Introduction to changepoint detection from first principles

A. Liehrmann and G. Rigaill

March 3, 2025

Abstract

This is a draft set of notes for a 3-4-day introduction on changepoints.

In recent years, there has been a proliferation of methods for detecting changepoints (also known as breakpoints or structural breaks) in data streams. This surge has been driven by the wide range of applications where changepoint methods are needed, including genomics, neuroscience, climate science, finance, and econometrics, among others. These notes serve as an introduction to multiple changepoint detection methods. This course first addresses the simpler task of detecting a single changepoint in the mean of a univariate data stream. This is crucial for understanding several state-of-the-art approaches designed for detecting multiple changepoints. Subsequently, we will delve into the fundamentals of multiple changepoint inference based on optimizing a penalized likelihood with a constant per changepoint penalty. In the final chapter, we characterize some statistical and computational limits of this approach and present relatively recent solutions to address these issues. We illustrate these concepts using the Python and R programming languages.

Chapter 1

Introduction

Detecting and locating changes in distribution within time series data presents a fundamental statistical challenge. The first studies on changepoint detection emerged in the 1940s [Wald, 1945, Page, 1954], but a significant increase in research activity in this area has occurred in recent decades (see [Venkatraman and Olshen, 2007, Killick et al., 2012, Fryzlewicz, 2014, Maidstone et al., 2017] among many others).

1.1 A simple illustration

Here we will consider data taken from [Lai et al., 2005] and easily available in R using the changepoint package [Killick and Eckley, 2014]. As explain in [Lai et al., 2005]:

"Array Comparative Genomic Hybridization (CGH) can reveal chromosomal aberrations in the genomic DNA. These amplifications and deletions at the DNA level are important in the pathogenesis of cancer and other diseases."

Abrupt changes in the intensity of the signal reveal these aberrations. In the left panel of figure 1.1, we observe a particular CGH/DNA copy number profile.

From a data-analysis point of view the questions are

- Are there changes in the intensity that reveal chromosomal aberrations?
- If so,
  - How many are there? In this particular example, this number could be interpreted as a measure of genomic instability and/or used as a feature to classify tumors [Vincent-Salomon et al., 2013].
  - Where are these changes located? The exact locations of these chromosome rearrangements can be biologically meaningful, as they may involve splitting or fusing certain key genes involved in tumor progression [Nam et al., 2007].
  - Which chromosomal regions are amplified or deleted? These may contain candidate oncogenes or tumor suppressor genes [Hyman et al., 2002].

From a statistical point of view to summarize, we aim to detect abrupt changes in the signal. In the right panel of figure 1.1, we represent the output obtained using a piecewise constant changepoint model with Gaussian errors and constant variance, inferred using a penalized maximum likelihood approach [Yao and Au, 1989, Auger and Lawrence, 1989, Maidstone et al., 2017]. This model identifies six abrupt changes corresponding to three amplified regions. In a bit more detail, the problem can be decomposed as follows:

Detection Has a change occurred?

How many If there are changes, how many are there?

No Spurious Related to the two previous points, we would like to avoid the detection of false changes (also called spurious changes).

Signal Estimation If there is a change, what is the intensity before and after?

Localization If there are changes, where are they, and how confident are we about their locations?

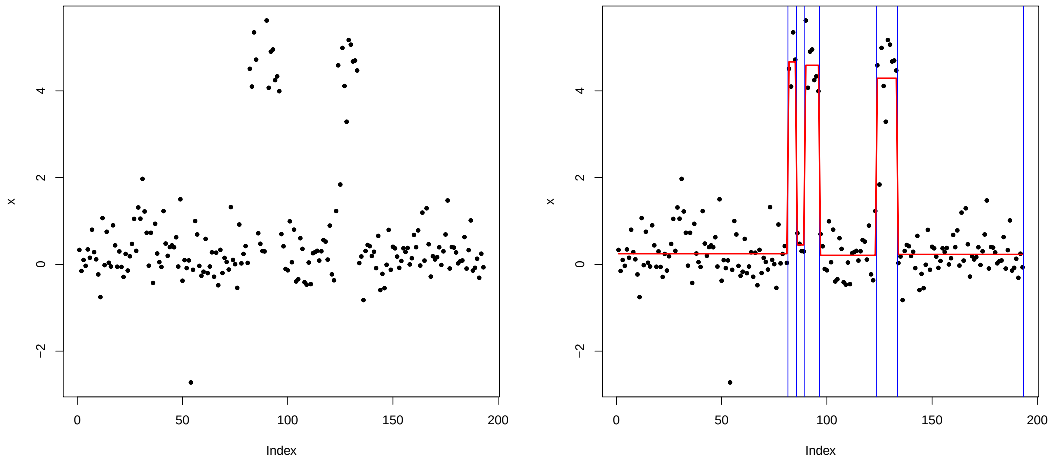

Figure 1.1:

1.2 Many types of changes

The previous, arguably simple example will serve as the basis to formalize the problem. Let's consider some chunk of the data

$$y_1, y_2, \ldots, y_n.$$

Assuming the y_i are Gaussian, we will say that there is a change at (some unknown) position τ if the mean of the signal before τ differs from the mean after τ. It should be clear that if n or the mean difference are too small there is no hope to detect this change. In the rest of this course we will spend a lot of time on this arguably simple model [1]. But changepoint detection goes beyond this. We now illustrate this point with a few examples from the literature.

1. In genomics, with the advent of sequencing techniques on typically aim to model counts using for example a Poisson distribution or a Negative Binomial model (e.g. [Cleynen and Lebarbier, 2014]).

2. In many cases changes are in another moment of the distribution. A classical scenario is the variance (see for example [Killick et al., 2010]

3. The change may also affect several parameters of the distribution say the mean and variance (see for example [Picard et al., 2005, Pishchagina et al., 2023])

4. In a multivariate (possibly high-dimensional) setting one could aim to detect changes in a regression model or graphical models.[Kovács et al., 2023, Enikeeva and Harchaoui, 2019]

5. Yet, another example would be the detection of changes in network structure [Enikeeva and Klopp, 2021, Schwaller and Robin, 2017]

6. Non-parametrically, one could aim to detect changes in distribution (e.g. [Garreau and Arlot, 2018, James and Matteson, 2015]

[1] As we will discuss a bit later, this model is much more complex than one would first think.

A principled approach As already stated, we will primarily consider the archetypical univariate change-in-mean model. Using this example, we hope to provide a solid understanding of the core principles of changepoint detection in a short amount of time. Some of the key challenges and difficulties related to changepoint detection are already present in this univariate change-in-mean model, and working with a Gaussian distribution facilitates mathematical exposition and implementation (in Python or R). Further to our knowledge, most modern approaches to changepoint detection explicitly (or implicitly in their proofs) recursively or iteratively apply ideas developed for the offline detection of at most one changepoint.

Before proceeding, we emphasize that, perhaps surprisingly, inferring this multiple change-in-mean model is far from straightforward. Despite its apparent simplicity, it remains an active area of research (see, for example,[Killick et al., 2012, Maidstone et al., 2017, Cho and Kirch, 2019, Verzelen et al., 2023]).

To understand why, let us briefly outline the key challenges. If we consider an At Most One Changepoint model (often referred as AMOC) the vanilla approach is to use maximum likelihood inference. Noting our data

$$y_1, y_2, \dots, y_n$$

and assuming the y_i are Gaussian

$$y_t = \mu_t + \varepsilon_t \qquad \varepsilon_t \sim \mathcal{N}(0,1),$$

as explained in detail in Section 2.2.2 to test for the presence of a change in the mean at τ we will compare the mean square error with a change at τ :

$$\sum_{t=1}^{\tau} (y_t - \bar{y_{1:\tau}})^2 + \sum_{t=\tau+1}^{n} (y_t - \bar{y_{\tau+1:n}})^2,$$

and the mean square error without a change

$$\sum_{t=1}^{n} (y_t - \bar{y_{1:n}})^2,$$

where

$$\bar{y_{i:j}} = \sum_{t=i}^{j} y_t/(j-i+1)$$

is the mean of segment $i : j = \{i, \dots, j\}$. Informally, considering just one change, a large difference between these two indicates a change.

The key difficulty is that we need to consider not just one change but $n-1$ possible changes. In other words, with n data points we aim to study/compare $n-1$ models. To do that at least conceptually we will compute all $n-1$ mean squared errors and we seek to control their variation to decide/infer whether there is a change or not. It should be clear that the mean squared error for a change at τ and $\tau+1$ are in most cases close and this should help inference. However, precisely exploiting this "link" computationally and/or statistically is not trivial. When considering more than one change the total number of model increases as $\binom{n}{K}$, with K the number of changes and the statistical and computational complexity of the problem both increase.

One last comment, on these notes before we begin, it is not a tutorial on how to use some particular approach/method for changepoint detection. Some tutorials already exist. Our hope is rather to provide a solid understanding of what to look for and test in these packages. Finally, we focus on approaches using optimizing globally or locally a penalized likelihood. This does not include several valid approaches to changepoint inference (notably regularization using for example the fused-lasso or Bayesian inference).

Chapter 2

At most one change offline: essential statistical and computational concepts

2.1 Problem set-up

Formally, single changepoint detection assumes the presence of at most one change in the underlying probability distribution generating the data $Y_{1:n}$. Let f_{Y_t} denote the probability density function (p.d.f) at the data point Y_t. Then the "at most one changepoint" (AMOC) detection task can be phrased as a hypothesis testing problem, where we test:

- **(H**$_0$**)**: There is *no changeoint* in $Y_{1:n}$, i.e.,
$$f_{Y_1} = f_{Y_2} = \cdots = f_{Y_n}.$$
- **(H**$_1$**)**: There is *one changepoint* at some unknown time $\tau \in \{1, \dots, n-1\}$, such that
$$f_{Y_1} = \cdots = f_{Y_\tau} \neq f_{Y_{\tau+1}} = \cdots = f_{Y_n}.$$

If we decide there is a change (**(H1)** is true), we aim to *estimate* the location of the change τ.

A first fairly generic idea to address this question is to construct/use a test statistic and apply it to all changepoints. In particular, given some likelihood, it is often possible to build a test comparing the likelihood of the data (or fit to the data) with a change at τ and the likelihood without any change. The precise test statistics depend on the assumption we will make about the distribution. Here are a few example assumptions we could make

- i.i.d Gaussian with known variance
- i.i.d Gaussian with known mean
- i.i.d Gaussian with unknown variance
- i.i.d Poisson, Exponential ...
- i.i.d Non-Parametric model (using a kernel for example)
- AR(1) Gaussian with known variance
- ...

