-step-

ahead predictive distribution of y_t given $y_{1:t-1}$ is:

$$y_t \mid y_{1:t-1}, \tau \geq t-1 \sim t_{2\nu^{t-1,0}} \left(h_0^\top \mu^{t-1,0}, \frac{\iota^{t-1,0}}{\nu^{t-1,0}} \left(h_0^\top \Sigma^{t-1,0} h_0 + 1 \right) \right),$$

where $t_{2\nu^{t-1,0}}$ denotes the Student-t distribution with $2\nu^{t-1,0}$ degrees of freedom.

A similar derivation works for the case where there is a change at time i, but now the dynamic linear model is

$$\begin{aligned} (\beta_t^\top, \gamma_t^\top)^\top &= (\beta_{t-1}^\top, \gamma_{t-1}^\top)^\top \\ y_t &= h_i^T (\beta_t^\top, \gamma_t^\top)^\top + \sigma \epsilon_t. \end{aligned}$$

The model follows the same dynamic linear structures, but with an augmented parameter vector. As such, the posterior and predictive quantities follow immediately by applying the standard DLM update formulas introduced earlier.

C.3 The total variation between two univariate Gaussian with known variance

Corollary C.3.1. *The posterior distributions for neighbouring $f_i(\mu, \sigma^2\Sigma^2)$ and $f_j(\mu, \sigma^2\Sigma^2)$ always intersects at two points as $\sigma^2\Sigma_i^2 > \sigma^2\Sigma_j^2$. Two points of intersection are:*

$$(c_1, c_2) = \frac{\mu_j \sigma^2 \Sigma_i^2 - \mu_i \sigma^2 \Sigma_j^2 \mp \sqrt{\sigma^2\Sigma_i^2 \sigma^2 \Sigma_j^2} \sqrt{(\mu_i - \mu_j)^2 - (\sigma^2\Sigma_i^2 - \sigma^2\Sigma_j^2) \log \frac{\sigma^2\Sigma_j^2}{\sigma^2\Sigma_i^2}}}{\sigma^2\Sigma_i^2 - \sigma^2\Sigma_j^2}.$$

Proposition C.3.2. *The total variation between two normal distributions $f_i(\mu, \sigma^2\Sigma^2)$*

and $f_j(\mu, \sigma^2\Sigma^2)$ *is*

$$2\left[\Phi(\frac{c_2-\mu_i}{\sqrt{\sigma^2\Sigma_i^2}}) - \Phi(\frac{c_1-\mu_i}{\sqrt{\sigma^2\Sigma_i^2}}) + \Phi(\frac{c_1-\mu_j}{\sqrt{\sigma^2\Sigma_j^2}}) - \Phi(\frac{c_2-\mu_j}{\sqrt{\sigma^2\Sigma_j^2}})\right]. \tag{C.1}$$

Let $a = \frac{\mu_i-\mu_j}{\sigma^2\Sigma_j^2-\sigma^2\Sigma_i^2}$ and $b = \frac{\sqrt{(\mu_i-\mu_j)^2+(\sigma^2\Sigma_j^2-\sigma^2\Sigma_i^2)\log\frac{\sigma^2\Sigma_j^2}{\sigma^2\Sigma_i^2}}}{\sigma^2\Sigma_j^2-\sigma^2\Sigma_i^2}$, Formula C.1 could be rewritten as:

$$2\left[\Phi(a\sqrt{\sigma^2\Sigma_i^2}+b\sqrt{\sigma^2\Sigma_j^2}) - \Phi(a\sqrt{\sigma^2\Sigma_i^2}-b\sqrt{\sigma^2\Sigma_j^2}) + \Phi(a\sqrt{\sigma^2\Sigma_j^2}-b\sqrt{\sigma^2\Sigma_i^2}) - \Phi(a\sqrt{\sigma^2\Sigma_j^2}+b\sqrt{\sigma^2\Sigma}\right.$$

C.4 Simulation results under different priors

C.4.1 Priors

For the change-in-mean case, we assume the probability of no change $Pr(\tau \geq 2000) = 0.9$, and the distribution of the changepoint location follows a uniform distribution $Pr(\tau = i|\tau < 2000) \propto 1$ for $i = 1, \cdots, 1999$. To simplify the updating procedure, we consider a model where the pre-change and post-change means are independent, through reparametrization. Specifically, we model the pre-change parameter (β) and post-change parameters ($\theta = \beta + \gamma$). If $\sigma^2 = 1$ is know, we have normal prior with parameters $(\mu, \sigma^2\Sigma)$; or normal-inverse-gamma prior with parameters $(\mu, \Sigma, \nu, \iota)$ if it is unknown. Table C.1 and Table C.2 in the Appendix present the detection power under different priors. In the main text, Table 5.1 shows the results under a Normal prior with parameters $(\mu = 0, \sigma^2\Sigma = 0.25^2)$ when the variance is known, and under a Normal-Inverse-Gamma prior with parameters $(\mu = 0, \Sigma = 10, \nu = 30, \iota = 30)$ when the variance is unknown.

For the change-in-slope case with known variance, we allow the model to detect the type of change, either continuous or discontinuous, assigning equal prior probabilities

$\Pr^{\text{conti}}(\tau < 2000) = \Pr^{\text{dis}}(\tau < 2000) = 0.1$. The parameters before and after the change, (β, γ), are jointly modelled with bivariate normal priors with hyper-parameters (μ_1, μ_2) and $(\delta_1, \delta_2, \delta_3, \delta_4)$ as specified in Section 5.2.1. Different prior settings are considered as shown in Table C.4 and C.3. In the main text, Table 5.2 reports the results for $\mu_1 = \mu_2 = 0$ and $\delta_1 = \delta_2 = \delta_3 = \delta_4 = 1$.

