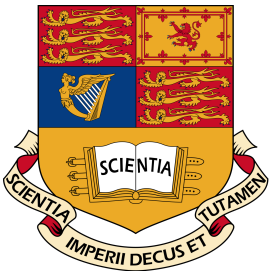
Imperial College London



Imperial College London

Department of Mathematics

An efficient non-parametric algorithm for robust offline changepoint detection

Author:
Karim Farahat

Supervisors:
Dr. Dean Bodenham
Professor Niall Adams

This dissertation presents an advanced research project associated with the degree of MSci Mathematics. This is my own original work, except where otherwise stated.

Abstract

Changepoint detection studies the development of algorithms with the purpose of identifying points in time under which the statistical properties of a stochastic process of interest undergo change. It is a subject of wide practical relevance. The problem can be viewed as a generalisation of goodness of fit testing and as such the results are often of interest as its own ends. Equally, it has use in conjunction with broader statistical methodology such as being used as a tool to identify model drift. This thesis begins with an independent review of the state of the art, including the presentation of simulation studies illustrating the practical performance of the core algorithms in a broad selection of scenarios. Thereafter a novel approach is introduced to address the limitations of one promising class of non-parametric approaches. It has been shown to improve upon previous iterations through meaningful improvements to classification accuracy while simultaneously improving computational efficiency.

June 13, 2021

Contents

Chapter 1

Introduction

1.1 Motivation

With the widespread availability of statistical computing software in accessible interpreted programming languages, analysing data, conducting statistical inference, and performing broader predictive modelling can all be implemented with relative ease. Building useful models to effectively solve live problems, however, is another issue entirely. There are several challenges one faces when deploying statistical methods in practice. One such fundamental issue is the quality of data any analyses are performed on. It is desirable for training data to resemble the situation of interest as closely as possible. Statistical methodology is therefore commonly underpinned by assumptions such as stationarity or that the population being modelled is homogenous. Changepoint detection algorithms are a useful tool to that ends, providing a systematic approach by which one can validate such assumptions. Specifically, changepoint detection provides one with algorithms to identify points in time where a stochastic process of interest undergoes change within one or more of its underlying statistical properties.

Furthermore, change detection can be viewed as a generalisation of goodness of fit testing and as such the results of deploying change detection algorithms often merit scientific interest. Changes in the underlying distribution of the data often corresponds to observable physical phenomena with an associated interpretation. The techniques discussed in detail within Chapters 2 and 4 of this project have recently been applied directly to datasets in the study of biology [1]-[3], economics [4], the environment [5]-[6], geology [7], history [8], infrastructure [9], sport [10], and transportation [11]-[12]. Changepoint detection is topical, tangible, and impactful.

1.2 Problem Specification

The detection of change has been studied as a problem in various forms. The first distinction to make is between online and offline detection. While coming from the same class of problem, the online vs. offline problems are fundamentally different. Online change detection studies the scenario of real-time monitoring, where one has access to an unending data stream. This formulation is characterised by its memory limitations, with the core priority being the detection of change with minimal time lag. Offline change detection allows access to the entire dataset and focuses on maximising classification effectiveness subject to constraints on time complexity. Applications of the offline problem are what the author has been alluding to in the preceding motivating discussion. The online problem is applied in fields such as quality process control. While also having a vast and rich literature, the scope of this project is constrained to the offline problem.

Beyond the division between online and offline detection, there are further variants of the problem which have been studied. Within the offline setting there is the question as to whether one is looking for a single change or multiple. A case which has received much attention is that of detecting a single change. This can be viewed as a special case of the problem which emphasises statistical methodology rather than questions of algorithm design. One of the central points of interest in this project involves the interaction between statistical methods and their algorithmic implementation. The case of interest therefore is the multiple detection case. It is interesting to note that any approach to detecting a single change can be readily extended to that of multiple by use of a greedy algorithm and as such the performance of methods within the single change case is nonetheless something which will receive attention in this project. Finally, there is a distinction to be made regarding the form of the data. In this thesis, the focus is on that of the univariate case.

1.3 Project Summary

The contributions of this thesis are threefold. The first lies in its exposition. An extensive literature review has been distilled and communicated in a descriptive manner, gradually developing the theory of changepoint detection. This could serve as a gentle introduction for those whom the problem is of practical significance but have limited background in the area.

Secondly, independent large-scale simulations illustrating the practical performance of various methods in a broad selection of scenarios have been conducted. Reproducing the results presented within the literature to date has been a core focus. In some instances, experimental results are found which present a more nuanced view of the problem than those available. This is intended to be of interest to anyone exploring the general algorithm design choices within changepoint detection.

Finally, an approach to address the limitations of one promising class of non-parametric approaches is introduced. It has been shown to improve upon previous iterations through both improvements to classification accuracy and computational efficiency. It performs reasonably when benchmarked against correctly specified parametric models, while having the notable upsides of being robust and able to detect arbitrary changes. This project closes by highlighting several interesting ways the methodology could be extended.

1.3.1 Chapter Breakdown

Chapter 2 discusses the problem in a general sense to provide background which would be helpful to those approaching the changepoint problem with minimal existing knowledge. Chapter 3 aims to expand on Chapter 2 by discussing the fundamentals of the most widely cited changepoint detection algorithms in detail. The base understanding conveyed in this chapter is essential for subsequent work. Chapter 4 briefly provides definitions for two non-parametric approaches. This is designed to be a self-contained reference for the novel approach subsequently developed. Chapter 5 contains the first presentation of simulation results. Its purpose serves to provide results which would be of interest to anyone studying questions of algorithm design in changepoint detection, while also highlighting themes to be revisited throughout Chapter 6. Chapter 6 contains detailed discussion of the construction of a novel approach which attempts to improve upon those in Chapter 4. It includes a reformulation of the cost function proposed to date and an approach to the efficient evaluation of the empirical cumulative distribution function in a Binary Segmentation framework more generally. Chapter 7 expands on that of Chapter 5 by presenting further simulation results. While there is broader relevance, greatest emphasis is placed upon robustness properties and evaluating the approach outlined in Chapter 6. Chapter 8 closes by first synthesising the general principles of algorithm design learned throughout the course of this thesis and secondly by outlining a number of interesting ways in which one could extend the novel methodology presented.

Chapter 2

Background

The purpose of this chapter is to describe aspects of the changepoint problem in a general sense. Various possible approaches to the problem are highlighted, alongside discussion of algorithm design and statistical methodology. The first objective of this chapter is to encourage familiarity with the problem by illustrating the themes of interest. Equally, it is to articulate a general philosophy and set of principles regarding the changepoint problem. This is of particular relevance to the construction of the approach introduced in Chapter 6.

2.1 Problem Definition

The primary object of study is a time series $\{X_t\}_{t=1:n}$. The realisation of interest will be referred to by $x_{1:n} = (x_0, x_1, ..., x_n) \in \mathbb{R}^n$ and its associated changepoints $\mathcal{T} = \{t_0, t_1, ..., t_{m+1}\}$ correspond to the points

$$X_{t_0}, X_{t_0+1}, \ ... \ , X_{t_1-1} \sim F_1$$

$$X_{t_1}, X_{t_1+1}, \ ... \ , X_{t_2-1} \sim F_2$$

$$\vdots$$

$$X_{t_m}, X_{t_m+1}, \ ... \ , X_{t_{m+1}-1} \sim F_j$$

where $F_i \neq F_j$. Note that $t_0 = 1$ and $t_{m+1} = n+1$ for notational convenience. Verbally, these are the points where the underlying distribution of the data generating mechanism changes. The stationary subsequences of this sample will be referred to as segments. Mathematically, segment i is $x_{t_i:t_{i+1}-1}$ with associated length $n_i = t_{i+1} - t_i$. The objective is to classify whether each point is a changepoint and hence produce estimates $\hat{\mathcal{T}} = \{\hat{t_0}, \hat{t_1}, ..., \hat{t_n}\}$ which well reflect $\mathcal{T}$. It is noted that in practice one does not know $\mathcal{T}$ and so this is an unsupervised learning problem.

To quantitatively approach the problem a cost function $C : \mathbb{T}_n \times \mathbb{R}^n \mapsto \mathbb{R}$, where $\mathbb{T}_n$ is the space of possible changepoint structures will need to be defined. Thereafter a rule which prompts the decision to classify a point as a changepoint will be introduced based on this function. There are two approaches to making such a definition. The first is to define $C(.)$ as a measure of difference between distributions such as with a statistic which arises from a hypothesis test. The other approach is to define $C(.)$ as a measure of model fit. The decision rule then arises by setting a threshold in the first case or by penalising and making relative comparisons in the second. It is noted that these are not necessarily different and often lead to equivalent or slightly modified ideas.

With basic definitions given, the next section will be devoted to discussion of the problem in a general sense. Any algorithm can be viewed as the combination of a cost function, a means of optimisation, and a decision rule. The purpose, rather than to enumerate all possible techniques, is to highlight broad themes of interest within the design of changepoint detection algorithms. Comprehensive overviews can be found in the papers [13] [14].

2.2 Algorithm Design

To begin, consider a cost function and decision rule given as it is useful to appreciate the context in which it is to be used prior to making any definitions. Obtaining an answer to which changepoint structure fits the dataset best with respect to the given measure will involve a combinatorial optimisation given the discrete nature of the problem. The simplest such method to find a solution, the naïve search, involves directly

evaluating and comparing the cost of every possible changepoint structure. To implement such a search in a structured way, the problem can be viewed as a multi-stage decision problem. Specifically, one has to decide whether or not there is a changepoint at every single time index excluding the first. Each choice leads to a unique solution and there is therefore 2^{n-1} possible changepoint configurations of a time series of length n. The combinatorial explosion arises from making decisions concurrently. This is one example of a broader concept known as the curse of dimensionality, termed by Richard Bellman [15] in his works outlining suitable approaches to such problems.

Given the breadth of possible applications, many inevitably occur in circumstances with abundant data. Exponential time complexity prior to having considered the burden of evaluating the cost function renders such an approach unusable in any situation of practical interest. Returning, as an example, to the changepoint analysis of genomic applications referred to in the introduction, the article [16] highlights the tremendous scale of data such projects can generate and [17] contains discussion of the software considerations in developing an R package for change detection with efficient use of memory to address this context. The need for efficiency is a core priority in the development of changepoint detection algorithms.

To this end, note that the majority of the possible changepoint structures considered by the naïve search are unreasonable solutions. That which introduces a new changepoint at every time-step, for example, is unlikely to ever be considered appropriate. While accepting their inevitable limitations, it is nonetheless reasonable to expect that one can employ heuristic methods to produce acceptable solutions without the same computational burden. The development and study of such heuristics has been a core focus within the literature. Many variations of related ideas have been proposed and two canonical examples, that of Binary Segmentation and a greedy Dynamic Programming approach, are explored within this project.

2.3 Cost Functions

Identifying differences between distributions given two samples of data is something which has been studied extensively. Classical hypothesis tests of this kind can be viewed as a special case of the single changepoint detection problem. Namely, that of detecting a single change given a known location. A similar notion holds for comparisons between the relative quality of different statistical models. Any statistic used for these purposes can be extended to form the basis of a change detection algorithm and therefore are possible definitions. There is a wealth of possible techniques which can be drawn upon. To motivate understanding of the problem, a general discussion of several approaches follows.

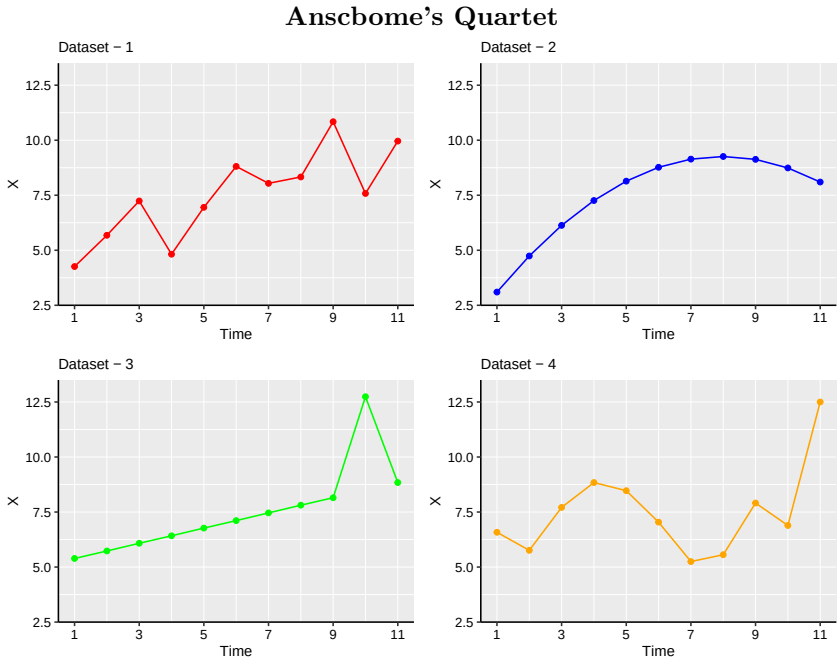
2.3.1 Parametric Approaches

A common feature of many elementary approaches to problems in statistical modelling are assumptions that the data being modelled is distributed according to some simple parametric form. Typically this will give rise to known analytic expressions for quantities of interest, such as moments or maximum likelihood estimates. For instance, much theory is underpinned by the iconic normality assumptions,

$$f(x|\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}.$$

Regardless of whether the distribution of the data strictly follows a given form or not, one can often find a suitable model this way. Moreover, in such circumstances it is not necessarily productive to apply more sophisticated tools. These make for convenient problems to solve. This is particularly relevant in the context of change detection as this gives rise to a concise way of describing a distribution. Namely, by specifying a limited number of parameters.

Approaches to identifying change can then be defined by reference to estimates of these parameters. A common choice being the penalised negative maximum likelihood, which, under independence assumptions, is



Dataset	Sample Mean	Sample Variance
1	7.5	4.1
2	7.5	4.1
3	7.5	4.1
4	7.5	4.1

Figure 2.1: A pertinent illustration of the possible pitfalls of taking an approach based upon assumptions of a simple parametric form. This figure was created with ggplot2 [18]

.

$$C(x_{t_0:t_1-1}, ..., x_{t_m:t_{m+1}-1}) = -\sum_{i=0}^{m} \sum_{j=t_i}^{t_{i+1}-1} \log(f(x_j|\hat{\theta}_i)) + \beta m$$

where $\hat{\theta}_i$ is the maximum likelihood estimate of the distribution parameters derived from the data contained in segment i. Subject to the truth of these assumptions, such an approach would be among the most powerful and efficient available. However, it is also worth highlighting the significance of this caveat. The foremost observation to make now is the generality of the problem being attempted. The definition as stated in Section 2.1 allows for arbitrary changes in distribution, so there is no constraint ensuring that every point in a given dataset be generated from a single parametric family.

Figure 2.1, known as Anscombe's Quartet [19], is a succinct illustration of a fundamental issue to contend with when attempting to detect change in parametric frameworks. Namely, it shows four datasets generated from different distributions with identical summary statistics. This is an inherent imperfection to the dimensionality reduction involved in describing a distribution, characterised in this context by a probability measure over $\mathbb{R}$, by a few metrics.

Alongside the introduction of the quartet, Anscombe highlights the utility of graphs to address such issues. In another context it may be considered sufficient to examine the dataset with a diagnostic plot and perform a statistical test to conclude the data follows a given distribution with fallible but nonetheless reasonable certainty. Within this context the use of such analyses must be done so with particular caution. It is entirely plausible to have a sizeable time series, the majority of which is normally distributed such that these criteria are satisfied, with regions resembling the features of Anscombe's Quartet. Any function of the mean and variance would have no discrimination power for such changes, meanwhile the user remains oblivious given that one does not know the nature of change in advance. In short, it is difficult to validate

distributional assumptions and verify the correctness of results. This does not mention other issues arising from misspecification such as robustness, a theme to be explored in detail in subsequent chapters.

2.3.2 Non-Parametric Approaches

A natural reaction to this would be to look to define a cost function with greater generality. Given that there could be arbitrary change, one could ask for a method which is able to handle this. Representing these ideas mathematically, the true nature of the problem with the normality assumptions in the last example is not strictly that they are not necessarily true, but that normal assumptions only capture changes in

$$\int_{\mathbb{R}} x dF \quad \text{and} \quad \int_{\mathbb{R}} x^2 dF,$$

while

$$\int_{\mathbb{R}} x dF_1 = \int_{\mathbb{R}} x dF_2$$

and

$$\int_{\mathbb{R}} x^2 dF_1 = \int_{\mathbb{R}} x^2 dF_2$$

does not imply that

$$F_1 = F_2.$$

Effectively, it is possible to adjust the shape of the distribution without shifting location or scale and this would be captured by higher moments. Asking for a method which can detect arbitrary change is equivalent to looking for a function $d : \mathbb{F} \times \mathbb{F} \mapsto \mathbb{R}$ satisfying

$$d(F_1, F_2) = 0 \implies F_1 = F_2$$

where $\mathbb{F}$ denotes the space of cumulative distribution functions over $\mathbb{R}$. Or, in other words, a function which satisfies the identifiability axiom of a metric.

Fortunately, functional analysis is foundational to much of mathematics and so is particularly well studied. As such, there is an abundance of such measures. Consider, for instance, one popular choice in a family of information theoretic measures, the KL Divergence. This was introduced in [20] with further details on its properties in [21]. For continuous random variables with CDFs F_1, F_2 one form is,

$$D_{KL}(F_1||F_2) = \int_{\mathbb{R}} \log\left(\frac{F_1}{F_2}\right) dF_1.$$

One can construct a cost function based on this and indeed the cost function associated with the novel algorithm introduced within Chapter 6 is one example of doing so. The foremost issue with such an approach is that one needs to estimate F_1 and F_2 in some way. There are multiple ways of doing so. Perhaps the simplest makes use of the empirical distribution function

$$\hat{F}(s) = \frac{1}{n} \sum_{j=1}^{n} 1_{X_j \leq s}.$$

A point to be discussed in Chapter 4 is that the empirical distribution function is a maximum likelihood estimator for the true distribution $F(s)$ with no apriori assumptions beyond that of X_i being independent and identically distributed. A widely known property of maximum likelihood estimators is their functional invariance [22]. Supported by arguments such as the consistency of maximum likelihood estimators [23] and the continuous mapping theorem [24], one could make the case that in the idealised case of infinite observations per distribution, any such approach would be highly effective.

The qualifying statement made within this last point is nonetheless akin to an assumption like any other. The main difference is that it is made implicitly. Specifically, it is that one has sufficient data per segment to detect changes with reasonable power. In the same vein that there was no restriction placed upon every underlying distribution being drawn from a particular parametric family, there is equally no restriction placed on the minimum segment size. Hence, again, making no assumptions is not necessarily a clear solution to the problem either.

2.3.3 Alternatives

A major distinction drawn within this thesis is the comparison between simple parametric vs. non-parametric approaches at their most general. Such approaches are on opposite ends of the spectrum and therefore such a distinction is an oversimplification. There is a variety of tools one could turn to beyond these in order to strike a compromise between the two.

For instance, in response to the simplicity of the normal distribution one could turn to more complicated parametric forms. There are numerous generalised distributions proposed within the literature which can be moulded to a desired form. Consider, for instance, a generalised normal distribution such as described in [25]. The flexibility to model a broader range of processes often is the primary motivator for such developments. Equally, one could consider instead composing several known distributions into a mixture model [26].

To the best of the author's knowledge, no approach of this form has been proposed within the literature. There is one notable disadvantage to taking such an approach, beyond the fact that it is simply difficult to model a process undergoing arbitrary change at arbitrary times. Instead, it is that the likelihood rapidly becomes analytically intractable. The problem with this is not that one is unable to maximise functions of this nature, as there exists tools such as the EM algorithm [27] in the case of the mixture model or numerous other iterative methods in the case of generalised distributions. Rather, the changepoint problem can become computationally intractable also given the themes highlighted in Section 2.2.1.

Alternatively, another non-parametric approach draws on the theory of Hilbert Spaces and Reproducing Kernels. Overviews of this can be found in [28]. A useful insight from this domain allows one to compute distance measures, such as those discussed so far, over spaces of distribution functions in a computationally efficient and powerful manner through statistics such as the Maximum Mean Discrepancy [29]. Note that one can only estimate particular norms, corresponding to Reproducing Kernels, with such an approach. The polynomial kernel, for example, allows one to account for $\mathbb{E}(X^d)$ up to a specified d. Choosing such a norm requires foresight which again incurs the risk of inappropriate selection for the potential upside of power gains.

2.3.4 Review

Within the changepoint problem there is a concurrent interest in power, generality, robustness, and computational efficiency. As such, not only are there particular constraints to pay attention to but in fact often these are conflicting priorities. Of the tools available to us, all have strengths and weaknesses. It is because of these tradeoffs it is scarcely the case that, subject to intelligent design, one method is uniformly superior to another. To credibly answer the question of which is best would require the definition of a utility function. This can reasonably be done. The issue, however, is again that it is entirely domain specific. Hence, much of this project involves exploring the conditions which make a method perform better or worse. Quality research includes introducing approaches which increase one of the desirable attributes without having to sacrifice an excessive amount of other desirable qualities.

This is the culmination of this thesis. Non-parametric approaches to changepoint detection have natural appeal given the generality of the problem. One fundamental issue with the existing non-parametric methods, however, is excessive computational cost. Chapter 6 introduces improvements to existing methods through efficiency improvements without having to make compromises regarding classification effectiveness. More generally, it is shown that this class of approach has particularly attractive robustness at the expense of minimal power relative to parametric approaches.

Chapter 3

A Review of Parametric Approaches

The purpose of this chapter is to discuss the elements of changepoint detection. The classical approaches within the field fall under this parametric framework, primarily being underpinned by normality assumptions. The methods presented include Binary Segmentation, Wild Binary Segmentation, Optimal Partitioning, and PELT. These are introduced in the papers [30], [31], [32], and [33] respectively. While discussion of the approaches to defining cost functions and regularisation is included throughout, most emphasis is placed on the various aspects of algorithm design within the context of changepoint detection.

3.1 Binary Segmentation

3.1.1 Overview

In Section 2.2 two fundamental characteristics of the changepoint problem were introduced. The first was the overwhelming number of possible changepoint structures. The second noted that the material imbalance between the number of changepoints and non-changepoints causes most solutions to be poor ones. Hence, it is generally unnecessary to consider every solution. A recurring theme within the study of algorithm design in situations such as this is to develop what is known as a greedy aproach [34]. This will typically involve making a convenient assumption, while fully acknowledging that it may be an oversimplification, in order to solve a simpler problem that may still provide reasonable solutions. All algorithms of interest within this thesis are greedy.

The first approach can be understood by way of a simple analogy. First, consider Figure 3.1. The data undergoes a change in mean after every 50 points sampled. The changes which occur at an odd multiple of 50 are relatively subtle, while those occurring at an even multiple of 50 are relatively apparent. Consider a situation where an individual was asked to examine this dataset and identify changepoints instinctively. Perhaps a natural way of doing so would be by taking several glances over the dataset, noting first the most abrupt changes, followed by a more careful inspection of the particular subsections of the data where one is less certain. Binary Segmentation greedily finds changepoints by implementing a similar procedure.

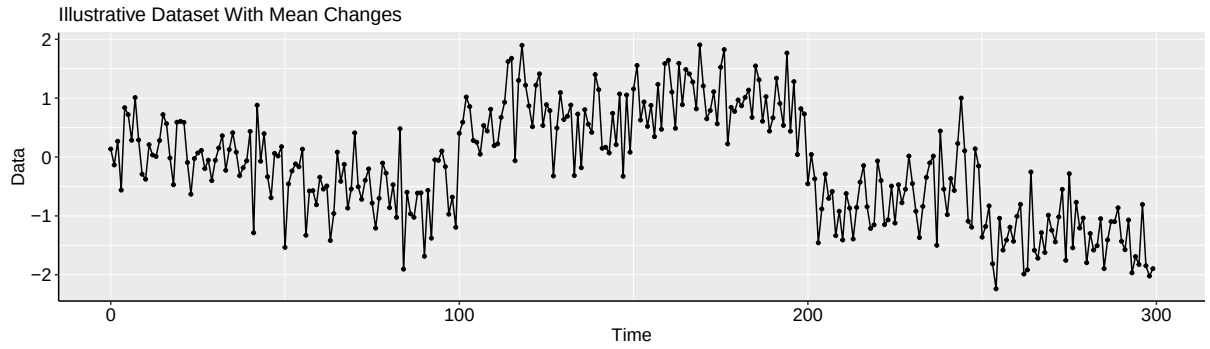


Figure 3.1: An illustration of a dataset with changes in mean. 50 points were sampled from each of $N(0, 0.5)$, $N(-0.5, 0.5)$, $N(-0.75, 0.5)$, $N(1, 0.5)$, $N(-0.5, 0.5)$, and $N(-1.25, 0.5)$ respectively.

3.1.2 Algorithm Outline

To translate the motivating analogy into mathematics, a cost function, $C(.)$, to quantify the degree of confidence one has in a potential changepoint is required. Given observations from $X_1, X_2, ..., X_j$ and $X_{j+1}, X_{j+2}, ..., X_n$ with distribution functions F_1 and F_2, the case of testing

$$H_0 : F_1 = F_2$$

versus

$$H_1 : F_1 \neq F_2$$

is well understood. Any statistic used to conduct such a test can be used as this cost function. The idea then is to scan the dataset for changepoints by splitting it into two segments at each time index and computing this statistic. The most apparent potential changepoint, t_i, is considered to be the index at which the statistic is maximised,

$$t_i = \arg\max_j C(x_{1:j-1}, x_{j:n}).$$

Whether this point should be classified as a changepoint is decided by setting a significance threshold of β and evaluating whether $C(x_{1:t_i-1}, x_{t_i:n}) > \beta$. If t_i is considered to be a changepoint, exploration of the dataset continues by applying the same process to both $x_{1:t_i-1}$ and $x_{t_i:n}$. Otherwise, one stops and concludes that all changepoints have been found. Pseudocode implementing this approach recursively follows.

Algorithm 1 Binary Segmentation

```
Procedure BinSeg(.)
Input:
Dataset (x_1, ..., x_n) ∈ ℝ^n
Cost function C(.)
Penalty β ∈ ℝ^+

Initialise:
Max_Cost ← 0
Max_Changepoint_Index ← 0
Changepoints ← Empty container for changepoint locations

Identifying Changepoint:
For t ∈ {1, 2, ..., n}:
    Current_Cost ← C(x_{1:t}, x_{t+1:n})
    If Current_Cost > Max_Cost:
        Max_Cost ← Current_Cost
        Max_Changepoint_Index ← t

Recursion:
If Max_Cost > β:
    Changepoints.insert(Max_Changepoint_Index)
    Changepoints.insert(BinSeg(Dataset[ : Max_Changepoint_Index], C(.), β))
    Changepoints.insert(BinSeg(Dataset[Max_Changepoint_Index : ], C(.), β))

Return Changepoints
```

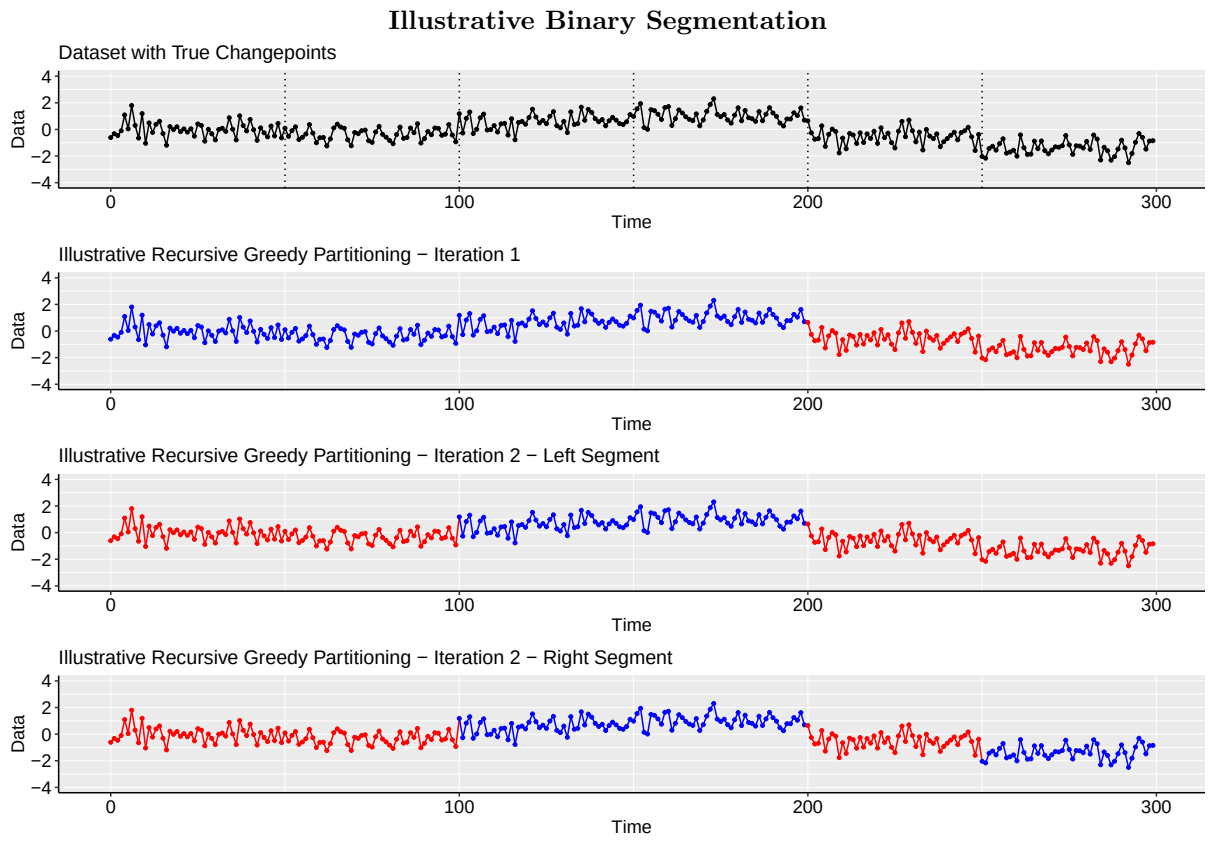


Figure 3.2: An illustration of Binary Segmentation. In the figure, a changepoint identified corresponds to a colour change. In each iteration, a linear sweep of a sub-segment is performed and either a point is chosen to further partition the data or exploration terminates.