2.1.1 A few examples

2.1.2 A change in mean with i.i.d Gaussian errors

We first consider the scenario in which the data $Y_{1:n}$ are assumed to be univariate independently and identically distributed (i.i.d.) Gaussian, with a potential change in the mean parameter μ_t. Formally, we assume

$$Y_t \sim \mathcal{N}(\mu_t, \sigma^2), \quad t = 1, \cdots, n$$

or equivalently

$$Y_t = \mu_t + \varepsilon_t \quad \text{with } \varepsilon_t \sim \mathcal{N}(0, \sigma^2), \quad t = 1, \cdots, n$$

with the associated p.d.f

$$f_{Y_t} = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\{-\frac{1}{2\sigma^2}(y_t - \mu_t)^2\}.$$

Under $\mathbf{H}_1$, there is *one changepoint* at some unknown time $\tau \in \{1, \ldots, n-1\}$, such that

$$\mu_1 = \cdots = \mu_\tau \neq \mu_{\tau+1} = \cdots = \mu_n.$$

Often, one assumes that the variance, σ^2 is known. This variance could be estimated within the test. A common practice (that we will detail a bit later), in particular for multiple changepoint detection, is to “pre-estimate” the variance using a difference-based estimator of the variance and then plug this estimate in the changepoint analysis. This model is schematically illustrated in Figure 2.1.

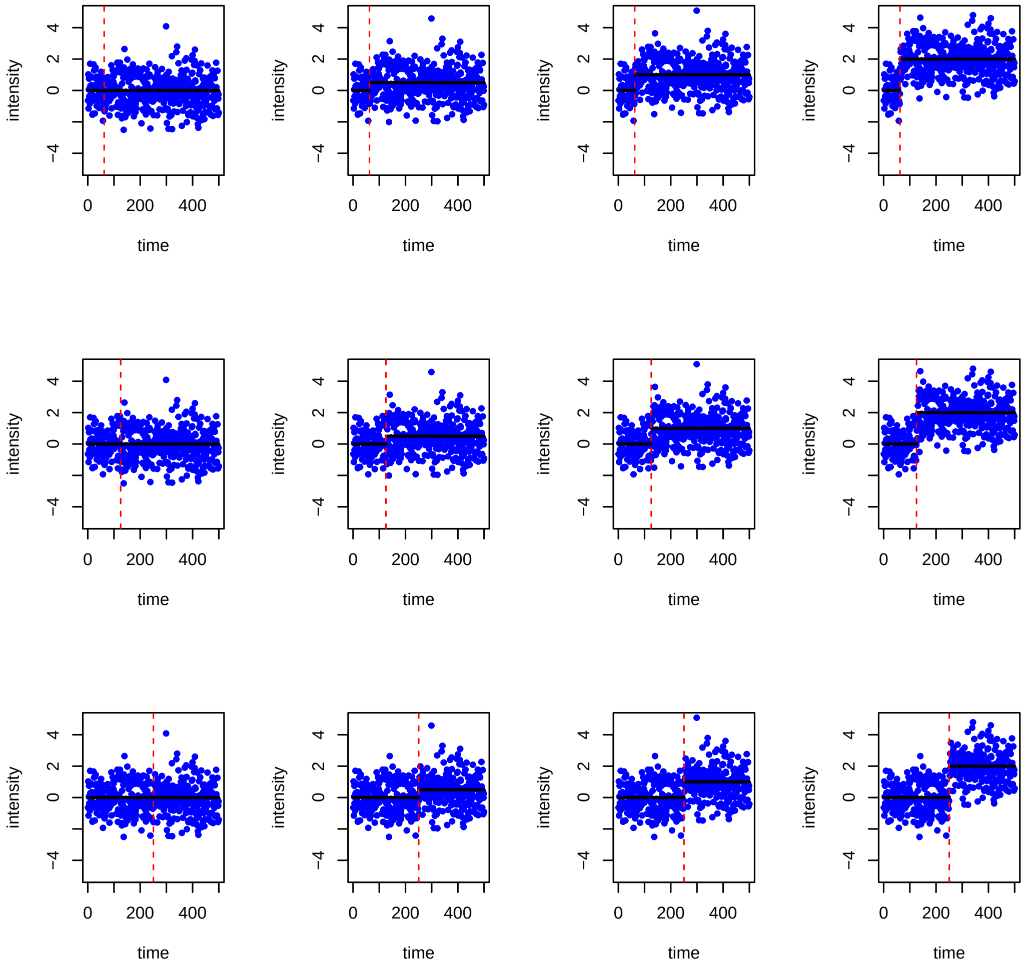

Figure 2.1: Some Gaussian simulated profiles with or without changes and a variance of 1

2.1.3 A change in the parameter of i.i.d Exponentials

Here we proceed and consider a change in the parameter of an Exponential distribution. Formally we have

$$Y_t \sim \text{Exp}(\lambda_t), \quad t = 1, \cdots, n$$

with the associated p.d.f

$$f_{Y_t} = \lambda_t e^{-\lambda_t y_t}.$$

Under $\mathbf{H}_1$, there is *one changepoint* at some unknown time $\tau \in \{1, \dots, n-1\}$, such that

$$\lambda_1 = \cdots = \lambda_\tau \neq \lambda_{\tau+1} = \cdots = \lambda_n.$$

This model is schematically illustrated in Figure 2.2

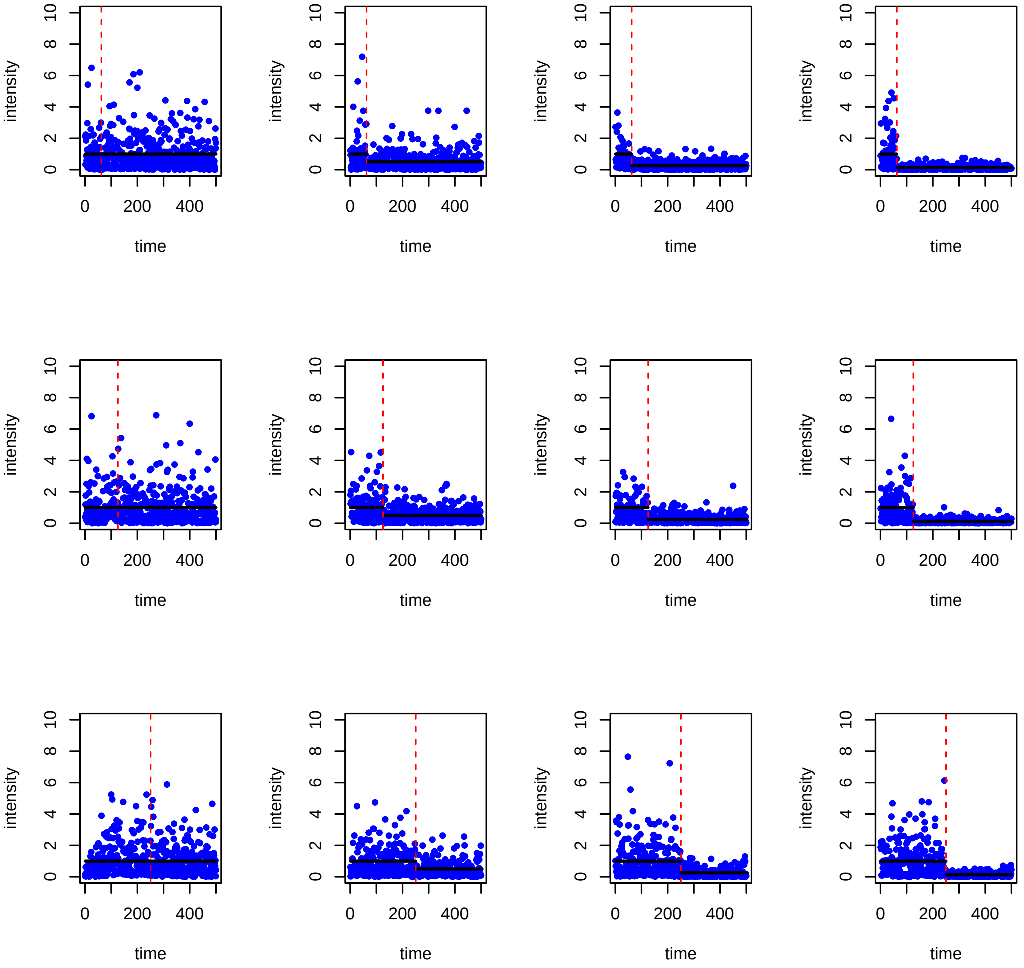

Figure 2.2: Some Exponential simulated profiles with or without changes

Stop and Think

Visualizing the previous plots, how do you think the difficulty of detecting a changepoint depends on the height of the change and the position of the changepoint?

Exercise 1:

Implement in Python a function that simulates i.i.d. Gaussian data with a single change in the mean. Then test this implementation with various values of n, τ, μ_1, μ_2, and σ. After implementing this pseudocode and obtaining the simulated data, you can plot (see Figure 2.3. A for an example with $n = 100$, $\tau = 50$, $\mu_1 = 0$, $\mu_2 = 3$ and $\sigma = 1$) :

- the mean vector as a line or step plot;
- a scatter plot of the observed data;
- add a vertical line at the change-point τ.

Exercise 2:

Do the same for a change in variance only (see Figure 2.3.B), and a change in both mean and variance (see Figure 2.3.C).