The detection thresholds for each algorithm are chosen to control the false alarm rate at 5%, defined as the probability of detecting a change before time 2000 when no change.

C.4.2 Detection power

Table C.1, Table C.2, Table C.4 and Table C.3 present the simulation results for each example under different prior choices. Overall, the results are consistent with those reported in Section 5.3.

C.4.3 Speed

Here, we calculate the average computational speed at each step of each approach shown in Section 5.3. The main code is written in R, while the merging part is written in C++. The code was executed in a virtualised environment using VMware with full virtualisation, on a machine powered by an Intel(R) Xeon(R) Gold 6248R CPU, featuring 4 physical cores operating at 3.00 GHz. The time is recorded through the R package "microbenchmark" (Mersmann, 2024) which can accurately measure and compare the execution time of R expressions. From Figure C.1, we can see that the exact approach is more expensive than the pruned approaches. The running time of our proposed approach is close to that of the benchmark approach at each time step.

Prior	M	Algorithm	Coverage	Error	Delay	Map
$N(0, 0.25^2)$		Exact	0.95	0.05	281±174	1008±110
	50	Bagel	0.95	0.04	283±176	1010±108
		Benchmark	0.38	0.08	308±198	1009±116
	100	Bagel	0.95	0.05	282±175	1009±107
		Benchmark	0.56	0.06	300±195	1012±102
	200	Bagel	0.95	0.05	281±175	1009±109
		Benchmark	0.74	0.05	294±186	1011±109
$N(0, 0.5^2)$		Exact	0.90	0.08	311±194	1023±106
	50	Bagel	0.91	0.08	314±196	1024±109
		Benchmark	0.41	0.10	332±209	1027±94
	100	Bagel	0.91	0.08	312±194	1023±105
		Benchmark	0.57	0.09	327±208	1029±109
	200	Bagel	0.91	0.08	311±194	1023±106
		Benchmark	0.74	0.09	319±200	1028±116
$N(0, 1^2)$		Exact	0.88	0.09	347±215	1033±114
	50	Bagel	0.89	0.10	348±216	1034±113
		Benchmark	0.40	0.14	358±219	1031±97
	100	Bagel	0.89	0.10	347±214	1034±113
		Benchmark	0.58	0.13	346±210	1031±96
	200	Bagel	0.88	0.09	347±215	1033±114
		Benchmark	0.72	0.11	349±215	1031±101

Table C.1: Simulation results for Example 1 (change-in-mean scenario with known variance) are based on 500 replicates when we vary the value of the variance. The simulated data follow a normal distribution $N(0, 1)$ before time $t = 1000$, and then follow $N(0.25, 1)$ after the change. The priors are specified as $p(\tau \geq 2000) = 0.9$ and $p = 0.1$. The prior column in the tables specifies that the distribution of pre-change and post-change, as they are independently and identically distributed.

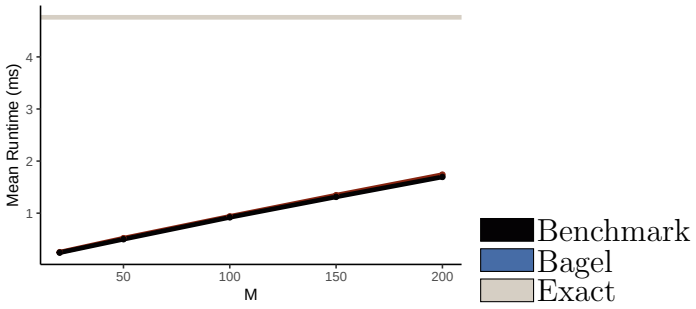


Figure C.1: Average running speed per time step for exact, Bagel, and benchmark approaches against different values of M on data $n = 1000$.

Prior	M	Algorithm	Coverage	Error	Delay	MAP
$NIG(0, 0.1, 30, 30)$		Exact	0.95	0.06	295 ± 189	1025 ±106
	50	Bagel	0.95	0.06	299 ± 192	1022 ± 105
		Benchmark	0.36	0.09	326 ± 213	1025 ± 106
	100	Bagel	0.95	0.06	296 ± 190	1024 ± 106
		Benchmark	0.56	0.08	317 ± 207	1023 ± 104
	200	Bagel	0.94	0.06	295 ± 190	1024 ± 106
		Benchmark	0.74	0.07	306 ± 202	1026 ± 111
$NIG(0, 10, 5, 5)$		Exact	0.87	0.19	409 ± 232	1036 ± 102
	50	Bagel	0.87	0.19	411 ± 232	1037 ± 105
		Benchmark	0.32	0.24	441 ± 250	1040 ± 102
	100	Bagel	0.88	0.18	410 ± 232	1036 ± 103
		Benchmark	0.57	0.22	420 ± 238	1038 ± 105
	200	Bagel	0.88	0.18	411 ± 234	1035 ± 102
		Benchmark	0.75	0.20	413 ± 236	1040 ± 114
$NIG(0, 100, 2.1, 2.1)$		Exact	0.87	0.23	465 ± 246	1040 ± 119
	50	Bagel	0.88	0.24	463 ± 245	1041 ± 119
		Benchmark	0.30	0.32	469 ± 244	1047 ± 121
	100	Bagel	0.87	0.23	466 ± 246	1040 ± 119
		Benchmark	0.54	0.28	465 ± 243	1038 ± 108
	200	Bagel	0.87	0.23	467 ± 247	1040 ± 119
		Benchmark	0.75	0.25	466 ± 247	1040 ± 118

Table C.2: Simulation results for Example 1 (change-in-mean scenario with unknown variance) are based on 500 replicates when we vary the value of the scaled covariance matrix and the prior on the variance. The simulated data follow a normal distribution $N(0, 1)$ before time $t = 1000$, and then follow $N(0.25, 1)$ after the change. The priors are specified as $p(\tau \geq 2000) = 0.9$ and $p = 0.1$. The prior column in the tables specifies that the distribution of pre-change and post-change, as they are independently and identically distributed.