3.1.3 Cost Function

While it has been noted that Binary Segmentation can be implemented with any statistic measuring differences between samples, it is most commonly associated with a statistic known as the cumulative sum (CUSUM) [35]. In this case,

$$C(x_{1:t_i-1}, x_{t_i:n}) = \left| \sqrt{\frac{(n-t_i)}{nt_i}} \sum_{i=1}^{t_i-1} x_i - \sqrt{\frac{t_i}{n(n-t_i)}} \sum_{i=t_i}^{n} x_i \right|.$$

This statistic is the square root of a likelihood ratio test where

$$X_1, X_2, ..., X_{t_i-1} \sim N(\mu_1, \sigma),$$

$$X_{t_i}, X_{t_i+1}, ..., X_n \sim N(\mu_2, \sigma),$$

and the hypothesis being tested is

$$H_0 : \mu_1 = \mu_2.$$

A derivation of this result can be found in Section 2.1.1 of the book [36]. Binary Segmentation is implemented with this statistic in the CRAN package WBS and it will be assumed that this is the implementation of interest unless otherwise stated.

Regarding this choice of cost, note that the likelihood is the foundation of much of modern statistical inference and will in fact underpin every method discussed within this thesis. Maximum likelihood estimates have several desirable properties, with examples being consistency and efficiency [22]. Also notable is the

Neyman-Pearson lemma [37], stating that likelihood ratio tests are the uniformly most powerful test for a certain class of hypothesis.

While the Neyman-Pearson lemma is a noteworthy motivating result, it should also be stated that this by no means guarantees that this is the best cost function for every situation. There are two possible issues. First, the hypothesis being tested arises from the greedy criteria which is being used to deliberately oversimplify the situation. Second, recall that misspecification is always of primary concern. The effectiveness will largely be a function of how well the normality assumptions are met. The discussion raised in Section 2.2 regarding Anscombe's Quartet is therefore still particularly relevant. A further point largely unaddressed with the discussion regarding Anscombe's Quartet is false positives. The thin tails of the normal distribution result in the cost function placing significant emphasis on large deviations. Robustness in the presence of heavy noise is therefore a particular issue.

3.1.4 Regularisation

After setting a cost function, the next decision is regarding the threshold. First, note that throughout the machine learning literature more generally, the tuning of hyperparameters is a recurring topic. There are several possible approaches to this. In a supervised learning context, often a variant of cross-validation [38] is used. Cross-validation has several attractive upsides. One benefit is parameters which are tailored to the specific context in which any method is to be deployed. Another is an estimate of performance. In this unsupervised context, however, such an approach is unavailable.

Another alternative is to frame this as a multiple testing problem. Binary Segmentation can be modified to do so by computing the p-value associated with each change and combining them through a method for dependent tests such as the harmonic-mean [39]. The algorithm would then terminate once a desired family-wise false positive rate has been reached. Note, however, that no CRAN implementation has done so and presumably this is due to one notable downside. The distribution of the CUSUM does not have a simple analytic form, thus requiring a numerical procedure to compute p-values. Details on the distribution can again be found in Section 2.1.1 of [36] and discussion of a similar example can be found in Section 8.3.2.

Given that the CUSUM is likelihood based, model selection criteria are also relevant. This instance of the problem can be viewed as deciding whether fitting a piecewise constant signal to the data with one or two means is preferable. Information criteria incorporate a measure of model fit and complexity in a simple analytic expression which can be used to make such judgements. The canonical examples are Akaike's Information Criteria [40],

$$\text{AIC} = 2p - 2\log(\hat{\mathcal{L}})$$

or the Bayesian (Schwarz's) Information Criteria [41]

$$\text{BIC} = p\log(n) - 2\log(\hat{\mathcal{L}})$$

where p is the number of model parameters and $\hat{\mathcal{L}}$ the maximised likelihood. A lower score is preferred and so $\text{BIC}_1 < \text{BIC}_2$ would suggest that the quality of model 1 is superior to that of model 2.

Owing to their generality and convenience, information criteria are also a foundational tool within statistical inference. The paper introducing AIC is famously among the most cited in modern mathematics. Interestingly, in specific regression contexts, these have been shown to be asymptotically equivalent to variants of cross-validation [42]. While motivating their use, it is equally of note that these results do not directly transfer to the changepoint problem [43] and hence there is a separate study of consistency properties.

In the case of Binary Segmentation and the CUSUM, the recommendation in [31] is a score strengthening the complexity penalisation of the BIC

$$\text{SSIC}_\alpha = \log^\alpha(n) - 2\log(\hat{\mathcal{L}})$$

where it is recommended to take $\alpha = 1.01$. Recall that the CUSUM arises from a likelihood ratio and so is a difference of log-likelihood functions. A comparison between model selection criteria can also readily be translated to a threshold in this context by differencing. In this case the recommendation is

$$\beta = \hat{\sigma}\sqrt{\frac{\log^{1.01}(n)}{2}}$$

where $\hat{\sigma}$ is a maximum likelihood estimate of the common variance. Recall that the square root arises from the derivation of the CUSUM.

Note, one point being alluded to through discussion of these various approaches is to highlight the difficulty and subjectivity in choosing a penalty. This discussion has focused on the strengthened BIC due to it being the best performing in the simulation studies presented alongside the implementation of interest. Equally, however, there were several recommendations given and the package includes functionality to set this in 5 alternative ways. Briefly, that which receives most discussion outside of the strengthened BIC is

$$\beta = C\hat{\sigma}\sqrt{2\log(n)}$$

where $\hat{\sigma}$ is instead the median absolute deviation. It has been stated that setting $C = 1$ or $C = 1.3$ works well.

3.1.5 Time Complexity

First, the time complexity of the cost function is briefly to be discussed. In the case of the CUSUM, the benefit of the precise form given over an equivalent way of expressing the same likelihood ratio is its ease of evaluation. One can compute and store the partial sums,

$$\sum_{i=1}^{k} X_i,$$

for $k = 1, ..., n$ with $\mathcal{O}(n)$ operations while initialising the algorithm. Thereafter any partial sums over indexes j_1 and j_2 can be computed with $\mathcal{O}(1)$ operations by referring back to these previous calculations. Any subsequent value of the CUSUM can therefore also be evaluated in $\mathcal{O}(1)$ time. Outside of the power arising from the use of the likelihood ratio when well specified, this is also one of its most desirable features.

Next, note that a single iteration of the algorithm corresponds to computing the cost function for every point within the segment considered. There is therefore n_i $\mathcal{O}(1)$ calls to the cost function which is bounded by $\mathcal{O}(n)$. The time complexity thereafter depends on the data as a function of the number of changepoints found. To speak precisely about an expected time complexity would therefore require the definition of a stochastic model for the changepoint generating process. Assumptions would solely dictate this time complexity and instead highlighting the scope of what is possible is most informative.

The worst case scenario is where a changepoint is found at every time within the dataset and this corresponds to n calls of the $\mathcal{O}(n)$ routine which identifies a single change and therefore an overall time complexity of $\mathcal{O}(n^2)$. Contrastingly, the best-case scenario is where no changepoints are found in which case there is only a single call to the change identifying routine and therefore time complexity of $\mathcal{O}(n)$. Both scenarios are somewhat rare in practice, though that of no changes is clearly less so than that of every point being declared a changepoint. A more commonly imposed assumption is that the number of changepoints found is proportionally logarithmic to the length of the dataset. Such a case would occur if the dataset were bisected precisely at its midpoint during every iteration for example. A similar result would hold were another fixed ratio to occur and the time complexity in these cases would be $\mathcal{O}(n\log(n))$. Note that for comparisons between algorithms, assigning a base time complexity is useful and this result shall be used for this purpose. However, to reiterate the sentiment of the preceding paragraph, the assumptions underpinning this result and the scope for variation are most important to note.

3.2 Wild Binary Segmentation

3.2.1 Overview

Recall two points raised in the discussion of Binary Segmentation. First, a motivation for the algorithm is the need to develop a greedy criteria to rapidly discard potential changepoints from the dataset. The approach introduced achieves this with an approach that resembles the Binary Search. Specifically, by potentially removing a sizeable segment of the data from consideration based on the binary outcome of $C(x_{1:t_i-1}, x_{t_i:n}) > \beta$. In contrast to the simpler deterministic problem considered by the Binary Search, however, in some cases this can be problematic.

Note now the second point, the basis of the algorithm is an assumption that oversimplifies the true situation of interest. Specifically, it computes a statistic which arises from a hypothesis test under the assumption that there is only a single change within a segment. There are cases where the statistic is demonstrated to lose power in the presence of multiple changes. One such example is contained in Appendix A.2. This allows for an approach which repeatedly solves a relatively simple single stage decision problem, rather than a more complicated multi-stage decision problem where the implications of past decisions are continually re-evaluated. A notable downside to this is that by never reconsidering a decision, failing to detect a single change in a segment of data once causes every changepoint within that segment to also be lost. In such a case, many practitioners would consider the loss in classification effectiveness to offset the computational benefit.

Wild Binary Segmentation (WBS) is a variant of Binary Segmentation (SBS) which aims to address this. The algorithm is largely the same in the sense that changepoints are still found by partitioning the dataset as illustrated in Figure 3.2. The difference lies in the way the specific point to partition the data is chosen. In addition to performing a linear sweep of each segment as per the classical approach, WBS also begins to consider subsegments which would potentially only be considered by SBS in the next iteration were there a changepoint found in the current segment. The changepoint candidate is then selected to be the point which presents the strongest evidence of change, regardless of whether that be when viewed from the perspective of the entire segment or of any subsegment.

Wild Binary Segmentation therefore introduces an approach to address the issue raised by proactively considering possible changepoints which otherwise would have been missed were there not a changepoint found in the total segment. In doing so, it retains the potential benefits of the general Binary Segmentation approach, opposed to addressing the issue by incurring the computational cost associated with reconsidering decisions.

3.2.2 Algorithm Outline

Recall that Binary Segmentation would consider the changepoint candidate maximising the value of the CUSUM which arises from a likelihood ratio test with

$$X_1, X_2, ..., X_{j-1} \sim N(\mu_1, \sigma),$$

$$X_j, X_{j+1}, ..., X_n \sim N(\mu_2, \sigma),$$

where the hypothesis considered is

$$H_0 : \mu_1 = \mu_2$$

over each $j \in \{1, 2, ..., n\}$.

On the other hand, Wild Binary Segmentation relaxes this by only assuming that

$$X_s, X_{s+1}, ..., X_{b-1} \sim N(\mu_1, \sigma),$$

$$X_b, X_{b+1}, ..., X_e \sim N(\mu_2, \sigma),$$

and considering the changepoint candidate, t_i, to be

$$t_i = \arg \max_{(s,e)\in M} \max_{s\leq b\leq e} \left| \sqrt{\frac{(e-b)}{(e-s+1)(b-s)}} \sum_{i=s}^{b-1} x_i - \sqrt{\frac{b-s}{(e-s+1)(e-b-1)}} \sum_{i=b}^{e} x_i \right|,$$

the maximum CUSUM over $M = M_{1,n} \cup \{(1, n)\}$, where $M_{1,n} \subset \{(s, e) : s, e \in \{1, .., n\} \text{ with } s < e\}$ consists of uniformly randomly sampled pairs of time indices. WBS therefore supplements SBS by considering randomly selected additional intervals over which to perform the same procedure as SBS.

The remainder of the procedure is then identical to that of SBS. The decision to introduce a changepoint is again evaluated by $C(x_{1:t_i-1}, x_{t_i:n}) > \beta$, where β is one of the penalties discussed in Section 3.1.4. If t_i is considered to be a changepoint, exploration of the dataset continues by applying the same process to both $x_{1:t_i-1}$ and $x_{t_i:n}$. Otherwise, one stops and concludes that all changepoints have been found. Pseudocode implementing this approach follows.

Algorithm 2 Wild Binary Segmentation

Procedure Wild_BinSeg(.)
Input:
Dataset $(x_1, ..., x_n) \in \mathbb{R}^n$
Cost function $C(.)$
Penalty $\beta \in \mathbb{R}^+$
A hyperparameter M dictating the size of the random sample

Initialise:
Max_Changepoint_Index $\leftarrow 1$
Max_Cost $\leftarrow 0$
Changepoints $\leftarrow$ Empty container for changepoint locations
$S \leftarrow \{(\min(s, e), \max(s, e))) : \text{ For } (s, e) \in \text{ sample}(\{1, 2, ..., n\}, \text{ size } = M)\}$

Identifying Changepoint:
For $(s, e) \in S$:
    Interval_Max_Cost $\leftarrow 0$
    For $b \in \{s, s + 1, ..., e\}$:
        Current_Cost $\leftarrow C(x_{s:b-1}, x_{b:e})$
        If Current_Cost $>$ Interval_Max_Cost:
            Interval_Max_Cost $\leftarrow$ Current_Cost
            Interval_Changepoint_Index $\leftarrow$ b
    If Interval_Max_Cost $>$ Max_Cost:
        Max_Cost $\leftarrow$ Interval_Max_Cost
        Max_Changepoint_Index $\leftarrow$ Interval_Changepoint_Index

Recursion:
If Max_Cost $> \beta$:
    Changepoints.insert(Max_Changepoint_Index)
    Changepoints.insert(Wild_BinSeg(Dataset[: Max_Changepoint_Index], $C(.)$, β, M))
    Changepoints.insert(Wild_BinSeg(Dataset[Max_Changepoint_Index :], $C(.)$, β, M))

Return Changepoints

3.2.3 Discussion

The differences in the theoretical properties of the classical versus Wild variants of Binary Segmentation arise primarily in time complexity and consistency. Briefly, the time complexity of WBS can be understood by realising that from this perspective it is equivalent to applying SBS M times and hence is $\mathcal{O}(Mn\log(n))$. Beyond the discussion contained in Section 3.1.5, the only additional point to note is the choice of parameter. Note that no recommendation with an explicit dependence on the length of the dataset is given. Instead, Section 4.1 of the paper [31] offers descriptive remarks. The author recommends choosing M to be as large as is tolerable computationally, while acknowledging that modest values can also be beneficial. The time complexity is therefore largely dictated by the practitioners preference.

Regarding the consistency, first consider the motivation behind the use of a random sample of intervals. It is noted that short segment lengths are a common cause of the false negative issue highlighted in Section 3.2.1. Of particular relevance is a result demonstrating that the consistency of SBS is contingent upon a minimum segment size between changepoints of $\mathcal{O}(n^{\frac{3}{4}})$. Comparatively, WBS permits a noticeably improved minimum spacing of $\mathcal{O}(\log(n))$. Further details can be found in Theorem 3.2 of [31]. Briefly, this is attributed in part to there being a high probability of sampling a segment under which the assumptions underpinning the CUSUM are true. Additionally, the equal emphasis placed on short and long segments is noted.

These consistency results highlight the possible upside to using WBS over SBS. Namely, an improved rate of convergence in the segment length. A point to note, however, is that consistency results are by nature idealised. Superior asymptotic properties by no means guarantee superior finite sample performance. These do not speak to performance under misspecification, for instance. A point of interest in Chapter 5 will be to observe how these theoretical properties translate to classification performance.

3.3 Dynamic Programming

3.3.1 Contrasting Approaches

The principal upside to using a Binary Segmentation based approach is its efficiency. Using a greedy heuristic, solutions were found by repeatedly solving one simple decision problem thereby reducing time complexity from exponential to log-linear. This simpler decision problem arose from a deliberate over-simplifying assumption which makes decisions as if there is only one changepoint within a given segment of the dataset and calculating the test statistic as such. Regardless, the approach can still be asymptotically consistent and is highly effective practically as will be demonstrated in Chapter 5.

Equally, note that any assumption has its drawbacks. As highlighted by Wild Binary Segmentation, multiple changepoints in close proximity can be masked under the one changepoint assumption when there are offsetting variations. Alternately, the local optimisation can result in the identification of a changepoint which is an amalgamation of several true changes, without sufficiently representing well any given individual change.

From a computational perspective, the heuristic is most useful when the stopping criterion correctly terminates the algorithm after a limited number of recursive steps. This, for instance, could correspond to a bisection of the dataset at the midpoint and the decision to disregard half of it within two iterations. Again, this relies on sufficient spacing between changepoints. Were there regularly spaced changepoints throughout the dataset at constant intervals of relatively small width, one has the somewhat cumbersome $\mathcal{O}(n^2)$ time complexity which arises from repeatedly performing linear scans of the dataset.

While it is not expected nor necessary that any method perform well in all situations, it is necessary to have an alternative an approach available that will be able to. To that ends, this section introduces another greedy heuristic to the changepoint problem which is an adaption of a Dynamic Programming based approach. Conveniently, it can handle such situations with the same effectiveness as Binary Segmentation in the case of few changepoints.

3.4 Optimal Partitioning

3.4.1 Overview

The notion that the solution space grows exponentially as the size of the dataset increases linearly was introduced alongside the naive search. A consequence of this is that optimising over any subset of the data is significantly simpler than optimising over the full dataset. Dynamic Programming [15] is a general framework for solving multiple decision problems which can exploit this feature.

Algorithms based on Dynamic Programming often bear resemblance to a greedy approach, but with key differences. In developing Binary Segmentation's greedy heuristic, the primary goal was to repeatedly solve an easier problem than the original. It is a consequence of the structure of the changepoint problem that one builds solutions by recursively exploring subsets of the dataset. The distinction between Binary Segmentation and any approach based on Dynamic Programming is that the former fixes a changepoint within its solution at every stage aside from when it terminates. The latter explores an increasing sequence of subsets of the data without fixing any decisions. Rather, the focus is on removing any solution known to be sub-optimal at each stage. Again, this is fruitful as a sub-optimal solution introduced at time k creates 2^{n-k} sub-optimal solutions within the final solution space. A decision is then made as to which changepoint structure to return at the final stage with relative ease.

A key ingredient to Dynamic Programming is known as optimal substructure. This involves creating an approach in which one can express solutions to the final problem as a function of solutions of sub-problems. This gives rise to a means by which one can combine solutions and a criteria to decide which solutions to rule out. There have been several approaches to invoking optimal substructure within the changepoint literature, such as one early variant being [44], known as the Segmented Neighbourhood and this approach will be discussed briefly within Chapter 4. A subsequent development was introduced in [32], known as Optimal Partitioning and it is this approach which will form the foundation of the remainder of this section.

3.4.2 Cost Function

The basis of this approach is a cost function, $C(.)$, which is additive, namely that

$$C(x_{t_0:t_1-1}, ..., x_{t_m:t_{m+1}-1}) = \sum_{i=0}^{m} C(x_{t_i:t_{i+1}-1}).$$

Changepoints are then identified by finding

$$\min_{(t_0,...,t_{m+1})} \left[\sum_{i=0}^{m-1} C(x_{t_i:t_{i+1}-1}) + \beta m \right].$$

Within the sections on Binary Segmentation, the form of the cost function was motivated by comparing it to a hypothesis test. Throughout the remaining section, the motivating idea is to consider the cost function solely as a measure of model fit. Clearly, these are not separate concepts and so these lead to variants of highly similar cost functions.

One point to draw attention to here is the specific linear form which the cost function must adhere too. Criteria for model selection typically consist of a term with some measure of the model fit and a regularisation term, due to uniform improvements in model fit with increasing complexity. Most measures of model fit are trivially adapted to satisfy this additive property by defining this as the cost of a segment with no changepoints. The regularisation term on the other hand has potential for issues. Well-known choices such as the $2k$ term in Aikake's Information Criteria or $k\log(n)$ in Schwarz's Information Criteria satisfy this. However, there have been recently proposed schemes within the changepoint literature involving non-linear terms which would not. This is a point made in passing here, with specifics to be expanded on in the context of PELT in the upcoming section.

3.4.3 Algorithm Outline

In this case, the cost of segmenting the dataset $x_{1:n}$ with changepoints at $(t_0, ..., t_{m+1})$ can be viewed as the cost of segmenting the dataset $x_{1:t_m-1}$ with changepoints at $(t_0, ..., t_m)$ summed with a term representing the cost of the additional data $x_{t_m:n}$.

This can be seen by

$$\underbrace{\sum_{i=0}^{m} C(x_{t_i:t_{i+1}-1}) + \beta m}_{\text{segmented cost of } x_{1:n}} = \underbrace{\sum_{i=0}^{m-1} C(x_{t_i:t_{i+1}-1}) + \beta(m-1) + C(x_{t_m:n}) + \beta}_{\text{segmented cost of } x_{1:t_{m-1}}}.$$

As a consequence

$$\min_{(t_0,...,t_{m+1})} \left[\sum_{i=0}^{m-1} C(x_{t_i:t_{i+1}-1}) + \beta m \right]$$

is identical to

$$\min_{(t_0,...,t_{m+1})} \left[\sum_{i=0}^{m-1} C(x_{t_i:t_{i+1}-1}) + \beta(m-1) + C(x_{t_m:n}) + \beta \right]$$

which is

$$\min_{t_m} \left[\min_{(t_0,...,t_m)} \left[\sum_{i=0}^{m-1} C(x_{t_i:t_{i+1}-1}) + \beta(m-1) \right] + C(x_{t_m:n}) + \beta \right].$$

As a result, $(t_0, ..., t_{m+1})$ being the minimal cost segmentation of the data $x_{1:n}$ implies that $(t_0, ..., t_m)$ is the minimal cost segmentation of $x_{1:t_m-1}$. More generally, the idea is that one can run the algorithm over any sub-segment contained within the solution one gets from running the algorithm over the entire dataset and in doing so no new changepoints should be found.

The utility of this idea then comes from the fact that it can be exploited for efficient computation. As a direct consequence of the additivity, the addition of a new point only changes the cost of the segment which it is contained in. It is therefore required that the position of the last changepoint be chosen again. Note, however, that it does not change the cost of any segment prior to the last. The optimum of these, therefore, must be a previously computed optimum to avoid a contradiction.

By denoting the magnitude of the minimal cost segmentation of $x_{1:k}$ by $F(k)$, it follows directly that

$$F(n) = F(t_m - 1) + C(x_{t_m:n}) + \beta$$

for some $t_m \in \{1, ..., n\}$. Suppose, for a moment, then that one had knowledge of $F(0), ..., F(n-1)$. It would then be an evaluation of

$$\min_{j \in \{0,...,n-1\}} F(j) + C(x_{j:n}) + \beta$$

to deduce $F(n)$.

However, the segmentation with no changepoints should not incur a penalty and so one has $F(0) = -\beta$. One can therefore recursively deduce each of the remaining minimal costs, $F(1), ..., F(n-1)$, to arrive at $F(n)$. Pseudocode outlining such an approach follows.

Algorithm 3 Optimal Partitioning

Input:
Dataset $(x_1, ..., x_n) \in \mathbb{R}^n$
Additive cost function $C(.)$
Penalty constant $\beta \in \mathbb{R}^+$

Initialise:
$n \leftarrow$ length(dataset)
Changepoints $\leftarrow$ empty container of size n storing vectors
$F \leftarrow$ empty vector of size $n + 1$
$F[0] \leftarrow -\beta$
Cp $\leftarrow 0$
Current_cost $\leftarrow 0$
Min_cost $\leftarrow 0$

Iterate for $i \in \{1, ..., n\}$:
    Iterate for $j \in \{0, ..., i\}$:
        Current_cost $\leftarrow F[j] + C(x_{j:i}) + \beta$
        If j == 1:
            Min_cost $\leftarrow$ Current_cost
            Cp $\leftarrow j$
        Else:
            If Current_cost $\leq$ Min_cost:
                Current_cost $\leftarrow$ Min_cost
                Cp $\leftarrow j$
    $F[i] \leftarrow$ Min_cost
    Changepoints[i] $\leftarrow$ (Changepoints[Cp], Cp)

Return Changepoints[n]

3.4.4 Time Complexity

The time complexity is understood by noting that at each time i, one needs to evaluate the cost of having a final changepoint at each of the i points before it. Given that no cost function has currently be set, this shall be disregarded temporarily by assuming that this is $\mathcal{O}(1)$. There is therefore

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2} = \mathcal{O}(n^2)$$

possible changepoint structures considered. Note, in contrast to those approaches seen so far, the approach of the algorithm has no dependence on the data. There is therefore no variation and this is the base time complexity across all cases.

3.5 Pruned Exact Linear Time

3.5.1 Overview

Optimal Partitioning is a fundamental to understand as the foundation of an alternative approach that again offers substantial improvements in efficiency beyond that of the naïve search. It, however, is not a complete changepoint detection algorithm. It does not offer any guidance as to how one should choose their cost function or penalty value. Moreover, the current formulation offers no benefits to efficiency beyond methods already introduced given that its base time complexity is the same as Binary Segmentation in its worst case. $\mathcal{O}(n^2)$ is still somewhat cumbersome for large datasets, given that this does not account for evaluation of

the cost function.

The primary determinant of this time complexity is that upon each time-step, the entire dataset up to this time-step is considered. In turn, this is because, while it recognises that the optimal segmentation of the whole series must contain an optimal sub-segmentation, the possibility is open that it could be any sub-segmentation. It, therefore, considers the cost of segmenting $x_{1:n+1}$ in ways which are drastically different to what we have concluded is optimal for $x_{1:n}$.

It would be intuitive to feel that the optimal changepoint structure of a dataset with an additional datapoint should have some resemblance to that of the prior dataset, and particularly so as the dataset begins to grow large. After all, adding an additional datapoint does not remove pre-existing changepoints from the structure of the dataset. Optimal Partitioning demonstrated how one can materially reduce time complexity by removing points from the solution space as soon as possible. Were one to know definitively of the existence of a changepoint at some position within their dataset, one could extend the approach to yield further reductions.

The issue is that in practice one cannot know with certainty whether a given changepoint has been correctly identified. However, with an appropriate cost function, there should be a meaningful cost advantage to introducing a correct changepoint which would suggest evidence of having done so. The magnitude of such a cost advantage will be determined by the magnitude of the change. Such a change could be arbitrarily large, with every iteration of Optimal Partitioning also keeping that change in place. Yet, by design, Optimal Partitioning continues to consider the possibility that such a change does not exist. PELT [33], the subject of this section, on the other hand may not.

PELT quantifies these notions by introducing a criterion to remove a changepoint whenever a sufficiently large cost advantage has been realised. Specifically, the criteria used is a decrease large enough to offset the penalty factor. Recall, in Binary Segmentation approaches one introduces a change under the same condition. As such, despite the different foundation it is nonetheless a similar greedy heuristic with its own associated set of advantages and disadvantages to be explored.

3.5.2 Algorithm Outline

As PELT is an extension of Optimal Partitioning, the two setups are similar. Once again one requires an additive cost function and identifies changepoints through the objective of finding

$$\min_{(t_0,\ldots,t_{m+1})} \left[\sum_{i=0}^{m-1} C(x_{t_i:t_{i+1}-1}) + \beta m \right].$$

However, where Optimal Partitioning would set

$$F(i) = \min_{j \in \{0,\ldots,i-1\}} F(j) + C(x_{j:i}) + \beta$$

at every stage, PELT performs a constrained minimisation over the set R_i by setting

$$F(i) = \min_{j \in R_i} F(j) + C(x_{j:i}) + \beta.$$

R_i is initialised to consider the entire dataset, however upon over every time-step checks the condition

$$F(j) + C(x_{j:i}) + \beta \geq F(i)$$

and removes all points $j < i$ from R_i which satisfy this. Pseudocode outlining this approach follows.