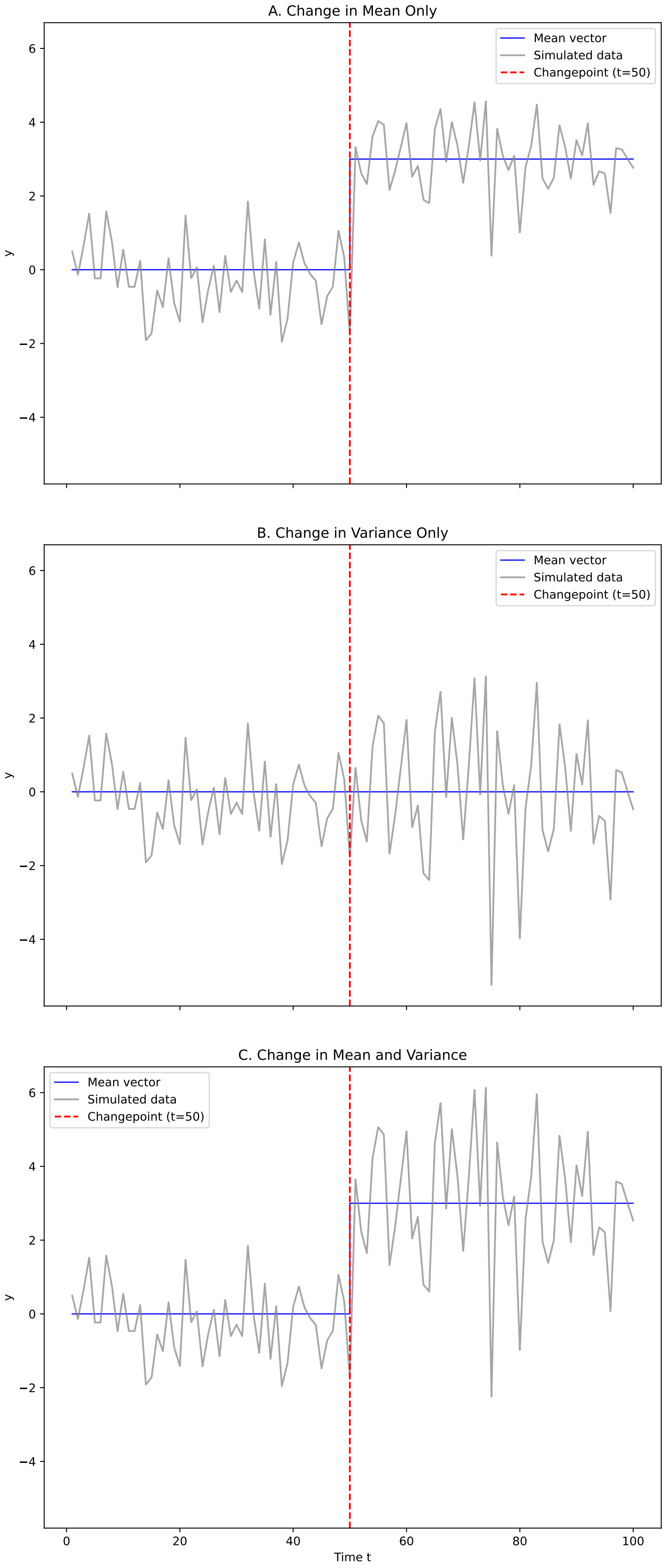

Figure 2.3: Examples of Gaussian-simulated profiles: (A) change in mean only, (B) change in variance only, and (C) change in both mean and variance.

2.2 Maximum Likelihood Inference

Assuming we have an i.i.d. parametric model, the 'vanilla' way to decide whether there is a change or not is to use maximum likelihood and derive a likelihood ratio test. In more detail,

$\mathbf{H}_0$: under $\mathbf{H}_0$ for some distribution f parameterized by θ the maximum likelihood is

$$\max_{\theta} \prod_{t=1}^{n} f_{Y_t}(\theta).$$

$\mathbf{H}_1$: under $\mathbf{H}_1$ for a pre-change parameter θ_1 and a post-change parameter θ_2 the maximum likelihood is

$$\max_{\theta_1,\theta_2}[(\prod_{t=1}^{\tau} f_{Y_t}(\theta_1))(\prod_{t=\tau+1}^{n} f_{Y_t}(\theta_2))]$$

Based on these two assumptions $\mathbf{H}_0$ and $\mathbf{H}_1$ and knowing the position of the change τ we can construct a Likelihood Ratio Test (LRT) to compare them. The LRT statistic at τ is given by :

$$LR_\tau = -2\log\left\{\frac{\max_\theta \prod_{t=1}^{n} f_{Y_t}(\theta)}{\max_{\theta_1,\theta_2}[(\prod_{t=1}^{\tau} f_{Y_t}(\theta_1))(\prod_{t=\tau+1}^{n} f_{Y_t}(\theta_2))]}\right\}$$

where the numerator and the numerator correspond to the likelihood of $\mathbf{H}_0$ and $\mathbf{H}_1$, respectively. Often the LRT statistic is expressed as a difference between the log-likelihoods :

$$LR_\tau = -2\left\{\max_\theta \sum_{t=1}^{n} \log(f_{Y_t}(\theta)) - \max_{\theta_1,\theta_2}[\sum_{t=1}^{\tau} \log(f_{Y_t}(\theta_1)) - \sum_{t=\tau+1}^{n} \log(f_{Y_t}(\theta_2))]\right\}.$$

In practice, the location of the changepoint is unknown and we thus need to consider all possible τ : there are $n-1$. To this end, one typically considers the maximum of all these LRT statistics:

$$LR_{\max} = \max_{\tau\in\{1,\cdots,n-1\}} LR_\tau.$$

Intuitively if $LR_{\max}$ is large enough, we will declare that there is a change, and estimate its position as the argmax.

$$\hat{\tau} = \text{argmax}_{\tau\in\{1,\cdots,n-1\}} LR_\tau.$$

More formally, we need to specify a threshold, β, such that we detect a change if $LR_{\max} \geq \beta$.

This idea of computing a test statistics for all changes and then considering the maximum is illustrated in figure 2.4

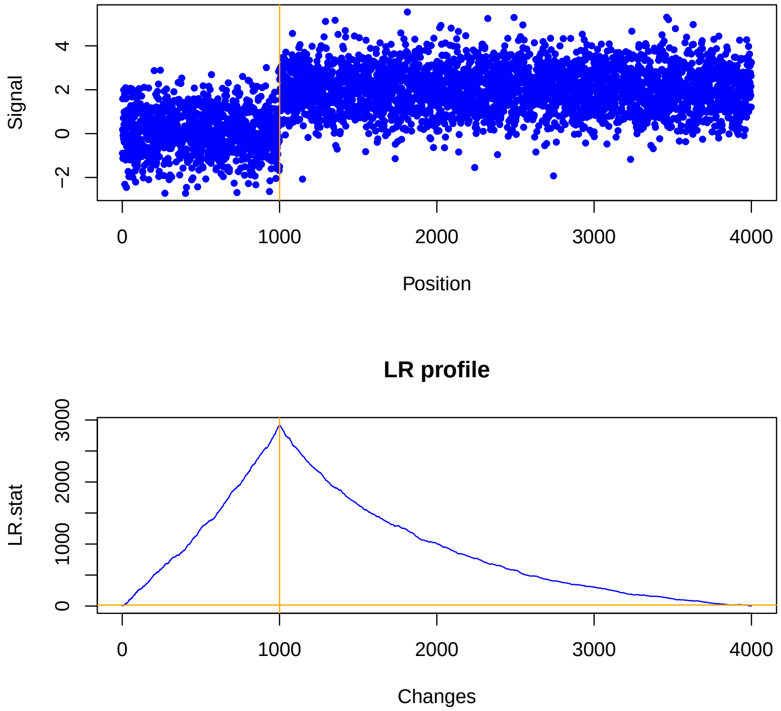

Figure 2.4: (Top) Some profile with a Gaussian Error and a change at $t = 1000$ (Bottom) Likelihood ratio statistics as a function of t.

2.2.1 Calibrating the threshold

In practice the choice of the threshold is crucial. We want this threshold to be large enough so that we don't detect a change when there is none, small enough that we can detect a change if there is a sufficiently large one, and ideally provide some robustness to model error.

Setting an "appropriate" threshold

To the best of our knowledge, mathematically deriving an appropriate/optimal threshold (achieving the minimax rate) for some generic assumptions on the distribution (e.g. univariate/multivariate subgaussian, change in the mean, variance, slope, ...) is an open question (and active area of research). The main difficulty is that the likelihood ratio test is non-regular (as we are maximizing over a discrete set of changes τ).

Under the Gaussian assumption, there are some fairly precise answers to that question based on asymptotic or non-asymptotic arguments. In a bit more details (I) the null distribution can be related to the maximum of a scaled Brownian bridge process (e.g. [Hinkley, 1971, Gombay and Horvath, 1990]) or (II) the LRT can be controlled (as we will see in an exercise a bit later) by non-asymptotic versions of the law of the iterated logarithm for sub-Gaussian random variables [Verzelen et al., 2023]. These theoretical results are very important as they provide insight into the difficulties of the problem and how we could overcome some of them; however, they typically do not provide explicit thresholds, and their results are given up to some constant that needs to be calibrated using simulations or heuristic arguments.

Setting the threshold using Monte Carlo

A practical solution is to use Monte Carlo methods to approximate the null distribution of our test statistic. Importantly it is easily applied to more complicated change-point scenarios. One drawback of this approach is that the threshold should typically depend on the size of the data and therefore for every new number of data points you get you need to re-run the simulation (this might be a problem depending on the precise application and size of n)

Setting the threshold with a Bonferonni correction

From a statistical perspective, a rather natural way (even if not optimal) of controlling our likelihood ratio statistics is to use a Bonferonni bound. Informally, assuming we return a p-value for each of the $(n-1)$ change τ we are considering we can control the family-wise error rate (the probability to make at least 1 error) at α setting our per p-value threshold at α/n.

A bit more formally, assume that we can control each statistics LR_τ at any specified level α: that is for any given τ and α we can provide a value $\beta_\tau(\alpha)$ such that

$$P(LR_\tau \geq \beta_\tau(\alpha)) \leq \alpha.$$

Assuming $\beta_\tau(\alpha)$ does not depends on τ $(\beta_\tau(\alpha) = \beta(\alpha)$ we control the probability that $LR_{\max}$ exceeds $\beta(\alpha/(n-1))$ using a union bound:

$$\begin{aligned}
P\left(LR_\tau \geq \beta\left(\frac{\alpha}{n-1}\right)\right) &= P\left(\max_{\tau\in\{1,\cdots,n-1\}} LR_\tau \geq \beta\left(\frac{\alpha}{n-1}\right)\right) \\
&= P\left(\bigcup_{\tau\in\{1,\cdots,n-1\}} (LR_\tau \geq \beta\left(\frac{\alpha}{n-1}\right)\right) \\
&\leq \sum_{\tau=1}^{n-1} P\left(LR_\tau \geq \beta\left(\frac{\alpha}{n-1}\right)\right) \\
&\leq \sum_{\tau=1}^{n-1} \frac{\alpha}{n-1} = \alpha
\end{aligned}$$

This bound is probabilistically crude as it doesn't consider the dependence between the $(n-1)$ statistics. In the Gaussian case (as we will see a bit later) it leads to a threshold of $2\log(n)$ much larger than the optimal $2\log\log(n)$. Therefore it is conservative. One might wonder whether it is too conservative. To answer this question we informally need to know how the threshold $\beta(\alpha/n)$ varies with n and how the test statistic increases as a function of n and τ for a given effect size (we will see later that for the Gaussian model that it has some power).