Prior	M	Algorithm	Coverage	Error	Delay	MAP
$\delta_1 = \delta_2 = \delta_3 = \delta_4 = 0.125^2$		Exact	0.86	0.05	304 ± 71	1045 ± 156
	50	Bagel	0.88	0.05	307 ± 73	1048 ± 139
		Benchmark	0.00	0.05	392 ± 104	1216 ± 111
	100	Bagel	0.87	0.05	305 ± 72	1047 ± 145
		Benchmark	0.01	0.05	350 ± 89	1146 ± 96
	200	Bagel	0.87	0.05	304 ± 71	1045 ± 144
		Benchmark	0.05	0.05	325 ± 79	1088 ± 121
$\delta_1 = \delta_2 = \delta_3 = \delta_4 = 0.25^2$		Exact	0.86	0.05	318 ± 73	1033 ± 187
	50	Bagel	0.87	0.05	323 ± 73	1040 ± 155
		Benchmark	0.00	0.05	401 ± 105	1206 ± 114
	100	Bagel	0.86	0.05	319 ± 73	1039 ± 153
		Benchmark	0.01	0.05	364 ± 91	1139 ± 110
	200	Bagel	0.86	0.05	318 ± 73	1031 ± 163
		Benchmark	0.05	0.05	341 ± 82	1079 ± 131
$\delta_1 = \delta_2 = \delta_3 = \delta_4 = 0.5^2$		Exact	0.87	0.05	329 ± 73	1037 ± 190
	50	Bagel	0.86	0.05	336 ± 76	1038 ± 152
		Benchmark	0.00	0.05	399 ± 96	1199 ± 101
	100	Bagel	0.88	0.05	331 ± 74	1042 ± 150
		Benchmark	0.00	0.05	373 ± 90	1140 ± 106
	200	Bagel	0.87	0.05	329 ± 74	1036 ± 159
		Benchmark	0.05	0.05	353 ± 83	1082 ± 136
$\delta_1 = \delta_2 = \delta_3 = \delta_4 = 1$		Exact	0.86	0.05	341 ± 75	1039 ± 167
	50	Bagel	0.75	0.05	350 ± 78	1042 ± 149
		Benchmark	0.00	0.05	414 ± 90	1203 ± 95
	100	Bagel	0.88	0.05	344 ± 75	1039 ± 140
		Benchmark	0.00	0.05	389 ± 89	1144 ± 106
	200	Bagel	0.88	0.05	342 ± 75	1039 ± 139
		Benchmark	0.05	0.05	367 ± 86	1080 ± 134

Table C.3: Simulation results for Example 2 (continuous change-in-slope scenario with known variance) are based on 500 replicates when we vary the value of the scaled covariance matrix. The simulated data follow a normal distribution $N(0, 1)$ before time $t = 1000$, and then undergo a continuous change, gradually shifting to follow $0.002 \times t + N(0, 1)$ after the change. For the known variance setting, the priors are specified as $p(\tau \geq 2000) = 0.8$, $p_{\text{dis}} = 0.1$, $p_{\text{conti}} = 0.1$, $\sigma^2 = 1$ and $\mu_\beta = (0, 0)^\top$.

Prior	M	Algorithm	Coverage	Error	Delay	MAP
$\delta_1 = \delta_2 = \delta_3 = \delta_4 = 0.125^2$		Exact	0.93	0.05	171 ± 63	965 ± 128
	50	Bagel	0.94	0.05	174 ± 64	964 ± 111
		Benchmark	0.07	0.05	246 ± 98	1076 ± 85
	100	Bagel	0.94	0.05	172 ± 64	967 ± 104
		Benchmark	0.29	0.05	206 ± 81	1022 ± 74
	200	Bagel	0.93	0.05	172 ± 63	962 ± 125
		Benchmark	0.63	0.05	186 ± 71	984 ± 92
$\delta_1 = \delta_2 = \delta_3 = \delta_4 = 0.25^2$		Exact	0.94	0.05	186 ± 66	953 ± 144
	50	Bagel	0.95	0.05	191 ± 68	958 ± 109
		Benchmark	0.08	0.05	254 ± 99	1071 ± 89
	100	Bagel	0.94	0.05	188 ± 67	958 ± 109
		Benchmark	0.26	0.05	223 ± 80	1016 ± 79
	200	Bagel	0.94	0.05	187 ± 66	952 ± 123
		Benchmark	0.61	0.05	202 ± 75	971 ± 108
$\delta_1 = \delta_2 = \delta_3 = \delta_4 = 0.5^2$		Exact	0.93	0.05	198 ± 68	959 ± 145
	50	Bagel	0.93	0.05	205 ± 70	949 ± 137
		Benchmark	0.05	0.05	253 ± 89	1062 ± 75
	100	Bagel	0.92	0.05	199 ± 69	954 ± 106
		Benchmark	0.22	0.05	231 ± 80	1017 ± 83
	200	Bagel	0.92	0.05	198 ± 68	949 ± 116
		Benchmark	0.55	0.05	214 ± 75	978 ± 105
$\delta_1 = \delta_2 = \delta_3 = \delta_4 = 1$		Exact	0.92	0.05	211 ± 69	959 ± 129
	50	Bagel	0.91	0.05	223 ± 75	953 ± 164
		Benchmark	0.04	0.05	267 ± 89	1065 ± 72
	100	Bagel	0.91	0.05	212 ± 70	952 ± 107
		Benchmark	0.21	0.05	247 ± 82	1021 ± 78
	200	Bagel	0.90	0.05	211 ± 69	952 ± 104
		Benchmark	0.52	0.05	228 ± 77	978 ± 103