The above condition is known as a pruning step. It is a greedy criteria inspired by Dynamic Programming. The $F(j)+C(x_{j:i})+\beta$ term is the cost of a given solution up to the most recent time-step in the dataset. The

condition is therefore checking whether the current minimum has sufficiently decreased the cost to invoke a future change and still be cheaper than a previous solution.

In effect, the moment one gains a discernible improvement in cost locally, the decision is made that this is the solution to pursue regardless of what occurs over the next iteration. This does not get reconsidered at any point. Where Optimal Partitioning would build solutions from the whole dataset, PELT builds solutions from those which have remained locally attractive throughout the traversal of the dataset. The benefit is potentially drastic improvements in time complexity.

Algorithm 4 PELT

Input:
Dataset $(x_1, ..., x_n) \in \mathbb{R}^n$
Cost function $C(.)$
Penalty constant $\beta \in \mathbb{R}^+$

Initialise:
$n \leftarrow$ length(dataset)
Changepoints $\leftarrow$ empty container of size n storing vectors
Final_changes $\leftarrow (0, 1)$
$F \leftarrow 0$ vector of size $n + 1$
$F[0] \leftarrow -\beta$
Cp $\leftarrow 0$
Current_cost $\leftarrow 0$
Min_cost $\leftarrow 0$

Iterate for $i = 1, ..., n$:
    Iterate for each $j \in$ Final_changes:
        Current_cost $\leftarrow F[j] + C(x_{j:i}) + \beta$
        If j == 1:
            Min_cost Current_cost $\leftarrow$ Min_cost
            Cp $\leftarrow j$
        Else:
            If Current_cost $\leq$ Min_cost:
                Current_cost $\leftarrow$ Min_cost
                Cp $\leftarrow j$
    Iterate for each $j \in$ Final_changes:
        If $F[j] + C(x_{j:i}) + \beta \leq$ Min_cost:
                Final_changes.remove(j)
    $F[i] \leftarrow$ Min_cost
    Changepoints[i] $\leftarrow$ (Changepoints[Cp], Cp)
    Final_changes $\leftarrow$ (Final_changes, $i + 1$)

Return Changepoints[n]

Consider the extreme case, for example, where every point is pruned and $|R_i| = 1$ for all i. This will involve an $\mathcal{O}(1)$ evaluation over n time steps leading to $\mathcal{O}(n)$ time complexity, a particularly notable result given that any algorithm which considers the possibility of a change occuring at all points within the dataset once would be of the same order. Contrastingly, however, consider the other extreme where no point is pruned. The algorithm is then Optimal Partitioning with unproductive additional calculations. In this case, it has, again, $\mathcal{O}(n^2)$ time complexity.

Understanding the time complexity generally is made difficult by the dependence on the data. The key is that, as per Optimal Partitioning, the number of comparisons involved in the minimisation over $|R_i|$ is the

primary determinant of the algorithms overall time complexity. In practice, this refers to the segment length between changepoints. An informative theoretical result follows. This is conditional on a given cost function and so is deferred until after the subsequent section.

3.5.3 Cost Function

While the algorithm as described can be implemented with any cost function, it is taken to be the twice the negative maximum likelihood under normality assumptions. As such, one has the option to search for changes in mean via

$$C(x_{t_i:t_{i+1}-1}) = \frac{1}{\hat{\sigma}^2} \sum_{j=t_i}^{t_{i+1}-1} x_j^2 - \frac{\left(\sum_{j=t_i}^{t_{i+1}-1} x_j\right)^2}{t_{i+1} - t_i},$$

where $\hat{\sigma}$ is estimated by the total dataset, changes in variance by

$$C(x_{t_i:t_{i+1}-1}) = (t_{i+1} - t_i) \left[\log \left(\frac{1}{t_{i+1} - t_i} \sum_{j=t_i}^{t_{i+1}-1} (x_j - \hat{\mu})^2 \right) + 1 \right],$$

where $\hat{\mu}$ is estimated by the total dataset, or mean and variance changes with

$$C(x_{t_i:t_{i+1}-1}) = (t_{i+1} - t_i) \left[\log \left(\frac{1}{t_{i+1} - t_i} \sum_{j=t_i}^{t_{i+1}-1} (x_j - \frac{\left(\sum_{j=t_i}^{t_{i+1}-1} x_j\right)^2}{t_{i+1} - t_i}) \right) + 1 \right].$$

PELT specifically recommends use of the SIC by default which gives rise to an objective of the form

$$\min_{(t_0,\ldots,t_{m+1})} \left[\sum_{i=0}^{m-1} C(x_{t_i:t_{i+1}-1}) + pm \log(n) \right]$$

with $C(.)$ taking one of the three forms described and p taking the value 1 or 2 depending on whether one is looking for changes in mean, variance, or both.

3.5.4 Time Complexity

With the cost function appropriately defined, attention is turned back toward the algorithms time complexity. The idea that PELT is linear time in the extreme where every point is pruned has been introduced. This, however, can be relaxed without negating the validity of this result. Specifically, this revolves around bounding $E(|R_i|)$ by a constant asymptotically. In this case, the minimisation performed over every step is $\mathcal{O}(1)$ and hence the algorithm $\mathcal{O}(n)$.

Visualising Dynamic Programming Approaches

Natural Solution Space

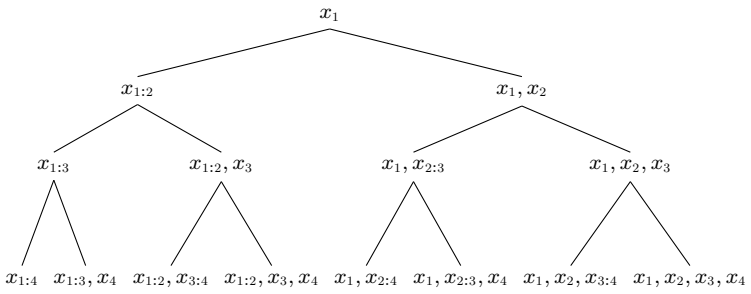


Figure 3.3: An illustration of the sub-problems were one to consider every possible changepoint structure.

Optimal Partitioning

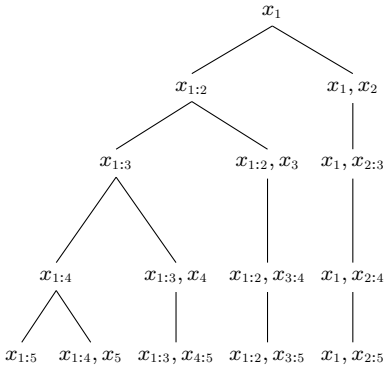


Figure 3.4: An illustration of the sub-problems considered by Optimal Partitioning.

PELT

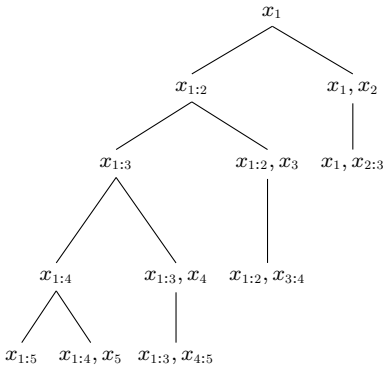


Figure 3.5: An illustration of the sub-problems considered by PELT.

Visualising the solution pursued by a Dynamic Programming approach is more difficult than that of Binary Segmentation given that nothing is fixed at each stage aside from those branches removed by PELT. The figures above correspond to a decision tree which shows the changepoint structures considered at each time-step. The right subtree corresponds to structures which add a changepoint at the latest time step to previous solutions and the left subtree those which do not. Optimal Partitioning keeps the solution space manageable by removing sub-optimal choices as they arise, ensuring that it grows linearly rather than multiplicatively. In addition to managing the solution space by the same means as Optimal Partitioning, PELT makes use of a greedy criteria to pre-emptively remove further branches.

Theorem 1: PELT - Linear Time Complexity

Consider $X_1, ..., X_n$ to be data generated with changepoint i at position

$$\sum_{j=1}^{i} S_j,$$

where the sequence S_j are I.I.D. random variables with common distribution S. Within segments, assume that the data is I.I.D. with common density function $f(x|\theta)$. Namely, one has

$$X_1, ..., X_{S_1-1} \sim f(x|\theta_1),$$

$$X_{S_1}, ..., X_{S_2-1} \sim f(x|\theta_2),$$

and so on. Assume further that the true parameters of each segment, $\theta_1, \theta_2, \theta_3$, are I.I.D. random variables drawn from density function $\pi(\theta)$. Let $\hat{\theta}_n$ be the maximum likelihood estimate for θ under the assumption of a single segment

$$\hat{\theta}_n = \arg\max_{\theta} \sum_{i=1}^{n} \log(f(x_i|\theta)).$$

Let θ^* be the value that maximises the expected log-likelihood,

$$\theta^* = \arg\max \int_{\mathbb{R}} \int_{\mathbb{R}} f(x|\theta) f(x|\theta_0) \pi(\theta_0) dx d\theta_0.$$

Under conditions

$$1.\ B_n = \sum_{i=1}^{n} \log(f(x_i|\hat{\theta}_n)) - \log(f(x_i|\theta^*)) \text{ has } \mathbb{E}(B_n) = o(n)$$

$$\text{and } \mathbb{E}([B_n - \mathbb{E}(B_n)]^4) = \mathcal{O}(n^2)$$

$$2.\ \mathbb{E}\left(\log(f(X_i|\theta_i)) - \log(f(X_i|\theta^*)\right)^4 < \infty$$

$$3.\ \mathbb{E}(S^4) < \infty$$

$$4.\ \mathbb{E}\left(\log(f(X_i|\theta_i)) - \log(f(X_i|\theta^*)\right) > \frac{\beta}{\mathbb{E}(S)}$$

PELT has $\mathcal{O}(n)$ time complexity.

The full details of the proof can be found in [33]. Rather than reproducing all steps, those which are most relevant to the design and implementation of change detection algorithms more generally will be discussed.

Broadly, the proof follows two parts. The first bounds the expected time complexity above by $nE(|R_n|)$. This is a consequence of noting that in general the last iteration of the algorithm involves the most expensive minimisation and that the compute time is bounded above by the algorithm which simply performs this n times.

In order then to deduce that the expected compute time of the bound is $\mathcal{O}(n)$, one needs to have $E(|R_n|)$ bounded by a constant as discussed. Now, one can write

$$|R_n| = \sum_{i=1}^{n} I_j$$

where I_j is an indicator for the pruning status of the point j. After invoking independence and that the expectation of an indicator is the probability of its event, one gets

$$E(|R_n|) = \sum_{i=1}^{n} Pr(I_j = 1).$$

Concluding that this is finite asymptotically then amounts to showing that this sum converges. Of the various approaches, the authors chosen means of doing so is by comparison with the known

$$\sum_{i=1}^{n} \frac{1}{n^2}$$

after showing that

$$Pr(I_j = 1) \sim \mathcal{O}(\frac{1}{n^2}).$$

Note again that this is the probability of the pruning criteria being met. The remainder of the proof therefore involves manipulations involving the likelihood function and the data, primarily making use of the assumptions and other results such as Markov's and Minkowski's to conclude the proof.

At this point it is worth reflecting on the assumptions. Note, the author equally could have weakened the assumptions of the theorem such that

$$Pr(I_j = 1) \sim \mathcal{O}(\frac{1}{n})$$

making

$$E(|R_n|) = \sum_{i=1}^{n} Pr(I_j = 1) \sim \log(n)$$

and resulting in overall $\mathcal{O}(n \log(n))$ time complexity. Or in other words, asymptotically equivalent to Binary Segmentation. Alternatively, one could alter the assumptions of Binary Segmentation such that there were only a fixed number of changes contained within the dataset asymptotically. This would ensure that the algorithm makes only finite bisections and also make this algorithm $\mathcal{O}(n)$.

The point of this is to highlight the need to be careful when attempting to translate implications regarding asymptotic behaviour back to finite time. Results are only as good as their conditions, the truth of which is impossible to know. Additionally, constants which are ignored asymptotically make a distinct difference in practice. Especially when one is attempting to study the differences between $\mathcal{O}(n)$ and $\mathcal{O}(n \log(n))$ given how slowly $\log(n)$ grows. As such, the $\mathcal{O}(n)$ result is of little utility in and of itself. The overall theorem, however, is informative. In particular, the study of the conditions under which the result holds, how those interact with $Pr(I_j = 1)$, and the broader performance of the algorithm is of interest.

Insofar as highlighting the implications more generally, there are several. The first being that for PELT's pruning criteria to yield meaningful improvements in efficiency, one needs a rather distinct changepoint structure. Given that one wants to prune only changepoints there needs to be a sufficient number within the dataset. Specifically, condition 3 implies that the number of changepoints needs to increase linearly with the length of the data. More than being present, it is also important changepoints are appropriately spaced.

Multiple changepoints in close proximity yield rapidly decreasing benefits to time complexity.

Moreover, beyond having an appropriate changepoint structure the speed of detection as a function of time is important. One needs more than points being pruned eventually. In particular, one needs more power than what a well-specified parametric likelihood is guaranteed to give. Hence conditions 1, 2, 4. This also is a meaningful point to note, given that model misspecification is a primary concern. This point regarding detection speed frames the basis of a major point of discussion in Chapter 6.

In closing, note that practically one will not be able to know if the conditions of the theorem are strictly true with any certainty. Strict truth, however, does not matter in practice. Under the correct specification of the likelihood function and a uniformly spaced changepoint structure with relative small width between changes, the computational performance of PELT is likely to be desirable. These conditions make for a heuristic approach of thinking about the problem which is more readily applied in practice.

3.5.5 Binary Segmentation vs. PELT

PELT is a valuable contribution to the changepoint literature as it is highly complementary to Binary Segmentation. From a time complexity perspective, the conditions for each algorithms best case behavior occurs under the conditions of the others worst case. A philosophy that has been articulated at several points within this project is that there is no best method, instead the goal is to develop a variety of tools which allows one to effectively solve any problem. Methods with opposite best and worst case behaviors make that possible.

The conditions required for PELT to be computationally more efficient than Binary Segmentation involves segment lengths remaining bounded by a constant, while the length of the dataset tends to infinity. When implementing these algorithms for practical purposes, this certainly should not be taken for granted. With modern computing, often the sampling frequency of the data occurs at a much greater rate than any process initiating change. Hence the number of changepoints in a dataset are often far outweighed by the overall length of the dataset. To this point, the authors of PELT explicitly say within the paper, "Whilst asymptotically PELT can be quicker, Binary Segmentation is quicker on the examples we consider and we believe this would be the case in almost all applications" [33].

The main argument the authors instead put forth in favour of using PELT is that it leads to a substantially more accurate segmentation than Binary Segmentation. The author supports this idea by a notion introduced as "exact searching" which has become prevalent within the change detection literature, being referred to in [45] and [46] for example. Specifically, this refers to the fact that if one imposes additional assumptions on the cost function and penalty, solutions from the Dynamic Programming approach outlined in Optimal Partitioning and that of the greedy approach outlined in PELT are always identical. In effect this is a situation where making decisions locally always results in the best outcome. More broadly throughout the optimisation literature, such an algorithm is referred to as a Greedoid. Across more general contexts, understanding where greedy algorithms result in Greedoids is a point of significance. To that ends, a full categorisation of the conditions under which such results hold in its most general are contained in [47].

In this case, the assumption required is sub-additivity of the cost function and penalty. If one considers a more general cost function and penalty than the one discussed thus far, instead defining one's total cost as

$$\sum_{i=0}^{m-1} C(x_{t_i:t_{i+1}-1}) + \beta f(i).$$

The condition required is sub-additivity of the term $C(x_{t_i:t_{i+1}-1}) + \beta f(i)$
such that

$$C(x_{t_i:s}) + C(x_{s:t_{i+1}-1}) + \beta f(i+1) \leq C(x_{t_i:t_{i+1}-1}) + \beta f(i).$$

If both the functions $C(.)$ and $f(.)$ are sub-additive, the overall sub-additivity trivially follows. Give that many measures of model fit are a sum of error terms and monotonically increase with the data, sub-additivity

of this is often true. The normal likelihood, per the author's recommendation, for instance satisfies this property as a result of the concavity of the logarithm function. Sub-additivity of the penalty, however, need not be true.

Again, sub-additivty could be replaced by the more familiar notion of concavity as any function concave function $f : \mathbb{R} \to \mathbb{R}$ with $f(0) \geq 0$ has

$$f(x + y) \leq f(x) + f(y),$$

or in other words that it is sub-additive.

The recommendation by the author of using SIC, corresponding to $f(m) = m$ trivially satisfies this as it is linear. However, there have been numerous other regularisation schemes in the literature. Typically these penalise more heavily as the number of changepoints increases. The logic being to avoiding circumstances where one overshoots and returns much too few or many changepoints. It is precisely this behaviour which could cause one to possibly regret making decisions locally.

Within Binary Segmentation one can easily construct functions to achieve this. For example, consider a penalty term of $\beta m \log(m)$ or βm^2 rather than βm. Both of these are convex rather than concave and so would not satisfy the property. Under these circumstances PELT would simply be a greedy algorithm rather than a Greedoid. While these serve as simple motivating examples, it would make little sense to use such a penalty with PELT given that it would bias changepoints toward the beginning of the dataset. Instead, for example, one would adjust the SIC to include a term which depends on the segment length [48]. A similar notion would hold in this case.

In returning to the author's recommendation of using the normal likelihood and SIC, the "exact" property as described does hold. Nonetheless, no supporting argument is ever given as to why this necessarily improves classification effectiveness. Recall, that this is the ultimate objective rather than to find the optimum model as described by an information criteria. This author would therefore argue that the distinction between greedy and Greedoid is far more relevant in the case of deterministic rather than stochastic optimisation. In the deterministic case, one is directly solving the problem of interest and the solution returned can be described as optimal. A canonical example of this is Kruskal's algorithm for finding a minimal spanning tree of a graph [49]. In a stochastic case such as this, however, it is a self defined notion of optimal relying on a degree of subjective judgement and foresight which one is by no means guaranteed to have.

Recall, the value of an information criteria means little in and of itself. These simply provide a means by which one can decide if a model is favourable relative to another. These do not, however, say anything about the quality of any of the models. In fact, every such model considered within the algorithm could vastly overfit or underfit the data. Furthermore, it raises the question of which information criteria one should choose. The optimal model under the AIC can be different to that under the BIC or other information criteria too numerous to mention.

In summary, the solution path between PELT and Binary Segmentation will certainly be different. It is not clear whether either of the algorithms structurally has an advantage over the other from a classification perspective. A central question of interest throughout Section 3.1.1 therefore will be to investigate this.

Chapter 4

A Review of Non-Parametric Approaches

The purpose of this chapter is to provide a brief account of a class of non-parametric approaches. Specifically, the methods to be discussed are Non-Parametric Multiple Changepoint Detection (NMCD) and Empirically Distributed Pruned Exact Linear time (ED-PELT) which are introduced in the papers [50] and [51] respectively. Note that these are predecessors of the approach which is the subject of Chapter 6. As such, this introduction is intentionally light on details. The objective is to define the statistical methodology underpinning these approaches. Aspects of algorithm design are compared thoroughly in Chapter 6.

4.1 Non-Parametric Multiple Change Detection

4.1.1 Overview

Within the broader hypothesis testing literature, there is a variety of non-parametric goodness of fit tests which have been proposed. Many of the best known non-parametric statistics arise from applying a function norm to empirical distribution functions estimated from different samples of data. For instance, the L_∞ or supremum norm gives rise to the two-sample Kolmogorov-Smirnov statistic [52] of

$$\sup_x |\hat{F}_1(x) - \hat{F}_2(x)|.$$

Other well-known examples include the two-sample Cramér-von-Mises [53] which arises from the L_2 norm,

$$n \int_{\mathbb{R}} \left(\hat{F}_1(x) - \hat{F}_2(x)\right)^2 d\hat{F}(x)$$

or a variant of this proposed by Anderson-Darling [54]

$$n \int_{\mathbb{R}} \left(\hat{F}_1(x) - \hat{F}_2(x)\right)^2 \frac{d\hat{F}(x)}{\hat{F}(x)(1-\hat{F}(x))},$$

where $\hat{F}$ represents the population empirical distribution function. Several of these statistics were proposed in roughly the same period as Fisher's seminal works on the likelihood. Note, the likelihood is most commonly associated with parametric model assumptions. Perhaps lesser-known, however, are the more recent extensions of the approach to the non-parametric setting and it is works of this nature which are of interest in this chapter. The papers [55] [56] introducing non-parametric goodness of fit tests based on the likelihood are particularly relevant.

4.1.2 Cost Function

Let F_i be the cumulative distribution function of segment i, with data $x_{t_i:t_{i+1}-1}$. The associated empirical cumulative distribution function shall be denoted by $\hat{F}_i(s)$, where

$$\hat{F}_i(s) = \frac{1}{t_{i+1} - t_i} \sum_{j=t_i}^{t_{i+1}-1} 1_{X_j \leq s}.$$

The key observation is that for an I.I.D. sample $X_1, X_2, ...X_n$, one has that $\hat{F}_i(s) \sim \text{Binom}(n_i, F(s))$. It is an elementary calculus exercise to then demonstrate that the corresponding likelihood function is maximised at $\hat{F}_i(s)$ and therefore a measure of segment fit is

$$\mathcal{L}(x_{t_i:t_{i+1}-1} \mid s) = (t_{i+1} - t_i)[\hat{F}_i(s) \log \hat{F}_i(s) + (1 - \hat{F}_i(s)) \log(1 - \hat{F}_i(s))].$$

One issue with defining cost based on the likelihood in this current formulation is that it depends on a single choice of s. To negate likelihood functions are combined by considering

$$\int_{\mathbb{R}} \mathcal{L}(x_{t_i:t_{i+1}-1} \mid s) dw(s),$$

for a particular weight function $dw(s)$. The issue now becomes choosing the weight to produce a powerful goodness of fit test. The authors suggest a parallel to the Anderson-Darling statistic by using

$$dw(s) = \frac{d\hat{F}(s)}{\hat{F}(s)(1-\hat{F}(s))}$$

where $\hat{F}(s)$ is the empirical cumulative density function for the unsegmented dataset $x_{1:n}$. The total cost for the segment $x_{t_i:t_{i+1}-1}$ is therefore

$$C(x_{t_i:t_{i+1}-1}) = n \sum_{s=1}^{n-1} \frac{\mathcal{L}(x_{t_i:t_{i+1}-1} \mid x_{(s)})}{s(n-s)}$$

where $x_{(s)}$ is order statistic s of the data $x_{1:n}$. Finally, the total cost of the changepoint structure $(t_0, t_1, ..., t_{m+1})$ is defined as

$$\left[\sum_{i=0}^{m} C(x_{t_i:t_{i+1}-1}) + \beta m \right]$$

with the recommendation of setting $\beta = \frac{1}{2}(\log(n))^2$. This is a strengthened form of the BIC, similar to that considered in Wild Binary Segmentation, which again arises from consistency results. Details of these results can be found in [50].

4.1.3 Review

The approach of NMCD is intended to solve a different but related problem to that which is of interest in this project. Namely, a case where the number of changepoints is known and one is seeking only to identify the location of change. As such, the searching method used is different than those which have been introduced thus far. It is a Dynamic Programming approach known as the Segmented Neighbourhood, the details of which can be found in [44].

Briefly, the algorithm requires one to specify a fixed number of changepoints for the method to return. One could apply such an approach to this problem by varying this parameter, but note that it has $\mathcal{O}(Mn^2)$ time complexity where M is the number of changepoints found. For the problem of interest, more efficient approaches exist. This motivates the development of the next method, ED-PELT, which looks to extend this approach to the unknown changepoint case.

4.2 Empirically Distributed Pruned Exact Linear Time

4.2.1 Overview

The primary motivation for ED-PELT is to reduce the time complexity of NMCD and several adjustments are made towards that ends. As the name suggests, the first is to adjust the searching algorithm from that of the Segmented Neighbourhood to PELT. The other is by adjusting the cost function. In particular, the issue of focus is that of the weight function

$$dw(s) = \frac{d\hat{F}(s)}{\hat{F}(s)(1-\hat{F}(s))}.$$

From the perspective of time complexity, this is problematic as computing the cost of a single segment then involves evaluating the likelihood function over all n terms observed within the dataset. Note that of all the approaches considered so far, segment costs have only ever included a summation of n_i terms.

4.2.2 Cost Function

Defining the cost function again begins by considering the likelihood of

$$\mathcal{L}(x_{t_i:t_{i+1}-1} \mid s) = (t_{i+1} - t_i)[\hat{F}_i(s) \log \hat{F}_i(s) + (1 - \hat{F}_i(s)) \log(1 - \hat{F}_i(s))].$$

First, to remove the need to evaluate the population distribution function a weight of $dw(s) = d\hat{F}(s)$ is considered. Next, in order to reduce the number of terms within the sum the author's use what they describe as a discrete approximation by evaluating the likelihood at particular quantiles of the population empirical distribution function. Specifically, the points considered are $x_1, x_2, ..., x_K$ where

$$x_i = \inf\{x : \hat{F}_0(x_i) > t_i\},$$

the corresponding t_i is given by

$$t_i = \left((2n-1) \exp\left(-\frac{\log(2n-1)(2i-1)}{K}\right)\right)^{-1},$$

and K is a hyperparameter that comes with a recommendation of $4 \log(n)$.

Recall, the weight used by NMCD was chosen to emphasise the tails of the distribution. At first, this feature was removed. However, the authors reiterate a perceived benefit from emphasising the tails. The purpose of choosing these quantiles is to again include a feature which does so. The final segment cost in this case is defined as

$$C(x_{t_i:t_{i+1}-1}) = \frac{2 \log(2n-1)}{K} \sum_{i=1}^{K} \mathcal{L}(x_{t_i:t_{i+1}-1} \mid x_i)$$

and the objective minimised by PELT is

$$\left[\sum_{i=0}^{m} C(x_{t_i:t_{i+1}-1}) + \beta m\right]$$

where it is again recommended to consider $\beta = \frac{1}{2}(\log(n))^2$.

4.2.3 Time Complexity

Regarding time complexity, within the paper it is briefly stated that in some conditions PELT is $\mathcal{O}(n)$ and secondly that evaluating the cost function can be performed in $\mathcal{O}(\log(n))$. However, within Chapter 5 it will become clear that the algorithm does not behave like an $\mathcal{O}(n \log(n))$ algorithm. The point regarding PELT's $\mathcal{O}(n)$ time is scrutinised in Section 6.1. For now, the cost function is of interest.

Note that no details are provided on the implementation of this algorithm within the paper nor the documentation of the CRAN package [57]. This and the practical performance are notable reasons for scepticism. Inspection of the source code suggests that it is implemented with a bin-counting approach and so it could be true. It is apparent, however, that there is minimal storage of computations and that it can be tangibly improved. This is the basis of Section 6.4.

Moreover, another distinct point to note is regarding the hyperparameter. A similar approach to adjusting computational time was successfully performed by WBS. This, however, is fundamentally different. With WBS, the minimalist choice reduces to that of SBS. Choosing the hyperparameter is relatively easier as one therefore has a degree of confidence in the results irrespective of the choice. Contrastingly, this adjusts an unproven cost function. Its only supporting results are relatively limited simulations provided by the author. There is a distinct risk therefore that one reduces compute time at the expense of classification performance in an undesirable way. Negating this would require tuning the parameter with a two-dimensional grid search which has a material computational cost. Were one to do so, the theoretical time complexity may be misleading.

Chapter 5

Empirical Studies

The purpose of this chapter is to illustrate how the theoretical properties of the algorithms discussed within the preceding chapters translate practically into performance. To that end, pairwise comparisons of the different approaches are given in chosen scenarios as an illustration of more general behaviour. This is accompanied by discussion of the inherent difficulty to assessing performance in a representative way and a review of the results presented in the literature thus far. Insights drawn from this chapter motivate design choices made within Chapter 6. A broader set of simulation studies benchmarking the performance of all algorithms discussed within this project are presented in Chapter 7.

5.1 Performance Evaluation

5.1.1 Overview

The intention of the forthcoming sections will be to present a retrospective analysis of performance to inform future decisions regarding which algorithm is most appropriate. There are two inherent issues to this. The most apparent is that finding insights which are generalisable and exploitable in practice is simply a difficult task. The second issue, the focus of this section, was briefly raised in Section 2.3.4. Namely, the utility of a given algorithm is largely context dependent. It will therefore be necessary to make assumptions regarding performance priorities.

The primary assumption will be that classification effectiveness is of the greatest concern. Practically, this translates to the foremost concerns upon receiving estimates of changepoint locations being regarding the number of locations which have been falsely declared as a point of change and conversely how many points of change have in fact been declared as a period of stationarity. It is assumed then that runtime is a secondary consideration. To provide a crude but tangible example, consider, for instance, an objective function in which one gains x_1 units for correctly identifying a changepoint, loses x_2 units for declaring a change exists where there does not, and the overall objective is to maximise utility within an allotted time t.