One reason this argument is interesting is that it is very generic: we control our false positive rate as long as each test does. This line of reasoning is often used in multiple changepoint methods (e.g., [Fryzlewicz, 2014]). In the Gaussian context, this justifies a penalty that is proportional to $\log(n)$. As discussed in Section 4.2.3 of [Verzelen et al., 2023], this is not optimal, and changes whose energy is smaller than $\sqrt{\log(n)}$ are not detected. However, penalties derived from this line of reasoning (possibly because their constants are easy to calibrate) typically yield good empirical performance in simulations and applications (see the results of fpop [Maidstone et al., 2017] with the penalty of [Yao and Au, 1989] in [Fearnhead and Rigaill, 2020]). To conclude, it is our experience that such approaches provide a good baseline for comparison.

Exercise 3:

Using the previous Bonferonni argument construct a procedure to detect a change in the mean (for a known variance). Test it on datasets of size $n = 2^{10}$. Check that is controlling H_0. Assess its power to detect changes at positions 2^i, for $i = 1, 2, ..., 9$. What is the computational complexity of your approach?

Exercise 4:

Propose a Monte Carlo-based calibration of your procedure. Compare its power to the Bonferonni version.

Exercise 5:

Test the two approaches for errors simulated as a student t statistic. Explore for various values of the degree of freedom.

2.2.2 LRT for i.i.d Gaussian change in mean model with known variance.

We now turn our attention to the Gaussian change in mean model with independent errors. We will first see how, in that case, the LRT statistic is related to the CUSUM statistic. Then we will consider how one can efficiently compute the LRT statistic. Finally, we will consider the calibration of the threshold and analyse the detection power of this approach.

We start by plugging in our Gaussian p.d.f (see Section 2.1.2) into the LR_τ above :

$$LR_\tau = -2\left[\min_\mu \left(-\frac{1}{2\sigma^2}\sum_{t=1}^{n}(y_t-\mu)^2\right) - \min_{\mu_1,\mu_2}\left(-\frac{1}{2\sigma^2}\left(\sum_{t=1}^{\tau}(y_t-\mu_1)^2 + \sum_{t=\tau+1}^{n}(y_t-\mu_2)^2\right)\right)\right] + \\ \tau\log(2\pi\sigma^2) + (n-\tau)\log(2\pi\sigma^2) - n\log(2\pi\sigma^2).$$

Up to some constants, this simplifies to:

$$\frac{1}{\sigma^2}\left[\min_\mu \sum_{t=1}^{n}(y_t-\mu)^2 - \min_{\mu_1,\mu_2}\left(\sum_{t=1}^{\tau}(y_t-\mu_1)^2 + \sum_{t=\tau+1}^{n}(y_t-\mu_2)^2\right)\right].$$

To solve the minimization problem over μ_1 and μ_2, we substitute the empirical mean of the entire sequence $\bar{y}_{1:n}$ into the first term and the empirical means of the left and right segments, i.e. $\bar{y}_{1:\tau}$ and $\bar{y}_{(\tau+1):n}$ into the second term, yielding:

$$LR_\tau = \frac{1}{\sigma^2}\left[\sum_{t=1}^{n}(y_t-\bar{y}_{1:n})^2 - \sum_{t=1}^{\tau}(y_t-\bar{y}_{1:\tau})^2 - \sum_{t=\tau+1}^{n}(y_t-\bar{y}_{(\tau+1):n})^2\right].$$

These empirical means are computed as:

$$\bar{y}_{u:l} = \frac{1}{l-u+1}\sum_{t=u}^{l} y_t.$$

Exercise 6:

Do the same calculation for a Gaussian change in mean model with unknown variance. Compare the $\arg\max$ with the one obtained for a known variance.

Exercise 7:

(AT HOME) Do the same calculation for a Gaussian change in variance only.

CUSUM-like reformulation

The CUSUM statistics date back to the seminal work of Page [Page, 1954] and have been designed to detect a change in an online setting. The general idea behind Page's CUSUM statistics is to detect changes comparing the likelihood of a model without a change and parameter θ_1 and a model with a change with a post-change parameter θ_2. Partly for computational reasons (easy update of the statistic for a new observation), the values of θ_1 and θ_2 are assumed to be known and this is written as

$$cusum_n(\theta_1,\theta_2) = \max_{\tau<n}\left\{\sum_{t=\tau+1}^{n}\log(f_{Y_t}(\theta_2)/f_{Y_t}(\theta_1))\right\}$$

This idea has led to numerous developments for changepoints [Aue and Kirch, 2024]. Importantly, assuming we would optimize the value of θ_1 and θ_2 and for the Gaussian model we are considering this CUSUM statistics can be rewritten as the likelihood ratio statistic. More formally, for a fixed τ, the CUSUM statistic compares the empirical mean of the data to the left of τ with the empirical mean of the data to the right:

$$C_\tau = \sqrt{\frac{\tau(n-\tau)}{n}}\left(\bar{y}_{1:\tau} - \bar{y}_{(\tau+1):n}\right).$$

The prefactor on the left serves as a rescaling term, ensuring that under the null, the statistic follows a normal random variable with variance σ^2. If no changepoint exists at τ, the statistic is distributed as a standard normal. Intuitively, if the mean μ remains constant throughout the sequence, the empirical means on both sides of any point τ should be similar. However, if there is a sufficiently large change in the mean, the two means will differ significantly, revealing the presence of a changepoint. More formally, we declare a change at τ if the squared and variance-normalized CUSUM statistic—equivalently, the LRT statistic—exceeds a suitably chosen threshold β:

$$LR_{\max} = C^2_{\max} = \max_{\tau \in \{1,\cdots,n-1\}} \frac{C^2_\tau}{\sigma^2} > \beta$$

Exercise 8:

Let us prove that the two statistics are indeed related.

Proof 1. (not too slow) We first consider the left terms that is with t from 1 to τ.

$$\begin{aligned} A_{left} &= \sum_1^\tau (y_t - \bar{y}_{1:\tau})^2 - \sum_1^\tau (y_t - \bar{y}_{1:n})^2 \\ &= \sum_1^\tau (2y_t - \bar{y}_{1:\tau} - \bar{y}_{1:n})(\bar{y}_{1:n} - \bar{y}_{1:\tau}) \\ &= \sum_1^\tau (y_t - \bar{y}_{1:n})(\bar{y}_{1:n} - \bar{y}_{1:\tau}) \\ &= \tau(\bar{y}_{1:n} - \bar{y}_{1:\tau})^2 \end{aligned}$$

Now we have

$$\bar{y}_{1:n} - \bar{y}_{1:\tau} = \frac{\tau \sum_1^n y_t - n \sum_1^\tau y_t}{n\tau} = \frac{\tau \sum_{\tau+1}^n y_t - (n-\tau)\sum_1^\tau y_t}{n\tau} = \frac{n-\tau}{n}(\bar{y}_{(\tau+1):n} - \bar{y}_{1:\tau})$$

And we get $A_{left} = \frac{\tau(n-\tau)^2}{n^2}(\bar{y}_{(\tau+1):n} - \bar{y}_{1:\tau})^2$. By symetry $A_{right} = \frac{\tau^2(n-\tau)}{n^2}(\bar{y}_{(\tau+1):n} - \bar{y}_{1:\tau})^2$. Summing we have $A_{left} + A_{right} = \frac{\tau(n-\tau)}{n}(\bar{y}_{(\tau+1):n} - \bar{y}_{1:\tau})^2$, and dividing by σ^2 we get the desired result.

Proof 2. (A bit slower but just as good) We start by writing down $\sigma^2 LR_\tau$:

$$\sigma^2 LR_\tau = \sum_{t=1}^n (y_t - \bar{y}_{1:n})^2 - \sum_{t=1}^\tau (y_t - \bar{y}_{1:\tau})^2 - \sum_{t=\tau+1}^n (y_t - \bar{y}_{(\tau+1):n})^2.$$

Now we need to expand each term. Starting with the first:

$$\sum_{t=1}^n (y_t - \bar{y}_{1:n})^2 = \sum_{t=1}^n y_i^2 - 2\bar{y}_{1:n} \sum_{t=1}^n y_t + n\bar{y}^2_{1:n}.$$

As $\sum_{t=1}^n y_t = n\bar{y}_{1:n}$, we notice that we can simplify the last two terms. We are left with:

$$\sum_{t=1}^n (y_t - \bar{y}_{1:n})^2 = \sum_{t=1}^n y_t^2 - n\bar{y}^2_{1:n}.$$

We proceed similarly for the other two terms:

$$\sum_{t=1}^\tau (y_t - \bar{y}_{1:\tau})^2 = \sum_{t=1}^\tau y_t^2 - \tau\bar{y}^2_{1:\tau}, \quad \sum_{t=\tau+1}^n (y_t - \bar{y}_{(\tau+1):n})^2 = \sum_{t=\tau+1}^n y_t^2 - (n-\tau)\bar{y}^2_{(\tau+1):n}.$$

Putting all together, and getting rid of the partial sums, we are left with:

$$\sigma^2 LRT = -n\bar{y}^2_{1:n} + \tau\bar{y}^2_{1:\tau} + (n-\tau)\bar{y}^2_{(\tau+1):n}.$$

Now recall that $\bar{y}_{1:n} = \frac{1}{n}[\tau\bar{y}_{1:\tau} + (n-\tau)\bar{y}_{(\tau+1):n}]$ and:

$$\bar{y}^2_{1:n} = \frac{1}{n^2}[\tau^2\bar{y}^2_{1:\tau} + 2\tau(n-\tau)\bar{y}_{1:\tau}\bar{y}_{(\tau+1):n} + (n-\tau)^2\bar{y}_{(\tau+1):n}].$$

Plugging in this into LR_τ, we obtain:

$$\begin{aligned}
\sigma^2 LR_\tau &= -\frac{\tau^2}{n}\bar{y}^2_{1:\tau} - \frac{2\tau(n-\tau)}{n}\bar{y}_{1:\tau}\bar{y}_{(\tau+1):n} - \frac{(n-\tau)^2}{n}\bar{y}_{(\tau+1):n} + \tau\bar{y}^2_{1:\tau} + (n-\tau)\bar{y}^2_{(\tau+1):n} \\
&= \frac{\tau(n-\tau)}{n}\bar{y}^2_{1:\tau} - \frac{2\tau(n-\tau)}{n}\bar{y}_{1:\tau}\bar{y}_{(\tau+1):n} + \frac{\tau(n-\tau)}{n}\bar{y}^2_{(\tau+1):n} \\
&= \frac{\tau(n-\tau)}{n}(\bar{y}^2_{1:\tau} - 2\bar{y}_{1:\tau}\bar{y}_{(\tau+1):n} + \bar{y}^2_{(\tau+1):n}) \\
&= \frac{\tau(n-\tau)}{n}(y_{1:\tau} - y_{(\tau+1):n})^2 \\
&= C^2_\tau
\end{aligned}$$

This give us $LR_\tau = \frac{C^2_\tau}{\sigma^2}$.