Table C.4: Simulation results for Example 2 (discontinuous change-in-slope scenario with known variance) are based on 500 replicates when we vary the value of the scaled covariance matrix. The simulated data follow a normal distribution $N(0, 1)$ before time $t = 1000$, and then undergo a continuous change, gradually shifting to follow $-1.75 + 0.002 \times t + N(0, 1)$ after the change. For the known variance setting, the priors are specified as $p(\tau \geq 2000) = 0.8$, $p_{\text{dis}} = 0.1$, $p_{\text{conti}} = 0.1$, $\sigma^2 = 1$ and $\mu_\beta = (0, 0)^\top$.

Bibliography

Adams, R. P. and MacKay, D. J. (2007). Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*.

Ahmad, S., Lavin, A., Purdy, S., and Agha, Z. (2017). Unsupervised real-time anomaly detection for streaming data. *Neurocomputing*, 262:134–147.

Ahmad, S. and Purdy, S. (2016). Real-time anomaly detection for streaming analytics. *arXiv preprint arXiv:1607.02480*.

Akaike, H. (1974). A new look at the statistical model identification. *IEEE transactions on automatic control*, 19(6):716–723.

Akman, V. E. and Raftery, A. E. (1986). Bayes factors for non-homogeneous poisson processes with vague prior information. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 48(3):322–329.

Alami, R. (2023). Bayesian changepoint detection for bandit feedback in non-stationary environments. In Khan, E. and Gonen, M., editors, *Proceedings of The 14th Asian Conference on Machine Learning*, volume 189 of *Proceedings of Machine Learning Research*, pages 17–31. PMLR.

Alrashdi, I., Alqazzaz, A., Aloufi, E., Alharthi, R., Zohdy, M., and Ming, H. (2019). Ad-IoT: Anomaly detection of IoT cyberattacks in smart city using machine learning.

In *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0305–0310. IEEE.

Altamirano, M., Briol, F.-X., and Knoblauch, J. (2023). Robust and scalable Bayesian online changepoint detection. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J., editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 642–663. PMLR.

Appadwedula, S., Veeravalli, V. V., and Jones, D. L. (2005). Energy-efficient detection in sensor networks. *IEEE Journal on Selected Areas in Communications*, 23(4):693–702.

Aue, A., Horváth, L., Kühn, M., and Steinebach, J. (2012). On the reaction time of moving sum detectors. *Journal of Statistical Planning and Inference*, 142(8):2271–2288.

Aue, A. and Kirch, C. (2024). The state of cumulative sum sequential changepoint testing 70 years after page. *Biometrika*, 111(2):367–391.

Auger, I. E. and Lawrence, C. E. (1989). Algorithms for the optimal identification of segment neighborhoods. *Bulletin of Mathematical Biology*, 51(1):39–54.

Banerjee, T. and Veeravalli, V. V. (2015). Data-efficient quickest change detection in sensor networks. *IEEE Transactions on Signal Processing*, 63(14):3727–3735.

Baranowski, R., Chen, Y., and Fryzlewicz, P. (2019). Narrowest-over-threshold detection of multiple changepoints and changepoint-like features. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 81(3):649–672.

Bardwell, L. and Fearnhead, P. (2017). Bayesian Detection of Abnormal Segments in Multiple Time Series. *Bayesian Analysis*, 12(1):193 – 218.

Barigozzi, M., Cho, H., and Trapani, L. (2024). Moving sum procedure for multiple change point detection in large factor models. *arXiv preprint arXiv:2410.02918*.

Barry, D. and Hartigan, J. A. (1993). A Bayesian analysis for change point problems. *Journal of the American Statistical Association*, 88(421):309–319.

Basseville, M. and Nikiforov, I. V. (1993). *Detection of Abrupt Changes: Theory and Application*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Bell, C., Gordon, L., and Pollak, M. (1994). An efficient nonparametric detection scheme and its application to surveillance of a bernoulli process with unknown baseline. *Lecture Notes-Monograph Series*, pages 7–27.

Benson, A. and Friel, N. (2018). Adaptive MCMC for multiple changepoint analysis with applications to large datasets. *Electronic Journal of Statistics*, 12(2):3365 – 3396.

Bhaduri, K., Das, K., and Votava, P. (2010). Distributed anomaly detection using satellite data from multiple modalitie. In *CIDU*, pages 109–123. Citeseer.

Broemeling, L. D. (2017). *Bayesian analysis of linear models*. CRC Press.

Carlin, B. P., Gelfand, A. E., and Smith, A. F. (1992). Hierarchical Bayesian analysis of changepoint problems. *Journal of the royal statistical society: series C (applied statistics)*, 41(2):389–405.

Chakar, S., Lebarbier, E., Lévy-Leduc, C., and Robin, S. (2017). A robust approach for estimating changepoints in the mean of an AR(1) process. *Bernoulli*, 23(2):1408 – 1447.

Chan, H. P. (2017). Optimal sequential detection in multi-stream data. *The Annals of Statistics*, 45(6):2736–2763.

Chen, Y., Wang, T., and Samworth, R. J. (2022). High-dimensional, multiscale online changepoint detection. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 84(1):234–266.