The analogy given motivates the evaluation metrics of choice. The best approach to method selection for this problem would first consist of visualising a curve of classification scores for each algorithm under variations of parameters on a simulated dataset closely replicating that which is of interest. Method selection would then consist of choosing the point on one of these curves which meets the constraints and gives the best score. A variation of this will be of primary focus. For the purpose of this thesis, however, instead of making any conclusions regarding which is best for a particular case, instead, the objective is to understand the conditions under which one method may outperform another. The scope of possible outcomes is therefore of greater interest. To that point, in order to study a broader range of settings, in some cases these curves will be reduced into a single metric and compared. With this background, specific definitions are to follow.

5.1.2 Evaluation Metrics

To give a mathematical definition of any metric, it first needs to be understood what is meant by a correctly identified changepoint. Commonly a tolerance threshold of ϵ is set and given true changepoints $\mathcal{T}_0 = \{t_0, ..., t_{m+1}\}$, a single estimate $\hat{t}_i$ is considered correct if there is a $t_j \in \mathcal{T}_0$ such that $|\hat{t}_i - t_j| < \epsilon$. To ensure that only one estimate can be considered correct for any true changepoint, after having classified the estimates $\hat{\mathcal{T}}_i = \{\hat{t}_0, \hat{t}_1, ...\hat{t}_i\}$ associated with the set of true changepoints $\mathcal{T}_i^c \subset \mathcal{T}_0$, the estimate $\hat{t}_{j+1}$ can only be considered correct if there is a $t_{i+1} \in \mathcal{T}_0 \setminus \mathcal{T}_i^c$ with $|\hat{t}_{i+1} - t_{j+1}| < \epsilon$.

Under this notation, the metrics described previously are known as Precision and Recall [26], which are defined by

$$P = \frac{|\mathcal{T}_i^c|}{|\hat{\mathcal{T}}_i|} \quad \text{and} \quad R = \frac{|\mathcal{T}_i^c|}{|\mathcal{T}_0|}$$

respectively. Another performance metric which shall be referred to and feature explicitly within Appendix A.1 is the F_1 score [26],

$$F_1 = \frac{2PR}{P+R},$$

a harmonic mean of Precision and Recall. The final quantity of interest shall be CPU time, the total time taken to run an algorithm on a given dataset. It is these which will perform the basis of performance assessment throughout the remainder of this thesis.

5.1.3 Parameter Selection

Prior to introducing the experimental setup, this subsection is devoted to motivating the choices made. One of the core objectives of this thesis is to present a consistent narrative on algorithm tradeoffs. This is considered of importance due to the lack of any such narrative within the literature to date. Whenever a novel algorithm is introduced typically this will be accompanied by results demonstrating all of its most positive features. While this is to be expected, it would be forgivable for a practitioner for whom change detection is a means to an end to read a limited number of papers and come to a misguided decision.

The question will always be as to whether the insights gained from a single experiment are generalisable and the true test of this is whether it is readily reproducible. Hence there is a notable need for independent review. Those presented so far, however, are somewhat inadequate. Of the few available, the results in [58] are judged by the MSE with much focus on the case of a known number of changepoints, whereas in [59] the true changepoints are marked by human judgement. The following results intend to serve this purpose.

The specific simulations presented are those which are considered to display characteristic features after having run wide-ranging simulations and distilled the results. The reader should be aware that however impartial one attempts to be, there is still the caveat that it is impossible to simulate every possible situation. A true characterisation of all possible behaviors is beyond the scope of one body of work and instead would come about incrementally. This therefore should be viewed as one step towards this ends.

Finally, it is to be noted what is meant by the subjective notion of "characteristic features". A fundamental set of results with significant consequences for this thesis are the "No Free Lunch in Search and Optimisation" theorems [60]. In summary, these suggest that no algorithm is inherently better than another unless it exploits a structural feature of the problem which the other does not. Were one to aggregate performance over all situations, differences would arise as a consequence of these structural features. The author's notion of "characteristic features" therefore relates to the theoretical properties discussed in Chapters 3 and 4. For instance, the first set of simulations quantifies the power benefit realised from the improved convergence rate in the Wild versus classical Binary Segmentation formulations. Note, the "No Free Lunch" theorems say nothing of the variance underlying the broad performance averages, and so it is not necessarily the case that there are no parameter choices for which the results are unexpected. However, to the best of the author's knowledge, this is the most principled way to differentiate signal from noise and construct a set of guiding principles regarding method selection which may be readily identified in practice and are consistently exploitable.

5.2 Simulation Results

5.2.1 Experimental Setup

The experiments are to be conducted on synthetic data in a supervised setting, where the locations of changepoints are known. In generating the data, first the length of the dataset is fixed. In all cases this is $n = 10,000$. The locations of changepoints t_i are decided by sampling from $S_i \sim Po(\lambda)$ and setting

$t_i = \sum_{j=1}^{i} S_i$ until $t_{i+1} > n$ where t_{i+1} is discarded. The parameter λ is to be varied and will be either of 50 or 200 as directed. The data X_t is set as $X_t = \mu_t + \epsilon_t$, where ϵ_t is a stochastic quantity drawn from either of $N(0,1)$, t_4, t_3 as will be directed. The quantity μ_t is deterministic and piecewise constant over the changepoints t_i. μ_0 is initialised as 0 and thereafter will be set by sampling D with equal probability from $\{-1, 1\}$ and computing $\mu_{t_i} = \mu_{t_i - 1} + 0.5D$. The context of interest is therefore mean shifts up or down by 0.5 under different noise processes and segment lengths. Finally, a threshold of $\epsilon = 5$ is considered throughout.

5.2.2 Binary Segmentation vs. Wild Binary Segmentation

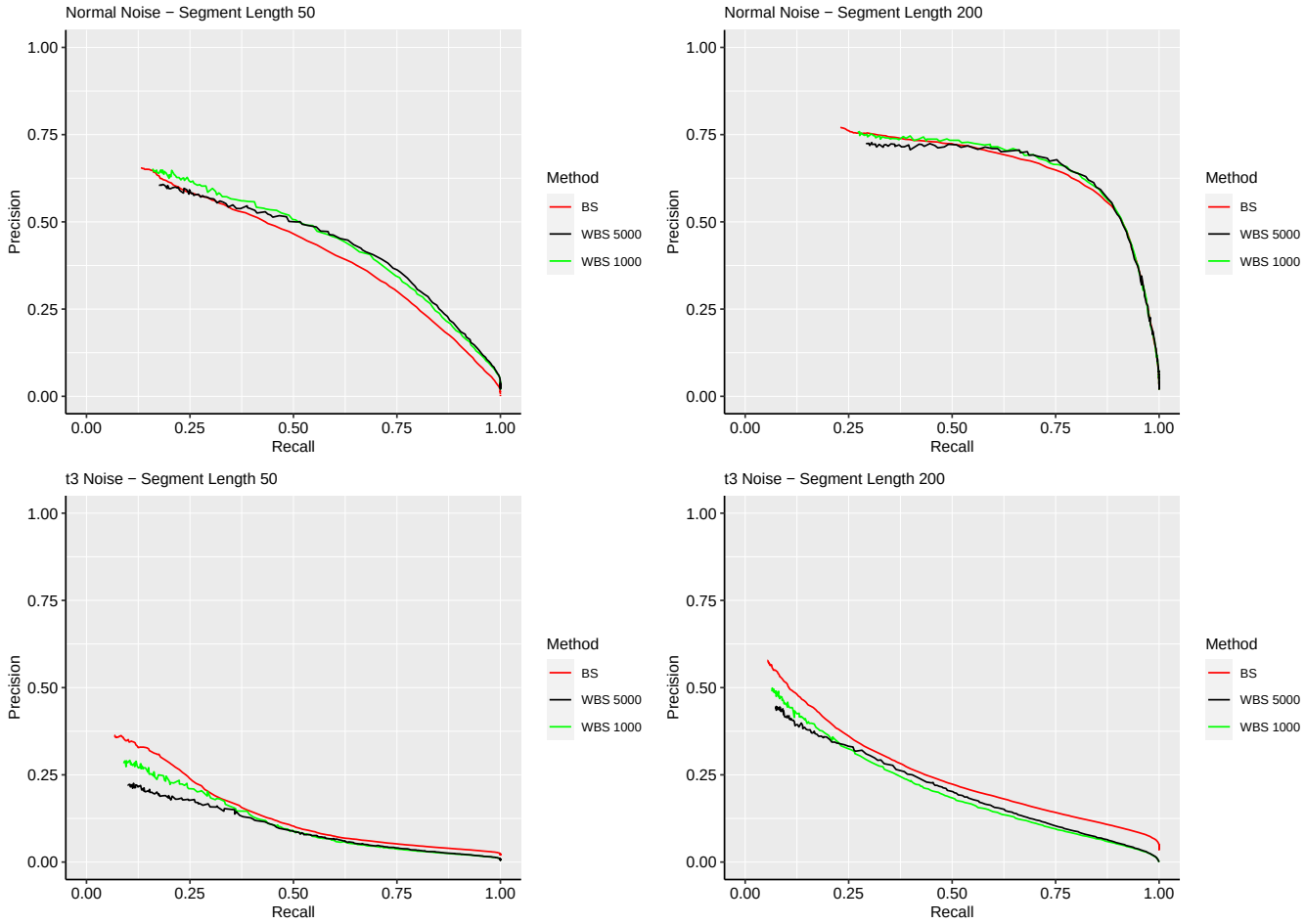


Figure 5.1: These figures were generated using the 'sbs' and 'wbs' methods of the WBS package. The values displayed are averages of Precision and Recall over 100 replications while varying the penalty parameters.

By design, the CUSUM has no discrimination power for non-mean changes and an example illustrating this for variance changes is in Appendix A.3. Differences in WBS versus SBS arise in power and robustness and this is therefore the focus. Prior to discussing specifics, note that across all figures the differences are modest. Largely this is due to Binary Segmentation, despite its simplicity, being consistent asymptotically and in this case it translates to effective performance subject to having sufficient data to perform on.

The effect of this caveat, however, becomes clear when comparing the normal subplots. In the case with expected segment lengths of 50, Wild Binary Segmentation outperforms across the full spectrum of penalties. This power advantage under correct specification is a notable merit of the WBS approach. Alternately, in cases with greater segment lengths differences in convergence rate are negated and these performance differences are negligible. Moreover, note that making more evaluations of the CUSUM does not solve all possible issues. The tradeoff becomes apparent in the t_3 case where the results are reversed. Where

the noise process is heavy, drawing random intervals to find a "favourable" one is outweighed by the risk of false positives. The core question in choosing between these two approaches therefore is whether the misspecification risk outweighs that of false negatives arising from short segments. Clearly, the answer is dataset dependent. Fortunately, these questions are approachable practically. The intensity of the noise process is visually distinguishable by the largest deviations, while the segment length can be thought about roughly by noting the spacings of solutions considered acceptable by diagnostics.

5.2.3 Binary Segmentation vs. PELT

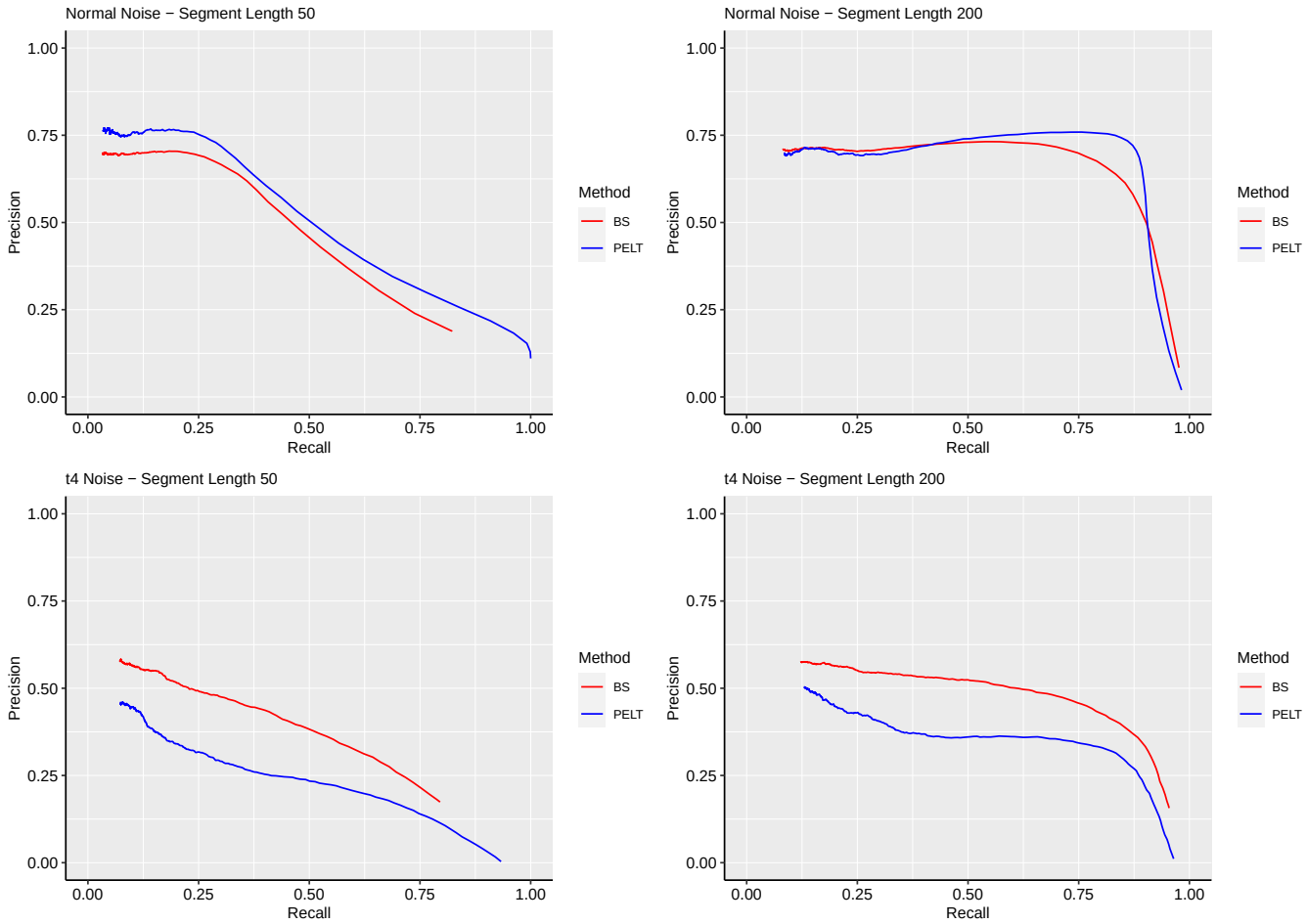


Figure 5.2: These figures were generated using the 'cpt.mean' function of the changepoint package with the method parameter set to "BinSeg" and "PELT" to enforce the same cost function. The values displayed are averages of Precision and Recall over 100 replications while varying the penalty parameters.

The differences between Binary Segmentation and PELT shown above are similar to that of SBS and WBS. Again, PELT offers the possibility of improved power at the expense of worse misspecification performance, but to a greater extent. PELT has a positive performance differential while correctly specified which persists even with significant segment lengths. Under weaker misspecification than WBS, via t_4 rather than t_3 noise, there is a greater performance advantage to Binary Segmentation. Structurally, PELT avoids the single change assumption and therefore weakens the sensitivity to segment lengths. In doing so, however, it increases the sensitivity to how well the cost function and penalisation scheme represents the data. To this point, within [33] the author presents examples where PELT achieves a superior score of model fit to that of Binary Segmentation and speaks only to its upsides. Yet, the balance between over and under fitting is of greatest concern. In cases of reasonable segment lengths where Binary Segmentation is known to fit the data well, there is an argument against incurring the heightened misspecification risk.

Chapter 6

A Novel Non-Parametric Approach

The changepoint detection literature is dominated by normality assumptions. The potential upside to any such approach is statistical power and computational efficiency. However, clearly not all stochastic processes can be modelled adequately in this manner. A theory underpinned purely by any such assumptions is therefore incomplete. While non-parametric approaches have been proposed within the literature to date, there are apparent areas for improvement. Perhaps the most notable is the excessive computational cost highlighted in the preceding chapter. The primary goal of this chapter is to address this. Building on the insights discussed within Chapter 5, an alternative treatment of the non-parametric likelihood is given. This will demonstrate the validity of such insights and in doing so the author also intends to illustrate broader best practices of algorithm design.

6.1 Principles of the Construction

To achieve the objectives set out, the cost function will be reformulated such that the optimisation can be performed through Binary Segmentation. To motivate the use of Binary Segmentation note two points. Within the ED-PELT algorithm, the motivation given for the use of PELT is that it is a searching method which is "exact" and "subject to some conditions" can be of time complexity $\mathcal{O}(n)$. The first point is that the author of ED-PELT is also suggesting an approximation to the cost function. By their own logic, the algorithm finds an "exact" approximation. Instead, this author would suggest that one disregard these notions of exactness. Rather, noting that within the preceding chapter, the performance of Binary Segmentation has been observed to be competitive among the state of the art in cases with long segment lengths. Given that sufficient data per segment is critical for the efficacy of a cost function based on the empirical cumulative distribution function, this is a natural approach to take.

The second point to note is that in the circumstances just described, Binary Segmentation is also the most computationally efficient approach available. It therefore aligns with the overall goals set. Note, the point regarding PELT's linear time complexity is irrelevant in this context. The author now refers the reader back to the discussion of the theorem presented in Section 3.3.2.4. The proof of this result involves a different cost function entirely. Moreover, the precise conditions under which it holds involve imposing additional assumptions on the data to improve the rate of pruning beyond what a correctly specified parametric likelihood is guaranteed to give. Within the literature, the condition requiring a changepoint structure that is uniformly spaced and scales in magnitude linearly with the dataset is often discussed. The others, however, are often disregarded as technical conditions. In this case it is important. A particularly low-order approximation to the likelihood based on the empirical cumulative distribution function will not prune sufficiently rapidly to be competitive with Binary Segmentation outside of the most extreme cases.

As a final note, it is worth highlighting that while switching the searching algorithm will be of incremental benefit to the time complexity of the algorithm, it will nonetheless still require the repeated computation of an expensive cost function. This is the most meaningful burden computationally, irrespective of searching method, given how often one must perform these operations. The highest priority therefore is to address this. Neither authors proposing the cost function to date acknowledge this issue, nor suggest how to resolve it. More generally, to the best of the author's knowledge, there has not been a similar approach outlined in the literature either. To this ends, first a modified definition of the cost function is given. Thereafter the remainder of this chapter is devoted to presenting such an approach.

6.2 Redefining Cost

In Chapter 3 it has been noted that the likelihood based cost functions are readily translated between Dynamic Programming and Binary Segmentation based approaches by alternating between thresholding likelihood ratios and optimising model selection criteria. In this case, under the null hypothesis one has that $n_i\hat{F}_i(s) \sim \text{Binom}(n_i, F_i(s))$. When considering the possibility of a changepoint at time j, of interest is

$$H_0 : F_i^{L_j}(s) = F_i^{R_j}(s)$$

versus

$$H_1 : F_i^{L_j}(s) \neq F_i^{R_j}(s).$$

The likelihood ratio takes the form

$$\mathcal{L}(x_{t_i:j-1} \mid s) + \mathcal{L}(x_{j:t_{i+1}-1} \mid s) - \mathcal{L}(x_{t_i:t_{i+1}-1} \mid s)$$

where

$$\mathcal{L}(x_{t_i:t_{i+1}-1} \mid s) = (t_{i+1} - t_i)[\hat{F}_i(s) \log \hat{F}_i(s) + (1 - \hat{F}_i(s)) \log(1 - \hat{F}_i(s))].$$

At this point a similar sentiment is echoed to that of the discussion of the non-parametric methods in Chapter 4. One does not want to focus exclusively on a single empirical probability estimate. The cost function is to be constructed analogously to a metric. Hence, multiple tests will be combined by considering a final cost function of the form

$$\int_{\mathbb{R}} \mathcal{L}(x_{t_i:j-1} \mid s) + \mathcal{L}(x_{j:t_{i+1}-1} \mid s) - \mathcal{L}(x_{t_i:t_{i+1}-1} \mid s) dw(s)$$

for a particular $dw(s)$. The next decision to be made is regarding this weight function. There are two factors which influence this decision. The more apparent of the two is the factor one wishes to multiply each test by. The more subtle is selecting the points at which one wishes to evaluate the likelihood ratio. This is where a fundamentally different approach is proposed to that of the predecessors. Sections 6.2.1 and 6.2.2 are devoted to discussion of this. Of the two issues raised, the latter is addressed first followed by the former.

6.2.1 Review of the Weight Function

6.2.1.1 Global vs. Local

The first iteration of this cost function in the changepoint literature was in NMCD, where it was proposed to take a weight function of the form

$$dw(s) = \frac{d\hat{F}_0(s)}{\hat{F}_0(s)(1 - \hat{F}_0(s))}$$

where $\hat{F}_0(s)$ is the population empirical cumulative distribution function. The motivation for the denominator of this weight is to emphasise the tails of the distribution. This is to be discussed in the next section. The point noted here is that a downside to this approach is the use of $\hat{F}_0$. This involves repeatedly evaluating the likelihood function at every point observed within the dataset.

The idea that one could retain the power of this cost function while reducing time complexity by making fewer evaluations was introduced alongside ED-PELT. The issue then becomes deciding where it is most meaningful to evaluate the likelihood function. The proposal within ED-PELT is that one should evaluate the likelihood at points $x_1, x_2, ..., x_K$ where

$$x_i = \inf\{x : \hat{F}_0(x_i) > t_i\},$$

the corresponding t_i is given by

$$t_i = \left((2n-1) \exp\left(-\frac{\log(2n-1)(2i-1)}{K}\right)\right)^{-1},$$

and K is a hyperparameter which is recommended to be set at $4\log(n)$. The motivation for the choice of t_i is again to emphasise the tails of the distribution. Note that while the issue regarding excess evaluations of the likelihood is resolved, this author suggests the solution is unsatisfactory as it instead has been replaced by other issues. In part, these are again due to the population distribution function $\hat{F}_0$.

There is nothing to suggest that the outer quantiles of the population, corresponding to the most extreme points observed, are the most meaningful to consider. For instance, consider an example where one has a dataset with frequent monotonic mean changes of significant magnitude. In this circumstance it is straightforward to appreciate that the outer quantiles need not even share points in common with many of the individual segments considered.

The author suggests that both of these approaches are somewhat crude. The likelihood is either being evaluated at every point possible, or at seemingly arbitrarily chosen points. Note, an interesting point is that there is an approach discussed within Chapter 3 that considers a similar issue that one can draw inspiration from. Namely, Wild Binary Segmentation.

Recall, Wild Binary Segmentation attempted to improve upon the likelihood ratio based on a change in mean under normal assumptions by randomly drawing additional points over which to compute additional likelihood ratios. The goal and means of this chapter are similar in nature. Instead of improving power, however, the quality of interest is robustness. Further, instead of considering the maximum of all likelihood ratios, several will be combined into a single statistic.

Within Wild Binary Segmentation, each additional likelihood ratio was considered over the local interval at each stage rather than globally. The author also suggests taking a localised approach. In this case, it will arise from use of a function depending on $\hat{F}_i(s)$ rather than $\hat{F}_0$. This is a compromise between the global and quantile based approaches proposed so far.

One final comment will be again on the comparison between this approach and Wild Binary Segmentation. One undesirable aspect of Wild Binary Segmentation is that one also sets the parameter for the number of additional likelihood ratios considered globally. One of the highly desirable aspects of Binary Segmentation is that it can bisect and disregard large portions of the dataset rapidly. When such behaviour arises, global choices again may not represent well what one otherwise would have chosen for the actual subset of the data one spends most time considering. In this case, a weight function based on $\hat{F}_i(s)$ avoids this.

6.2.1.2 Anderson-Darling vs. Cramér-von-Mises

The next choice regarding the weighting is the functional form. As noted in the previous section, the motivation for using

$$dw(s) = \frac{d\hat{F}_0(s)}{\hat{F}_0(s)(1-\hat{F}_0(s))}$$

is that it places more emphasis on the tails of the distribution. The most apparent alternative is $dw(s) = d\hat{F}_0(s)$ and the difference results in a final cost function of the form

$$\frac{1}{n_i}\sum_{k=t_i}^{t_{i+1}-2}\left[\frac{n_i^2}{k(n_i-k)}\mathcal{L}(x_{t_i:j-1} \mid x_{(k)}) + \mathcal{L}(x_{j:t_{i+1}-1} \mid x_{(k)}) - \mathcal{L}(x_{t_i:t_{i+1}-1} \mid x_{(k)})\right],$$

where $x_{(k)}$ is the kth order population order statistic, versus

$$\frac{1}{n_i}\sum_{s=t_i}^{t_{i+1}-2}\left[\mathcal{L}(x_{t_i:j-1} \mid x_s) + \mathcal{L}(x_{j:t_{i+1}-1} \mid x_s) - \mathcal{L}(x_{t_i:t_{i+1}-1} \mid x_s)\right].$$

The difference is a weighted versus simple average. The author notes this idea regarding the tails of the distribution holding greater importance is central to both of NMCD and ED-PELT and per the preceding

discussion both have features emphasising this. To summarise Section 3.2 of the paper [50], the authors motivate this with two reasons. First, it is stated that as most observations occur around the median of the distribution, both detect changes of this nature sufficiently well. However, with fewer tail observations the simple average lacks sensitivity to these changes.

While this argument is plausible, note that it is descriptive only. It is surprising to the author that for an idea which is central to both approaches, no numerical or theoretical results are presented to support the argument. One could construct another contradictory descriptive argument that is also plausible. Given that the argument is contingent on few tail observations, one could suggest that tail probability estimates are less reliable and so should not be over-emphasised. This is particularly relevant in the context of heavy noise processes given the interest in robustness.

Weight functions of this form for goodness of fit tests were first introduced by Cramér, von-Mises, Anderson, and Darling before that of NMCD. A particularly relevant point is that this was in the context of the L_2 norm rather than the likelihood. Regarding these other goodness of fit tests, no consensus has ever arisen as to which is "best". For instance, in the comparative study [61] the performance of the Cramér-von-Mises versus Anderson-Darling statistics are equivalent. Stochastic questions of this nature are often nuanced and it is most likely that both descriptions are grossly oversimplifying. Nonetheless, a decision is to be made. This is to be resolved empirically.

Single Change Experiment - Mean Change

Abbreviation	$N(0,1)$	t_4	t_3	t_2	t_1
Average NPL	57.5 %	46.2 %	43.8 %	39.4 %	27.8 %
Weighted Average NPL	50.5 %	41.2 %	38.0 %	33.8 %	22.6 %

Figure 6.1: 300 points were generated from the specified noise process. The data undergoes a mean change of 0.5 at t = 250. The statistics displayed in the table record the proportion of times the maximum of the specified statistic was reached within a window of 5 either side of the changepoint over 500 repetitions.

Single Change Experiment - Mean and Variance Change

Abbreviation	$N(0,1)$ to $N(0.25,1)$	$N(0,1)$ to $N(0.25,0.5)$	$N(0,1)$ to $N(0.25,0.25)$
Average NPL	29.6 %	52.4 %	74.8 %
Weighted Average NPL	30.0 %	35.6 %	52.4 %

Figure 6.2: 300 points were generated from the specified noise process. The data undergoes the changes at t = 250. The statistics displayed in the table record the proportion of times the maximum of the specified statistic was reached within a window of 5 either side of the changepoint over 500 repetitions.

Interestingly, the results contained in Figures 6.1 and 6.2 portray a clear choice. It is noted that while simple, the single change case controls for differences between local and global weight functions based on $\hat{F}_i(s)$ versus $\hat{F}_0(s)$. Additionally, power comparisons of this form should generalise appropriately given that Binary Segmentation consists of recursively considering such a case. The interested reader can find a broader comparison with the CUSUM and a range of additional non-parametric statistics within appendix A.3. As such, a final weight function of $dw(s) = d\hat{F}_i(s)$ and corresponding test statistic of

$$\frac{1}{n_i}\sum_{s=t_i}^{t_{i+1}-1}\left[\mathcal{L}(x_{t_i:j-1} \mid x_s) + \mathcal{L}(x_{j:t_{i+1}-1} \mid x_s) - \mathcal{L}(x_{t_i:t_{i+1}-1} \mid x_s)\right]$$

shall be used throughout.

6.2.2 Properties of the Cost Function

Prior to constructing a decision rule, several properties of the cost function are to be highlighted. The first expands on the discussion in the previous section.