Computation

One critical aspect of modern (multi)-changepoint approaches is their runtime complexity. You want the approach to be both statistically efficient and computationally scalable. The offline AMOC is rather simple, but a good opportunity to present a simple yet key trick.

Stop and Think

What is the computational complexity of a naive implementation of the CUSUM statistic when computed iteratively along a time series?

An example. Let us compute the CUSUM for the vector

$$y_{1:4} = (0.8, 1.2, 4.5, 4.3),$$

assuming the observations are Gaussian with $\sigma^2 = 1$. The possible changepoint positions are

$$\tau \in \{1, 2, 3\}.$$

Before computing the CUSUM statistics for each potential changepoint, we first need to determine the empirical means for each segment. Specifically, for each τ, we compute the means of the subsequences:

$$\bar{y}_{1:\tau} = \frac{1}{\tau}\sum_{i=1}^{\tau} y_i, \quad \bar{y}_{(\tau+1):4} = \frac{1}{4-\tau}\sum_{i=\tau+1}^{4} y_i.$$

We compute these means for each possible τ:

$$\begin{aligned}
\bar{y}_{1:1} &= y_1 = 0.8, & \bar{y}_{2:4} &= \frac{1.2+4.5+4.3}{3} = 3.33, \\
\bar{y}_{1:2} &= \frac{0.8+1.2}{2} = 1.0, & \bar{y}_{3:4} &= \frac{4.5+4.3}{2} = 4.4, \\
\bar{y}_{1:3} &= \frac{0.8+1.2+4.5}{3} = 2.17, & \bar{y}_{4:4} &= y_4 = 4.3.
\end{aligned}$$

The squared CUSUM statistic for each τ is then given by:

$$C^2_\tau = \frac{\tau(4-\tau)}{4}(\bar{y}_{1:\tau} - \bar{y}_{(\tau+1):4})^2.$$

Computing the values:

$$C_1^2 = \frac{3 \times 1}{4}(0.8 - 3.33)^2 = 0.75 \times 6.4 = 4.8,$$
$$C_2^2 = \frac{2 \times 2}{4}(1.0 - 4.4)^2 = 11.56,$$
$$C_3^2 = \frac{1 \times 3}{4}(2.17 - 4.3)^2 = 0.75 \times 4.53 = 3.4.$$

Thus, the maximum CUSUM statistic is:

$$C_{\max}^2 = 11.56 \text{ at } \tau = 2.$$

Recall, that to call a changepoint, we would compare $C_{\max}^2$ to a threshold value c. If $C_{\max}^2 > c$, we conclude that there is a changepoints $\hat{\tau} = 2$.

Algorithmic Formulation of the CUSUM Statistic. A naive implementation of the CUSUM statistic (re)computes the means $\bar{y}_{1:\tau}$ and $\bar{y}_{\tau+1:n}$ for each τ in $\mathcal{O}(n)$ leading to an overall computational complexity of $\mathcal{O}(n^2)$. However, we can be much faster by sequentially computing partial sums,

$$S_t = \sum_{t=1}^{n} y_t.$$

By computing partial sums incrementally, we avoid redundant calculations and achieve a linear time complexity of $\mathcal{O}(n)$, making the method scalable for large datasets.

Algorithm 1 Efficient CUSUM Algorithm

```
Require: Time series $y = (y_1, \cdots, y_n)$, threshold $c$, variance $\sigma^2$
Ensure: Changepoint estimate $\hat{\tau}$, maximum CUSUM statistic $C_{\max}^2$
$n \leftarrow$ length of $y$
$C_{\max}^2 \leftarrow 0$
$\hat{\tau} \leftarrow 0$
$S_n \leftarrow \sum_{t=1}^{n} y_t$
$S \leftarrow 0$
for $t = 1$ to $n - 1$ do
    $S \leftarrow S + y_t$
    $\bar{y}_{1:t} \leftarrow \frac{S}{t}$
    $\bar{y}_{(t+1):n} \leftarrow \frac{S_n - S}{n - t}$
    $C_t^2 \leftarrow \frac{t(n-t)}{n}(y_{1:t} - \bar{y}_{(t+1):n})^2$
    if $C_t^2 > C_{\max}^2$ then
        $C_{\max}^2 \leftarrow C_t^2$
        $\hat{\tau} \leftarrow t$
    end if
end for
if $\frac{C_{\max}^2}{\sigma^2} > c$ then
    return $\hat{\tau}, C_{\max}^2$                    ▷ Changepoint detected
else
    return NULL, $C_{\max}^2$                    ▷ No changepoint detected
end if
```

This incremental update of the mean is key to the efficiency of many algorithms and methods for multiple changepoint detection in particular approaches based on dynamic programming [Auger and Lawrence, 1989, Killick et al., 2012, Rigaill, 2015] and approach based on local optimization or isolation [Fryzlewicz, 2014, Fryzlewicz, 2020, Kovács et al., 2023, Anastasiou and Fryzlewicz, 2022]. This incremental approach works for many other distributions than just the Gaussian. However, there are some models for which it does not work.

Exercise 9:

Give an example of a distribution, or model for which the CUSUM trick does not work.

Exercise 10:
Implement the *efficient CUSUM Algorithm* in Python. Your function should:

1. **Return:**
   - The estimated changepoint, $\hat{\tau}$.
   - The maximum CUSUM statistic, $C^2_{\max}$.
   - The full sequence $\{C^2_\tau\}$ for $\tau = 1, \ldots, n-1$.
2. **Test** your implementation on multiple synthetic datasets generated by the function developed in *Exercise 1*.

Calibrating the threshold

We now turn back to the problem of controlling the CUSUM statistics (which is here equivalent to the LRT statistic). Our goal is first to find a threshold β such that we declare a changepoint if $\max_{\tau<n} C_\tau > \beta$

A Bonferonni-like bound Assuming the y_t are all Gaussian with $\sigma^2 = 1$ (without loss of generality), the C_τ are $\mathcal{N}(0,1)$ and we can use a sub-Gaussian concentration bound. Combined with a union or Bonferonni bound we get

$$
\begin{aligned}
P\left(\max_{\tau<n} |C_\tau| > \beta\right) &= P\left(\bigcup_{\tau<n} |C_\tau| > \beta\right) \\
&\leq \sum_{\tau<n} P\left(|C_\tau| > \beta\right) \\
&\leq \sum_{\tau<n} 2\exp -\frac{\beta^2}{2}
\end{aligned}
$$

Taking $\beta^2 = 2\log(n-1) - 2\log(\alpha)$. We get

$$
P\left(\max_{\tau<n} |C_\tau| > \beta\right) \leq 2\alpha
$$

As explained earlier (see Section 2.2.1) this is likely conservative. In detail, based on [Gombay and Horvath, 1990] or [Verzelen et al., 2023] the threshold should be of order $\log(\log(n))$. Nonetheless, this simple approach is somewhat powerful for sufficiently large changepoints. Indeed consider a change of size δ at τ^*, that is the mean is 0 before τ^* and δ after. Let's try to bound the probability that C_{τ^*} exceeds our $2\log(n/\alpha)$. C_{τ^*} can be rewritten as

$$
C_{\tau^*} = \delta\sqrt{\frac{\tau^*(n-\tau^*)}{n}} + \sqrt{\frac{\tau^*(n-\tau^*)}{n}}(\bar{\varepsilon}_{\tau^*:n} - \bar{\varepsilon}_{1:\tau^*})
$$

Note that $\sqrt{\frac{\tau^*(n-\tau^*)}{n}}(\bar{\varepsilon}_{\tau^*:n} - \bar{\varepsilon}_{1:\tau^*})$ is $\mathcal{N}(0,1)$ so using a sub-Gaussian concentration bound with probability at least $1 - 2\exp -.5x^2$ we get the following bound

$$
\left|\sqrt{\frac{\tau^*(n-\tau^*)}{n}}(\bar{\varepsilon}_{\tau^*:n} - \bar{\varepsilon}_{1:\tau^*})\right| \leq x
$$

Considering $x = \sqrt{-2\log(\alpha)}$ we get that if

$$
\delta\sqrt{\frac{\tau(n-\tau)}{n}} - \sqrt{-2\log(\alpha)} > \sqrt{2\log((n-1)/\alpha)}
$$

or

$$
\delta\sqrt{\frac{\tau(n-\tau)}{n}} > \sqrt{2\log((n-1)/\alpha)} + \sqrt{-2\log(\alpha)}
$$

then with probability at least $1-\alpha$, C_{τ^*} would pass the threshold and we would detect a change.

From this, we get two important information.

1. The power depends on the position of the change, with change close to the border harder do detect. If we consider τ to be proportional to n, $\tau = a \cdot n$ (with a in $(0,1)$ we detect the change if
$$\delta\sqrt{a(1-a)n} \geq \sqrt{2\log((n-1)/\alpha)} + \sqrt{-2\log(\alpha)}.$$
Asymptotically (letting n grow) this will always happen eventually as $\sqrt{n}$ groes faster than $\log(n)$.

Stop and Think

Using the previous power calculation (assuming a variance of 1) plot the minimum jump size you could detect as a function of n and then as a function of τ^* for $n = 10^3, 10^4, 10^5$. Repeat using a $\sqrt{2\log\log(n)}$ threshold. What do you think?

Exercise 11:

- Check the distribution of $\hat{\tau}$ returned by the CUSUM procedure with the Bonferroni-like bound on simulated datasets of size $n = 100$ under H_0.
- Assess the power of the procedure to detect changes at positions 2^i, for $i = 1, 2, ..., 9$, on profiles of size $n = 100$ under H_1. What do you notice?
- Does the distance between the true and estimated changepoints vary with the position of the true changepoint?

Exercise 12:

In this exercise, we will consider a restriction on the size of the segments (say at least 10 datapoints). Propose a Monte Carlo-based calibration of this new procedure. Test its H0 control and power on data simulated with i.i.d Gaussian errors, then with i.i.d. Student's t-distributions with degrees of freedom 3. Compare to the results you had without the restriction.