Chen, Y., Wang, T., and Samworth, R. J. (2024). Inference in high-dimensional online changepoint detection. *Journal of the American Statistical Association*, 119(546):1461–1472.

Chernoff, H. and Zacks, S. (1964). Estimating the current mean of a normal distribution which is subjected to changes in time. *The Annals of Mathematical Statistics*, 35(3):999–1018.

Chib, S. (1996). Calculating posterior distributions and modal estimates in markov mixture models. *Journal of Econometrics*, 75(1):79–97.

Chib, S. (1998). Estimation and comparison of multiple changepoint models. *Journal of Econometrics*, 86(2):221–241.

Chin Choy, J. and Broemeling, L. (1980). Some Bayesian inferences for a changing linear model. *Technometrics*, 22(1):71–78.

Cho, H. and Fryzlewicz, P. (2014). Multiple changepoint detection for high dimensional time series via sparsified Binary Segmentation. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 77:475–507.

Cho, H. and Kirch, C. (2022). Bootstrap confidence intervals for multiple changepoints based on moving sum procedures. *Computational Statistics & Data Analysis*, 175:107552.

Cho, H. and Kirch, C. (2024). Data segmentation algorithms: univariate mean change and beyond. *Econometrics and Statistics*, 30:76–95.

Chu, C.-S. J., Stinchcombe, M., and White, H. (1996). Monitoring structural change. *Econometrica: Journal of the Econometric Society*, pages 1045–1065.

Detommaso, G., Hoitzing, H., Cui, T., and Alamir, A. (2019). Stein variational online changepoint detection with applications to hawkes processes and neural networks. *arXiv preprint arXiv:1901.07987*.

Eichinger, B. and Kirch, C. (2018). A MOSUM procedure for the estimation of multiple random changepoints. *Bernoulli*, 24(1):526–564.

Enikeeva, F. and Harchaoui, Z. (2019). High-dimensional changepoint detection under sparse alternatives. *The Annals of Statistics*, 47(4):2051–2079.

Fearnhead, P. (2006). Exact and efficient Bayesian inference for multiple changepoint problems. *Statistics and Computing*, 16:203–213.

Fearnhead, P. and Fryzlewicz, P. (2022). Detecting a single changepoint. *arXiv preprint arXiv:2210.07066*.

Fearnhead, P. and Liu, Z. (2007). Online inference for multiple changepoint problems. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 69(4):589–605.

Fearnhead, P. and Liu, Z. (2011). Efficient Bayesian analysis of multiple changepoint models with dependence across segments. *Statistics and Computing*, 21:217–229.

Fearnhead, P., Maidstone, R., and Letchford, A. (2019). Detecting changes in slope with an l 0 penalty. *Journal of Computational and Graphical Statistics*, 28(2):265–275.

Fearnhead, P. and Rigaill, G. (2019). Changepoint detection in the presence of outliers. *Journal of the American Statistical Association*, 114(525):169–183.

Fisch, A. T., Bardwell, L., and Eckley, I. A. (2022a). Real time anomaly detection and categorisation. *Statistics and Computing*, 32(4):55.

Fisch, A. T. M., Eckley, I. A., and Fearnhead, P. (2022b). A linear time method for the detection of collective and point anomalies. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 15(4):494–508.

Fryzlewicz, P. (2014). Wild binary segmentation for multiple changepoint detection. *The Annals of Statistics*, 6:2243 – 2281.

Fryzlewicz, P. (2020). Detecting possibly frequent changepoints: Wild Binary Segmentation 2 and steepest-drop model selection. *Journal of the Korean Statistical Society*, 49(4):1027–1070.

Gallant, A. R. (2009). *Nonlinear statistical models*. John Wiley & Sons.

Gombay, E. and Serban, D. (2009). Monitoring parameter change in AR (p) time series models. *Journal of Multivariate Analysis*, 100(4):715–725.

Gordon, L. and Pollak, M. (1995). A robust surveillance scheme for stochastically ordered alternatives. *The Annals of Statistics*, pages 1350–1375.

Gordon, L. and Pollak, M. (1997). Average run length to false alarm for surveillance schemes designed with partially specified pre-change distribution. *The Annals of Statistics*, 25(3):1284–1310.

Gösmann, J., Kley, T., and Dette, H. (2021). A new approach for open-end sequential changepoint monitoring. *Journal of Time Series Analysis*, 42(1):63–84.

Gösmann, J., Stoehr, C., Heiny, J., and Dette, H. (2022). Sequential change point detection in high dimensional time series. *Electronic Journal of Statistics*, 16(1):3608–3671.

Green, P. J. (1995). Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82(4):711–732.

Gundersen, G. W., Cai, D., Zhou, C., Engelhardt, B. E., and Adams, R. P. (2021). Active multi-fidelity Bayesian online changepoint detection. In *Uncertainty in Artificial Intelligence*, pages 1916–1926. PMLR.

Haynes, K., Fearnhead, P., and Eckley, I. A. (2017). A computationally efficient nonparametric approach for changepoint detection. *Statistics and computing*, 27:1293–1305.

Held, L., Hofmann, M., Höhle, M., and Schmid, V. (2006). A two-component model for counts of infectious diseases. *Biostatistics*, 7(3):422–437.

Hoadley, K. A., Yau, C., Hinoue, T., Wolf, D. M., Lazar, A. J., Drill, E., Shen, R., Taylor, A. M., Cherniack, A. D., Thorsson, V., et al. (2018). Cell-of-origin patterns dominate the molecular classification of 10,000 tumors from 33 types of cancer. *Cell*, 173(2):291–304.