6.2.2.1 The Asymptotic Null Distribution

Recall in Chapter 4 it was noted that these methods arose in a series of works studying goodness of fit tests in a different context. This cost function was considered in the paper [56] and it is shown that under the null hypothesis, one has that

$$\frac{1}{n_i}\sum_{s=t_i}^{t_{i+1}-1}\left[\mathcal{L}(x_{t_i:j-1}\mid x_s)+\mathcal{L}(x_{j:t_{i+1}-1}\mid x_s)-\mathcal{L}(x_{t_i:t_{i+1}-1}\mid x_s)\right]\rightarrow_d \frac{1}{2}\int_0^1\frac{B^2(t)}{t(1-t)}dt$$

where $B(t)$ denotes the standard Brownian Motion. To briefly motivate this result, one can note that the Brownian Motion is the continuous time analogue of a sum of I.I.D. normal variables. One property is that $B(t)-B(0)\sim N(0,t)$ for instance [24]. Further one can note that

$$\int_{\mathbb{R}} B^2(F(t))g''(F(t))dF(t)=\int_0^1\frac{B^2(t)}{t(1-t)}dt$$

where $g:[0,1]\mapsto\mathbb{R}$ with

$$g(x)=x\log(x)+(1-x)\log(1-x)$$

represents the likelihood function and that

$$\sqrt{n}|\hat{F}-F|_\infty\rightarrow_d \sup_{t\in\mathbb{R}}(|B(F(t))|).$$

The result arises from considering the asymptotics of a second order series expansion. This is a standard approach in statistical theory owing to the standard $\sqrt{n}$ rate of convergence one gets from classical results such as the central limit theorem. Canonical examples of this include the proof of Wilk's theorem [62] and the efficiency of maximum likelihood estimators [22].

A result of particular significance is that the Anderson-Darling statistic also has [54]

$$n\int_{\mathbb{R}}\frac{(\hat{F}(x)-F(x))^2}{F(x)(1-F(x))}dF(x)\rightarrow_d\int_0^1\frac{B^2(t)}{t(1-t)}dt.$$

In light of Section 6.2.1.2 it is interesting to note that this likelihood ratio and the Anderson-Darling statistic have the same asymptotic distribution. The likelihood ratio considered naturally emphasises the tails of the distribution. While it has been argued successfully that the L_2 norm can benefit from including such an adjustment, it should equally be of note that, to the best of the author's knowledge, no widely accepted approach has yet to propose further emphasising the tails as per the work reviewed in Chapter 4. This is a suggestion of the two authors considering the changepoint problem, neither of whom provide results demonstrating its worth.

6.2.2.2 Further Properties

Finally, the relationship between the proposed cost and other widely known results are highlighted. First, note the discrete form of the KL-Divergence over the probability space $\mathcal{X}$ is

$$D_{KL}(F_1||F_2)=\sum_{x\in\mathcal{X}}F_1(x)\log\left(\frac{F_1(x)}{F_2(x)}\right).$$

If one notes that

$$\hat{F}_i(x_k)=\frac{j\hat{F}_i^{L_j}(x_k)+(n_i-j)\hat{F}_i^{R_j}(x_k)}{n_i}$$

it follows directly from substituting this into the definition of the likelihood function that this cost function

is also

$$D_{KL}(\hat{F}_i^{L_j}||\hat{F}_i) + D_{KL}(\hat{F}_i^{R_j}||\hat{F}) + D_{KL}(1-\hat{F}_i^{L_j}||1-\hat{F}_i) + D_{KL}(1-\hat{F}_i^{R_j}||1-\hat{F}_i).$$

Recall that $\hat{F}_i$ is a maximum likelihood estimator and as a result of the functional invariance of maximum likelihood estimates, the cost function can therefore also be viewed as a maximum likelihood estimate for this symmetrised KL-Divergence. Pseudo-distance notions of this form have been studied more generally and have interesting properties. One similar example is the Jensen-Shannon Divergence. More details can be found in [63]. Despite its popularity in the broader machine learning literature, within change detection the KL-Divergence has mainly been applied in the multivariate case [64].

Finally, a related note is that as the function g is convex, the cost is

$$\sum_{k=t_i}^{t_{i+1}-1}\left[\frac{jg(\hat{F}_i^{L_j}(x_k)) + (n_i-j)g(\hat{F}_i^{R_j}(x_k))}{n_i} - g\left(\frac{j\hat{F}_i^{L_j}(x_k) + (n_i-j)\hat{F}_i^{R_j}(x_k)}{n_i}\right)\right]$$

which quantifies the cumulative Jensen's Gap of the function g evaluated over the functions $\hat{F}_i^{L_j}$, $\hat{F}_i^{R_j}$, and $\hat{F}_i$.

6.3 Creating a Decision Rule

Given a cost function and means of optimisation, the final aspect of any change detection algorithm is the decision rule. In this case, the decision to introduce a changepoint corresponds to the test statistic exceeding a significance threshold. There are two possible approaches to this. The first is via model selection criteria and this has been the approach of all methods discussed. Note, one could also draw on traditional hypothesis testing theory with steps to account for multiple comparisons. The approach recommended follows the precedent set previously. However, translating the algorithm to a multiple comparison procedure would be a particularly interesting direction of future research and detailed discussion of this is contained within Section 8.3.2.

6.3.1 Information Criteria

Recall that this chapter began by considering whether the model with $n_i\hat{F}_i(s)$ being binomially distributed with one success probability is a superior fit to the data than that of the model with two success probabilities. In the case of the Bayesian Information Criterion, the quantity being compared is therefore

$$\text{BIC}_j = 2\log(n_i) - 2(\mathcal{L}(x_{t_i:j-1} \mid s) + \mathcal{L}(x_{j:t_{i+1}-1} \mid s))$$

versus

$$\text{BIC}_0 = \log(n_i) - 2\mathcal{L}(x_{t_i:t_{i+1}-1} \mid s).$$

Choosing the better model then comes down to the simple case of asking if $\max_j \text{BIC}_j < \text{BIC}_0$. Or, if

$$\mathcal{L}(x_{t_i:j-1} \mid s) + \mathcal{L}(x_{j:t_{i+1}-1} \mid s) - \mathcal{L}(x_{t_i:t_{i+1}-1} \mid s) > \frac{1}{2}\log(n_i).$$

A complication to the approach taken here is that one wants to combine various likelihood ratio tests. Specifically, the quantity of interest is in the average of all likelihood ratios observed

$$\frac{1}{n_i}\sum_{t_i}^{t_{i+1}-1}\left[\mathcal{L}(x_{t_i:j-1} \mid s) + \mathcal{L}(x_{j:t_{i+1}-1} \mid s) - \mathcal{L}(x_{t_i:t_{i+1}-1} \mid s)\right]$$

One simple approach would be to ask whether that the likelihood ratio exceeds the same quantity on average. Specifically,

$$\frac{1}{n_i}\sum_{k=t_i}^{t_{i+1}-2}\left[\mathcal{L}(x_{t_i:j-1} \mid x_k) + \mathcal{L}(x_{j:t_{i+1}-1} \mid x_k) - \mathcal{L}(x_{t_i:t_{i+1}-1} \mid x_k)\right] > \frac{1}{2}\log(n_i)$$

and it is this which is recommended by default.

6.3.1.1 Comparisons with Predecessors

Translating the decision rule and weighting used by NMCD to this context would give

$$\sum_{k=t_i}^{t_{i+1}-2}\left[\frac{n_i}{k(n_i-k)}\mathcal{L}(x_{t_i:j-1}\mid x_{(k)})+\mathcal{L}(x_{j:t_{i+1}-1}\mid x_{(k)})-\mathcal{L}(x_{t_i:t_{i+1}-1}\mid x_{(k)})\right]>\frac{1}{2}(\log(n_i))^2$$

where $x_{(k)}$ represents the kth order statistic of the population. It is interesting to note that for the statistic introduced within this chapter, this decision rule implies that

$$\frac{1}{n_i}\sum_{k=t_i}^{t_{i+1}-2}\left[\mathcal{L}(x_{t_i:j-1}\mid x_{(k)})+\mathcal{L}(x_{j:t_{i+1}-1}\mid x_{(k)})-\mathcal{L}(x_{t_i:t_{i+1}-1}\mid x_{(k)})\right]>\frac{(\log(n_i))^2}{4n_i(\log(n_i-1)+1)}.$$

Specifically, the point of interest is

$$\frac{\log(n_i)}{2}>\frac{(\log(n_i))^2}{4n_i(\log(n_i-1)+1)},$$

and hence that deciding to introduce a changepoint with the average likelihood ratio exceeding the threshold given by the BIC is a materially more stringent choice. For the derivation of this result, see Appendix B.1. Surprisingly, however, such an approach nonetheless works effectively in practice. On that note, the author raises two points. The first is to note how this particular $\frac{1}{2}(\log(n_i))^2$ arose. Within the paper [50] introducing NMCD this was a choice of penalty which allowed the proposed method to be consistent asymptotically. One particular limitation of this consistency result is that one of its assumptions involves the number of changepoints remaining finite while the length of the total dataset tends to infinity. The implication the author has drawn from this is that it is a case of idealised asymptotic results translating poorly to finite time.

The second point to raise is that the author's of ED-PELT still use a penalty of $\frac{1}{2}(\log(n_i))^2$ by default also. Recall now that the cost function of ED-PELT is implemented with a sum over $4\log(n_i)$ terms, compared to that of NMCD who sum over every point observed within the dataset. Given that the likelihood ratio terms are strictly positive it is therefore necessarily the case that the statistic of NMCD has to be far in excess of the statistic used by ED-PELT. This would again suggest a similar finding to the first point. Specifically, that the penalty proposed by NMCD does not penalise sufficiently.

The cost function and decision rule proposed by ED-PELT translated to this context would be

$$\frac{2\log(2n-1)}{4\log(n)}\sum_{i=1}^{4\log(n)}\left[\mathcal{L}(x_{t_i:j-1}\mid t_i)+\mathcal{L}(x_{j:t_{i+1}-1}\mid t_i)-\mathcal{L}(x_{t_i:t_{i+1}-1}\mid t_i)\right]>\frac{(\log(n_i))^2}{2}$$

where the t_i's are as defined previously. The author is therefore asking that the average of the likelihood ratio over each t_i exceeds the quantity

$$\frac{(\log(n))^2}{4\log(2n-1)}.$$

The ratio of $\log(n)$ to $2\log(2n-1)$ is between 0.44 and 0.48 for $10<n<10^9$ and so for practical purposes the proposed thresholds differ by a constant magnitude. Aside from different points being taken in the average, these are comparable.

6.4 The Mathematics of Computational Efficiency

6.4.1 Overview

The discussion contained so far has been a reformulation of the cost function. Recall, the objective of the author is to reduce computational time without sacrificing classification effectiveness. Section 4.2 demonstrated one way of reducing computational time. It is this second point where the author believes the previous approach falls short. ED-PELT merely reduces the number of computations performed, without regard for whether the results are meaningful. Instead, the author believes that this new cost function isolates the computations which are most informative. Now, the objective is to formulate a minimalist way of computing these. Finally, note that for the sake of clarity it is primarily the conclusion of any manipulation which is presented within this section. Full derivations for every result are deferred to the corresponding section of Appendix B.

6.4.2 Memoization

There are several mathematical properties of the expression

$$\frac{1}{n_i}\left[\sum_{k=t_i}^{t_{i+1}-2}\mathcal{L}(x_{t_i:j}\mid x_k)+\mathcal{L}(x_{j+1:t_{i+1}-1}\mid x_k)-\mathcal{L}(x_{t_i:t_{i+1}-1}\mid x_k)\right]$$

which allow us to compute it efficiently. To do so, define functions $H:\mathbb{Z}_+\times\mathbb{Z}_+\mapsto\mathbb{R}$ by

$$H(k,j)=\begin{cases}0 & k\leq 0\\ k\log(k)+(j-k)\log(j-k) & 0<k<j\\ 0 & k\geq j\end{cases},$$

$G:\mathbb{Z}_+\times\mapsto\mathbb{R}$ by

$$G(k,j)=k\log(k)$$

so that one has

$$H(k,j)=G(k)+G(j-k),$$

$S:\mathbb{Z}_+\mapsto\mathbb{R}$ by

$$S(i)=\sum_{k=1}^{i}G(k),$$

and finally $D:\mathbb{Z}_+\times\mapsto\mathbb{R}$ by

$$D(k)=G(k)-G(k-1).$$

The function H is introduced to ease notation. It will be demonstrated that evaluating the cost function will reduce to making evaluations of the remaining functions G, S, and D at particular points. Note that as all of these are defined over integers, one knows in advance every value of the logarithm which one could need at any point of the algorithm. These can therefore be computed upfront and stored, avoiding the need to make repeated calls to the logarithm function and hence reducing compute time. This is a common technique known as a memoization [34]. The Dynamic Programming approaches seen within Chapter 3 can be viewed as an elaborate application of memoization.

6.4.3 Data Independent Terms

Returning back to the likelihood, many terms of this cost function depend on the position of the changepoint considered and the overall length of the data, rather than having any explicit dependence on the data itself. These can be computed outright through memoization with no additional work.

6.4.3.1 Full Segment Cost

First note that

$$-\sum_{k=t_i}^{t_{i+1}-2} \mathcal{L}(x_{t_i:t_{i+1}-1} \mid x_k)$$

is constant with respect to each changepoint and therefore that this need only be evaluated once after obtaining the minimum of the segmented likelihoods. Specifically, this sum is equal to

$$-\sum_{k=1}^{n_i-1} \left[k \log\left(\frac{k}{n_i}\right) + (n_i - k) \log\left(\frac{n_i - k}{n_i}\right) \right]$$

and within the appendix it is shown that this is equal to

$$(n_i - 1)G(n_i) - 2S(n_i - 1).$$

6.4.3.2 Denominators

If one denotes $R_{L_j}(x)$ by the rank of point x within segment $x_{t_i:j}$,

$$R_{L_j}(x) = \sum_{k=t_i}^{j} 1_{X_k \leq x}$$

and $R_{R_j}(x)$ similarly the rank within segment $x_{j+1:t_{i+1}-1}$, it can be shown that an analogous result to that of the full segment applies to the remaining likelihood functions. By using logarithm properties to remove the denominators of the empirical cumulative distribution function, these sums can also be written as

$$-(n_i - 1)(G(j) + G(n_i - j)) + \sum_{k=1}^{n_i-1} H\left(R_{L_j}(x_k)), j\right) + H\left(R_{R_j}(x_k)), n_i - j\right).$$

The last term again is a function of the length of each sub-segment requiring no insight into the values of the data to compute. This is useful as it allows one to only consider the integer values of the ranks rather than rationals. Specifically, it gives rise to the remaining summation over the likelihoods being

$$\sum_{k=t_i}^{j-1} H\left(R_{L_j}(x_k)), j\right) + H\left(R_{R_j}(x_k)), n_i - j\right) + \sum_{k=j}^{t_{i+1}-1} H\left(R_{R_j}(x_k)), n_i - j\right) + H\left(R_{L_j}(x_k)), j\right).$$

6.4.3.3 Within Segment Costs

Now, focusing on the first term in each of the sums,

$$\sum_{k=t_i}^{j-1} H\left(R_{L_j}(x_k)), j\right) + \sum_{k=j}^{t_{i+1}-1} H\left(R_{R_j}(x_k)), n_i - j\right)$$

attempting these summations term by term is difficult because one does not know $R_{L_j}(x_k)$ or $R_{R_j}(x_k)$ for each k in advance. One does, however, know the number of points in each segment and, subject to adjustment for tied ranks, that there must be a bijection between the rank function and the sets $\{1, 2, \ldots, j\}$ and $\{1, 2, \ldots, n_i - j\}$ respectively. These sums are therefore

$$\sum_{k=1}^{j} H(k, j) + \sum_{k=1}^{n_i-j} H(k, n_i - j)$$

or

$$2S(j) + 2S(n_i - j) = \sum_{k=1}^{j} G(k) + G(j - k) \sum_{k=1}^{n_i-j} G(k) + G(n_i - j - k)$$

which again has no dependence on the data but the lengths of the sub-segments.

6.4.4 Cost Function Update Rules

The terms so far have been based on the cost of a given segment length. These are baseline model fit terms which arise. The essence of this cost function, however, lies in the terms spanning across sub-segments. These amount to evaluating the likelihood of observing a point in one segment given the empirically observed distribution of the alternate segment. If one observes this to be low enough in aggregate, the cost function suggests a changepoint should be introduced.

These terms are represented mathematically within the second terms of the above summations,

$$\sum_{k=t_i}^{j-1} H\left(R_{R_j}(x_k), n_i - j\right) + \sum_{k=j}^{t_{i+1}-1} H\left(R_{L_j}(x_k), j\right).$$

While the details differ for each specific segment, there are two primary insights to be aware of that motivate every result within the next section. The first is that

$$R_{L_j}(x_k) + R_{R_j}(x_k) = R(x_k),$$

where $R(x)$ represents the segment rank. One could therefore equally write the total summation as

$$\sum_{k=t_i}^{j} H\left(R(x_k) - R_{L_j}(x_k), n_i - j\right) + \sum_{k=j}^{t_{i+1}-1} H\left(R(x_k) - R_{R_j}(x_k), j\right).$$

This is used such that one only ever need to track the rank of a point within the segment it is contained in. The other point to note is that

$$\hat{F}_{j+1}^{L_j}(x) = \frac{j}{j+1}\hat{F}_j^{L_j}(x) + \frac{1}{j+1}1_{X_{j+1}\leq x}$$

and that

$$\hat{F}_{j+1}^{R_j}(x) = \frac{n_i - j}{n_i - (j+1)}\hat{F}_j^{R_j}(x) - \frac{1}{n_i - (j+1)}1_{X_{j+1}\leq x}.$$

Note that the denominators of each distribution function have already been removed within the previous section and so within the cost function terms involving functions need only be adjusted by adding and subtracting one to specific ranks.

The objective now is to consider how the cost function will specifically change as one considers introducing a changepoint at $j+1$ after having considered introducing a changepoint at j. The next section details the preicse means by which one can apply these results to adjust the cost function.

6.4.4.1 Left Cross-Segment Updates

In the subsequent cost function, the term associated with the left segment is

$$\sum_{k=t_i}^{j+1} H\left(R(x_k) - R_{L_{j+1}}(x_k), n_i - (j+1)\right)$$

This sum can be computed with relative ease by breaking it down into separate cases. Specifically, as $x_k < x_{j+1}$ implies $R_{L_{j+1}}(x_k) = R_{L_j}(x_k)$, one has that in adjusting from one segment to the next the cost is

$$\sum_{k=t_i, x_k<x_{j+1}}^{j+1} H\left(R(x_k) - R_{L_j}(x_k), n_i - j\right) - D\left(n_i - j - (R(x_k) - R_{L_{j+1}}(x_k))\right).$$

The consequence is that as a new point is added to the left segment, updating the cost of the points contained in the segment with rank lower than the additional point consists of subtracting one term to reflect the decreased length of the alternate segment.

By examining the second term of the sum,

$$\sum_{k=t_i, x_k \geq x_j}^{j+1} H\left(R(x_k) - R_{L_{j+1}}(x_k), n_i - (j+1)\right),$$

one can see that a similar relationship holds. This time note that $x_k \geq x_j$ implies that $R_{L_{j+1}}(x_k) = R_{Lj}(x_k) + 1$. The second term of the summation can be written as

$$\sum_{k=t_i, x_k \geq x_j}^{j+1} H\left(R(x_k) - R_{L_j}(x_k), n_i - j\right) - D\left(R(x_k) - (R_{L_{j+1}}(x_k) - 1)\right).$$

In order to update the second term of the cost, again, one can start by adjusting the previous cost through subtraction of one term to reflect the decreased observed frequency of points with magnitude above that which has been removed.

One difference in this case, however, is that one has the additional point x_{j+1} contained within this sum. Rather than updating the cost of x_{j+1} in the same fashion as all other points, it is more sensible to consider the cost which the term x_{j+1} would have previously given rise to. This would have been

$$H\left(R(x_{j+1}) - R_{R_j}(x_{k+1}), j\right)$$

compared to the current cost of

$$H\left(R(x_k) - (R_{L_{j+1}}(x_k), n_i - (j+1)\right).$$

The role of considering terms as a function of

$$R_{L_j}(x) = R(x) - R_{R_j}(x)$$

is so that one does not have to maintain track of the rank of each point within the segment it does not belong to. In this case, however, the point has switched segment and so it is in fact simpler to view the difference in cost between these points instead as

$$H\left(R(x_{j+1}) - R_{L_{j+1}}(x_{j+1}), n_i - (j+1)\right) - H\left(R_{L_j}(x_{j+1}), j\right).$$

This is still simple to update again by the identity

$$R_{L_{j+1}}(x_{j+1}) = R_{L_j}(x_{j+1}) + 1.$$

giving rise to an update rule combining

$$G\left(R(x_{j+1}) - R_{L_{j+1}}(x_{j+1})\right) + G\left(n_i - (j+1) - \left(R(x_{j+1}) - R_{L_{j+1}}(x_{j+1})\right)\right)$$

with

$$-\left(G\left(R_{L_{j+1}}(x_{j+1}) - 1\right) + G\left(j - \left(R_{L_{j+1}}(x_{j+1}) - 1\right)\right)\right).$$

6.4.4.2 Right Cross-Segment Updates

In turning our attention toward the right segment, a similar set of results can be used to update the cost of the points $x_{j+2:t_{i+1}-1}$. These should be relatively simple to understand as the reverse of the updates given for the left segment. In this case, the length of the alternate segment is incremented by 1 while the ranks of all points greater than the point removed are decremented by 1.

The cost associated with the right segment is

$$\sum_{k=j+2}^{t_{i+1}-1} H\left(R(x_k) - R_{R_{j+1}}(x_k), j+1\right)$$

A relevant identity for the summation is that for $x_k < x_{j+1}$, $R_{R_{j+1}}(x_k) = R_{R_j}(x_k)$. The first sum therefore can be viewed as

$$\sum_{k=j+2, x_k<x_{j+1}}^{t_{i+1}-1} H\left(R(x_k) - R_{R_j}(x_k), j+1\right)$$

which is

$$\sum_{k=j+2, x_k<x_j+1}^{t_{i+1}-1} H\left(R(x_k) - R_{R_j}(x_k), n_i - j\right) + D\left(j+1-(R(x_k) - R_{R_{j+1}}(x_k))\right).$$

Within the right segment, the cross-segment costs are updated through addition rather than subtraction. This term reflects a decrement to the segment length. The final cost term to consider is

$$\sum_{k=j+2, x_k\geq x_{j+1}}^{t_{i+1}-1} H\left(R(x_k) - R_{R_{j+1}}(x_k), j+1\right).$$

The relevant identity in this case is that for $x_k \geq x_{j+1}$, $R_{R_{j+1}}(x_k) = R_{R_j}(x_k) - 1$. The above term is

$$\sum_{k=j+2, x_k\geq x_{j+1}}^{t_{i+1}-1} H\left(R(x_k) - R_{R_j}(x_k), j\right) + D\left(R(x_k) - R_{R_{j+1}}(x_k)\right).$$

Again, the final term can be updated by addition. This is to reflect the decreased observed frequency of points.

6.4.5 Review

To review, within the algorithm there are six updates in total to adjust the cost of changepoint j to that of changepoint $j+1$. In the following, the cost associated with x_k when introducing a changepoint at j is denoted by $C_j(k)$.

The left cross-segment updates are

$$1) \text{ For } x_k \in \{x_{t_i} : j, x_k < x_{j+1}\}, C_{j+1}(k) = C_j(k) - D\left(n_i - j - (R(x_k) - R_{L_{j+1}}(x_k))\right)$$

$$2) \text{ For } x_k \in \{x_{t_i} : j, x_k > x_{j+1}\}, C_{j+1}(k) = C_j(k) - D\left(R(x_k) - (R_{L_{j+1}}(x_k) - 1)\right),$$

whereas the right cross-segment updates are

$$3) \text{ For } x_k \in \{x_{j+2} : t_{i+1} - 1, x_k < x_{j+1}\}, C_{j+1}(k) = C_j(k) + D\left(j+1-(R(x_k) - R_{R_{j+1}}(x_k))\right)$$

$$4)\ \text{For } x_k \in \{x_{j+2} : t_{i+1} - 1, x_k > x_{j+1}\}, C_{j+1}(k) = C_j(k) + D\left(R(x_k) - R_{R_{j+1}}(x_k)\right).$$

The update for the term which switches segment, x_{j+1}, involves differencing

$$5i)\ G\left(R(x_{j+1}) - R_{L_{j+1}}(x_{j+1})\right) + G\left(n_i - (j+1) - \left(R(x_{j+1}) - R_{L_{j+1}}(x_{j+1})\right)\right)$$

$$5ii)\ G\left(R_{L_{j+1}}(x_{j+1}) - 1\right) + G\left(j - \left(R_{L_{j+1}}(x_{j+1}) - 1\right)\right).$$

The remaining terms are independent of the data and can be expressed as a function of the respective segment lengths. The baseline terms are $2S(j)$ and $2S(n_i - j)$, while the denominators within the logarithms are $-(n_i - 1)G(j)$ and $-(n_i - 1)G(n_i - j)$ giving rise to a final update rule of

$$6)2\left(G(j) - G(n_i - j)\right) - (n_i - 1)\left(D(j) - D(n_i - j)\right).$$

It is worth highlighting, many of these update rules work because one is using a simple average of the likelihoods. Were one using a weighted version, the update rules would be more complex. The adjustments required would not affect the asymptotic time complexity, but it nonetheless would require additional operations and be somewhat less efficient. This is another benefit to using this approach, a simple average of the likelihood ratios, rather than what has been seen in the literature so far. Although, it is also worth stating that this is certainly a secondary benefit to that of the additional robustness one gains from placing less emphasis on those tests most prone to distortion from noise.

6.5 Implementation

6.5.1 Overview

The mathematics outlined above has important consequences for the algorithm. It is a minimalist formulation of the cost function. Essentially, there are only four changes which occur upon examining a new changepoint from a previous one. Specifically, the ranks of each segment are incremented up and down by one and the segment lengths are incremented up and down by one. After computing the initial cost, one needs to update these four sums as a result. Computing the cost function without making use of these properties would require both n^2 calls to the logarithm function and knowledge of any given points rank in both the sub-segment to which it belongs and to the alternate sub-segment. With this, however, one need only know its rank within the segment as a whole and the sub-segment to which it belongs.

It is worth noting that the formulation does not explain how this can be implemented algorithmically. Details regarding the computation of ranks have been unaddressed. While the memoization approach introduced reduces the problem to one involving sums of stored numbers, the algorithm could nonetheless have unacceptable time complexity were this performed inefficiently. This is the case with ED-PELT, for instance. Differences in searching methodology aside, the discrete approximation involving only $\mathcal{O}(\log(n))$ terms would almost certainly still be computationally faster than a summation over $\mathcal{O}(n)$ stored terms despite making repeated calls to the logarithm. The remaining discussion is therefore devoted to the implementation of these computations.

6.5.2 Dynamic Sorting

Efficient implementation in this case is contingent upon the use of an appropriate data structure. The idea behind data structures is that if one knows in advance that they are to repeatedly perform particular operations, it can be preferable to store data in a structured way instead of using a simple array. There is an upfront cost associated with imposing any structure. However, if it subsequently allows one to use more a efficient algorithm for a particular operation the investment may be worth it. In this case the interest is in rank statistics, which are defined by the positions of elements once the data is sorted. There are several properties which allow this to be performed efficiently.

First, note that any given points rank within the total segment is independent of how one subsequently segments the dataset. This can therefore be stored inexpensively by sorting the data once at the outset and performing an update upon each interval bisection. Instead, the main computational expense in computing the cost function then comes from maintaining the ordering of each respective sub-segment. Within segment ranks require frequent updates given that the segment changes each time a new changepoint is considered. The simplest possible means of computing the rank of each point would simply be to manually count the number of elements with magnitude less than the given point. A slightly more sophisticated approach would be achieved by repeatedly sorting the data. This can still be improved upon.

In a linear sweep of the dataset each segment changes only by the removal and addition of one point at a time. To an extent, the rank structures are preserved. Specifically, to again sort the data with the addition of one incremental point it is only necessary to insert it within the correct position. This therefore is the main operation of interest and choosing a data structure to support efficient insertion is the primary concern.

Efficient insertion into a sorted list is a task for which algorithms have been studied extensively. Note, this is equivalent to searching for the position of an element within a sorted list. Perhaps the best known approach to this is the binary search. The Binary Search Tree is an extension of this. A full account of the details can be found in [34]. Briefly, the idea is to store all points in a tree structure such that any point greater than a given node is placed in the right sub-tree, otherwise it is placed in the left tree. Hence, when searching for any point one can rule out an entire subtree with a simple binary comparison. Of main concern is keeping the tree balanced. One way of doing so is with the red-black tree which stores a counter to ensure that no path is more than twice as long as any other. Searching for a point in a list, and insertion by extension, is then supported in $\mathcal{O}(\log(n))$. It is this variant which is used within this approach.

6.5.2.1 Pseudocode

Pseudocode drawing on all of the aspects discussed to construct the algorithm in full follows. An implementation with an R interface and C++ backend was created using Rcpp [65]. It is available freely for distribution in the github repository located at [66]. This code also contains step by step comments and it is recommended to view this where possible.