Exercise 13:

(AT HOME) In this exercise, we will consider the case where the variance is unknown and explore two strategies. In the first strategy, we use the statistic defined earlier in *Exercise 6* and calibrate it using Monte Carlo simulation. In the second strategy, we pre-estimate the variance using the Median Absolute Deviation (MAD) estimator [Rousseeuw and Croux, 1993], defined below and available via the `scipy.stats.median_abs_deviation` function in Python:

$$\mathrm{MAD} = \mathrm{median}\Big(\big|D_i - \mathrm{median}(D)\big|\Big),$$

with $D_i = \frac{Y_{i+1}-Y_i}{\sqrt{2}}$. The MAD can be scaled to provide a consistent estimator for the standard deviation:

$$\hat{\sigma} = \frac{\mathrm{MAD}}{0.67449}.$$

Exercise 14:

(AT HOME) This exercise is for the more mathematically inclined and can be skipped. It should give some intuition as to why $\sqrt{\log(\log(n))}$ from [Gombay and Horvath, 1990, Verzelen et al., 2023] is feasible. In [Verzelen et al., 2023] the following lemma is proven.

- Use this lemma to show that a threshold of order $\sqrt{\log(\log(n))}$ does control H_0.
- Assume you only want to test changepoints that are sufficiently far from the borders (say in $(an : (1-a)n)$ for $0 < a < 1$. Does it change your bound?

Lemma 1 *Let $\epsilon_1, \ldots, \epsilon_n$ be independent centered sub-Gaussian random variables such that*

$$E[e^{s\epsilon_i}] \leq e^{\frac{s^2}{2}}, \quad \textit{for any } i \geq 1 \textit{ and any } s > 0.$$

Then, for any integer $d > 0$, any $\alpha > 0$, and any $x > 0$,

$$P\left[\max_{k \in [d,(1+\alpha)d]} \sum_{i=1}^{k} \epsilon_i \frac{1}{\sqrt{k}} \geq x\right] \leq \exp\left(-\frac{x^2}{2(1+\alpha)}\right).$$

Conclusion We conclude this chapter on the detection power of a single change here and do not further discuss the localization error of the CUSUM procedure or the construction of a confidence interval for the change. While some proposals have been made, their derivations are typically a bit technical and outside the scope of this class. Arguably, a natural way to address this question and cope with the discrete nature of the changepoint is to adopt a Bayesian framework and construct credibility intervals.

Chapter 3

Multiple changepoints offline: Essential Statistical and Computational Concepts

3.1 A baseline approach?

In this chapter, we present the detection of multiple changepoints using a penalized maximum likelihood approach with a constant penalty per changepoint, along with its implementation in the Gaussian case utilizing an $O(n^2)$ dynamic programming algorithm [Bellman, 1961, Auger and Lawrence, 1989, Jackson et al., 2005] and the SIC-like penalty proposed by [Yao and Au, 1989].

Assuming we have a likelihood (or some measure of fit to the data), we argue that optimizing the likelihood is a natural approach and likely the first idea that would come to the mind of a statistician not trained in changepoint detection. Importantly, when properly tuned, this approach is empirically competitive and typically yields good statistical performance in applications and simulations (see, for example, the results of FPOP [Maidstone et al., 2017] with the penalty of [Yao and Au, 1989] in [Fearnhead and Rigaill, 2020]).

The quadratic complexity is certainly a concern for large datasets, but assuming the implementation is done in C or another reasonably low-level programming language, it is our experience that the code runs in several tens of minutes for $n = 10^4$ and one or a few hours for $n = 10^5$ (obviously, this varies depending on the penalty). Furthermore, for most models with independent errors, the computation can be accelerated using inequality-based pruning [Maidstone et al., 2017], as in PELT [Killick et al., 2012]. Using PELT, the complexity is roughly of order $O(n^2/\hat{K})$, where $\hat{K}$ is the estimated number of changepoints[1] For specific models (the univariate i.i.d. Gaussian in particular), the computation can be further accelerated using functional pruning ideas, as in pDPA or FPOP [Rigaill, 2015, Maidstone et al., 2017, Runge et al., 2023, Pishchagina et al., 2024]. Using FPOP, the complexity is roughly of order $O(n \log(n))$, regardless of the number of changes, making it feasible to segment very large datasets (about 1 minute for $n = 10^8$).

For all these reasons, we believe that the maximum likelihood approach with a constant penalty serves as a good baseline for entering the field of changepoint detection.

Nonetheless, we would like to stress that from a theoretical perspective (as also discussed in the previous and following chapter) this approach is not ideal. To be specific, in the Gaussian case, this approach (when properly tuned) does not detect spurious changes, does detect high-energy changepoints (that is, those that are high compared to $\sqrt{\log(n)}$), but it may miss some low-energy changepoints that would be detected by (properly tuned) multi-scale penalty approaches such as [Verzelen et al., 2023, Pein et al., 2017, Cho and Kirch, 2019] (see the simulation results in [Liehrmann and Rigaill, 2024]).

Further, in many cases, maximum likelihood inference is not possible because dynamic programming does not apply or because it is too slow. Therefore, there is a need for a generic solution to circumvent

[1] More theoretically, PELT can be shown to be $O(n)$ if the true number of changepoints is linear in n for some properly tuned penalty. It is $O(n^2)$ if the number of true changepoints is constant with respect to n.

these problems. Ideally, this solution should be easy to implement (to handle various models and penalization schemes), computationally fast, and provide good statistical guarantees (at least for simple models). In recent years, a simple, elegant, and popular idea in this direction is to apply an AMOC LRT/CUSUM-like strategy not to the whole dataset recursively (as in Binary Segmentation), but to many different chunks of the data of varying sizes and positions. Intuitively, if these chunks are diverse enough, it is guaranteed that each true (detectable) changepoint will be present in at least one of those chunks alone (without any other true change). Many recent approaches are based on this idea, they differ in the exact statistics they are using, the way they sample and aggregate all chunks, and the particular threshold or penalty they are using (e.g. [Fryzlewicz, 2014, Cho and Kirch, 2019, Anastasiou and Fryzlewicz, 2022, Fryzlewicz, 2020, Verzelen et al., 2023]).

3.2 Problem set-up

3.2.1 Model and Likelihood

Multiple changepoint detection is an extension of the single changepoint detection problem (see chapter 2) that allows for more than one changepoint. Let us consider a time series $Y_{1:n}$ and assume each data point Y_t follows a probability distribution f_{Y_t}. Under this setup, the null hypothesis $\mathbf{H_0}$ is the same and the alternative hypothesis $\mathbf{H_1}$ can be redefined as follows:

- **($\mathbf{H}_0$)**: There is *no changepoint* in $Y_{1:n}$, meaning the distribution remains constant throughout the sequence:
$$f_{Y_1} = f_{Y_2} = \cdots = f_{Y_n}.$$
- **($\mathbf{H}_1$)**: There exist K *changepoints* at unknown positions $0 < \tau_1 < \cdots < \tau_K < n$, such that the distribution of Y_t changes at these points:
$$f_{Y_1} = \cdots = f_{Y_{\tau_1}} \neq \cdots \neq f_{Y_{\tau_k+1}} = \cdots = f_{Y_{\tau_{k+1}}} \neq \cdots \neq f_{Y_{\tau_K+1}} = \cdots = f_{Y_n}.$$

Under the alternative hypothesis ($\mathbf{H_1}$), the informal objective is to estimate both the number of changepoints K and their locations represented as a vector of size K: $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_K)$. As discussed in [Verzelen et al., 2023], a more refined goal is to detect high-energy changepoints (in a sense to be defined in chapter **??**) while not detecting spurious changepoints (that is not detecting any other changepoints).

In the following to simplify notations and equations, we define for any segmentation (vector of change $\boldsymbol{\tau}$): $\tau_0 = 0$ and $\tau_{K+1} = \tau_{\#\boldsymbol{\tau}} = n$. Further, we define

- $\mathcal{M}^K_{1:n}$ (or $\mathcal{M}^K_n$) the set of all segmentations with K changes of n data-points (with cardinality $\#\mathcal{M}^K_{1:n}$).
- $\mathcal{M}_{1:n}$ (or $\mathcal{M}_n$) the set of all segmentations of n data-points (with cardinality $\#\mathcal{M}_{1:n}$).

Exercise 15:
Let us count the number of segmentations with K changes ($\#\mathcal{M}^K_n$) and the number of segmentations ($\#\mathcal{M}_n$).
Bonus: count the number of segmentation in $K+1$ segments with length at least r.

3.2.2 Penalized Likelihood

Given some vector of change $\boldsymbol{\tau}$ of size K, $K+1$ parameters $\theta_1, \ldots, \theta_{K+1}$, one for each segment and some distribution f parameterized by θ, assuming the observations are independent, the likelihood can be written as

$$\prod_{k=1}^{k=\#\boldsymbol{\tau}+1} \prod_{t=\tau_{k-1}+1}^{\tau_k} f_{Y_t}(\theta_k)$$

Minus the log-likelihood can then be written as

$$\sum_{k=1}^{k=\#\boldsymbol{\tau}+1} \sum_{t=\tau_{k-1}+1}^{\tau_k} -\log(f_{Y_t}(\theta_k))$$

In some applications, the exact number of changepoint is known and arguably it makes sense to try and optimize the likelihood (minimizing minus the log-likelihood) for this given number of changepoints:

$$\mathcal{L}^K_{1:n} = \min_{\boldsymbol{\tau} \in \mathcal{M}^K_n} \left\{ \sum_{k=1}^{k=\#\boldsymbol{\tau}+1} \min_{\theta_k} \left(\sum_{t=\tau_{k-1}+1}^{\tau_k} -\log(f_{Y_t}(\theta_k)) \right) \right\}$$

Often however the number of changepoints is not known. Optimizing over all possible segmentations in $\mathcal{M}_n$ doesn't make sense as it would always return a segmentation in $(n-1)$ changes. From this, it should be clear that one need a criterion to balance the "complexity" of the obtained segmentation. As for the threshold in the at most one change case, to the best of our knowledge, mathematically deriving an appropriate/optimal penalty for some generic assumptions on the distribution is an open question (and an active area of research). The simplest approach is certainly to pay a fixed price for any additional change, say $\beta > 0$. This is the approach we will be considering in the rest of this chapter.

Mathematically, the penalized maximum likelihood problem is written as

$$\mathcal{L}_{1:n} = \min_{\boldsymbol{\tau} \in \mathcal{M}_n} \left\{ \sum_{k=1}^{k=\#\boldsymbol{\tau}+1} \min_{\theta_k} \left(\sum_{t=\tau_{k-1}+1}^{\tau_k} -\log(f_{Y_t}(\theta_k)) + \beta \right) \right\}. \tag{3.1}$$

Intuitively, for some sufficiently large β the output of this minimization should be different from the trivial segmentation with $(n-1)$ changes. The previous equation raises two questions:

- How should we set β? It should be large enough that we don't detect spurious changes and but small enough that we do detect sufficiently large changepoints. This is a model selection problem.