Horváth, L. and Hušková, M. (2012). Changepoint detection in panel data. *Journal of Time Series Analysis*, 33(4):631–648.

Horváth, L., Hušková, M., Kokoszka, P., and Steinebach, J. (2004). Monitoring changes in linear models. *Journal of statistical Planning and Inference*, 126(1):225–251.

Horváth, L., Kühn, M., and Steinebach, J. (2008). On the performance of the fluctuation test for structural change. *Sequential Analysis*, 27(2):126–140.

Horváth, L., Liu, Z., and Lu, S. (2022). Sequential monitoring of changes in dynamic linear models, applied to the US housing market. *Econometric Theory*, 38(2):209–272.

Hsu, F.-H., Chen, H.-I. H., Tsai, M.-H., Lai, L.-C., Huang, C.-C., Tu, S.-H., Chuang, E. Y., and Chen, Y. (2011). A model-based circular binary segmentation algorithm for the analysis of array cgh data. *BMC Research Notes*, 4:1–12.

Hušková, M. and Kirch, C. (2012). Bootstrapping sequential changepoint tests for linear regression. *Metrika*, 75(5):673–708.

Iglewicz, B. and Hoaglin, D. C. (1993). *Volume 16: how to detect and handle outliers*. Quality Press.

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(2):105–108.

Jesmeen, M., Hossen, J., and Aziz, A. B. A. (2021). Unsupervised anomaly detection for energy consumption in time series using clustering approach. *Emerging Science Journal*, 5(6):840–854.

Katser, I. D. and Kozitsin, V. O. (2020). Skoltech anomaly benchmark (skab). https://www.kaggle.com/dsv/1693952.

Kengne, W. and Ngongo, I. S. (2022). Inference for nonstationary time series of counts with application to changepoint problems. *Annals of the Institute of Statistical Mathematics*, 74(4):801–835.

Kiefer, N. M. and Vogelsang, T. J. (2002a). Heteroskedasticity-autocorrelation robust standard errors using the Bartlett kernel without truncation. *Econometrica*, 70(5):2093–2095.

Kiefer, N. M. and Vogelsang, T. J. (2002b). Heteroskedasticity-autocorrelation robust testing using bandwidth equal to sample size. *Econometric Theory*, 18(6):1350–1366.

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(500):1590–1598.

Kim, J. and Cheon, S. (2010). Bayesian multiple changepoint estimation with annealing stochastic approximation monte carlo. *Computational Statistics*, 25(2):215–239.

Kim, J., Oh, H.-S., and Cho, H. (2024). Moving sum procedure for change point detection under piecewise linearity. *Technometrics*, pages 1–10.

Kirch, C. and Kamgaing, J. T. (2015). On the use of estimating functions in monitoring time series for changepoints. *Journal of Statistical Planning and Inference*, 161:25–49.

Kirch, C. and Weber, S. (2018). Modified sequential change point procedures based on estimating functions. *Electronic Journal of Statistics*, 12(1):1579–1613.

Kovács, S., Bühlmann, P., Li, H., and Munk, A. (2023). Seeded binary segmentation: a general methodology for fast and optimal changepoint detection. *Biometrika*, 110(1):249–256.

Krieger, A. M., Pollak, M., and Yakir, B. (2003). Surveillance of a simple linear regression. *Journal of the American Statistical Association*, 98(462):456–469.

Kurozumi, E. (2020). Asymptotic properties of bubble monitoring tests. *Econometric Reviews*, 39(5):510–538.

Kurozumi, E. (2021). Asymptotic behavior of delay times of bubble monitoring tests. *Journal of Time Series Analysis*, 42(3):314–337.

Kurozumi, E. (2023). Fluctuation-type monitoring test for explosive behavior. *Econometrics and Statistics*.

Lai, T. L. (1998). Information bounds and quick detection of parameter changes in stochastic systems. *IEEE Transactions on Information theory*, 44(7):2917–2929.

Lavin, A. and Ahmad, S. (2015). Evaluating real-time anomaly detection algorithms–the numenta anomaly benchmark. In *2015 IEEE 14th international conference on machine learning and applications (ICMLA)*, pages 38–44. IEEE.

Leisch, F., Hornik, K., and Kuan, C.-M. (2000). Monitoring structural changes with the generalized fluctuation test. *Econometric Theory*, 16(6):835–854.

Liu, K., Zhang, R., and Mei, Y. (2019). Scalable SUM-shrinkage schemes for distributed monitoring large-scale data streams. *Statistica Sinica*, 29:1–22.

Lorden, G. (1971). Procedures for reacting to a change in distribution. *The Annals of Mathematical Statistics*, pages 1897–1908.

Lorden, G. and Pollak, M. (2008). Sequential changepoint detection procedures that are nearly optimal and computationally simple. *Sequential Analysis*, 27(4):476–512.

Lung-Yut-Fong, A., Lévy-Leduc, C., and Cappé, O. (2012). Distributed detection/localization of changepoints in high-dimensional network traffic data. *Statistics and Computing*, 22(2):485–496.

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

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR.

Mei, Y. (2005). Information bounds and quickest change detection in decentralized decision systems. *IEEE Transactions on Information Theory*, 51(7):2669–2681.

Mei, Y. (2010). Efficient scalable schemes for monitoring a large number of data streams. *Biometrika*, 97(2):419–433.

Mei, Y. (2011). Quickest detection in censoring sensor networks. In *2011 IEEE International Symposium on Information Theory Proceedings*, pages 2148–2152.

Mersmann, O. (2024). *microbenchmark: Accurate Timing Functions*. R package version 1.4.10.