Algorithm 5 Non-Parametric Likelihood Based Binary Segmentation
Main Routine Implementing Binary Segmentation

Procedure NPL_BS(.)
Required Input:
Dataset $(x_1, ..., x_n) \in \mathbb{R}^n$
Penalty $\beta \in \mathbb{R}^+$

Optional Input:
Global_Tree, a binary search tree with pairs (data, rank) sorted in ascending order by data
Log_Lookup, an array containing $k \log(k)$ for $k \in \{1, 2, ..., n\}$
Log_Diff_Lookup, an array containing $k \log(k) - (k-1) \log(k-1)$ for $k \in \{1, 2, ..., n\}$
Log_Sum_Lookup, an array containing $\sum_{j=1}^{k} k \log(k)$ for $k \in \{1, 2, ..., n\}$

Initialise:
If Optional Input == Null:
    Sorted_Data ← sort(Dataset)
    Rank ← 0
    For Data in Sorted_Data:
        Rank += 1
        Global_Tree.insert(Data, Rank)
        Log_Lookup[Rank - 1] ← Rank * log(Rank)
        Log_Diff_Lookup[Rank - 1] ← Log_Lookup[Rank - 1] - Log_Lookup[Rank - 2]
        Log_Sum_Lookup[Rank - 1] ← Log_Sum_Lookup[Rank - 2] + Log_Lookup[Rank - 1]
Optional_Input ← [Log_Lookup, Log_Diff_Lookup, Log_Sum_Lookup, Global_Tree]
Changepoints ← Empty ordered set used to store changepoint locations
Change_Out ← Empty container storing the changepoint candidate, test statistic, and segment trees

Recursion:
Change_Out ← NPL_Single_Change(Dataset, Optional_Input)
If Change_Out[1] > β:
    Changepoints.insert(Change_Out[0])
    Update Tree:
    Rank ← 0
    Left_Tree ← Change_Out[3]
    For Data in Left_Tree:
        Rank += 1
        Left_Tree[Data] ← Rank
    Optional_Input[3] ← Left_Tree
    Changepoints.insert(NPL_BS(Dataset[: (Change_Out[0]-1)], β, Optional_Input))
    Update Tree:
    Rank ← 0
    Right_Tree ← Change_Out[4]
    For Data in Right_Tree:
        Rank += 1
        Right_Tree[Data] ← Rank
    Optional_Input[4] ← Right_Tree
    Changepoints.insert(NPL_BS(Dataset[Change_Out[0] :], β, Optional_Input))

Return Changepoints

Algorithm 6 Non-Parametric Likelihood Based Binary Segmentation
Changepoint Identifying Subroutine

Procedure NPL_Single_Change(.)
Required Input:
Dataset $(x_1, ..., x_n) \in \mathbb{R}^n$
Penalty $\beta \in \mathbb{R}^+$
Global_Tree, a binary search tree with pairs (data, rank) sorted in ascending order by data
Log_Lookup, an array containing $k\log(k)$ for $k \in \{1, 2, ..., n\}$
Log_Diff_Lookup, an array containing $k\log(k) - (k-1)\log(k-1)$ for $k \in \{1, 2, ..., n\}$
Log_Sum_Lookup, an array containing $\sum_{j=1}^{k} k\log(k)$ for $k \in \{1, 2, ..., n\}$

Initialise:
N ← length(Dataset)
Changepoint_Index ← 1
Max_Changepoint_Index ← 1
Left_Tree ← Empty tree container with pairs (data, global rank) for the left segment
Right_Tree ← Global_Tree, tree container with pairs (data, global rank) for the right segment
L_Iter ← Null iterator for the left binary search tree
R_Iter ← Null iterator for the right binary search tree
Max_Left_Tree ← Null binary search tree container
Max_Right_Tree ← Null binary search tree container
Rank ← Global_Tree[Dataset[0]]
Left_Tree.insert(Dataset[0], Rank)
Right_Tree.remove(Dataset[0])
Current_Cost ← Log_Lookup[Rank] + Log_Lookup[N - Rank]
Current_Cost += 2*Log_Sum_Lookup[N-1] - (N-1)*Log_Lookup[N]
Max_Cost ← Current_Cost

Main Loop:
For Data in Dataset[1 :]:
    Changepoint_Index += 1
    Update Tree:
    Left_Tree.insert(Data, Global_Tree[data])
    Left_Tree_Iterator ← Left_Tree.find(Data)
    Right_Tree.remove(Data)
    Right_Tree_Iterator ← Right_Tree.upper_bound(Data)
    Cross-Segment Updates:
    Current_Cost += Cost_Update(Left_Tree, Right_Tree, L_Iter, R_Iter, Log_Lookup, Log_Diff_Lookup)
    Within-Segment Updates:
    Current_Cost += 2 * Log_Lookup[Current_Changepoint_Index - 2]
    Current_Cost -= 2 * Log_Lookup[N - Current_Changepoint_Index - 2]
    Denominators:
    Current_Cost -= (N-1) * Log_Diff_Lookup[Current_Changepoint_Index - 1]
    Current_Cost += (N-1) * Log_Diff_Lookup[N - Current_Changepoint_Index - 1]
    If Current_Cost > Max_Cost:
        Max_Cost ← Current_Cost
        Max_Changepoint_Index ← Changepoint_Index
        Max_Left_Tree ← Left_Tree
        Max_Right_Tree ← Left_Tree
Baseline:
Max_Cost ← (1/N)*(Max_Cost - 2 * Log_Sum_Lookup[N - 2] + (N - 1) * Log_Lookup[N - 1])

Return Max_Changepoint_Index, Max_Cost, Max_Left_Tree, Max_Right_Tree

Algorithm 7 Non-Parametric Likelihood Based Binary Segmentation
Cost Update Subroutine

Procedure Cost_Update(.)
Required Input:
Left_Tree, a binary search tree with pairs (data, global rank) for a segment
Right_Tree, a binary search tree with pairs (data, global rank) for a segment
Left_Iterator, an iterator pointing to the new position of the point switched
Right_Iterator, an iterator pointing to the old position of the point switched
Log_Diff_Lookup, an array containing $k\log(k) - (k-1)\log(k-1)$ for $k \in \{1, 2, ..., n\}$
Log_Lookup, an array containing $k\log(k)$ for $k \in \{1, 2, ..., n\}$

Initialise:
Cost_Update ← 0
Rank ← 0
Alternate_Seg_Rank ← 0
L ← Left_Tree.size()
R ← Right_Tree.size()

Left Tree Updates:
For Pair in Segment_Tree to Left_Iterator:
    Rank += 1
    Alternate_Seg_Rank ← Pair[1] - Rank
    Cost_Update -= Log_Diff_Lookup[R + 1 - Alternate_Seg_Rank]
Rank += 1
Alternate_Seg_Rank += *(Tree_Iterator)[1] - Rank
Cost_Update += Log_Lookup[Alternate_Seg_Rank] + Log_Lookup[R - Alternate_Seg_Rank]
Cost_Update -= Log_Lookup[Rank - 1] + Log_Lookup[L - Rank]
Left_Iterator.increment()
Rank += 1
Alternate_Seg_Rank ← *(Left_Iterator)[1] - Rank
While Alternate_Seg_Rank < R & Left_Iterator != End:
    Cost_Update -= Log_Diff_Lookup[Alternate_Seg_Rank + 1]
    Rank += 1
    Left_Iterator.increment()
    Alternate_Seg_Rank ← Left_Iterator[1] - Rank
Cost_Update -= (R - Rank + 1) * Log_Diff_Lookup[L - 1]
Right Tree Updates:
Rank ← 0
For Pair in Segment_Tree to Right_Iterator:
    Rank += 1
    Alternate_Seg_Rank ← Pair[1] - Rank
    Cost_Update += Log_Diff_Lookup[L - Alternate_Seg_Rank]
Rank += 1
Alternate_Seg_Rank ← Right_Iterator[1] - Rank
While Alternate_Seg_Rank < L & Right_Iterator != End:
    Cost_Update += Log_Diff_Lookup[Alternate_Seg_Rank]
    Rank += 1
    Right_Iterator.increment()
    Alternate_Seg_Rank ← Right_Iterator[1] - Rank
Cost_Update += (L - Rank + 1) * Log_Diff_Lookup[R - 1]

Return Cost_Update

6.5.3 Time Complexity

The purpose of this section is first to give an account of the cost of each of these routines. Thereafter the scope of possible values for the total algorithm is discussed and a final time complexity assigned.

6.5.3.1 Routine Breakdown

The main routine consists of sorting the data and rebuilding the global trees. While not a main point of interest, it is noted that the sort is performed using Introspective Sort [67] which has $\mathcal{O}(n\log(n))$ time complexity. Rebuilding the trees involves $\mathcal{O}(n_i)$ operations each time a changepoint is found. Outside of a limited number of $\mathcal{O}(1)$ operations, the changepoint identifying subroutine primarily consists of operations with Binary Search Trees. There are 5 operations consisting of insertion, deletion, setting iterators twice, and look-up by key. Each of these are bounded above by $\mathcal{O}(\log(n_i))$ and are performed n_i times. These do not impact the leading term in the time complexity and in fact amortise to $\mathcal{O}(1)$ time. The cost update subroutine has been formulated such that it consists of only a single addition of a stored value for each element contained within the segment. The remaining operations include incrementing iterators and its time complexity is therefore $\mathcal{O}(n_i)$. This is called n_i times each time a single change is identified.

6.5.3.2 Algorithm Complexity

Note, it is the number of calls to the cost update subroutine which dominates the time complexity and therefore discussion of this will be the focus. Relative to other variants of Binary Segmentation, the dependence of this on the segment length makes it more difficult to understand. The worst-case scenario arises when a changepoint is found at the end of the interval during each iteration of the algorithm. In this situation, the cost of these calls consists of $\sum_{i=0}^{n-1}(n-i)^2 \sim \mathcal{O}(n^3)$. The other extreme is where no changepoints are found in which there $\mathcal{O}(n*n) = \mathcal{O}(n^2)$ updates.

Consider now a case where an arbitrary number of changepoints are found but the interval is bisected precisely in half at each stage. In this case the cost is $\sum_{i=0}^{n-1} 2^{-2i}n^2$ and the convergence of $\sum_{i=0}^{\infty} 2^{-2i}$ ensures that the total cost is always of $\mathcal{O}(n^2)$. It can be shown that a similar result holds for any other fixed ratio, as $0 < r < 1$ implies the convergence of

$$\sum_{n=1}^{\infty} r^{4n} + 2(r^{2n})(1-r)^{2n} + (1-r)^{4n} = \sum_{n=1}^{\infty}(r^{2n} + (1-r)^{2n})^2.$$

These scenarios were presented to illustrate that the time complexity largely depends on the position within the segment where the data is bisected. If one were to assume that upon each iteration one side of the interval is truncated by a fixed number of points, the time complexity will be $\mathcal{O}(Mn^2)$ where M represents the number of changepoints found. In such a case one could reasonably expect time complexities of $\mathcal{O}(n^2\log(n))$ or $\mathcal{O}(n^{\frac{3}{2}})$. Alternately, as has been suggested, were the interval to be bisected in a fixed proportion the time complexity will be of $\mathcal{O}(n^2)$. As the assumptions underpinning Binary Segmentation's $\mathcal{O}(n\log(n))$ time complexity is comparable to this, $\mathcal{O}(n^2)$ is deemed the base time complexity. However, as per before, the underlying assumptions and possible scenarios are most important to note.

6.5.3.3 Comparison

First, consider the approach relative to Wild Binary Segmentation. The faster of the two will be dictated by the choice of hyperparameter for WBS. The ability to readily scale the compute time while using WBS is a notable advantage. Throughout the simulation studies it was noted that parameters as low as $\sqrt{n}$ were highly effective and evidently in such a case WBS would be faster. Recall, however, that the author of WBS has recommended a parameter of 5000 for datasets "no longer than a few thousand", which is a multiple of $\mathcal{O}(n)$. This algorithm has the advantage of scaling the number of likelihood ratios evaluated proportionally to the segment length considered. Were one to follow such a recommendation, this approach would be expected to be faster. Relative to Dynamic Programming approaches, $\mathcal{O}(n^2)$ time complexity parallels that of Optimal Partitioning. Whether NPL-BS or ED-PELT is faster therefore depends on whether the gains in compute time from PELT's pruning criteria are outweighed by the expense of evaluating the cost function.

Chapter 7

Further Empirical Studies

The purpose of this chapter is to provide broad benchmarks of all algorithms introduced within this thesis. The performance evaluation framework is consistent with that of Chapter 5. A core difference lies in the scenarios considered, which now consists of a range of change types and noise processes. Further discussion of the core methods is presented in addition to particular emphasis on evaluating the novel approach introduced.

7.1 Parameter Selection

Rather than to merely present cases showing that NPL-BS can perform well, the intention is to demonstrate where it fits within the broader method selection recommendations given so far. Non-parametric cost functions can be viewed as taking a minimax approach to the changepoint problem [26]. In summary, this acknowledges that method selection can be a subtle and misleading problem. Selecting an approach based on one's expectation of absolute performance can therefore be intractable. One alternative is to instead choose the method where the expected worst case performance is as favourable as possible.

A distinct feature shown in some cases within Chapter 5 is that often the power gains from deploying a favourable algorithm are much outweighed by the performance loss under misspecification. In such circumstances, a minimax approach can be highly fruitful. The first set of experiments presented within this section are dedicated to determining whether this applies to NPL-BS. Specifically, the plots present a worst case scenario for the relative performance of NPL-BS. Asking for a cost function to detect arbitrary change inherently reduces its sensitivity to any given change. On the other hand, the performance is considered relative to those on the other extreme. Namely, where one correctly makes parametric assumptions and chooses only to look for the precise type of change which is occurring. This will largely speak to the approaches overall utility. Within the next set of plots a presentation of the central benefit of the approach is given, by highlighting its effectiveness in noise processes where the state of the art is incapable of performing adequately.

7.2 Simulation Results

7.2.1 Experimental Setup

Changepoints t_i are generated by first fixing n and λ, then sampling from $S_i \sim Po(\lambda)$ and setting $t_i = \sum_{j=1}^{i} S_i$ until $t_{i+1} > n$. All simulations presented within Sections 7.2.2 and 7.2.3 are on datasets of length $n = 10,000$, with $\lambda = 100$. Those of Section 7.2.4 vary n from 10^3 to 10^6 with λ set to either of $n/\log(n)$, $\sqrt{n}$, or 200 and these are referred to as the proportionally logarithmic, proportionally square root, and proportionally linear cases respectively. A threshold of $\epsilon = 5$ is considered throughout.

Within Section 7.2.2, the data X_t is generated from $N(\mu_t, \sigma_t)$ where μ_t and σ_t are piecewise constant. In all cases updating these parameters includes drawing a sample D from $\{-1, 1\}$ with equal probability. Mean only changes consist of, $\sigma_t = 1$, $\mu_0 = 0$, and setting $\mu_{t_i} = \mu_{t_i - 1} + 0.5D$. Variance only changes consist of $\mu_t = 0$, $\sigma_0 = 1$, and setting $\sigma_{t_i} = \sigma_{t_i - 1} + 0.5D$ unless $\sigma_{t_i - 1} = 0.5$ where $\sigma_{t_i} = 1$. Simultaneous mean and variance changes consist of $\mu_0 = 0$, $\sigma_0 = 1$, computing $\mu_{t_i} = \mu_{t_i - 1} - 0.5D$ and $\sigma_{t_i} = \sigma_{t_i - 1} + 0.5D$ unless $\sigma_{t_i - 1} = 0.5$ where $\sigma_{t_i} = 1$ and $\mu_{t_i} = \mu_{t_i - 1} - 0.5$. Finally, in the case of mean, variance, or mean and variance changes a sample is drawn with equal probability from $\{1, 2, 3\}$ to indicate one of the previously given change types. The mean only procedure is also used to generate the data within Section 7.2.4. In Section 7.2.3, $X_t = \mu_t + \epsilon_t$, where ϵ_t is a stochastic quantity drawn from either of t_4, t_3, t_2, t_1 as will be directed. The quantity μ_t is piecewise constant, initialised at $\mu_0 = 0$, and updated by $\mu_{t_i} = \mu_{t_i - 1} + D$.

7.2.2 Mean and Variance Shifts in the Gaussian Case

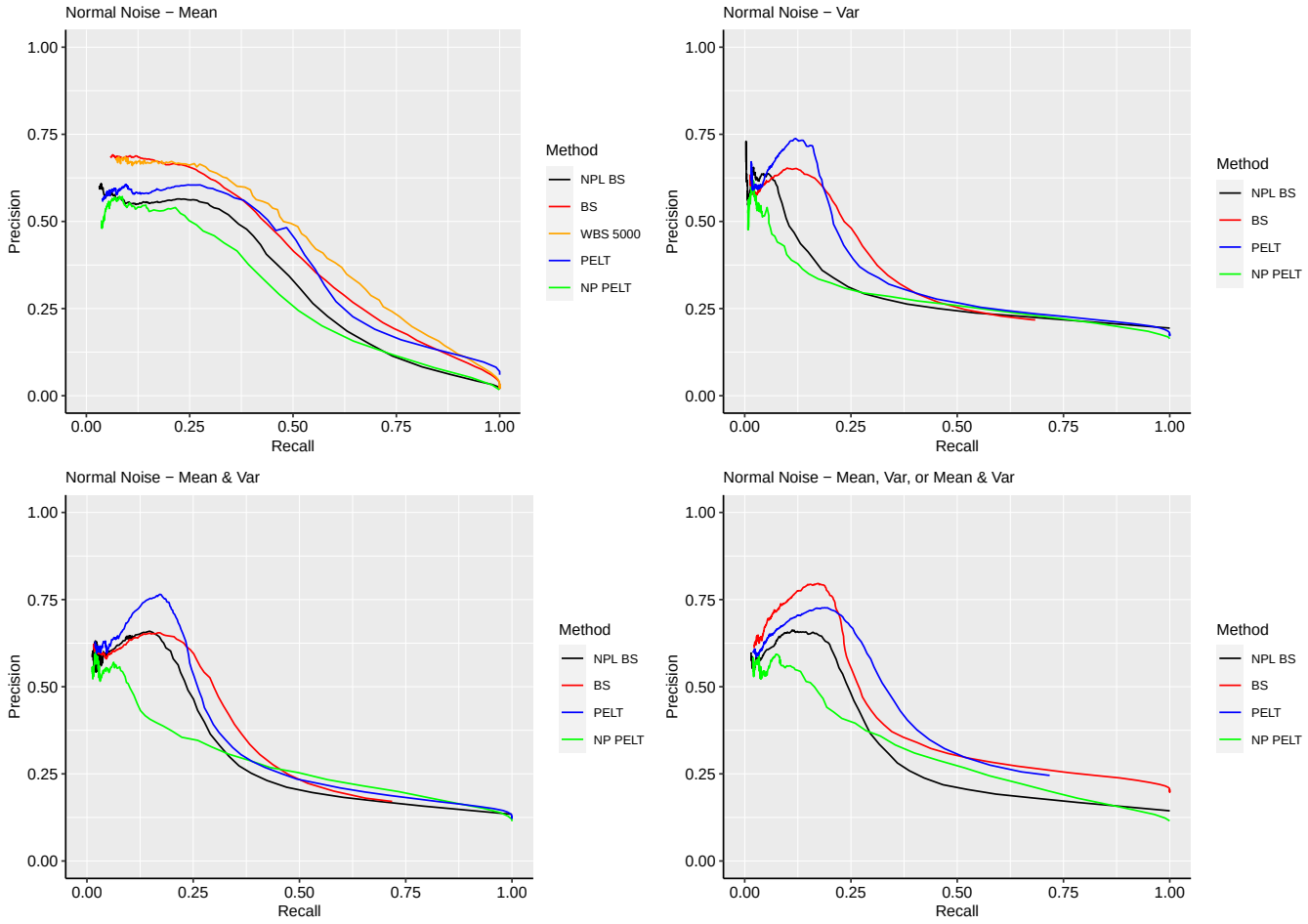


Figure 7.1: ED-PELT was called with the 'cpt.np' method of the changepoint.np package. In the mean only case, the 'sbs' and 'wbs' methods of the WBS package were used for BS and WBS 5000 with the 'cpt.mean' method of the changepoint package for PELT. Otherwise, 'cpt.var' and 'cpt.meanvar' were used with parameters of 'BinSeg' and 'PELT' to call BS and PELT. The values displayed are averages of Precision and Recall over 100 replications while varying the penalty parameters.

First note that the performance of Binary Segmentation and PELT is comparable in all cases but that of mean changes, where WBS outperforms over the full spectrum of penalties. Additionally, NPL-BS outperforms ED-PELT over the full spectrum of penalties in the mean and variance only cases. In the other cases, NPL-BS and ED-PELT are comparable with both exhibiting similar regions of outperformance.

Returning to the discussion of the minimax properties of the non-parametric approaches, it is noted that there is a meaningful power reduction in all cases. Interestingly, the relative performance reduction in using NPL-BS over SBS in part resembles that of using SBS where WBS is preferred, though to a slightly greater extent. Given the respectable robustness of SBS and WBS under modest misspecification, the argument for using a non-parametric cost function as one's base algorithm choice is weak. Were one also concerned by the possibility of changes in higher moments this may hold weight. By no means does this undermine the utility of NPL-BS however. Instead, it is the author's view that Binary Segmentation should remain the basis of a practitioners toolkit with this being a variant designed to address one of its weaknesses analogously to that of WBS or PELT.

7.2.3 Mean Shifts in Heavy Noise Processes

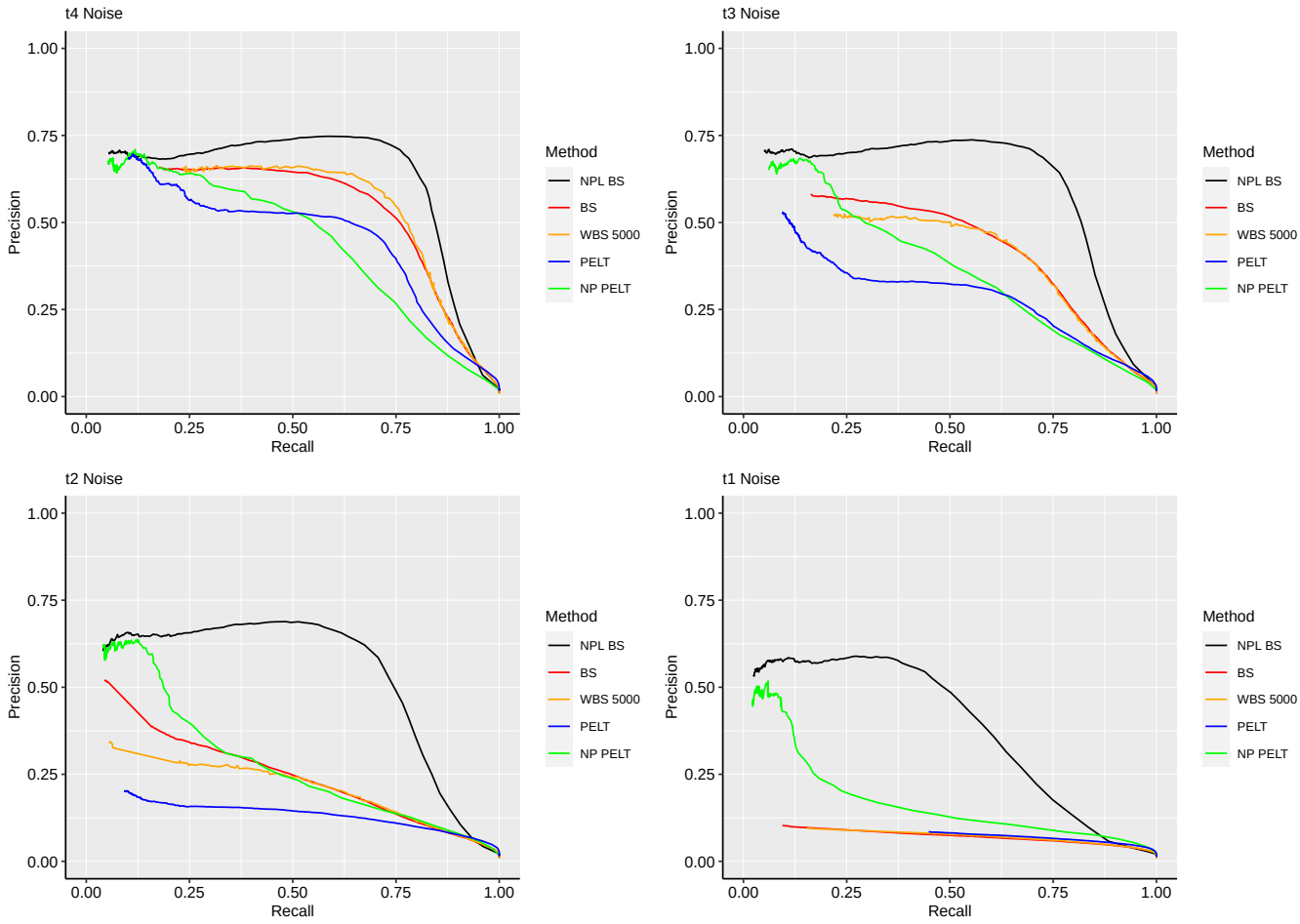


Figure 7.2: ED-PELT was called with the 'cpt.np' method of the changepoint.np package. BS and WBS were called with the 'sbs' and 'wbs' methods of the WBS package. PELT was called from the 'cpt.mean' method of the changepoint. The values displayed are averages of Precision and Recall over 100 replications while varying the penalty parameters.

First, the most apparent points are that NPL-BS is the best performing while PELT is the worst performing across all cases presented. A particularly interesting point is that in the noise processes with finite variance, t_4 and t_3, both SBS and WBS outperform ED-PELT. The cost function proposed within ED-PELT often leads to a performance benefit to that of PELT while misspecified. However, its absolute performance is still undesirable. In particular it is the idea that one need only consider $\mathcal{O}(\log(n))$ points biased toward the extremes of the dataset to compute the likelihood. Where the distribution is heavy-tailed, focusing on the extreme deviations merely inadequately represents each segment. In the Gaussian cases presented previously, this can be acceptable.

ED-PELT, despite being non-parametric, therefore still has this strong dependence on the underlying distribution. This is an unintended consequence of the author's approach to improving computational time. By analytically manipulating the cost function, this thesis has presented an alternative approach with the same computational expense while negating this dependence. At present, the author believes this is among the most effective approaches to detecting change in heavy noise processes available.

7.2.4 Timing Benchmark

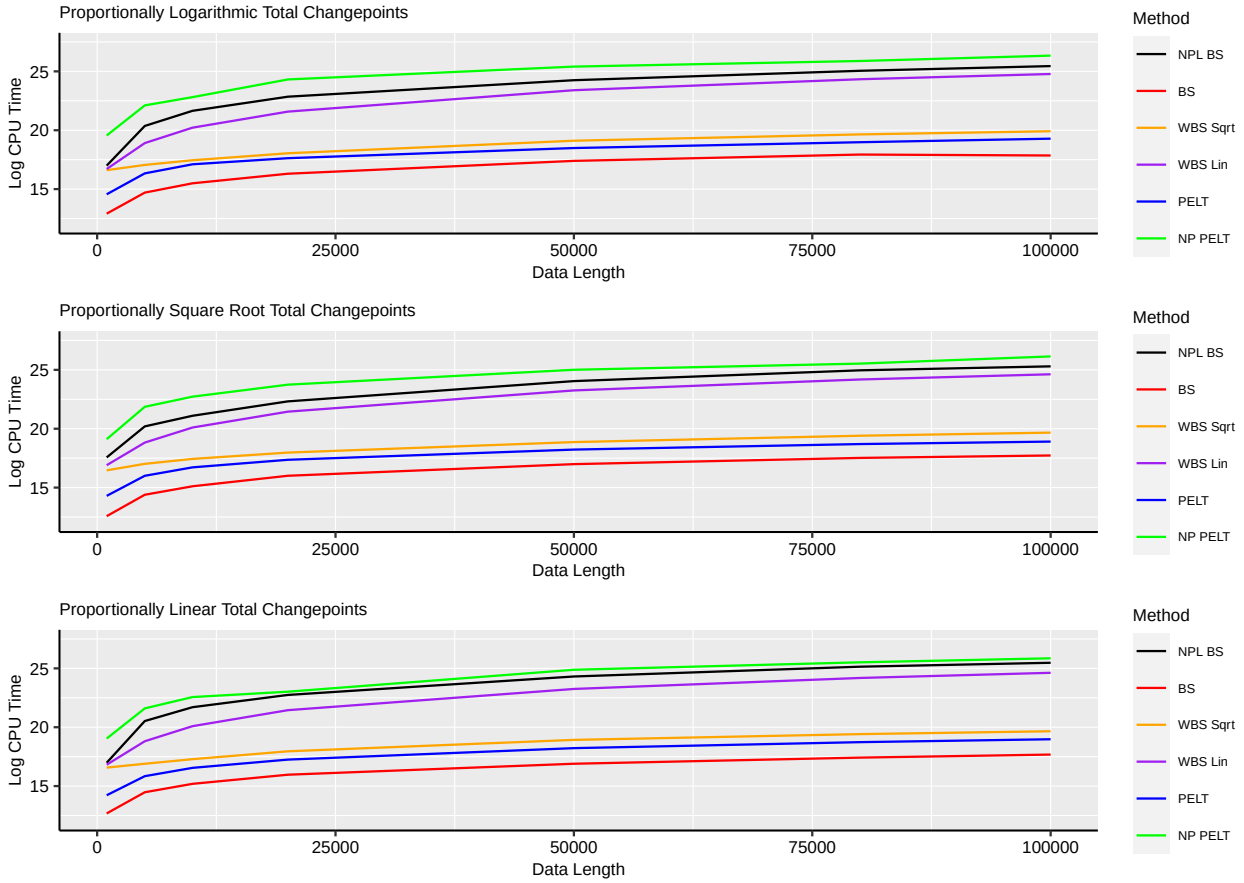


Figure 7.3: ED-PELT was called with the 'cpt.np' method of the changepoint.np package. BS was called with method 'sbs', where WBS Lin and WBS Sqrt were called with method 'wbs' with $M = n$ and $M = \sqrt{n}$ of the WBS package. PELT was called with the 'cpt.mean' method of the changepoint. The values displayed are the median runtime over 100 replications as recorded by the package microbenchmark [68].