- Given some β or some range of β how do we recover the segmentation optimizing the penalized likelihood. This is an algorithmic problem.

3.2.3 Three key ingredients of any multiple changepoint detection approach

The literature on multiple changepoint detection is vast. We will not attempt to make a review here. Nonetheless, we believe (as also discussed in [Truong et al., 2020]) that it is useful to classify methods based on three criteria. The penalized maximum likelihood framework we just described perfectly fits in this classification.

1. **A model or loss function** measuring the likelihood "fit to the data" or the homogeneity. This encode the type of changes we are looking for.

2. **A model selection criteria** to balance the goodness-of-fit with the complexity of the segmentation (typically encoded as a penalty taking into account the number of changes, the length of the segments, the variance...)

3. **An algorithm** to explore the segmentation space and output one or several candidate segmentations with a good trade off between likelihood and complexity.

In what follows we will consider what we consider the baseline/archetypical approach:

Model: univariate i.i.d. Gaussian errors;

Penalty: $c_1 \log(n) + c_2$ for some $c_1 \geq 2$ and $c_2 \geq 0$;

Algorithm: $O(n^2)$ optimal partitioning algorithm.

3.2.4 The univariate change in mean model with i.i.d Gaussian errors

Recall the i.i.d. Gaussian model in mean for a single changepoint (with known variance). In the multiple changepoint setting, under $\mathbf{H_1}$, there exist K *changepoints* at unknown positions $\boldsymbol{\tau}$, such that

$$\theta_1 = \cdots = \theta_{\tau_1} \neq \cdots \neq \theta_{\tau_k+1} = \cdots = \theta_{\tau_{k+1}} \neq \cdots \neq \theta_{\tau_K+1} = \cdots = \theta_n.$$

In that case, the penalized maximum likelihood problem simplifies to minimizing the mean-squared error:

$$\mathcal{L}_{1:n} = \min_{\boldsymbol{\tau} \in \mathcal{M}} \left\{ \sum_{k=1}^{k=\#\boldsymbol{\tau}+1} \left(\sum_{t=\tau_{k-1}+1}^{\tau_k} (Y_t - \bar{Y}_{\tau_{k-1}:\tau_k})^2 + \beta \right) \right\}. \tag{3.2}$$

Exercise 16:
Derive the previous simplification.

Exercise 17:
Implement in Python, a function that simulates i.i.d. Gaussian data with a several change in the mean. It should take as a parameter the position of the changes and the mean of each segment.

Exercise 18:
(AT HOME) Do the same for a change in the scale parameter of an Exponential distribution.

3.3 Dynamic Programming for multiple changepoint.

3.3.1 The basic recursion

Looking at equation 3.1 or even 3.2 for the first time it is not completely obvious how this can be solved because the possible number of segmentations we have to consider is very large. For example taking $n = 1000$ and $K = 3$ we get $\binom{1000-1}{3} > 1.66 * 10^8$. A naive search of the best segmentation (according to likelihood) is thus not feasible even for fairly small n and K.

The key property that we will exploit (in the coming algorithm) is that if we knew the position of just one change say t (in practice we will consider the last) then we get the solution by combining the solution to the same problem restricted to data points from 1 to t and the solution to the same problem restricted to data points from $t+1$ to n. This is possible because the log-likelihood writes as a sum overall segments.

Let us formalize this idea. We call $\ell_{i:j}$ the optimal minus log-likelihood of segment $i:j$. That is

$$\ell_{i:j} = \min_{\theta} \{ \sum_{t=i}^{j} -\log(f_{Y_t}(\theta) \}.$$

In the Gaussian case this simplifies to $\ell_{i:j} = \sum_{t=i}^{j} (Y_t - \bar{Y}_{i:j})^2$. With this notation minus the log-likelihood of a segmentation $\boldsymbol{\tau}$ writes as

$$\sum_{k=1}^{\#\boldsymbol{\tau}+1} \left(\ell_{(\tau_{k-1}+1):\tau_k} + \beta \right)$$

We claim that

$$\mathcal{L}_{1:n} = \min_{\tau < n} \{ \mathcal{L}_{1:\tau} + \ell_{\tau+1:n} + \beta \} \tag{3.3}$$

The previous update rule essentially states that if you know the solution up to any τ smaller than n then you can compute the solution at n. This update or recursion has been found multiple times with some little variations (see [Bellman, 1961, Auger and Lawrence, 1989, Jackson et al., 2005] to cite just a few) and is the basis for the dynamic programming algorithm we will describe in the next section.

Exercise 19:

- The proof of equation (3.3) is by contradiction. Prove it.

- How many times do you need to apply the update to get $\mathcal{L}_{1:n}$?
- What is the complexity of computing $\mathcal{L}_{1:n}$?

3.3.2 Keeping it low in memory

As we will see a bit later the update rule (3.3) leads to an $O(n^2)$ complexity in time assuming we have access to all $\ell_{i:j}$. To apply this recursion it seems at first that one needs to compute and store all $\ell_{i:j}$ for all $1 \leq i < j \leq n$. Storing all these values is a problem as it scales as $\mathcal{O}(n^2)$ leading to swapping issues and slow empirical runtimes. In most cases this is avoidable. In particular, for low-dimensional models, it is often the case that minus the log-likelihood of a segment $\ell_{i:j}$ can be computed efficiently using some well-chosen summary statistics.

For example, in the change in mean model with i.i.d Gaussian errors

$$\ell_{i:j} = \sum_{t=i}^{j}(Y_t - \bar{Y}_{i:j})^2 = \sum_{t=i}^{j} Y_t^2 - \frac{1}{j-i+1}(\sum_{t=i}^{j} Y_t)^2.$$

Therefore if pre-compute in $O(n)$ time and memory $S_j^{(1)} = \sum_{t=1}^{j} Y_t$ and $S_t^{(2)} = \sum Y_t^2$ we can compute $\ell_{i:j}$ on the fly in $O(1)$ time using:

$$\ell_{i:j} = G(i,j) = (S_j^{(2)} - S_{i-1}^{(2)}) - \frac{1}{j-i+1}(S_j^{(1)} - S_{i-1}^{(1)})^2.$$

We have already seen this trick in the previous chapter when computing the CUSUM statistics. Importantly, it works for many models, for example: changes in the parameter of distribution in the exponential family, changes in the regression coefficients... For some models, this trick does not apply. Sometimes it is nonetheless possible to align the dynamic programming recursion with the calculation of the $\ell_{i:j}$ in such a way that we never need to store more than n $\ell_{i:j}$ and without additional calculation (see [Celisse et al., 2018]).

3.3.3 The optimal partitioning algorithm

We are now ready to present the Optimal Partitioning or OP algorithm [Jackson et al., 2005]. In practice, it is not sufficient to recover the optimal likelihood but we also need the optimal set of changepoints. To this end, we also store the arg min at each step : $\mathcal{T}_t$ enabling back-tracking once the OP-recursion is finished.

We first write the algorithm without the memory trick described in subsection 3.3.2, therefore assuming we have pre-computed all $\ell_{i:j}$.

Algorithm 2 The generic Optimal Partitioning algorithm

```
Require: $\ell_{i:j}$ and $\beta$
Ensure: Optimal minus log-likelihood $L_t$ and Argmin $\mathcal{T}_t$
$\mathcal{L}_{1:1} \leftarrow 0$
for $t \in \{2, \dots, n\}$ do
    $\mathcal{L}_{1:t} \leftarrow \min_{\tau<t}(\mathcal{L}_{1:\tau} + \ell_{\tau+1:t} + \beta)$
    $\mathcal{T}_t \leftarrow \arg\min_{\tau<t}(\mathcal{L}_{1:\tau} + \ell_{\tau+1:t} + \beta)$
end for
```

In the Gaussian case, we now write the version where we provide the cumulative sum and cumulative sum of square of the data (see equation 3.3.2 for the definition of the function G)

Algorithm 3 The Optimal Partitioning algorithm in the Gaussian case

```
Require: Cumulative sum and sum of square of the data $S_t^{(1)}$, $S_t^{(2)}$
Ensure: Optimal minus log-likelihood $L_t$ and Argmin $\mathcal{T}_t$
$\mathcal{L}_{1:1} \leftarrow 0$
for $t \in \{2, \dots, n\}$ do
    $\mathcal{L}_{1:t} \leftarrow \min_{\tau<t}(\mathcal{L}_{1:\tau} + G(\tau+1, t) + \beta)$
    $\mathcal{T}_t \leftarrow \arg\min_{\tau<t}(\mathcal{L}_{1:\tau} + +G(\tau+1, t) + \beta)$
end for
```

Exercise 20:

Backtracking: Given the vector $\mathcal{T}_t$ for all t in $1 : n$ how would you recover the optimal segmentation?

Exercise 21:

Implement the OP algorithm for i.i.d Gaussian errors in Python and the corresponding backtracking algorithm.

Exercise 22:

Test your algorithm on some simulated data and check its runtime complexity for various values of β.

3.3.4 Model selection

One important question when applying the previous algorithm is to set the value of β. As for the AMOC problem, a practical solution is to use simulation, essentially choosing a value such that the probability of detecting changepoints when there are none is low enough.

For a changepoint in the mean model with i.i.d. Gaussian errors, it has been shown in [Yao and Au, 1989] that a SIC-like penalty of $2\sigma^2 \log(n)$ is consistent in infill asymptotics [Yao and Au, 1989]. The precise proof of this result is, to the best of our efforts, mathematically involved (see [Fearnhead and Fryzlewicz, 2024] for a sketch). Below to provide some intuition of this important result and, as an exercise, we take a slightly different route to justify that a larger penalty of about $6\log(n)$ will, with high probability, not underestimate the number of changepoints and will not grossly overestimate it.