Mouliere, F., Piskorz, A. M., Chandrananda, D., Moore, E., Morris, J., Smith, C. G., Goranova, T., Heider, K., Mair, R., Supernat, A., et al. (2017). Selecting short dna fragments in plasma improves detection of circulating tumour dna. *BioRxiv*, page 134437.

Moustakides, G. V. (1986). Optimal stopping times for detecting changes in distributions. *The Annals of Statistics*, 14(4):1379–1387.

Newey, W. K. and West, K. D. (1986). *A simple, positive semi-definite, heteroskedasticity and autocorrelationconsistent covariance matrix*. National Bureau of Economic Research Cambridge, Mass., USA.

Ninomiya, Y. (2015). Changepoint model selection via aic. *Annals of the Institute of Statistical Mathematics*, 67:943–961.

Olshen, A. B., Bengtsson, H., Neuvial, P., Spellman, P. T., Olshen, R. A., and Seshan, V. E. (2011). Parent-specific copy number in paired tumor–normal studies using circular binary segmentation. *Bioinformatics*, 27(15):2038–2046.

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(4):557–572.

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

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

Pinsker, M. S. (1964). Information and information stability of random variables and processes. *Holden-Day*.

Pinto, G. and Castor, F. (2017). Energy efficiency: A new concern for application software developers. *Communications of the ACM*, 60(12):68–75.

Pishchagina, L., Romano, G., Fearnhead, P., Runge, V., and Rigaill, G. (2023). Online multivariate changepoint detection: Leveraging links with computational geometry. arXiv:2311.01174.

Politis, D. N. and Romano, J. P. (1994). The stationary bootstrap. *Journal of the American Statistical association*, 89(428):1303–1313.

Pollak, M. and Siegmund, D. (1991). Sequential detection of a change in a normal mean when the initial value is unknown. *The Annals of Statistics*, 19(1):394–416.

Polunchenko, A. S. and Tartakovsky, A. G. (2010). On optimality of the shiryaev–roberts procedure for detecting a change in distribution. *The Annals of Statistics*, 38:3445–3457.

Rago, C., Willett, P., and Bar-Shalom, Y. (1996). Censoring sensors: a low-communication-rate scheme for distributed detection. *IEEE Transactions on Aerospace and Electronic Systems*, 32(2):554–568.

Rigaill, G. (2015). A pruned dynamic programming algorithm to recover the best segmentations with 1 to $k\_\{max\}$ changepoints. *Journal de la Société Française de Statistique*, 156(4):180–205.

Ritov, Y. (1990). Decision theoretic optimality of the CUSUM procedure. *The Annals of Statistics*, pages 1464–1469.

Robbins, M. W., Gallagher, C. M., and Lund, R. B. (2016). A general regression changepoint test for time series data. *Journal of the American Statistical Association*, 111(514):670–683.

Romano, G., Eckley, I. A., and Fearnhead, P. (2023a). A log-linear non-parametric online changepoint detection algorithm based on functional pruning. *IEEE Transactions on Signal Processing*.

Romano, G., Eckley, I. A., Fearnhead, P., and Rigaill, G. (2023b). Fast online changepoint detection via functional pruning cusum statistics. *Journal of Machine Learning Research*, 24:1–36.

Romano, G., Rigaill, G., Runge, V., and Fearnhead, P. (2022). Detecting abrupt changes in the presence of local fluctuations and autocorrelated noise. *Journal of the American Statistical Association*, 117(540):2147–2162.

Ross, G. J. (2013). Modelling financial volatility in the presence of abrupt changes. *Physica A: Statistical Mechanics and its Applications*, 392(2):350–360.

Runge, V., Pascucci, M., and de Boishebert, N. D. (2020). Change-in-slope optimal partitioning algorithm in a finite-size parameter space. *arXiv preprint arXiv:2012.11573*.

Saatçi, Y., Turner, R. D., and Rasmussen, C. E. (2010). Gaussian process changepoint models. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 927–934.

Salazar, D. (1982). Structural changes in time series models. *Journal of Econometrics*, 19(1):147–163.

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics*, pages 461–464.

Scott, A. J. and Knott, M. (1974). A cluster analysis method for grouping means in the analysis of variance. *Biometrics*, pages 507–512.

Shamp, W., Varbanov, R., Chicken, E., Linero, A., and Yang, Y. (2021). Computationally efficient Bayesian sequential function monitoring. *Journal of Quality Technology*, 54(1):1–19.

Soch, J. and Allefeld, C. (2016). Kullback-leibler divergence for the normal-gamma distribution. *arXiv preprint arXiv:1611.01437*.

Stephens, D. A. (1994). Bayesian retrospective multiple-changepoint identification. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 43(1):159–178.

Tartakovsky, A., Nikiforov, I., and Basseville, M. (2014). *Sequential analysis: Hypothesis testing and changepoint detection*. CRC press.

Tartakovsky, A. G. and Kim, H. (2006). Performance of certain decentralized distributed change detection procedures. In *2006 9th International Conference on Information Fusion*, pages 1–8. IEEE.

Tartakovsky, A. G. and Polunchenko, A. S. (2008). Quickest changepoint detection in distributed multisensor systems under unknown parameters. In *2008 11th International Conference on Information Fusion*, pages 1–8. IEEE.

Tartakovsky, A. G., Rozovskii, B. L., Blažek, R. B., and Kim, H. (2006). Detection of intrusions in information systems by sequential changepoint methods. *Statistical methodology*, 3(3):252–293.

Tartakovsky, A. G. and Veeravalli, V. V. (2002). An efficient sequential procedure for detecting changes in multichannel and distributed systems. In *Proceedings of the Fifth International Conference on Information Fusion. FUSION 2002. (IEEE Cat.No.02EX5997)*, volume 1, pages 41–48 vol.1.