The relative difference between NPL-BS and ED-PELT is comparable to that of BS and PELT. Note that NPL-BS has a cost function with a sum over n terms rather than $\log(n)$. Both NPL-BS and ED-PELT are written in C++ or C and so this does not impact the comparison. Instead, the differences are attributable to the analytical results provided within Chapter 6 addressing the implementation of ED-PELT. Further, note that this does not consider the need to tune the extra parameter of ED-PELT which would also multiply the compute time by a potentially sizeable constant factor. Considering also the previously demonstrated improvements to classification performance, the author believes NBL-BS should supersede ED-PELT.

Note, NPL-BS is still meaningfully slower than parametric methods. It behaves analogously to WBS with a parameter of $O(n)$. There should be good reason for one to incur this level of computational expense. The only other non-parametric R implementations include those within ECP [69] which are designed for multivariate data. These are usable in the univariate case. However, the mathematics presented in Section 6.4 exploited properties of $\mathbb{R}^1$ which are not available to those considering $\mathbb{R}^n$. These are consequently less efficient than that of ED-PELT. Detecting changes in noise processes which one otherwise could not as efficiently as possible is therefore such a good reason. However, the compute time is still a notable downside to the approach to the extent that it is restrictive practically. For reference, on a desktop with a 3.20GHz quad-core processor and 8GB of RAM, running NPL-BS took 138s on a dataset of length $n = 10^6$ with 10^3 evenly spaced changepoints. SBS took 2.4s, PELT took 3.7s, and WBS with $M = \sqrt{n}$ took 5.1s, whereas ED-PELT took 191s. NPL-BS is useful for datasets of moderate size and this represents progress. However, further efficiency improvements should be the priority for future research and this is explored in Chapter 8.

Chapter 8

Conclusion

The first aim of this chapter is to summarise the main experimental findings and concisely articulate an approach to algorithm selection. Secondly, it is to review the progress made toward improving the robustness and efficiency of non-parametric methods. In closing, it is noted that much of the work completed within Section 6.4 is readily generalisable and an outline of an extension is provided. This is intentionally light on details. The reader familiar with Chapter 6 should nonetheless grasp the main points and appreciate its promise.

8.1 Final Review

Binary Segmentation is the earliest method introduced. It employs a simple greedy heuristic making it somewhat unassuming. Nonetheless, it is highly effective and should be the foundation of a practitioner's toolkit. While it is a good starting point to any problem, it certainly has its weaknesses and therefore is not the only tool one should be aware of. A variety of approaches are subsequently introduced, all of which address a particular weakness by introducing a different tradeoff. First are Wild Binary Segmentation and PELT, which reduce segment length sensitivity by increasing misspecification sensitivity and also increasing time complexity.

Subject to correct specification, Wild Binary Segmentation with a modest parameter value, such as $\mathcal{O}(\sqrt{n})$, achieves classification performance equal to that of any approach reviewed across a broad range of scenarios. Given that lower parameter values have superior robustness and efficiency relative to both higher parameters and PELT, this approach offers a particularly attractive base choice. There are however cases where this last statement does not hold, and often these are associated with consistently short segments rather than those interspersed among longer ones. One approach to negate this is to increase the parameter up to values comparable to $\mathcal{O}(n)$ for instance. The other is to use PELT. From the perspective of time complexity, the circumstances described present ideal conditions for the manner in which PELT linearly traverses the dataset. It is considerably more efficient and moreover differences in robustness are negated by high parameter values. Under these circumstances, this is the approach recommended.

Finally, when deviating from normal noise, NPL-BS has been demonstrated to perform uniformly better than those within the literature. Were the data of modest size, this is recommended. The key drawback, however, is its time complexity. This becomes noticeable on examples such as that considered in Section 7.2.4, where it took minutes to run on a datasets of length $n = 10^6$ rather than seconds for parametric approaches. Were the compute time intolerable and misspecification modest, the classical variant of Binary Segmentation offers a reasonable alternative.

8.2 A Review of The Non-Parametric Likelihood

8.2.1 Classification Performance

There is a power cost to using a non-parametric cost function which is inherent to making no assumptions. Based on the simulation results in Section 7.2.1, it is not apparent that the construction fails in this regard. Moreover, in the presence of heavy noise processes it clearly outperforms the state of the art. Largely this is attributed to the different focus of Binary Segmentation and PELT, and the undesirable algorithm design of ED-PELT. From that perspective, the project has been successful. In the short-term, the author believes the algorithm presented is the most viable non-parametric approach available. However, it is also expected that this will be transient. Instead, this thesis should be viewed analogously to Optimal Partitioning. The contributions made in that paper were the concise presentation of simple but powerful ideas which were

readily expanded upon.

Moving forward, it is certainly plausible that this cost function could be improved upon. Equally, no obvious means of doing so has arisen directly from the results of this work. It is hard to credibly speak of the merits of this approach without conducting much broader benchmarks. The limiting factor in this regard is the relative immaturity of non-parametric change detection. There are dozens of alternative non-parametric cost functions which one could implement and this would make for informative comparison. Were the non-parametric likelihood found to be consistently among the most powerful, there are equally alternative means of estimating the KL-Divergence. Of foremost importance therefore is simply the need for further exploration. Again, owing to the vast unchartered territory, the author would expect this to be rather productive.

8.2.2 Efficiency

A distinct disadvantage is the time complexity of the approach. The steps taken throughout this thesis have reduced the computational burden beyond the non-parametric approaches it has been benchmarked against. Nonetheless, the approach is still orders of magnitude slower than parametric methods. The choice regarding whether to use this approach will often be contingent upon asking whether the additional steps required to transform the raw data onto order statistics and then manipulate the likelihood is worth the improved robustness.

This issue regarding compute time is structural. For a contrasting example consider the CUSUM. This consists of summing the values observed within the segment directly. As a result, one can compute partial sums over the dataset upfront and then adjust the cost function from one changepoint to the next with simple $O(1)$ operations. Contrastingly, there are two points which are burdensome for the time complexity of NPL-BS. The first is that one needs to continually update the ranks over each segment as it reduces the scope for memoization. The approach to the CUSUM involving an $\mathcal{O}(n)$ initialisation and $\mathcal{O}(1)$ updates throughout would be redundant if X_i changed each time a new changepoint was considered. However, the insights presented in Section 6.4 involving storing global ranks and using a Binary Search Tree reduced this to requiring a limited number of $\mathcal{O}(\log(n))$ operations.

The most fundamental complexity constraint arises with the cost updates. There is a need to complete an $\mathcal{O}(n)$ sum upon every single evaluation of the cost function. Now note that an elegant use of another data structure, known as the Fenwick Tree [70], can in fact allow one to compute partial sums in $\mathcal{O}(\log(n))$ time. Often this is highly useful. The primary reason it is not in this case is because the updates involve sum functions of $(k+1)\log(k+1) - k\log(k)$ for particular values of k which are data dependent and not readily stored.

8.3 Future Work

8.3.1 Non-Parametric Binary Segmentation

Were the author to continue research, the first priority would be to reduce the computational cost of non-parametric approaches further. Regarding the cost function presented, the most obvious means of improving efficiency is to use less terms within the sum. One could question whether it is strictly necessary to perform a likelihood ratio test over every point within the interval. This is the line of thinking taken by the author's in ED-PELT and a valid question. In the pseudocode presented, each branch stored in a tree corresponds to one likelihood ratio test hence the need to iterate over the tree. One could not include particular branches of the tree within the final calculation. The main issue then is which terms to take. The cost function in this formulation has noticeably superior classification power beyond that of ED-PELT and therefore it is not quite as simple as taking points within the tails. Perhaps an interesting approach would be to sample points randomly, drawing further inspiration from Wild Binary Segmentation.

Generally, the author would instead suggest it may be preferable to consider an alternative cost function. The description of the Binary Search Tree data structure given within Chapter 6 is the most basic possible and

does not do full justice to the utility of tree based data structures in general. There are several structures which could enhance efficiency were one instead using a polynomial based cost function such as the two sample Cramér-von-Mises

$$C(x_{1:t_i-1}, x_{t_i:n}) = t_i \sum_{i=1}^{t_i} (r_i - i)^2 + (n - t_i) \sum_{i=t_i}^{n} (r_i - (i - t_i))^2$$

where r_i represents the global rank of x_i. Section 6.4 can still serve as a step-by-step guide to computing this efficiently. Removing the data independent terms and introducing memoization is straightforward. Of greatest importance are the cost update rules. To that ends, first note that the global ranks r_i are independent of the changepoint considered. The dynamic element is the i term, corresponding to the within-segment ranks, which are incremented by positive or negative 1 depending on whether the point falls prior to or post the point inserted or deleted. For illustrative purposes consider just one of these updates,

$$\sum_{i=k}^{t_i} (r_i - (i + 1))^2$$

the simple polynomial form allows for a binomial expansion to

$$\sum_{i=k}^{t_i} r_i^2 + (i + 1)^2 - 2ir_i - 2r_i.$$

Note then that the terms r_i^2 and $2ir_i$ are data independent and differ only due to the insertion or deletion, whereas the adjustment to $(i + 1)^2$ is a function of the rank of the newest point. Of most significance to the update is the inclusion of

$$\sum_{i=k}^{t_i} 2r_i.$$

Recall now that the approach presented in Chapter 6 involves maintaining a Binary Search Tree including all r_i terms for a given segment. The distinct appeal of this approach is that one could similarly use a Fenwick Tree to compute this sum in only $\mathcal{O}(\log(n))$ time.

There is, however, one further issue to contend with. Previously there was never a need to compute the rank of any point within its own segment by anything other than incrementing a count while iterating. The cumbersome $\mathcal{O}(n)$ cost updates are convenient in this regard. Given that this is what one is trying to remove, an alternative is required. Fortunately, another tree based data structure, namely the Order Statistic Tree [34], also allows this to be computed readily in $\mathcal{O}(\log(n))$ time. It is also possible for a Fenwick Tree to be augmented with the methods of an Order Statistic Tree such that only one need to be maintained.

In total, this outline would give rise to a non-parametric variant of Binary Segmentation with time complexity $\mathcal{O}(n(\log(n))^2)$, a notable reduction from that introduced in Chapter 6. In the author's view, the most exciting aspect of this work is instead that it provides a clear framework to develop generalisations such as this. There is no reason to expect that polynomial cost functions could not be equally as effective as the non-parametric likelihood. Further efficiency improvements with minimal power loss foreshadow the next frontiers of non-parametric change detection. Were the research period extended, the author would be highly interested in adjusting the software developed to trial this.

8.3.2 Alternative Penalties

A secondary priority for the author would be to investigate alternative penalties. When model selection criteria were first introduced in Section 3.2.1, it was by way of comparison to cross-validation with commentary regarding the differences between supervised and unsupervised learning. It was highlighted that unsupervised learning problems pose an additional challenge in that one has no means by which to judge the effectiveness of a given method when deployed practically. Similarly, it was noted that one benefit of

cross-validation is that it provides an estimate of performance with respect to a given metric.

While accepting the inevitable limitations that arise in a stochastic problem, framing this problem in a multiple testing framework also has such benefits. By no means would this guarantee superior classification tradeoffs beyond using simple information criteria. One useful aspect, however, is where information criteria prove inadequate. Tuning thereafter becomes cumbersome. One option is to gather alternate information criteria and try several. Another would be to perform a grid search over a somewhat arbitrary range and consider a heuristic such as the 'elbow method' [71]. Neither of these are particularly elegant. On the other hand, in the multiple testing framework one can tune by reference to an estimated performance metric in the form of a theoretical false positive rate. The reader can find an interesting example of such an approach to the changepoint problem in [72], which introduces Simultaneous Multiscale Changepoint Inference (SMUCE).

For the non-parametric likelihood the distribution of interest is

$$A = \int_0^1 \frac{B^2(t)}{t(1-t)} dt$$

where $B(t)$ represents Brownian Motion. While the upsides to multiple testing procedures have been described, there is a notable downside. Determining a p-value associated with a given value in such a distribution is a complicated statistical problem in and of itself. The cumulative distribution function of this statistic takes the form

$$Pr(A \leq z) = \frac{\sqrt{2\pi}}{z} \sum_{j=1}^{\infty} \binom{\frac{-1}{2}}{j} (4j+1) e^{-(4j+1)^2 \frac{\pi^2}{8z}} \int_0^{\infty} e^{-(4j+1)^2 \frac{\pi^2}{8z}}$$

and evaluating this requires a numerical procedure [73]. Given that one would only be making a limited number of evaluations of this distribution function, it is possible to take such an approach. It is noted that within this last reference the authors explicitly state that the time taken to perform the Monte-Carlo simulations required to evaluate another Brownian Motion related distribution can exceed the run-time of the remaining algorithm. Future research could look to strike a compromise between these two approaches by pursuing a stochastic bound. Upon a further look at the test statistic, one can envisage a bound involving

$$\frac{j(n-j)}{n} |\hat{F}_i^{L_j} - \hat{F}_i^{R_j}|_{\infty}^2 \quad \text{and} \quad \sum_{k=1}^{n-1} \frac{1}{k},$$

the discrete time analogues of

$$K^2 = \sup_{t \in [0,1]} (B(t)^2) \quad \text{and} \quad \int \frac{1}{t}$$

respectively. The summation can be bounded by $\log(n)$ which arises when considering information criteria and would therefore attach a probability interpretation to these. The term involving K^2 represents the Kolmogorov Distribution and has a much simpler series representation of [52]

$$Pr(K \leq z) = 1 - 2\sum_{k=1}^{\infty} (-1)^{k-1} e^{-2k^2 z^2} \approx 1 - 2e^{-2z^2} + 2e^{-8z^2}.$$

One could then readily tune the penalty by scaling an information criteria by a desired constant. The core reason that pursuing such a bound is a secondary objective is that it is cost function dependent, and so one would want to establish the merits of the choice above any alternatives before doing so. While this discussion has focused on the non-parametric likelihood, recall that in Chapter 4 it was noted that non-parametric statistics generally involve applying a functional norm to estimated distribution functions and similar relationships bounding the distance between points in alternate spaces hold. In the case of the non-parametric likelihood, the asymptotic result for the distribution involved a second order series expansion and this also suggests an approach to obtaining such a bound. The curious reader can find a partial solution to this applying the Mean Value Theorem and a convexity argument to bound the second derivative within Appendix B.

Appendix A

Additional Simulation Results

A.1 Aggregated F1 Scores

Considered within this section is an extension of the mean only changes presented within the main text. Changepoints t_i are generated by first fixing $n = 5000$ and λ, then sampling from $S_i \sim Po(\lambda)$ and setting $t_i = \sum_{j=1}^{i} S_i$ until $t_{i+1} > n$. $X_t = \mu_t + \epsilon_t$, where ϵ_t is a stochastic quantity drawn from the noise process stated. The quantity μ_t is piecewise constant, initialised at $\mu_0 = 0$, and updated by $\mu_{t_i} = \mu_{t_i - 1} + \mu D$ where D is uniformly sampled from $\{-1, 1\}$. The parameter λ is varied across $\{1000, 666, 400, 200, 100, 50\}$, while the parameter μ is varied across $\{0.25, 0.5, 0.75, 1, 1.25, 1.5, 1.75, 2\}$. For each dataset, the F_1 score is computed over a range of penalties and the maximal found. The average of this over 100 replications is considered as the Monte Carlo estimate of $\max F_1$ for a given λ and μ. The results presented in the table are the average $\max F_1$ over all pairs of λ and μ.

Abbreviation	$N(0,1)$	t_4	t_3	t_2	t_1
PELT	0.80	0.63	0.48	0.21	0.09
BinSeg	0.77	0.66	0.60	0.43	0.21
Wild BinSeg, M = 250	0.78	0.66	0.57	0.38	0.18
Wild BinSeg, M = 5000	0.79	0.65	0.54	0.26	0.11
NPL BinSeg	0.76	0.70	0.67	0.64	0.54
ED-PELT	0.76	0.66	0.59	0.47	0.35

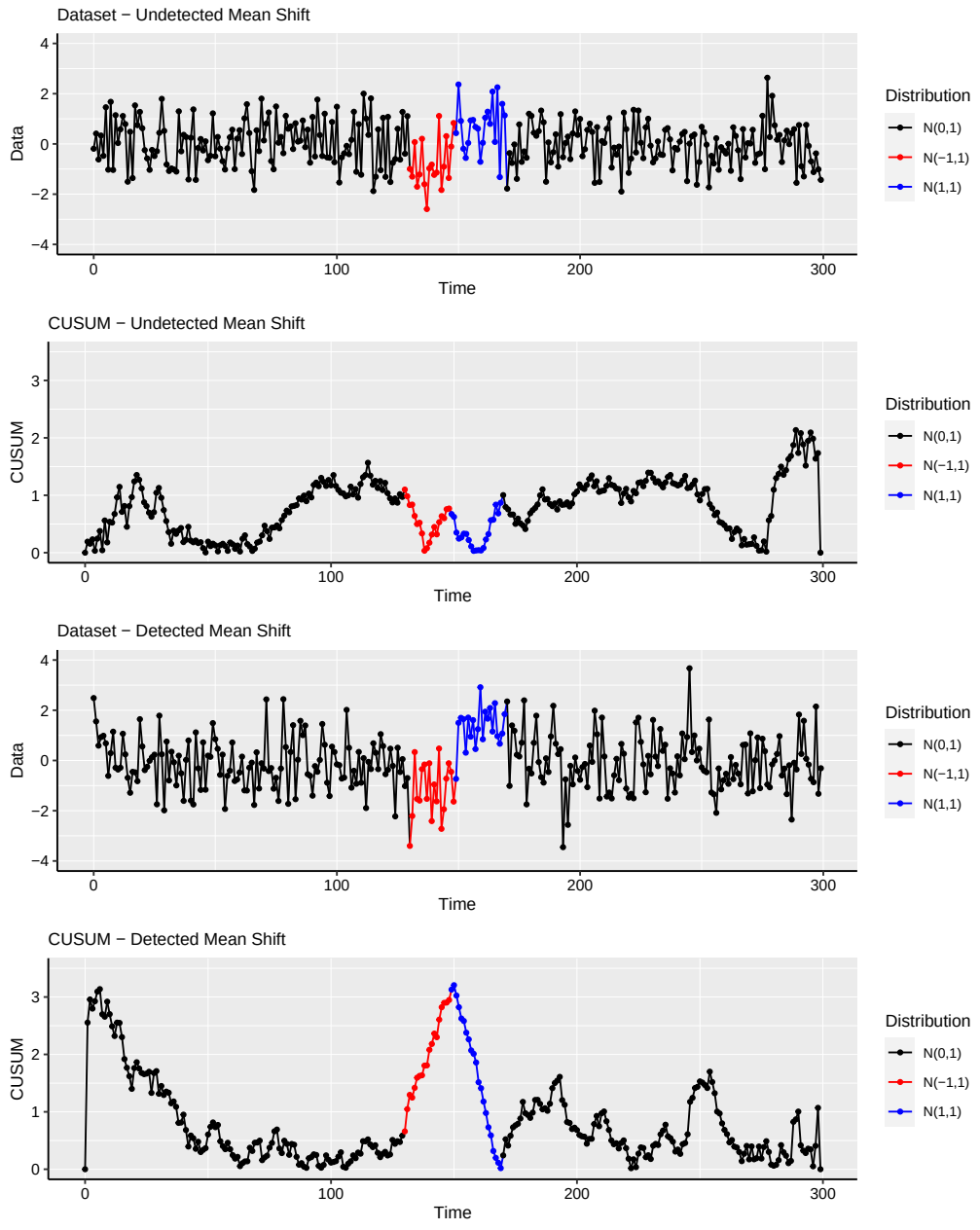
In general, the results presented above should be recognised for both their merits and flaws. This approach has allowed for the consideration of vastly more experimental parameters than what one could otherwise have considered through Precision-Recall curves. That particular trends are still evident presents strong evidence in favour of the generalisability of that which has been discussed throughout. In particular, note that the relatively weaker robustness of PELT and WBS with higher parameter values is evident. Also, the relatively strong performance of WBS with a low parameter and non-parametric methods in that the misspecification cost typically outweighs that of not pinpointing the precise best algorithm for a given situation.

Equally, there are also limitations to evaluating performance in this manner. Most notably, rigidly interpreting the values presented can be misleading in practice given that one does not have the benefit of tuning the hyperparameter in this supervised manner. In general, focusing on evaluation metrics based on a single penalty is advised against for this reason. It should be accepted that information criteria are fallible and so when comparing methods one wants insights which hold across a range of penalties as these can be relied on practically. Additionally, in averaging too broadly one considers performance evaluated in settings which may not represent those of the given context the algorithm is being applied. Given that there can be significant variance across parameters, those with an appreciation for the structure of the particular dataset may be better served by noting those features in common with situations studied in the Precision-Recall curves. Finally, it should be noted that implicit within the F_1 score is an assumption that both types of misclassification costs are equal when often in practice this is not so.

On the whole, the results have been presented within the Appendix specifically as a result of this. It is intended that these be treated as something of interest which does present some insight but equally which should not be of foremost consideration.

A.2 CUSUM Masking

This is the motivation for the development of Wild Binary Segmentation. It illustrates a false negative which arises as a weakness of the assumption of searching for one changepoint at a time. It is noted that this is particularly sensitive to the data generating parameters as the accuracy rate rapidly increases as one considers even marginally longer segment lengths.



Segment Length	10	15	20	25	30	35	40
CUSUM Accuracy Rate	25.0 %	45.4 %	65.2 %	80.1 %	88.5 %	94.4 %	97.8 %

Figure A.1: Replicating Figure 1 of the paper [31]. 130 points are drawn from N(0,1), 20 from N(-1,1), 20 from N(1,1), and 130 from N(0,1). The statistics displayed in the table record the proportion of times the maximum was reached within a window of 5 either side of the changepoint over 500 repetitions as the length of the small segment is varied above and below 20.

A.3 Non-Parametric Cost Functions

Name	Abbreviation	R Package	References
Cumulative Sum	CUSUM	WBS	[31]
Unweighted Non-Parametric Likelihood	UNPL	Chapter 6*	-
Weighted Non-Parametric Likelihood	WNPL	Changepoint.np, NMCD	[50], [51]
Wasserstein Distance	WD	-	[74]
Energy Distance	ED	ECP	[69]
Epps-Singleton	ES	-	[75]
Kolmogorov-Smirnov	KS	ECP	[69]

Note that within the following experiments, all of WD, ED, ES, KS were SciPy implementations. The CUSUM, UNPL, WNPL were the author's personal implementations.

Abbreviation	$N(0,1)$	t_4	t_3	t_2	t_1
CUSUM	61.5 %	38.8 %	31.6 %	19.6 %	8.0 %
UNPL	57.5 %	46.2 %	43.8 %	39.4 %	27.8 %
WNPL	50.5 %	41.2 %	38.0 %	33.8 %	22.6 %
WD	21.5 %	11.8 %	7.2 %	6.5 %	3.8 %
ED	19.5 %	12.8 %	8.8 %	6.5 %	2.2 %
ES	13.5 %	8.2 %	7.8 %	6.0 %	3.0 %
KS	11.5 %	8.0 %	7.5 %	5.6 %	4.6 %

Figure A.2: 300 points were generated from the specified noise process. The data undergoes a mean change of 0.5 at t = 250. The statistics displayed in the table record the proportion of times the maximum of the specified statistic was reached within a window of 5 either side of the changepoint over 500 repetitions.

Abbreviation	$N(0,1)$ to $N(0.25,1)$	$N(0,1)$ to $N(0.25,0.5)$	$N(0,1)$ to $N(0.25,0.25)$
CUSUM	30.8 %	30.4 %	30.0 %
UNPL	29.6 %	52.4 %	74.8 %
WNPL	30.0 %	35.6 %	52.4 %
WD	1.2 %	1.6 %	0.8 %
ED	0.8 %	1.6 %	2.8 %
ES	0.8 %	10.4 %	28.0 %
KS	0.0 %	0.4 %	6.0 %

Figure A.3: 300 points were generated from the specified noise process. The data undergoes the changes at t = 250. The statistics displayed in the table record the proportion of times the maximum of the specified statistic was reached within a window of 5 either side of the changepoint over 500 repetitions.

A problem with many non-parametric cost functions in their original formulation is that they exhibit undesirable behaviour of growing large toward the end of an interval. In essence, the small sample properties of such test statistics often are a notable limitation. This does not preclude such statistics from being useful. Instead one can perform modifications to the statistics before applying them to the changepoint problem. Another approach one could take is to incorporate a minimum segment size. The ECP package implementing the KS-test and Energy Distance has set a minimum segment size by default.