A penalty of order $\log(n)$

For any i, τ, j, we consider a local log-likelihood ratio statistic:

$$\sum_{t=i}^{j}(Y_t - \bar{Y}_{i:j})^2 - \sum_{t=i}^{\tau}(Y_t - \bar{Y}_{i:\tau})^2 - (Y_t - \bar{Y}_{\tau+1:j})^2$$

Assuming there are no changepoints in the data between i and j, a good penalty β should ensure that with high probability this statistic is smaller than β. To this end, recall from Chapter 2 that the previous statistic can be rewritten as the square of a CUSUM statistic, namely:

$$C_{i,\tau,j} = \sqrt{\frac{(\tau - i + 1)(j - \tau)}{j - i + 1}}(\bar{Y}_{\tau+1:j} - \bar{Y}_{i:\tau}).$$

Under the null hypothesis, the previous statistic can be controlled using a sub-Gaussian bound:

$$P(|C_{i,\tau,j}| > x) \leq e^{-1/2x^2}.$$

We have $\binom{n+1}{3} \leq n^3$ possible choice of triplets satisfying $1 \leq i \leq \tau < j \leq n$. Plugging $x^2 = 2\log\left(\binom{n}{3}\right) - 2\log(\alpha) \leq 6\log(n) - 2\log(\alpha)$ into the previous bound and summing over all possible triplets, we control

$$P\left(\exists\ i < \tau < j \text{ such that } |C_{i,\tau,j}| \geq x\right) \leq \alpha.$$

Our union bound is very crude and ignores the dependency between all $C_{i,\tau,j}$. Therefore, our $6\log(n)$ is conservative. It can be shown that the constant 2 (obtained in [Yao and Au, 1989]) is optimal in the sense that no smaller constant would work [Wainwright, 2019].

Exercise 23:

Assuming $\beta = 6\log(n) - 2\log(\alpha)$, show that the segmentation $\boldsymbol{\tau}$ obtained by minimizing (3.2) (the output of the OP algorithm) returns at most 2 changes between two true changes τ_k^* and τ_{k+1}^*.

Lower bound The result of the previous exercise states that with high probability $(1-\alpha)$, the OP algorithm with a penalty of $6\log(n) - 2\log(\alpha)$ returns $2\boldsymbol{\tau^*} + 2$ changes. In the following exercise, we aim to get a lower bound.

Exercise 24:

Assuming $\beta = 6\log(n) - 2\log(\alpha)$, show that the segmentation $\boldsymbol{\tau}$ obtained by minimizing (3.2) (the output of the OP algorithm) has at least one changepoint in the interval $\left(\frac{\tau_{k-1}^*+\tau_k^*}{2}, \frac{\tau_k^*+\tau_{k+1}^*}{2}\right)$, assuming the size of the change is sufficiently large. That is, assuming for some sufficiently large C we have:

$$|\delta|\sqrt{2\frac{(\tau_{k-1}^* + \tau_k^*)(\tau_k^* + \tau_{k+1}^*)}{(\tau_{k-1}^* + \tau_{k+1}^*)}} \geq \sqrt{C\beta}.$$

Upper bound The result of the previous exercise states that with high probability we detect at least all changepoints.

3.3.5 Some practical exercices

Exercise 25:

Test the algo on some simulated data and measure its H0 control and power to detect changes as a function of their height.

Exercise 26:

Same thing considering, that we do not know the variance and pre-estimating it with MAD.

Exercise 27:

(AT HOME) Refine your implementation of the OP algorithm by incorporating a minimum segment length constraint. Then, investigate the algorithm's performance in the presence of outliers.

Bibliography

[Anastasiou and Fryzlewicz, 2022] Anastasiou, A. and Fryzlewicz, P. (2022). Detecting multiple generalized change-points by isolating single ones. *Metrika*, 85(2):141–174.

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

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

[Bellman, 1961] Bellman, R. (1961). On the approximation of curves by line segments using dynamic programming. *Communications of the ACM*, 4(6):284.

[Celisse et al., 2018] Celisse, A., Marot, G., Pierre-Jean, M., and Rigaill, G. (2018). New efficient algorithms for multiple change-point detection with reproducing kernels. *Computational Statistics & Data Analysis*, 128:200–220.

[Cho and Kirch, 2019] Cho, H. and Kirch, C. (2019). Localised pruning for data segmentation based on multiscale change point procedures. *arXiv preprint arXiv:1910.12486*.

[Cleynen and Lebarbier, 2014] Cleynen, A. and Lebarbier, É. (2014). Segmentation of the poisson and negative binomial rate models: a penalized estimator. *ESAIM: Probability and Statistics*, 18:750–769.

[Enikeeva and Harchaoui, 2019] Enikeeva, F. and Harchaoui, Z. (2019). High-dimensional change-point detection under sparse alternatives.

[Enikeeva and Klopp, 2021] Enikeeva, F. and Klopp, O. (2021). Change-point detection in dynamic networks with missing links. *arXiv preprint arXiv:2106.14470*.

[Fearnhead and Fryzlewicz, 2024] Fearnhead, P. and Fryzlewicz, P. (2024). The multiple change-in-gaussian-mean problem. *arXiv preprint arXiv:2405.06796*.

[Fearnhead and Rigaill, 2020] Fearnhead, P. and Rigaill, G. (2020). Relating and comparing methods for detecting changes in mean. *Stat*, 9(1):e291.

[Fryzlewicz, 2014] Fryzlewicz, P. (2014). Wild binary segmentation for multiple change-point detection.

[Fryzlewicz, 2020] Fryzlewicz, P. (2020). Detecting possibly frequent change-points: Wild binary segmentation 2 and steepest-drop model selection. *Journal of the Korean Statistical Society*, 49(4):1027–1070.

[Garreau and Arlot, 2018] Garreau, D. and Arlot, S. (2018). Consistent change-point detection with kernels.

[Gombay and Horvath, 1990] Gombay, E. and Horvath, L. (1990). Asymptotic distributions of maximum likelihood tests for change in the mean. *Biometrika*, 77(2):411–414.

[Hinkley, 1971] Hinkley, D. V. (1971). Inference about the change-point from cumulative sum tests. *Biometrika*, 58(3):509–523.

[Hyman et al., 2002] Hyman, E., Kauraniemi, P., Hautaniemi, S., Wolf, M., Mousses, S., Rozenblum, E., Ringnér, M., Sauter, G., Monni, O., Elkahloun, A., et al. (2002). Impact of dna amplification on gene expression patterns in breast cancer. *Cancer research*, 62(21):6240–6245.

[Jackson et al., 2005] 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.

[James and Matteson, 2015] James, N. A. and Matteson, D. S. (2015). ecp: An r package for nonparametric multiple change point analysis of multivariate data. *Journal of Statistical Software*, 62:1–25.

[Killick and Eckley, 2014] Killick, R. and Eckley, I. A. (2014). changepoint: An r package for changepoint analysis. *Journal of statistical software*, 58:1–19.

[Killick et al., 2010] Killick, R., Eckley, I. A., Ewans, K., and Jonathan, P. (2010). Detection of changes in variance of oceanographic time-series using changepoint analysis. *Ocean Engineering*, 37(13):1120–1126.

[Killick et al., 2012] 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.

[Kovács et al., 2023] 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.

[Lai et al., 2005] Lai, W. R., Johnson, M. D., Kucherlapati, R., and Park, P. J. (2005). Comparative analysis of algorithms for identifying amplifications and deletions in array cgh data. *Bioinformatics*, 21(19):3763–3770.

[Liehrmann and Rigaill, 2024] Liehrmann, A. and Rigaill, G. (2024). Ms. fpop: a fast exact segmentation algorithm with a multiscale penalty. *Journal of Computational and Graphical Statistics*, pages 1–11.

[Maidstone et al., 2017] Maidstone, R., Hocking, T., Rigaill, G., and Fearnhead, P. (2017). On optimal multiple changepoint algorithms for large data. *Statistics and computing*, 27:519–533.

[Nam et al., 2007] Nam, R. K., Sugar, L., Wang, Z., Yang, W., Kitching, R., Klotz, L. H., Venkateswaran, V., Narod, S. A., and Seth, A. (2007). Expression of tmprss2: Erg gene fusion in prostate cancer cells is an important prognostic factor for cancer progression. *Cancer biology & therapy*, 6(1):40–45.

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

[Pein et al., 2017] Pein, F., Sieling, H., and Munk, A. (2017). Heterogeneous change point inference. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 79(4):1207–1227.

[Picard et al., 2005] Picard, F., Robin, S., Lavielle, M., Vaisse, C., and Daudin, J.-J. (2005). A statistical approach for array cgh data analysis. *BMC bioinformatics*, 6:1–14.

[Pishchagina et al., 2024] Pishchagina, L., Rigaill, G., and Runge, V. (2024). Geometric-based pruning rules for change point detection in multiple independent time series. *Computo*.

[Pishchagina et al., 2023] Pishchagina, L., Romano, G., Fearnhead, P., Runge, V., and Rigaill, G. (2023). Online multivariate changepoint detection: Leveraging links with computational geometry. *arXiv preprint arXiv:2311.01174*.

[Rigaill, 2015] Rigaill, G. (2015). A pruned dynamic programming algorithm to recover the best segmentations with 1 to $k\_\{max\}$ change-points. *Journal de la Société Française de Statistique*, 156(4):180–205.

[Rousseeuw and Croux, 1993] Rousseeuw, P. J. and Croux, C. (1993). Alternatives to the median absolute deviation. *Journal of the American Statistical Association*, 88(424):1273–1283.

[Runge et al., 2023] Runge, V., Hocking, T. D., Romano, G., Afghah, F., Fearnhead, P., and Rigaill, G. (2023). gfpop: an r package for univariate graph-constrained change-point detection. *Journal of Statistical Software*, 106:1–39.

[Schwaller and Robin, 2017] Schwaller, L. and Robin, S. (2017). Exact bayesian inference for off-line change-point detection in tree-structured graphical models. *Statistics and Computing*, 27:1331–1345.

[Truong et al., 2020] Truong, C., Oudre, L., and Vayatis, N. (2020). Selective review of offline change point detection methods. *Signal Processing*, 167:107299.

[Venkatraman and Olshen, 2007] 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.

[Verzelen et al., 2023] Verzelen, N., Fromont, M., Lerasle, M., and Reynaud-Bouret, P. (2023). Optimal change-point detection and localization. *The Annals of Statistics*, 51(4):1586–1610.

[Vincent-Salomon et al., 2013] Vincent-Salomon, A., Benhamo, V., Gravier, E., Rigaill, G., Gruel, N., Robin, S., de Rycke, Y., Mariani, O., Pierron, G., Gentien, D., et al. (2013). Genomic instability: a stronger prognostic marker than proliferation for early stage luminal breast carcinomas. *PLoS One*, 8(10):e76496.

[Wainwright, 2019] Wainwright, M. J. (2019). *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge university press.

[Wald, 1945] Wald, A. (1945). Sequential tests of statistical hypotheses. In *Ann. Math. Statist. (June, 1945)*, pages 117–186.

[Yao and Au, 1989] Yao, Y.-C. and Au, S.-T. (1989). Least-squares estimation of a step function. *Sankhyā: The Indian Journal of Statistics, Series A*, pages 370–381.