Thies, S. and Molnár, P. (2018). Bayesian change point analysis of bitcoin returns. *Finance Research Letters*, 27:223–227.

Thomson, J. R., Kimmerer, W. J., Brown, L. R., Newman, K. B., Nally, R. M., Bennett, W. A., Feyrer, F., and Fleishman, E. (2010). Bayesian change point analysis of abundance trends for pelagic fishes in the upper san francisco estuary. *Ecological Applications*, 20(5):1431–1448.

Tickle, S. O., Eckley, I., Fearnhead, P., and Haynes, K. (2020). Parallelization of a common changepoint detection method. *Journal of Computational and Graphical Statistics*, 29(1):149–161.

Truong, C., Oudre, L., and Vayatis, N. (2020). Selective review of offline change point detection methods. *Signal Processing*, 167:107299.

Tveten, M., Eckley, I. A., and Fearnhead, P. (2022). Scalable changepoint and anomaly detection in cross-correlated data with an application to condition monitoring. *The Annals of Applied Statistics*, 16(2):721–743.

Varghese, B., Wang, N., Barbhuiya, S., Kilpatrick, P., and Nikolopoulos, D. S. (2016). Challenges and opportunities in edge computing. In *2016 IEEE International Conference on Smart Cloud (SmartCloud)*, pages 20–26.

Veeravalli, V. V. (2001). Decentralized quickest change detection. *IEEE Transactions on Information Theory*, 47(4):1657–1665.

Venkatraman, E. and Olshen, A. B. (2007). A faster circular binary segmentation algorithm for the analysis of array cgh data. *Bioinformatics*, 23(6):657–663.

Verma, B. K., Verma, M., Verma, V. K., Abdullah, R. B., Nath, D. C., Khan, H. T., Verma, A., Vishwakarma, R. K., and Verma, V. (2020). Global lockdown: An effective safeguard in responding to the threat of covid-19. *Journal of Evaluation in Clinical Practice*, 26(6):1592–1598.

Wald, A. (1945). Sequential tests of statistical hypotheses. *The Annals of Mathematical Statistics*, 16(2):117–186.

Waldmann, K.-H. (1996). Design of double CUSUM quality control schemes. *European Journal of Operational Research*, 95(3):641–648.

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

Ward, K., Dilillo, G., Eckley, I., and Fearnhead, P. (2023). Poisson-focus: An efficient online method for detecting count bursts with application to gamma ray burst detection. *Journal of the American Statistical Association*, pages 1–13.

Ward, K., Romano, G., Eckley, I., and Fearnhead, P. (2024). A constant-per-iteration likelihood ratio test for online changepoint detection for exponential family models. *Statistics and Computing*, 34(3):1–11.

Weber, S. M. (2017). *ChangePoint Procedures for Multivariate Dependent Data*. PhD thesis, Karlsruher Institut für Technologie (KIT).

West, M. and Harrison, J. (2006). *Bayesian forecasting and dynamic models*. Springer Science & Business Media.

White, H. and Domowitz, I. (1984). Nonlinear regression with dependent observations. *Econometrica: Journal of the Econometric Society*, pages 143–161.

Williams, C. K. and Rasmussen, C. E. (2006). *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA.

Wu, H., Hu, J., Sun, J., and Sun, D. (2019). Edge computing in an IoT base station system: Reprogramming and real-time tasks. *Complexity*, 2019:4027638:1–4027638:10.

Xie, Y. and Siegmund, D. (2013a). Sequential multi-sensor changepoint detection. *The Annals of Statistics*, 41(2):670–692.

Xie, Y. and Siegmund, D. (2013b). Sequential multi-sensor changepoint detection. In *2013 Information Theory and Applications Workshop (ITA)*, pages 1–20. IEEE.

Xuan, X. and Murphy, K. (2007). Modeling changing dependency structure in multivariate time series. In *Proceedings of the 24th international conference on Machine learning*, pages 1055–1062.

Yang, Z., Eckley, I. A., and Fearnhead, P. (2024). A communication-efficient, online changepoint detection method for monitoring distributed sensor networks. *Statistics and Computing*, 34(3):1–16.

Yao, Y.-C. (1988). Estimating the number of changepoints via schwarz'criterion. *Statistics & Probability Letters*, 6(3):181–189.

Yao, Y.-C. and Davis, R. A. (1986). The asymptotic behavior of the likelihood ratio statistic for testing a shift in mean in a sequence of independent normal variates. *Sankhyā: The Indian Journal of Statistics, Series A*, pages 339–353.

Yau, C. Y., Sze Him Isaac, L., and Ng, W. L. (2017). Sequential changepoint detection in time series models based on pairwise likelihood. *Statistica Sinica*, 27.

Yu, Y., Padilla, O. H. M., Wang, D., and Rinaldo, A. (2020). A note on online change point detection. *arXiv preprint arXiv:2006.03283*.

Zeileis, A., Leisch, F., Kleiber, C., and Hornik, K. (2005). Monitoring structural change in dynamic econometric models. *Journal of Applied Econometrics*, 20(1):99–121.

Zhang, R. and Mei, Y. (2018). Asymptotic statistical properties of communication-efficient quickest detection schemes in sensor networks. *Sequential Analysis*, 37(3):375–396.

Zhao, L., Guo, D., Xie, J., Luo, L., and Shen, Y. (2023). A closed-loop hybrid supervision framework of cryptocurrency transactions for data trading in iot. *ACM Transactions on Internet of Things*, 4(1):1–26.