Appendix B

Non-Parametric Likelihood Derivations

B.1 NMCD Bound on NPL

First note that the likelihood function can be written as

$$\sum_{k=t_i}^{t_{i+1}-2}\left[\frac{jg(\hat{F}_i^{L_j}(x_k))+(n_i-j)g(\hat{F}_i^{R_j}(x_k))}{n_i}-g(\hat{F}_i(x_k))\right]$$

where the function $g:[0,1]\mapsto\mathbb{R}$ is $g(x)=x\log(x)+(1-x)\log(1-x)$. Recall the identity

$$\hat{F}_i(x_k)=\frac{j\hat{F}_i^{L_j}(x_k)+(n_i-j)\hat{F}_i^{R_j}(x_k)}{n_i}$$

so the cost function is also

$$\sum_{k=t_i}^{t_{i+1}-2}\left[\frac{jg(\hat{F}_i^{L_j}(x_k))+(n_i-j)g(\hat{F}_i^{R_j}(x_k))}{n_i}-g\left(\frac{j\hat{F}_i^{L_j}(x_k)+(n_i-j)\hat{F}_i^{R_j}(x_k)}{n_i}\right)\right].$$

As g is convex, Jensen's inequality states that

$$\frac{jg(\hat{F}_i^{L_j}(x_k))+(n_i-j)g(\hat{F}_i^{R_j}(x_k))}{n_i})-g\left(\frac{j\hat{F}_i^{L_j}(x_k)+(n_i-j)\hat{F}_i^{R_j}(x_k)}{n_i}\right)\geq 0.$$

The positivity of the summand justifies the following quantity as a bound on the NMCD statistic

$$\left(\sum_{k=1}^{n_i-1}\frac{n_i}{k(n_i-k)}\right)\left(\sum_{k=t_i}^{t_{i+1}-2}\left[L(x_{t_i:j-1}\mid x_{(k)})+L(x_{j:t_{i+1}-1}\mid x_{(k)})-L(x_{t_i:t_{i+1}-1}\mid x_{(k)})\right]\right).$$

If one then notes that

$$\sum_{k=1}^{n_i-1}\frac{n_i}{k(n_i-k)}=\sum_{k=1}^{n_i-1}\frac{1}{k}+\frac{1}{(n_i-k)}$$

and that a well-known result regarding the harmonic series is its logarithmic growth

$$\sum_{k=1}^{n_i-1}\frac{1}{k}\leq\log(n_i-1)+1,$$

a further bound on the NMCD statistic is

$$2(\log(n_i-1)+1)\sum_{k=t_i}^{t_{i+1}-2}\left[L(x_{t_i:j-1}\mid x_{(k)})+L(x_{j:t_{i+1}-1}\mid x_{(k)})-L(x_{t_i:t_{i+1}-1}\mid x_{(k)})\right].$$

The truth of the NMCD decision rule

$$\sum_{k=t_i}^{t_{i+1}-2}\left[\frac{n_i}{k(n_i-k)}L(x_{t_i:j-1}\mid x_{(k)})+L(x_{j:t_{i+1}-1}\mid x_{(k)})-L(x_{t_i:t_{i+1}-1}\mid x_{(k)})\right]>\frac{1}{2}(\log(n_i))^2,$$

would therefore imply that the statistic used in this formulation satisfies

$$\frac{1}{n_i}\sum_{k=t_i}^{t_{i+1}-2}\left[L(x_{t_i:j-1}\mid x_{(k)})+L(x_{j:t_{i+1}-1}\mid x_{(k)})-L(x_{t_i:t_{i+1}-1}\mid x_{(k)})\right]>\frac{(\log(n_i))^2}{4n_i(\log(n_i-1)+1)}.$$

B.2 Data Independent Terms

B.2.1 Full Segment Cost

First note that

$$-\sum_{k=t_i}^{t_{i+1}-2}L(x_{t_i:t_{i+1}-1}\mid x_k)$$

is equal to

$$-\sum_{k=1}^{n_i-1}\left[k\log\left(\frac{k}{n_i}\right)+(n_i-k)\log\left(\frac{n_i-k}{n_i}\right)\right].$$

By the property of logarithms this is

$$-\sum_{k=1}^{n_i-1}\left[k\log(k)+(n_i-k)\log(n_i-k)-(k+(n_i-k))\log(n_i)\right]$$

which is

$$(n_i-1)n_i\log(n_i)-\sum_{k=1}^{n_i-1}\left[k\log(k)+(n_i-k)\log(n_i-k)\right]$$

or

$$(n_i-1)G(n_i)-2S(n_i-1).$$

B.2.2 Denominators

Note that

$$L(x_{t_i:j}\mid x_k)=(j+1-t_i)[\hat{F}^L_{ij}(x_k)\log\hat{F}^L_{ij}(x_k)+(1-\hat{F}^L_{ij}(x_k))\log(1-\hat{F}^L_{ij}(x_k))]$$

is equal to

$$R_{L_j}(x_k)\log\frac{R_{L_j}(x_k)}{j+1-t_i}+(j+1-t_i-R_{L_j}(x_k))\log\frac{j+1-t_i-R_{L_j}(x_k)}{j+1-t_i}$$

and equivalently

$$L(x_{t_i:j}\mid x_k)=(t_{i+1}-j)[\hat{F}^R_{ij}(x_k)\log\hat{F}^R_{ij}(x_k)+(1-\hat{F}^R_{ij}(x_k))\log(1-\hat{F}^R_{ij}(x_k))]$$

is

$$R_{R_j}(x_k)\log\frac{R_{R_j}(x_k)}{t_{i+1}-j}+(t_{i+1}-j-R_{R_j}(x_k))\log\frac{t_{i+1}-j-R_{R_j}(x_k)}{t_{i+1}-j}$$

By using the same properties of logarithms as before it follows that these can equally be written as

$$R_{L_j}(x_k)\log(R_{L_j}(x_k))+(j+1-t_i-R_{L_j}(x_k))\log(j+1-t_i-R_{L_j}(x_k))-G(j+1-t_i)$$

and

$$R_{R_j}(x_k)\log(R_{R_j}(x_k))+(t_{i+1}-j-R_{R_j}(x_k))\log(t_{i+1}-j-R_{R_j}(x_k))-G(t_{i+1}-j).$$

The corresponding sums within the total cost function can therefore be written as

$$-(n_i-1)(G(j)+G(n_i-j))+\sum_{k=1}^{n_i-1}H\left(R_{L_j}(x_k)),j\right)+H\left(R_{R_j}(x_k)),n_i-j\right).$$

B.3 Cost Function Update Rules

B.3.1 Left Cross-Segment Updates

The cross-segment term associated with the left segment is

$$\sum_{k=t_i}^{j+1} H\left(R(x_k) - R_{L_{j+1}}(x_k), n_i - (j+1)\right).$$

As $x_k < x_{j+1}$ implies $R_{L_{j+1}}(x_k) = R_{L_j}(x_k)$, one has that the first summation is equal to

$$\sum_{k=t_i, x_k < x_{j+1}}^{j+1} H\left(R(x_k) - R_{L_j}(x_k), n_i - (j+1)\right)$$

which is

$$\sum_{k=t_i, x_k < x_{j+1}}^{j+1} G\left(R(x_k) - R_{L_j}(x_k)\right) + G\left(n_i - (j+1) - (R(x_k) - R_{L_j}(x_k))\right).$$

This is also

$$\sum_{k=t_i, x_k < x_{j+1}}^{j+1} H\left(R(x_k) - R_{L_j}(x_k), n_i - j\right) - G\left(n_i - j - (R(x_k) - R_{L_j}(x_k))\right) + G\left(n_i - (j+1) - (R(x_k) - R_{L_j}(x_k))\right),$$

or

$$\sum_{k=t_i, x_k < x_{j+1}}^{j+1} H\left(R(x_k) - R_{L_j}(x_k), n_i - j\right) - D\left(n_i - j - (R(x_k) - R_{L_{j+1}}(x_k))\right).$$

Now consider

$$\sum_{k=t_i, x_k \geq x_j}^{j+1} H\left(R(x_k) - R_{L_{j+1}}(x_k), n_i - (j+1)\right),$$

one can see that a similar relationship holds. This time note that $x_k \geq x_j$ implies that $R_{L_{j+1}}(x_k) = R_{Lj}(x_k) + 1$.

Now, the second term of the summation can be written as

$$\sum_{k=t_i, x_k \geq x_j}^{j+1} H\left(R(x_k) - (R_{L_j}(x_k) + 1), n_i - (j+1)\right)$$

which is

$$\sum_{k=t_i, x_k \geq x_j}^{j+1} G\left(R(x_k) - (R_{L_j}(x_k) + 1)\right) + G\left(n_i - j - (R(x_k) - R_{L_j}(x_k))\right).$$

This is also

$$\sum_{k=t_i, x_k \geq x_j}^{j+1} H\left(R(x_k) - R_{L_j}(x_k), n_i - j\right) - G(R(x_k) - R_{L_j}(x_k)) + G(R(x_k) - (R_{L_j}(x_k) + 1))$$

or

$$\sum_{k=t_i, x_k \geq x_j}^{j+1} H\left(R(x_k) - R_{L_j}(x_k), n_i - j\right) - D\left(R(x_k) - (R_{L_{j+1}}(x_k) - 1)\right).$$

B.3.2 Right Cross-Segment Updates

The cost associated with the right segment is

$$\sum_{k=j+2}^{t_{i+1}-1} H\left(R(x_k) - R_{R_{j+1}}(x_k), j+1\right)$$

An identity is that for $x_k < x_{j+1}$, $R_{R_{j+1}}(x_k) = R_{R_j}(x_k)$. The first sum is therefore

$$\sum_{k=j+2, x_k<x_{j+1}}^{t_{i+1}-1} H\left(R(x_k) - R_{R_j}(x_k), j+1\right)$$

which is

$$\sum_{k=j+2, x_k<x_{j+1}}^{t_{i+1}-1} G\left(R(x_k) - R_{R_j}(x_k)\right) + G\left(j+1-(R(x_k) - R_{R_j}(x_k))\right).$$

This is also

$$\sum_{k=j+2, x_k<x_{j+1}}^{t_{i+1}-1} H\left(R(x_k) - R_{R_j}(x_k), j\right) + G(j+1-(R(x_k) - R_{R_j}(x_k))) - G(j-(R(x_k) - R_{R_j}(x_k)))$$

or

$$\sum_{k=j+2, x_k<x_j+1}^{t_{i+1}-1} H\left(R(x_k) - R_{R_j}(x_k), n_i - j\right) + D\left(j+1-(R(x_k) - R_{R_{j+1}}(x_k))\right).$$

The final cost term to consider is

$$\sum_{k=j+2, x_k\geq x_{j+1}}^{t_{i+1}-1} H\left(R(x_k) - R_{R_{j+1}}(x_k), j+1\right).$$

The relevant identity in this case is that $x_k \geq x_{j+1}$, $R_{R_{j+1}}(x_k) = R_{R_j}(x_k) - 1$. The above term is therefore

$$\sum_{k=j+2, x_k\geq x_{j+1}}^{t_{i+1}-1} H\left(R(x_k) - (R_{R_j}(x_k) - 1)), j+1\right).$$

which is

$$\sum_{k=j+2, x_k\geq x_{j+1}}^{t_{i+1}-1} G\left(R(x_k) - (R_{R_j}(x_k) - 1)\right) + G\left(j+1-(R(x_k) - (R_{R_j}(x_k) - 1))\right).$$

This is also

$$\sum_{k=j+2, x_k\geq x_{j+1}}^{t_{i+1}-1} H\left(R(x_k) - R_{R_j}(x_k), j\right) - G\left(R(x_k) - R_{R_j}(x_k)\right) + G\left(R(x_k) - (R_{R_j}(x_k) - 1)\right).$$

or

$$\sum_{k=j+2, x_k\geq x_{j+1}}^{t_{i+1}-1} H\left(R(x_k) - R_{R_j}(x_k), j\right) + D\left(R(x_k) - R_{R_{j+1}}(x_k)\right).$$

B.4 P-Value Thresholds for the Non-Parametric Likelihood

Recall, the statistic is

$$\frac{1}{n_i}\max_j\left[\sum_{k=t_i}^{t_{i+1}-2} L(x_{t_i:j}\mid x_k)+L(x_{j+1:t_{i+1}-1}\mid x_k)-L(x_{t_i:t_{i+1}-1}\mid x_k)\right]$$

where

$$L(x_{t_i:t_{i+1}-1}\mid s)=(t_{i+1}-t_i)[\hat{F}_i(s)\log\hat{F}_i(s)+(1-\hat{F}_i(s))\log(1-\hat{F}_i(s))].$$

If one introduces the definition of $g:[0,1]\mapsto\mathbb{R}$ to ease notation, where

$$g(x)=x\log(x)+(1-x)\log(1-x),$$

then the statistic can be viewed as

$$\frac{1}{n_i}\max_j\left[\sum_{k=t_i}^{t_{i+1}-2} j\left(g(\hat{F}_i^{L_j}(X_k))-g(\hat{F}_i(X_k))\right)+(n_i-j)\left(g(\hat{F}_i^{R_j}(X_k))-g(\hat{F}_i(X_k))\right)\right].$$

As g is continuous over $[0,1]$ and differentiable over $(0,1)$ it satisfies the Mean Value Property, that for any $(a,b)\in[0,1]$ one has that there exists a $c\in(a,b)$ such that

$$g(b)-g(a)=g'(c)(b-a).$$

Applying this to each of the pairs

$$\left(\min(\hat{F}_i^{L_j}(X_k)),\hat{F}_i(X_k),\max(\hat{F}_i^{L_j}(X_k),\hat{F}_i(X_k))\right)$$

and

$$\left(\min(\hat{F}_i^{R_j}(X_k),\hat{F}_i(X_k),\max(\hat{F}_i^{R_j}(X_k)),\hat{F}_i(X_k))\right)$$

for every $k\in 1,2,...,n$ one therefore has the statistic is equally

$$\frac{1}{n_i}\max_j\left[\sum_{k=t_i}^{t_{i+1}-2} j\left(g'(C_{L_k})|\hat{F}_i^{L_j}(X_k)-\hat{F}_i(X_k)|\right)+(n_i-j)\left(g'(C_{R_k})|\hat{F}_i^{R_j}(X_k))-\hat{F}_i(X_k)|\right)\right]$$

for some C_{L_k} and C_{R_k} between each of the respective pairs.

If one notes that

$$\hat{F}_i(x)=\frac{1}{n_i}\left[j\hat{F}_i^{L_j}(x)+(n_i-j)\hat{F}_i^{R_j}(x)\right]$$

the statistic is also

$$\max_j\frac{j(n_i-j)}{n_i^2}\left[\sum_{k=t_i}^{t_{i+1}-2}|\hat{F}_i^{L_j}(X_k)-\hat{F}_i^{R_j}(X_k)|(g'(C_{L_k})+g'(C_{R_k}))\right]$$

and a further bound is

$$\max_j\frac{j(n_i-j)}{n_i^2}\sup_x|\hat{F}_i^{L_j}(X_k)-\hat{F}_i^{R_j}(X_k)|\left[\sum_{k=t_i}^{t_{i+1}-2} g'(C_{L_k})+g'(C_{R_k})\right].$$

Note that $g'(x) = \log(x) - \log(1-x)$ and if one denotes $m_1 = \max(C_{L_k}, C_{R_k})$ and $m_2 = \min(C_{L_k}, C_{R_k})$, the statistic is bounded by

$$\max_j \frac{j(n_i - j)}{n_i^2} \sup_x |\hat{F}_i^{L_j}(X_k) - \hat{F}_i^{R_j}(X_k)| \left[\sum_{k=t_i}^{t_{i+1}-2} \log(m_1) + \log(m_2) - \log(1-m_2) - \log(1-m_1) \right].$$

Then note that for a concave function such as $f(x) = \log(x)$ one has that

$$f(y) - f(x) \leq f'(x)(y-x)$$

so the statistic is dominated by

$$\max_j \frac{j(n_i - j)}{n_i^2} |\hat{F}_i^{L_j}(X_k) - \hat{F}_i^{R_j}(X_k)| \left[\sum_{k=t_i}^{t_{i+1}-2} \frac{m_1 - m_2}{m_2} + \frac{m_1 - m_2}{1-m_1} \right].$$

Now note that one could use the outer boundaries of the pairs to show $m_1 - m_2 < |\hat{F}_i^{L_j}(X_k) - \hat{F}_i^{R_j}(X_k)|$, which suggests a bound of

$$\max_j \frac{j(n_i - j)}{n_i^2} |\hat{F}_i^{L_j}(X_k) - \hat{F}_i^{R_j}(X_k)|^2 \left[\sum_{k=t_i}^{t_{i+1}-2} \frac{1}{m_2} + \frac{1}{1-m_1} \right].$$

The first term of this product involves K^2 where K is the Kolmogorov distribution and has the convenient series representation alluded to in the main text. The primary focus therefore is on finding a sensible bound for the sum. Note that m_1, m_2 depend on C_{L_k} and C_{R_k} with

$$C_{L_k} \in \left(\min(\hat{F}_i^{L_j}(X_k)), \hat{F}_i(X_k), \max(\hat{F}_i^{L_j}(X_k), \hat{F}_i(X_k)) \right)$$

and

$$C_{R_k} \in \left(\min(\hat{F}_i^{R_j}(X_k), \hat{F}_i(X_k), \max(\hat{F}_i^{R_j}(X_k)), \hat{F}_i(X_k)) \right).$$

The idea would be to use this to construct a bound involving the sums

$$\sum_{i=1}^{n-1} \frac{1}{\hat{F}_i(X_k)} = n \sum_{i=1}^{n-1} \frac{1}{i} \leq n \log(n),$$

$$\sum_{i=1}^{j-1} \frac{1}{\hat{F}_i^{R_j}(X_k)} = j \sum_{i=1}^{j-1} \frac{1}{i} \leq j \log(j),$$

and

$$\sum_{i=1}^{n-j-1} \frac{1}{\hat{F}_i^{L_j}(X_k)} = (n-j) \sum_{i=1}^{n-j-1} \frac{1}{i} \leq (n-j) \log(n-j).$$

Noting also that

$$\sum_{i=1}^{j-1} \frac{1}{\hat{F}_i^{R_j}(X_k)} + \sum_{i=1}^{n-j-1} \frac{1}{\hat{F}_i^{L_j}(X_k)} \leq n \log(n)$$

by the concavity of the logarithm. The factor of n leading $\log(n)$ may be of concern but note that the Kolmogorov distribution has a factor of $\frac{j(n-j)}{n}$ where the denominator of the above bound is n^2 and therefore this would cancel.

Bibliography

[1] T. Yokoyama, F. Miura, H. Araki, K. Okamura4, & T. Ito (2015), "Changepoint detection in base-resolution methylome data reveals a robust signature of methylated domain landscape", Springer Nature, BMC Genomics vol. 16, pp. 594 - 604

[2] J. Chen & Y. Wang (2009), "A Statistical Change Point Model Approach for the Detection of DNA Copy Number Variations in Array CGH Data", IEEE Transactions on Computational Biology and Bioinformatics, vol. 6 pp. 529 – 541

[3] J. F. Beausang, Y. E. Goldman, & P. C. Nelson (2011), "Changepoint Analysis for Single-Molecule Polarized Total Internal Reflection Fluorescence Microscopy Experiments", Elsevier, Methods in Enzymology, Computer Methods – Part C, vol. 487, pp. 431 – 463

[4] E. S. Silva (2015), "A short note on the accuracy of structural break tests at detecting oil price shocks in advance", International Journal of Energy and Statistics, vol. 3, 1550009

[5] K. R. Ryberga, G. A. Hodgkinsb, & R. W. Dudley (2020), "Change points in annual peak streamflows: Method comparisons and historical change points in the United States", Elsevier, Journal of Hydrology, vol. 583

[6] M. Broomfield, T. Buckland, L. Gaston, B. Holmes, G. Martin, K. Morrow, & K. Whelan (2019), "Onshore oil and gas monitoring: assessing the statistical significance of changes", HM Government Department for Environment, Food & Rural Affairs, Environment Agency, Project SC160020, Section 3.4, pp. 53-56

[7] R. Lykou, G. Tsaklidis, & E. Papadimitriou (2020), "Change point analysis on the Corinth Gulf (Greece) seismicity", Elsevier, Physic A, vol. 541, 123630

[8] B. T. Fagan, M. I. Knight, N. J. MacKay, A. J. Wood (2020), "Change point analysis of historical battle deaths", Journal of the Royal Statistical Society, vol. 183, pp. 909-933

[9] R. Amaya-Gomez, E. Bastidas-Arteaga, F. Schoef, F. Munoz, & M. Sanchez-Silva (2020), "A condition-based dynamic segmentation of large systems using a changepoints algorithm: A corroding pipeline case", Elsevier, Journal of Structural Safety, vol. 84, 101912

[10] D. M. Corbett, A. J. Sweeting, & S. Robertson (2019), "A change point approach to analysing the match activity profiles of team-sport athletes", Taylor & Francis Group, Journal of Sports Sciences, vol. 37, pp. 1600-1608

[11] Q. Yan, Z. Sun, Q. Gan, W. Jin (2018), "Automatic identification of near-stationary traffic states based on the PELT changepoint detection", Elsevier, Transportation Research Part B: Methodological, vol. 108, pp. 39-54

[12] M. Vangoli & R. Remenyte-Prescott (2018), "An ensemble-based change-point detection method for identifying unexpected behaviour of railway tunnel infrastructures", Elsevier, Tunnelling and Underground Space Technology, vol. 81, pp. 68-82

[13] A. Samaneh & D. Cook(2017) "A Survey of Methods for Time Series Change Point Detection" Knowledge and information systems vol. 51.2, pp. 339-367

[14] C. Truong, L. Oudre, & N. Vayatis (2019) "A selective review of offline change point detection methods" IEEE Signal Processing, vol. 167, 107299

[15] R. Bellman (1961), "Adaptive Control Processes, a Guided Tour", Princeton University Press, Chapters 1, 3, 5

[16] E. Tu, & D. Patterson (2016), "Interactive Exploration of Large Genomic Datasets", European Bioinformatics Institute, EECS Technical Report Series, PMID 29308454

[17] T. Hocking, G. Rigaill, P. Fearnhead, G. Bourque, "Generalized Functional Pruning Optimal Partitioning (GFPOP) for Constrained Changepoint Detection in Genomic Data", Preprint arXiv:1810.00117, Accepted for publication in Journal of Statistical Software

[18] H. Wickham (2009), "ggplot2: elegant graphics for data analysis", Springer New York

[19] Anscombe, F. J. (1973), "Graphs in Statistical Analysis," The American Statistician, vol. 27, pp. 17 – 21

[20] S. Kullback & R. A. Leibler (1951), "On Information and Sufficiency", The Annals of Mathematical Statistics, vol. 22, pp. 79 - 86

[21] S. Kullback (1959), "Information Theory and Statistics", John Wiley & Sons, Chapters 1-3

[22] Mark J. Schervish (1995), "Theory of Statistics", Springer's Series in Statistics, ISBN: 978-1-4612-4250-5

[23] J. Kiefer & J. Wolfowitz (1956), "Consistency of the Maximum Likelihood Estimator in the Presence of Infinitely Many Incidental Parameters", Annals of Mathematical Statistics, vol. 27, pp. 887-906

[24] P. Billingsley (1995), "Probability and Measure", ISBN 978-8126517718

[25] S. Nadarajah (2005), "A generalized normal distribution", Taylor & Francis, vol. 32, pp. 685-694

[26] A. Webb (2011), "Statistical Pattern Recognition", Wiley, ISBN 9780470682289

[27] A. Dempster, N. Laird, & D. Rubin (1977), "Maximum Likelihood from Incomplete Data via the EM Algorithm", Journal of the Royal Statistical Society, vol. 39 pp. 1 – 38

[28] N. Aronszajn (1950), "Theory of Reproducing Kernels", Transactions of the American Mathematical Society, vol. 68 pp. 337–404

[29] A. Gretton, K. Borgwardt, M. Rasch, B. Scholkopf, & A. Smola (2012), "A Kernel Two-Sample Test", Journal of Machine Learning Research, vol. 13, pp. 723-773

[30] A. Scott & M. Knott (1974), "A cluster analysis method for grouping means in the analysis of variance" , Biometrics, vol. 30, pp. 507 – 512

[31] P. Fryzlewicz (2014), "Wild binary segmentation for multiple change-point detection", Annals of Statistics, vol. 42, pp. 2243–2281

[32] B. Jackson et al. (2012), "An algorithm for optimal partitioning of data on an interval," IEEE Signal Processing Letters, vol. 12, pp. 105-108

[33] R. Kilick, P. Fearnhead, & I. A. Eckley, "Optimal Detection of Changepoints With a Linear Computational Cost", Journal of the American Statistical Association, vol. 107, pp. 1590 - 1598

[34] T. Cormen, C. Leiserson, R. Rivest, & S. Clifford (2009), "Introduction to Algorithms", MIT Press and McGraw-Hill, ISBN 0-262-03384-4

[35] E. Page, "Continuous inspection schemes", Biometrika, vol. 41, pp. 100–115

[36] C. Jie & A. Gupta (2012), "Parametric Statistical Change Point Analysis", Birkhäuser Basel, Springer Science & Business Media, ISBN 978-0-8176-4801-5

[37] J. Neyman and E. Pearson (1933), "On the problem of the most efficient tests of statistical hypotheses", Journal of the Royal Statistical Society, vol. 231, pp. 694-706

[38] Stone, M (1974). "Cross-Validatory Choice and Assessment of Statistical Predictions". Journal of the Royal Statistical Society, vol. 36, pp. 111 – 147

[39] D. Wilson (2019), "The harmonic mean p-value for combining dependent tests", vol. 116, pp. 1195 - 1200

[40] H. Akaike (1998), "Information Theory and an Extension of the Maximum Likelihood Principle", Springer, Selected papers of Hirotugu Akaike, pp. 199-213

[41] G. Schwarz (1978), "Estimating the dimension of a model" Annals of Statistics, vol. 6, pp. 461 – 464

[42] Stone, M (1977), "An Asymptotic Equivalence of Choice of Model by Cross-Validation and Akaike's Criterion", Journal of the Royal Statistical Society, vol. 39 pp. 44–47

[43] Y. Yao, "Estimating the number of change-points via Schwarz' criterion", Statistics & Probability Letters, Elsevier, vol. 6, pp. 181 - 189

[44] I. Auger & C. Lawrence (1989), "Algorithms for the optimal identification of segment neighborhoods", Bulletin of Mathematical Biology, vol. 51, pp. 39 – 54

[45] R. Maidstone, T. Hocking, & G. Rigail (2017), "On optimal multiple changepoint algorithms for large data", Springer, Statistics and Computing, vol 27, pp. 519–533

[46] S. Tickle, I. Eckley, P. Fearnhead & K. Haynes (2020), "Parallelization of a Common Changepoint Detection Method", Taylor & Francis, Journal of Computational and Graphical Statistics, vol 29, pp. 149-161

[47] P. Helman, B. Moret, & H. Shapiro (1993), "An exact characterization of greedy structures", SIAM Journal on Discrete Mathematics, vol 6, pp. 274–283

[48] N. Zhang & D. Siegmund (2007), "A modified Bayes information criterion with applications to the analysis of comparative genomic hybridization data", Biometrics, vol 63, pp. 22-32

[49] J. Kruskal (1956) "On the shortest spanning subtree of a graph and the traveling salesman problem", Proceedings of the American Mathematical Society, vol 7, pp. 48–50

[50] C. Zou, G. Yin, L. Feng, & Z. Wang (2014), "Nonparametric maximum likelihood approach to multiple change-point problems", The Annals of Statistics, vol 42, pp. 970–1002

[51] K. Haynes, P. Fearnhead, & I. Eckley (2017), "A computationally efficient nonparametric approach for change-point detection", Springer, Journal of Statistics and Computing, vol 27, pp. 1293–1305

[52] F. Massey (1951), "The Kolmogorov-Smirnov Test for Goodness of Fit", Journal of the American Statistical Association, Vol. 46, pp. 68-78

[53] Cramér, H. (1928). "On the Composition of Elementary Errors". Scandinavian Actuarial Journal, vol 1, pp. 13–74

[54] T. Anderson & D. Darling (1954), "A Test of Goodness-of-Fit", Journal of the American Statistical Association, vol 49, pp. 765–769

[55] J. Zhang (2002), "Powerful goodness-of-fit tests based on the likelihood ratio", Journal of the Royal Statistical Society, vol 64, pp. 281-294

[56] J. Einmahl & I. Mckeague (2003), "Empirical likelihood based hypothesis testing", Bernoulli, vol 9, pp. 267 - 290

[57] K. Haynes, "changepoint.np: Methods for Nonparametric Changepoint Detection", R package version 0.0.2

[58] P. Fearnhead & G. Rigaill (2020), "Relating and comparing methods for detecting changes in mean", Wiley, Stat, vol 9

[59] G. van-den-Burg & C. Williams (2020), "An Evaluation of Change Point Detection Algorithms", arXiv preprint arXiv:2003.06222

[60] Wolpert, D.H., Macready, W.G. (1997), "No Free Lunch Theorems for Optimization", IEEE Transactions on Evolutionary Computation, vol 67

[61] F. Laio (2004), "Cramer–von Mises and Anderson-Darling goodness of fit tests for extreme value distributions with unknown parameters", Wiley, Water Resources Research Vol. 40 Issue 9

[62] S. Wilks (1938), "The Large-Sample Distribution of the Likelihood Ratio for Testing Composite Hypotheses", Annals of Mathematical Statistics, vol 9, pp. 60 - 62

[63] M. Endres & J. Schindelin (2003), "A new metric for probability distributions", IEEE Transactions, Information Theory, vol 49 pp. 1858–1860

[64] S. Liu, M. Yamada, N. Collier, & M. Sugiyama (2012), "Change-Point Detection in Time-Series Data by Relative Density-Ratio Estimation", Springer - Structural, Syntactic, and Statistical Pattern Recognition, vol 7626

[65] D. Eddelbuettel & R. Francois (2011), "Rcpp: Seamless R and C++ Integration", Journal of Statistical Software, vol 40, pp. 1–18

[66] https://github.com/karimfarahat1/NPL-BS

[67] D. Musser (1997), "Introspective Sorting and Selection Algorithms", Software: Practice and Experience, vol. 27, pp. 983 – 993

[68] Mersmann O (2019), "microbenchmark: Accurate Timing Functions", CRAN package version 1.4-7

[69] N. James & D. Matteson (2014), "A nonparametric approach for multiple change point analysis of multivariate data", Journal of the American Statistical Association, vol 109, pp. 334–345

[70] P. Fenwick (1994), "A new data structure for cumulative frequency tables", Software: Practice and Experience, vol 24, pp. 327 – 336

[71] S. Arlot (2019), "Minimal Penalties and the Slope Heuristics - A Survey", Journal de la société française de statistique, vol 3, pp. 1 - 106

[72] K. Frick, A. Munk, & H. Sieling (2014), "Simultaneous Multiscale Change-Point Inference", Journal of the Royal Statistical Society, vol 76, pp. 495-580

[73] G. Marsaglia (2004), "Evaluating the Anderson-Darling Distribution", Journal of Statistical Software, vol 9, pp. 730–737

[74] A. Ramdas, N. Garcia, & M. Cuturi (2017), "On Wasserstein Two-Sample Testing and Related Families of Nonparametric Tests", Entropy, vol 19

[75] S. Goerg & J. Kaiser (2009), "Nonparametric Testing of Distributions - the Epps–Singleton Two-Sample Test using the Empirical Characteristic Function", The Stata Journal, vol 9, pp. 454 - 465